An Inter-disciplinary Study on Open Source Software Development in Developing Countries A Case Study of Chinese Linux

Yinhua Zhou

Doctor of Philosophy

The University of Edinburgh

2012

Declaration

I hereby declare that this thesis has been composed by myself, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institutes of higher learning, except where due acknowledgment has been made in the text.

Yinhua Zhou

Abstract

This research provides a detailed account of Open Source Software (OSS) development in the context of developing countries (DCs) by exploring the specific case of Chinese indigenous Linux design and development. It builds an interdisciplinary, socio-technical, analytical framework from the perspective of science and technology studies (STS), in particular the social shaping of technology (SST), infrastructural studies and international technology transfer. It also covers the fields of economic analysis, policy studies and development studies. The research investigates the adaptation process of a unique OSS with infrastructural features – Linux in the context of China by conducting case studies on both embedded Linux and platform Linux products developed by two Chinese Linux providers.

Drawing upon the concepts developed in the SST perspective and infrastructural studies, this research addresses both the dynamism and continuity of OSS. In order to avoid the shortcomings of existing social scientific study on OSS, we applied social and biography of artefacts (BoA) approaches to examine the evolution of Chinese Linux by mapping out the key actors, investigating the linkages between them, and probing deeply into the intricate interplays among these actors over time.

A detailed longitudinal and contextual analysis has been undertaken through a qualitative historical case study of the evolution of both Chinese embedded Linux and platform Linux from 1998-2008. The empirical data reveals that the local adaptation and further innovation of Chinese Linux is a ‘generification’ process, i.e. a process of design and developing generic Linux solutions for diverse local users. Theoretically, the understanding of the socio-technical interfaces of the software (seeking, identifying, categorising local users/intermediaries, as well as collaborating with key players associated with the particular software) are central elements for software technology transfer and local technological capabilities building.

The findings also throw the light on the crucial importance of government role in providing incentives and institutional measures for Linux adaptation in China. In particular, it highlights the challenges concerning the socio-technical specificities of infrastructural software, like Linux OS (operating system) and the particular relevance to DCs as technology adapters.

Finally, this study throws light on the policy and practice for China’s future Linux development, and the implications for other DCs.

Acknowledgements

I would like to thank the people who have helped in the creation of this PhD thesis. There are many who should be mentioned, and as usual there is not enough room to mention them all. Therefore, to begin with I would like to thank all the people who are not named below but who have been very helpful in creating this thesis.

In the first place, I would like to thank Dr. Xiaobai Shen and Dr. James Stewart for guiding me through the PhD process with tremendous support and encouragement.

Special thanks are dedicated to Mr. Kenneth Wright for reviewing and improving my English writing. I also would like to thank the organisations and informants in China for their cooperation and contributions to my PhD research.

Last but not least, I would like to thank all the members of my family for their constant love and both financial and moral support

Abbreviations

ASP	Application Service Provider
API	Application Programming Interface
BoA	Biography of Artefacts
BSD	Berkeley Software Distribution
BUAA	Beijing University of Aeronautics and Astronautics also known as Beihang University
CAS	Chinese Academy of Science
CASTED	Chinese Academy of Science and Technology for Development
CCB	China Construction Bank
CCID	China Centre for Information Industry Development
CCIDVC	China Centre for Information Industry Development Venture Capital
CESI	China Electronic Standardisation Institute
CPC	Communist Party of China
CS2C	China Standard Software Co. Ltd
CS&S	China National Software and Service Co. Ltd
CSIS	Centre for Strategic and International Studies
DC	Developing Country
DCC	Data Centre Consolidation
DWDAS	Dynamic Website Development and Administration System
EDK	Embedded Development Kit
ERP	Enterprise Resource Planning
EU	European Union
FLOSSPOLS	Free/Libre/Open Source Software: Policy Support
FPS	Frames per Second
GNU	"GNU's Not Unix
GPL	General Public License

GUI	Graphic User Interface
IC	Integrated Circuit
ICT	Information and Communication Technology
IDC	International Data Corporation
IDE	Integrated Development Environment
IPC	Intellectual Property Centre
ISCAS	Institute of Software in Chinese Academy of Science
IT	Information Technology
LUPA	Leadership of Open Source University Promotion Alliance
MIT	Massachusetts Institute of Technology
MNE	Multinational enterprise
MS	Microsoft Corporation
NC	Network Computer
NCRE	National Computer Rank Examination
OA	Office Application
OS	Operating System
OSS	Open Source Software
PC	Personal Computer
Qualipso	Quality Platform for Open Source Software
R&D	Research and Development
SCOT	Social Construction of Technology
SDK	Software Development Kit
SETC	State Economic and Trade Commission
SLTI	Social Learning in Technology Innovation
SME	Small and Medium Enterprise
SSICT	Social Shaping of Information and Communication Technology
SST	Social Shaping of Technology
STS	Science & Technology Studies

VOD	Video on Demand

Figures and tables

Contents

Chapter 1 Introduction

1.1 Focus and Scope

Over the last decade, open source software (OSS) has attracted wide attention from not only the business communities, but also governments, in particular in developing countries (DCs). The understanding of OSS phenomenon however, was theorised mainly from the social scientific studies conducted in developed world, in particular the United States. The researchers have probed deeply into the managerial structures of OSS communities (Demil and Lecocq, 2006; von Krough and Spaeth, 2007), the reasons for programmers participating in OSS design (Bonaccorsi and Rossi, 2003; Campbell-Kelly and Garcia-Swartz, 2009), the licenses of OSS (Edwards, 2005; von Krough and Spaeth, 2007), and the competition between OSS and traditional proprietary software by building mathematic models (Hicks and Pachamanove, 2007; Economides and Katsamakas, 2006), etc. There are also some studies related to the open source movement in the context of DCs. They paid most attention to the benefits that can be reaped by implementing and applying OSS (Cook and Horobin, 2006; Camara and Fonseca, 2006; Pan and Bonk, 2007). We argue that existing social scientific research on DCs' open source movement is over-simplified and based on a naïve view: a globally designed OSS can easily be applied in a context of DC.

However, the open source movement in DCs can be seen as the continuous process of global open source movement, in which most overseas-designed OSS was transferred from abroad and adapted in the context of DCs. From a perspective of international technology transfer, transferring technologies is not a simple spatial move; instead local adaptation of technologies in DCs requires a great deal of efforts, both technically and socially (Shen, 1999; Hobday and Rush, 2007; Gallagher, 2006). Apart from the roles of technology transferors and transferees, other actors such as R&D actors, government, and financial agencies are also crucial for successful technology transfer.

Consequently, one shortcoming of research design for existing studies of OSS in DCs is that it focuses mainly on OSS implementation, and therefore overlooks the local design and further innovation process. The other shortcoming is that the broader social context in DCs was nearly neglected. There are many non-technical elements that cannot be changed by OSS suppliers and users, which resulted in unpredictable tensions and dynamics surrounding OSS adaptation. These two shortcomings of the methodological framework create an unsatisfactory understanding of the open source movement in the context of DCs.

Furthermore, in DCs, the reality is that the OSS is introduced to the domestic market with government backing. However, there is a dichotomy on government intervention: government should not issue any particular policy to support OSS; and government should involve in the open source movement by employing wise measures. This phenomenon calls for further and deeper research on government intervention.

In order to tackle shortcomings and examine the government role, a new research framework is needed, which can link both OSS design and implementation processes in DCs over a long-term, and address the social elements in a broader context.

The objective of this study is not about the ideology behind the open source movement, but the uptake of OSS in the context of a DC. The study illustrates Chinese indigenous Linux evolution as case studies, and gives a detailed account on the process of its adaptation and further development. In particular, attention is given to the supplier’s strategy and effort for cultivating the ‘generic Linux solution’ for Chinese local users. This research is based on two empirical cases: one is Linux developed in China as an operating system (OS) for embedded computing system (Chapter 4); the other is Linux adapted as a general-purpose OS – platform Linux for traditional computers, such as PC and server computers (Chapter 5).

Drawing upon the social shaping of technology (SST) perspective (MacKenzie and Wajcman 1985; Williams and Edge 1996), an inter-disciplinary methodological framework is built, in order to carry out this study through a longitudinal and contextual analysis of Chinese indigenous Linux evolution (chapter 3). Both the interplays between suppliers and other social groups and the broader context in which Chinese Linux is designed and used are taken into account.

This introductory chapter introduces the terms of OSS and Linux, and continues to introduce the social and technological context of this study by briefly describing the Chinese traditional socialist economic and science & technology (S&T) systems and China's later transition. This discussion is followed by an explanation of the historical background of Chinese OS development. Finally, this chapter ends by introducing the structure of this thesis.

1.2 Open Source Software and Linux

Software is a computer program which offers a set of instructions to control computers' operations. It is written by humans but implemented by computers so that the software exists into two forms: source code and object code. Source code is written in a computer language which is readily human-readable for experienced programmers. Object code is the machine-executable form which is binary code (a sequence of zeros and ones) compiled from source code. The machine code is difficult for human to read and to translate into source code as well (Lerner and Tirole, 2005; Miller, 2002). For a given piece of program, source code can be regarded as the knowledge which is considered a valuable resource, rare, inimitable, and non-substitutable and that contributes to sustain competitive advantages. By this premise, source code is a scarce resource under protection by its developer. Traditional proprietary software firms make profits from selling software so they must protect their intellectual property rights and maintain the trade secrets of how the software works to prevent others from copying or modifying the software. Most suppliers today keep source code as a secret during software distribution and only provide users with object code.

In contrast to the traditional proprietary software, OSS enables the user and fellow computer programmers to access and modify the source code. Prior to the emergence and clear success of OSS projects, it was assumed that a requirement to voluntarily contribute one's innovation to the public goods would destroy the incentive to innovate and inevitably lead to market failure. However, OSS communities attracted thousands of voluntary programmers to participate and make contributions to OSS development and innovation. Some OSS has become the competitive alternatives to proprietary software, such as GNU/Linux, freeBSD in the OS field, Apache in the web server area, OpenOffice in the office productivity suite domain, MySQL in the database field, etc.

OSS has nowadays been widely adopted in various fields and countries. A report from International Data Corporation (IDC) – "European End-User Survey: 2005 Spending Priorities, Outsourcing, Open Source, and Impact of Compliance" shows that 73.3% of firms adopted OSS into their business life. Walli, Gynn and Von Rotz (2005) reported that 87% of their surveyed companies in the United States were using OSS. In Asia, Linux accounted for 14% of servers and 5% of PCs in 2004 in the Asia-Pacific region (Ghosh, 2006).

One extremely interesting phenomenon is that many powerful multinational enterprises (MNEs), like Hewlett-Packard, IBM, Intel, Novell, Oracle, SAP, Sun Microsystems, LENOVO, etc. joined the forces in making significant investment to promote OSS (Shen, 2005b). For example, IBM developed hundreds of open source projects ranging from OS to complementary application software; Sun Microsystems shifted their new version OS – Solaris v. 10 to open source strategy. In addition, Microsoft released several pieces of source code as OSS.

Not only did firms pay attention to OSS, but also governments have a strong interest in the deploying of OSS. According to a CSIS's (centre for strategic and international studies) publication, there were 265 policy initiatives towards open source around the world by the end of 2005 (CSIS, 2006). At present, OSS is widely employed in the public sector in developed countries, e.g.79.3% of 995 public sector organisations in 13 EU countries have employed open source software by the end of 2004 (FLOSSPOLS Government Survey 2005). Furthermore, governments in the developed world have also conducted research on OSS development. For example, the European Commission granted €10.42 million euros to the Qualipso project (under the EU's sixth framework) which is a collaboration between four countries from the EU, China, Japan and Brazil (http://libresoft.es/node/250 and http://www.qualipso.org/).

Additionally, both government and local business entities in DCs are committing to developing OSS to suit their own purposes. Take China for example, in line with China’s software industry development strategy, some Chinese domestic companies and MNEs, encompassing IBM, Hewlett-Packard, Intel, Novell, Red-Flag, CS2C (China Standard Software Co. Ltd), Kingsoft, etc. formed the “China Open Source Software Promotion Alliance” to encourage the development of China’s OSS industry (Legard, 2005; Marson, 2005); over 30% of Chinese Linux income has come from government procurement since 2003; the Chinese government also invested in a great number of research projects on OSS related technology R&D, in particular Linux and open source office suite.

Linux perhaps can be seen as a representative for OSS in many DCs. In traditional OSS literature, in particular from a social science perspective, Linux is a loose term. Some research is only based on the ‘Linux kernel’ which is a core component of Linux OS; others regard Linux as a whole OS. Strictly, Linux does not refer to a completed OS but a well-defined OS kernel. The kernel is the central component of computer OS, which schedules tasks, including the execution of end-user applications by allocating a computer's system resources to the programs in execution. It can be thought of as the bridge connecting the application software to the hardware of a computer. The detailed functions of the kernel are handling process management and scheduling, inter-process communication, device Input/Output, and memory management for OS (Casadesus-Masanell and Ghemawat, 2006; Lee and Cole, 2003). Apart from the kernel, an OS consists of several components or modules, like user interface, system tools, network tools, compilers, etc (Valimaki and Oksanen, 2005; Camara and Fonseca, 2007). Therefore, Linux OS theoretically should be called GNU/Linux since it is a combination of software components from “Free Software Foundation” and the Linux kernel developed by Linus Torvalds (Varian and Shapiro, 2003). This research examines the process of adapting Linux in the Chinese context as a whole OS which is ready to be adopted by Chinese users.

1.3 Context – China in Transition

Drawing upon the experience of the Soviet Union, China established a highly centralized planned economy system in the 1950's. This economy system concentrated the resources of the whole country to facilitate on the development of heavy industries by a planned resource allocation system, when China was a capital scarce economy. Although the planned system made remarkable achievements in the development of Chinese heavy industries, its weaknesses were also apparent over time, such as distorted industrial structure, poor work incentives, and low allocative efficiency (Lin, Cai and Li, 2002). Under the traditional highly centralized planned economy system, the manufacture, supply and consumption of products were under the control of government, rather than market demand so that technological innovation was neglected without market competition. Furthermore, all the research institutes and universities were established and controlled by the government. Their research projects were completely organised and financed by the government, which focused mainly on theoretical research. As a result, both the enterprises and research institutes/universities paid little attention to market demands. Concentrating much on theoretical research resulted in a gap between industry, and research institutes and universities.

China therefore, decided to reform its national economy system by introducing market mechanisms when the third Plenary Session of the Central Committee of the CPC (Communist Party of China) was held in Beijing in 1978. A major objective of economic reforms is to promote dynamism in the technological innovation system. In 1985, the Chinese government published "Decision on the Reform of the Science and Technology Management System". The Decision was seen as the beginning of Chinese science and technology (S&T) reform which continues today. Based on the past ten years' experience of reform, CPC Central Committee and State Council announced "Decision on Accelerating Science and Technology Development". This document emphasised the important role of S&T in economic growth and further recognised the priority position of information and communication technology (ICT). It also encouraged indigenous enterprises to conduct research & development (R&D) activities according to market demands, and establish close ties with research institutes and universities.

1.4 Government Role in OS and Linux Development

China's government has a long tradition in intervening in indigenous OS research and development. Chinese public research institutes and universities started developing OS in the 1960s. For example, the GX-73 OS developed for Aerospace Ocean-going Measuring Fleet and the YHOS (Yin-He OS) for the Galaxy-I (Yin-He I) Supercomputer. However, these indigenous OS development projects were initiated and supported by the Chinese government under the traditional planning economic system (Wang, 2010). The purpose of designing and developing them was either for scientific research or specific hardware platforms, rather than for the mass market. Therefore, all Chinese indigenous OS products were too advanced for Chinese consumers because they were not compatible with mainstream computer hardware structures, such as the most popular X86 architecture. From a marketing perspective, this is the main reason why China's OS market was dominated by foreign proprietary products.

Since China's "9th Five Year Plan[1]" (1996-2000), IT attracted much attention from the government. Some government agencies established their own websites to collect, exchange and publish information. In order to provide more convenient services for citizens, the Chinese government initiated a series of projects to develop e-government systems. Influenced by a slogan created by the Chinese government – "driving industrialization through informatisation[2]", both stated-owned and private enterprises began to apply

[1] Influenced by the success of Soviet Union's Five Year Plans, China introduced her own Five Year Plan from 1953 in order to boost national industries.

[2] The term 'informatisation' is one of a most common word used in China, which means the modernization of China's IT infrastructure and applying IT in various sectors of national economy.

information technology (IT) to their business processes. Soon after, the information industry was perceived as the fundamental industry of the whole national economy.

The software industry is the core of the information industry and its forceful development bears great significance. Software is characterised by strong penetration: it can be applied in all traditional industries and sectors of national economy. As a result, software can accelerate the renewal and upgrading of industries and improve production efficiency considerably. By acknowledging the pivotal role of the software industry, China is committed to establishing an independent software industry.

A need to develop a self-controlled OS was recognised by both China’s software industry and the Chinese government. Chinese users, in particular government users, on the one hand put forward more rigorous requirements for information security; and on the other hand expressed distrust in the existing information system, which was based on ‘black-boxed’ OS provided by foreign companies. Professor Ni (1999), a senior member of the central government think-tank, suggests that the information security of a computer system is subject to the security of the OS on which the whole system was built.

Nevertheless, China’s software technology, and higher education in computing and software R&D, has lagged far behind western countries. Both indigenous software enterprises and public research institutes and universities had no intellectual property (IP) on any OS, and their software R&D activity was merely limited to applied research. All these factors are inevitable barriers for developing a self-controlled OS. Therefore, the Chinese government and indigenous software enterprises decided to develop own OS based on Linux.

I conducted a qualitative study of the Chinese government policies towards Linux (Chapter 6). This research tries to understand the role of the Chinese government in supporting indigenous Linux, the detailed administrative measures adopted by government and the wider implications which can be drawn from this study for governments in DCs. By drawing upon the concepts of infrastructure studies, I introduced the term ‘infrastructural software’ to provide understanding of social elements embodied in a special type of OSS –

Linux. Platform Linux can be regarded as an infrastructure because it can be developed as a shared resource to support various computer-based applications and for achieving the informatisation of China. For DCs, developing Linux requires strong government support by linking players and coordinating their resource in order to design appropriate Linux products for local user communities and competing with foreign products.

1.5 Research Questions

Based on the introduction above, the following research questions have been developed: the first group of research questions is designed to investigate the dynamism and complexity of Chinese Linux's adaptation and innovation, map the main actors involved and the interactions between these actors; the second group of questions focuses on an academic debate in which one voice is against any government intervention in the open source movement, while the other side favours government supports toward OSS; the last question is about the implication of Chinese Linux adaptation, which can be drawn on by other DCs.

1. How has Linux adapted in China's broader context?

(1) Which struggles and tensions did the Chinese suppliers face through the development of Chinese Linux? Which strategies did they use to enable them to deal with these problems?

(2) How did the Chinese socio-economic context and institutional arrangements influence the way in which Chinese Linux was adapted and developed?

2. What is the role of the Chinese government in Linux adaptation and development?

(1) Are government interventions necessary for Chinese Linux adaptation and development?

(2) What are the political measures adapted by the Chinese government?

3. What are the wider implications that other DCs can draw from China's experience in relation to adapting an infrastructural OSS into their local environment?

1.6 Structure of the Thesis

This thesis is divided into eight chapters. Chapter 1 provides a general introduction to this thesis.

Chapter 2 selectively reviews the OSS literature and discusses the issues which are associated with our study on OSS, as well as the literature from Technology Studies area, aiming to form the basis on which to build a research framework. This chapter points out that social scientific research on OSS development in DCs is still lacking and the long-term evolutional history of OSS needs to be further discussed. In order to frame the analytical framework, the review of Technology Studies is driven by a concern for adopting the concepts and theoretical tools developed from the SST perspective, infrastructural studies, and technology transfer.

Chapter 3 describes the research design and the methodology of this study. Firstly, it elaborates an integrated analytical framework and the way to use it, as well as the rationale of case selection. Secondly, this chapter presents the data collection and process by linking the research questions. The chapter ends with a discussion of how to conduct field work in China.

Chapter 4 examines the historical data of Chinese embedded Linux development. It starts with the general background which provides details for understanding the particular technical features of embedded systems and OS. The remaining sections are organised according to the biographical stages of embedded Linux by which we are able to trace the development history of Linux and explore the interactions between key players in milestone projects over time. The aim of this chapter is to demonstrate a detailed account of the generic Chinese embedded Linux solution development in relation to its supplier's ambition and strategies.

Chapter 5 presents the biography of another type of Chinese indigenous Linux – platform Linux. It illustrates how the Chinese supplier CS2C managed the dynamics and tensions surrounding the platform Linux development as an infrastructure in China. Although Linux is transferred in China as a generic OS, the biography of Chinese platform

Linux shows that the supplier still requires addressing the particular local work practices and everyday life in order to compete with foreign proprietary OS. Many generic platform Linux solutions are designed to fit various user groups. This chapter also shows that the Chinese government played an active and crucial role in every biographical stage of Chinese platform Linux localisation and further innovation.

Chapter 6 explores the initiatives and administrative measures of the Chinese government to support indigenous Linux, in particular platform Linux. This chapter uncovers that government supporting measures varied over time, which reflected a learning process of policy-making.

Chapter 7 is an analytical chapter. It links the empirical research findings to the theoretical framework and concerns of this study, and then presents the main findings. It discusses that the adaptation of Linux is a ‘generification’ process in which suppliers have to deal with particular requirements of Chinese local users by employing different strategies over time. As a result, understanding and categorising users, as well as enrolling them in the design is an essential element for local technological capabilities building, in order to transfer foreign software successfully. Additionally, the role of the Chinese government is also important for Linux adaptation. At each biographic stage, the Chinese government issued different policies and employ different measures to favour Linux development. However, not all of these policies and measures were effective. This research therefore also suggests that the government should choose wise polices.

Chapter 8 is the final chapter which presents the overarching conclusion of this study. A few limitations that resulted from the research interests, research strategy and data collection of this study is discussed. The reflections on these limitations can make contributions to the possible future research. This chapter also presents the wider implication which draws from the case studies both for China’s future S&T policy-making and for OSS development in other DCs.

Chapter 2 Literature Review

2.1 Introduction

Chapter 2 describes the intellectual context of this study, including the academic debates on existing OSS literature and theoretical tools based on the research framework adopted herein. Generally, this research spans two main disciplinary areas: OSS studies and technology studies. Section 2.2 reviews the mainstream literature relevant to the social science study of OSS and explores the academic debates related to the empirical cases of Chinese OSS – Linux adaptation and development. Section 2.3 is focused on selected literature from technology studies. First, this section examines the range of relevant approaches within technology studies, especially the idea of SST and SSICT (social shaping of information and communication technology) perspectives, and social learning and the BoA approach. Second, it addresses the conventional perspectives on and analytical approaches toward infrastructural studies and ends with the discussion of international technology transfer and local indigenous technological capabilities. Section 2.4 discusses the current academic debates and empirical theoretical limitations as the bases for research questions presented in Chapter 3.

2.2 Open Source Software (OSS)

2.2.1 Ideology behind the open source movement

The open source movement emerged in the context of software developers becoming overprotective of their work, as associated knowledge tends to be enclosed in the software. The advocates of the open source movement claim that restricting the use of software and not sharing the source code is unethical (Schmidt and Schnitzer, 2003). The ideology of the open source movement can be traced back to hacker ethics. The early hackers in

Massachusetts Institute of Technology (MIT), which is known as the cradle of hacker culture (Linus, 1998), believed that sharing information was a virtue and that there existed a moral duty to disseminate one's knowledge in the form of computer programs and to further develop the infrastructure where the programmes are used. As a result, they made the source code of their software available to fellow members of the computing community (Nikulainen, 2004) and knowledge sharing was established as an essential foundation for the open source movement.

In contrast to proprietary software which keeps the source code secret, OSS is the software for which the source code is open; it is therefore freely available to the public: everybody has the right not only to use the software, but also to extend it, to adapt it to individual needs, and to redistribute the original or modified software to others. OSS involves a copyright-based license to keep private intellectual property claims out of the way of both software innovators and software adopters while at the same time preserving a common software code that everyone can access (O'Mahony, 2003). Open source, by nature, opens its development process and operation to a variety of worldwide knowledge centres or ideas that could potentially improve the innovation, and then benefits from boundless distributed knowledge (Awazu and Desouza, 2004). In relation to OSS development, knowledge sharing comprises a wide range of exchanges among peer developers revolving around a number of epistemic tasks such as bug fixing, code writing, new-feature implementation and the like (Kuk, 2006).

OSS communities consist of people who contribute to the public good of OSS by writing code for the project. These contributors can distribute their works widely and essentially without cost via the Internet (von Krough and von Hippel, 2006).

2.2.2 Actors' strategies in and governance structure of the open source movement

Motivations of OSS developers

Any developer can make returns from investing in or developing traditional commercial software using a regime of intellectual property rights. On the contrary, OSS licenses guarantee the source code of OSS is freely available; everybody has the right not only to use the software, but also to extend it, to adapt it to meet users' own needs, and to redistribute the original or modified software to others. A wide array of studies has dealt extensively with the question of the incentives and motivations that lead individuals to make contributions to OSS.

The phenomenon of voluntary developer's contributions to OSS projects is a puzzling and fascinating subject that has spurred numerous theories. According to von Hippel and von Krough (2003), participants in OSS development obtain private benefit from writing code for their own use, sharing their code, and collectively contributing to the development and improvement of software. These private benefits have been tied to intrinsic or extrinsic rewards (Kogut and Metiu, 2001). Intrinsic rewards are related to altruism and gift economies (people receive satisfaction from gift-giving, especially in OSS communities, where they feel a kind of 'kinship amity') and 'fun' challenges (open source programming sometimes is regarded as an artistic satisfaction associated with solving complex computer problems) (Zeitlyn, 2003, Sanders, 1998).

However, it is very unlikely that intrinsic motivation might be sufficient to explain the behaviour of people who have devoted considerable resources of time and intellect into OSS development (Bonaccorsi and Rossi, 2003). The need to customise and improve software to meet developers' particular requirements is the most frequently cited reason related to extrinsic incentives (Raymond, 1999; Hertel *et al*, 2003; Frank and von Hippel, 2003). Lerner and Tirole (2002) develop theoretical frameworks by adopting labour economics in their most cited paper. They suggest that the motivation of voluntary participants in open source projects is a ‘signalling incentive’ which encompasses ego gratification and career concern. The ego gratification incentive stems from a desire for peer recognition. The career concern incentive refers to a programmer solving a difficult problem, or writing and sharing high-quality software in the public domain, thereby signalling exceptional abilities to the outside world. Peer recognition can result in better job offers from current or prospective employers, invitations to participate in commercial open source projects, or easier access to the venture capital market in starting one’s own business.

Consistent with the theory of the signalling incentive, Dalle and David (2003) reinforce that the signalling incentive was not an inducement to all contributors, but only to sophisticated users and IT professionals working on technically important and sophisticated modules, and to those project entrepreneurs initiating important and prevalent open source projects. Likewise, Roberts *et al* (2006) verify that extrinsically motivated contributors provided more substantive contributions to the project than those developers motivated by intrinsic incentives.

Schmidt and Schnitzer (2003) find that some programmers devote time to develop OSS as part of assignments from their employers. Some Linux distribution suppliers, like Red Hat, provide products and service that are complementary to Linux but that are not efficiently supplied by the Linux or other OSS communities. In order to guarantee that their commercial products and service are complementary to OSS, these firms have an incentive to invest in the development of OSS. Thus, Red Hat gains access to a larger market as Linux becomes more widely adopted. Similarly, some IBM developers also received payment to develop OSS. Due to also being complementary to Linux, various IBM middleware began to be run on non-IBM equipment, which generated significant software revenue for IBM (Capek, Frank, Gerdt and Shields, 2005; Campbell-Kelly and Garcia-Swartz, 2009).

Governance structure

Research has revealed that distributed contributors worldwide have participated in open source projects in various ways according to their diverse interests (Frank and Jungwirth, 2003; Koch and Schneider, 2002). As a result, open source projects have often been described by the media as anarchical communities, which has prompted researchers to examine the governance structure of the open source projects.

The structure of open source projects is depicted as a 'decentralized bazaar', as opposed to the 'cathedral' of typical commercial business product development, in Eric Raymond's manifesto – "The Cathedral and the Bazaar" (1999). In the paper, Raymond states that the bazaar offers a superior model for software development because the 'software crisis[3]' can be solved in effect. In the decentralised bazaar model, there is no central administrative authority with which to negotiate, so the software development is self-organising: the modular nature of open source software projects enables any group of individuals and organisations worldwide to develop software for diverse purposes, to

[3] The tasks of adding new features, adding support for new hardware devices and platforms, system tuning, and defect fixing, all become more difficult as a system ages and grows.

contribute debugging efforts, as well as to provide results to the public under certain open source licenses (Raymond, 1999; Tovalds, 1999; Evers, 2000). For example, Linux has a small and compact kernel that implements the basic operating tasks such as recognising input from the peripheral, sending output, managing files and directories on the disk, and controlling peripheral devices. The system capabilities can be extended by independent modules to meet particular jobs. Therefore, one group of developers can work on the Linux kernel while other groups develop the modules, allowing a multifunctional system without interfering with each other.

However, the bazaar metaphor is criticised by authors who stress that OSS development is not completely decentralized to this to extreme. The existence of a mixed model (bazaar and cathedral) can be identified in OSS project management (Bezroukov 1999; Connell 2000; Demil and Lecocq, 2006). Connell argues that the bazaar asserted by Raymond does not exist. In fact, open source projects use strong central control, which is crucial to their success. To support his viewpoint, Connell (2000) illustrated Linux as an example: "*Linus Torvalds made all major decisions, assigned subsystem to a few trusted people, resolved conflicts between competing ideas, and inspired his followers. The Linux projects were built by single, strong architects with lots of help. Cathedrals were guided by one person, over many years, with inexpensive help from legions of workers.*" Bezroukov (1999) emphasises that large projects like OS projects require that the core of the system (kernel) develops in a highly centralised (cathedral) fashion. In the meantime, the activities of developing add-on features and application modules are classified as belonging to the decentralised (bazaar) model. Some kind of hierarchical structure and coordination is necessary for carrying out open source projects while preventing forking or abandonment. Thus, a credible leader or small core group of developers typically shepherd software design, identification of crucial problems and appropriate solutions as well as the release of new software versions (Lerner and Tirole, 2001, 2003; Bonaccorsi and Rossi, 2003; von Krough and Spaeth, 2007).

2.2.3 The diffusion of OSS

The growing research on diffusion of OSS has mostly focused on two main areas: the network externalities and the competitive dynamics between OSS and commercial software vendors.

Software, in particular OS, is a networked technological product exhibiting strong effects of network externalities (Bresnahan, 2001; Brynjolfsson and Kemeter, 1996; Cowan, 1992). Network externalities arise if there is a proportional relationship between the adoption of a product by current and potential customers: increased adoption benefits existing users and increases the incentive to adopt. The software market is characterised by both direct and indirect network externalities: the former implies that users can obtain more benefits from using a software product with an increasing number of other users purchasing and using the same or compatible software; indirect network externalities is associated with compatibility, which refer to the utilities that users derive from using a software, which increases as more plentiful complementary application software is available in the market (Katz and Shapiro, 1985; Economides, 1996; Schmidt and Schnitzer, 2003).

Bonaccorsi and Rossi (2003) explain the widespread diffusion of OSS in an environment dominated by established proprietary standards. On the one hand, they find that diffusion of OSS contradicts the theory derived from economic analysis of standards, which states that the diffusion phenomenon is subject to the path dependence and lock-in effect (Arthur, 1994). On the other hand, the authors state that efficient diffusion of OSS in an established software standards dominated market greatly depends on the sources of network externalities, such as critical mass of consumers (also see Bresnahan and Greenstein, 1999). Developing a model to study the competition between open source and proprietary software in the market with an emphasis on the demand side, Lin (2007) suggests that the significant portion of users with high skills which enable them to customise the OSS to satisfy their needs is the key condition for OSS surviving in strong networks where externalities influence the market. Analysing data from multiple sources on a specific OSS – Freenet development, von Krough, et al. (2003) shows that a critical mass of user-contributors may sustain OSS innovation. The existence of compatibility is not only identified as an important reason for achieving critical mass of users of OSS products, but is also a pivotal factor affecting network externalities of OSS directly. In contrast to the giants in the software market who have already achieved network externalities for their products and sustained market dominance by limiting compatibility with other products from rivals, OSS can enhance its network externalities by offering broad compatibilities (Hicks and Pachamanova, 2007).

Because OSS and commercial software are likely to continue to coexist in the software market, the direct competition between the two paradigms has received a fair amount of attention. Kuan (2001) analyses the competition between proprietary and open source software by modelling the decision of agents choosing between buying software and producing software based on OSS. He finds that the OSS would be more valuable if the agents were programmers. Bitzer (2004) investigates the question of why some commercial proprietary software producers support OSS development and others do not. The level of support tends to be related to the level of heterogeneity between the developed products and OSS. For example: the firms producing software with low heterogeneity to Linux, such as proprietary variant of UNIX which have same or similar commands, network protocols, file systems, etc. to Linux would like to support Linux, while vendors which developed software with high heterogeneity to Linux, like MS-Windows do not support Linux. Casadesus-Masanell and Ghemawat (2006) analyse the competition between OSS and proprietary software in a dynamic mixed duopoly with demand-side learning. They suggest that if this demand-side learning is strong enough, then OSS will be higher quality and will replace conventional software eventually.

Mustonen (2003) finds that monopolists supplying commercial proprietary software face the constraints in the programmer labour market and competition from a substitute OSS in the consumer market. A sufficient low implementation cost of OSS is crucial for competition with proprietary software: it compels the monopolist to reduce the price of its product, and generates entry barriers to monopolists in certain market segments. On the other hand, Economides and Katsamakas (2006) investigate the competition between proprietary systems (e.g. MS-Windows) and a system based on open source platform (e.g. Linux) both in the platform market and applications market, and show that the proprietary system most likely dominates market share and profitability, even though the cost of switching to Linux is zero. This finding explains Microsoft's dominance in the OS market: the demand for Windows is larger than the demand for Linux because Microsoft controlled most of the significant applications (such as Office productivity software and PC games) for users.

2.2.4 OSS license and corresponding business model

A software license comprises the permissions, rights and restrictions imposed on software use. Under a software license, the licensee is permitted to use the licensed software in compliance with the specific terms of the license. Software licenses are classified in two main types: proprietary (closed source) and open source.

Proprietary software is typically licensed under very restrictive conditions, which only offers the right to use a piece of software. To protect intellectual property rights and maintain trade secrets in the software industry, incumbents distribute the software as closed source. In general, any copy, distribution or modification on software is forbidden (Edwards, 2005; Schmidt and Schnitzer, 2003). The main income of traditional proprietary software vendors comes from license sales.

There are around 60 approved open source licenses (opensource.org). Amongst them, GNU GPL (General Public License) is the most common and restrictive license under which Linux and over 80% of open source software are released (Bonaccorsi and Rossi, 2003; Johnson, 2002). It allows people to obtain the source code, as well as to use, copy, modify and distribute the software freely under the original license. GPL is characterised by a viral effect in the sense that any program incorporating GPL-covered source code is required to license under GPL as well (www.gnu.org). However, the viral effect has incurred criticism and scepticism. The viral effect of GPL may lead to consumer welfare loss by encouraging the development of two incompatible networks (Schmidt and Schnitzer 2003). Consequently, the OSS should be covered by a liberal license such as BSD (Berkeley Software Distribution). Casadesus-Masanell and Ghemawat (2006) argue that the GPL has not been tested in court yet (also see Lerner and Tirole, 2002). In reality, some companies have removed all GPL code and many other commercial companies have begun using multi-licensing models to avoid GPL violations.

In contrast with GPL, some other permissive open source licenses like the prevalent BSD exemplify a more liberal license to source code. The BSD license distinguishes itself from GPL by allowing derivative work to be distributed under a different license, even a proprietary license. Therefore, it allows individuals or firms to appropriate and commercialise derivative work, and to distribute the modifications as closed source.

Research on comparisons between commercial proprietary licenses, restrictive GPL license and permissive OSS licenses from the perspective of market competition, product innovation and corporation strategy, uncovers that it is difficult to show that certain licenses are superior to others because the adoption of licenses is dependent on the internal and external environment of the corporation (West, 2003; Edwards, 2005; Mustonen, 2003; Valimaki and Oksanen, 2005).

Since OSS vendors cannot generate income by using traditional proprietary software licenses, they must find other means to make a profit based on the value that is provided to the customer. Certain types of hybrid business models are subject to open source licenses (Hecher, 1999, Schiff, 2002). The emergence of a hybrid business model which commercially provides complementary services and products that are not supplied efficiently by the open source community, such as packaging, consultancy, maintenance, updating and training has been widely adopted by new entrants in the software industry (von Krough and Spaeth, 2007; Bonaccorsi, Rossi, 2003). This model is especially suitable for the restrictive GPL. Most GPL/Linux vendors such as Red Hat and SuSe in the desktop and server segments rely not on the profits that software generates, but on the revenues from charges for the services they provide (Wu and Lin, 2001). Large software companies, including IBM, Oracle and Sun, not only support Linux development, but also make their proprietary enterprise application software run on Linux (Bitzer, 2004; Leibovitch, 1999). These distributors and systems integrators who participate in the open source community show convincing credentials by providing their customers with expertise and demonstrating sufficient technology at software development, which helps them attract business opportunities.

A variant of the hybrid business model is based on the permissive open source licenses. This model enables the independent proprietary applications and system tools to be developed, run and even sold with proprietary licenses affiliated with the OSS (West, 2003). In this respect, some commercial software vendors develop their software on top of an existing permissive licence (e.g. BSD) OSS. For instance, Apple built a kernel of its Mac OS X OS – 'Darwin' based on FreeBSD and kept the source open. Thus, the open source movement saved Apple substantial sums in development investment while ensuring steady compatibility and on going improvement from BSD open source communities. That said, the most valuable parts, the graphical user interface of Mac OS X are commercially licensed by Apple (Deignan, 2001).

2.2.5 Government policy to OSS

Governments in both the developed and developing worlds have adopted explicit policies to foster the open source movement and encourage the deployment of OSS in public administration. At present, open source software is widely employed in public sectors, e.g.79.3% of 995 public sector organisations in 13 EU countries used open source software in 2004 (Glott and Ghosh, 2005); in China, over 30% of income from Chinese Linux came from government procurement from 2003 to the present day. Many governments have begun to conduct research on OSS. Examples include the European Commission investing the Qualipso project with € 10 million and the Chinese government launching a series of "863 Programme[4]" key research projects on Linux-based system development.

The reasons for governments' positive attitude toward open source software are several: governments can reduce the procurement costs, prevent vendor lock-in, support local software industry, and solve security issues (Waring and Maddocks, 2005; Applewhite, 2003; von Krough and Spaeth, 2007).

However, free-market advocates, proponents with a libertarian background of the open source movement, distrust the government interference and that believe government should not interfere with the software market except for antitrust policies (Raymond, Taylor, and *et. al*, 2000). In contrast, the academic evidence is mixed. On the one hand, most of the academic economists are sceptical about government support of OSS because there has

[4] In March of 1986, to meet the global challenges of new technology revolution and competition, four Chinese scientists jointly proposed to accelerate China's high-tech development. The Chinese leader Mr. Xiaoping DENG approved the National High-tech R&D Programme, namely the 863 Programme. This programme intends to stimulate the development of advanced technologies in a wide range of fields for the purpose of rendering China independent of financial obligations for foreign technologies.

been no market failure in the software market. Evans (2002) points out that the software industry has worked extremely well without government intervention and cites the software industry of the United States as an example. He further claims that GPL licensed software should not be supported by governments owing to its viral nature, which would damage incentives for firms to invest in innovation. Schmidt and Schnitzer (2003) propose that making decisions on software development should be left to the market. At the same time, there is no need for the government to sponsor specific open source software by means of direct subsidies because the development of most software including OSS, is applied R&D as software products normally have well-defined technical features to fulfil the needs of potential user groups. Therefore, the development incentive for software is provided by the market. What is more, after analysing the competition between OSS and proprietary closed source software under the conditions of strong and weak network effects respectively, it is found that government intervention which promotes the public procurement of open source products, and forces the public sectors and government agents to deploy OSS, may damage the social welfare benefits discussed earlier (Schmidt and Schnitzer, 2003). Ultimately, due to losing market share, proprietary software vendors are likely to increase their product price and build compatible barriers to OSS.

On the other hand, Shapiro and Varian (2003) argue that open source platform software (like Linux) should be adopted by governments because of its technically superior and open nature as the adaptation of it in the public sector can further greatly affect the economic development of a country’s entire software industry. Comio and Manent (2005) studied three types of government intervention widely proposed in many countries: (1) mandated adoption – the government obliges public agencies, schools and universities to adopt OSS; (2) subsidy to adoption – the government provides money to consumers if they adopt OSS; and (3) information provision – the government informs consumers who were uninformed about the existence and features of OSS. In this research, most consumers are presumed uninformed consumers, in particular those who are less sophisticated and have higher search costs in the mass market segment. The reason is that OSS providers normally have fewer financial resources to advertise than traditional proprietary software suppliers, due to no license fees. Through Comio and Manent's numerical simulations under conditions with and without network externalities, the subsidies towards OSS always reduce social welfare while the measures of mandated adoption and information provision enhance welfare. Furthermore, Comio and Manent also recommend that government intervention should guarantee a level playing field rather than picking out the winner. That is, the government should not espouse mandated adoption since it is seen as an intrusive measure damaging fair play in market competition. Providing knowledge of OSS to uninformed consumers in the mass market can simply put them in the best position to choose, and leave them free to adopt their favourite software according to their needs (Comio and Manent, 2005).

2.2.6 OSS in developing countries

Since technology holders in the developed world have sought to strengthen technology protection in developing countries, the latter face great pressure from the International Intellectual Property (IP) regime. The beneficiaries of current IP protection are Multi-National Enterprises (MNEs) which have a large number of technologies under IP protection in developed countries. The International IP regime is regarded as an obstacle for DCs improving indigenous technological capabilities, and thus enables MNEs to dominate the market in DCs (Colman and Nixson, 1994). This means that, DCs are locked into a dependent relationship with the developed world (Shen, 2005a). To bring all DCs into the same IP regime, the developed has implemented a series of agreements on trade related intellectual property rights (TRIPS).

Due to the lack of core software, such as OS in the software market, DCs depend greatly on the MNEs in the developed world. Once users in DCs install MNE's owned property software, they are forever forced to pay for upgrades or to buy new versions. To ameliorate this disadvantageous position, DCs, have begun to embrace the OSS.

The context of DCs is largely distinct from developed countries where most OSS design, development and innovation takes place. DCs not only lag in economic development but also in technology development. Another significant characteristic of DCs is their domestic software market domination by the giant players from the developed world. Therefore, it comes as no surprise that the studies on OSS in DCs are mainly focused on the benefits which DCs acquire by drawing upon OSS. It has been reiterated that there are at least four outstanding elements: affordability, adaptability, independence and security.

(1). Affordability. The economic benefit of selecting OSS over proprietary options is the most significant issue. It is an inevitable issue that most of the research papers address (see Bokhari and Rehman, 1999; James, 2002; Ksheteri, 2004; Cook and Horobin, 2006; May, 2006; Pan and Bonk, 2007 etc). The proportion of proprietary software in the total cost of the whole computing system has greatly increased in recent years; sometimes the cost of software even exceeds the price of the computer hardware. In this respect, if DCs choose OSS they can save much money on software expenditure because nearly all of the OSS can be downloaded freely and legally from the Internet without paying a license fee. Therefore OSS helps DCs solve the international and domestic digital divide without infringing upon intellectual property rights (Cook and Horobin, 2006; James, 2003; von Krough, and Spaeth, 2007; Salvador, Sherry and Urrutia, 2005).

(2). Adaptability. OSS can be customised and adopted for highly specific local needs; the availability of source code provides important and obvious advantages (Camara and Fonseca, 2006; Ksheteri, 2004; Kogut and Metiu, 2001).

(3). Independence. Like other high-tech fields, the foreign software giants have acquired a nearly monopolistic position in the DC software market. Developing software based on OSS is seen as an efficient way to escape software monopoly from foreign software companies (Cook and Horobin, 2006; von Krough, and Spaeth, 2007; James, 2002).

(4). Security. In order to keep trade secrets, the proprietary software firms today only provide users with an object code, which means that bugs and security problems are hard to perceive and remedy without accessing the source code. The OSS is generally distributed with human readable source code so that the conventional software bug and ‘intentional’ backdoor can be identified and fixed (Camara and Fonseca, 2006; Ksheteri, 2004; Pan and Bonk, 2007).

A large body of literature on open source software and Linux in China, at the same time has moved on considerably (Cao, 2005; Da, 2005; Gao and Xie, 2004; Gao, 2005; Li, Lin and Xia, 2005; Ni, 2005 and 2006; Shen, 2005a and 2005b; Pan and Bonk, 2007; Qian, 2995; Shi and Liu, 2005; Zhang, 2006). China has a huge software market: it not only has over 100 million personal and enterprise users, but also has countless demands in e-government which is generally in its infancy in China. Thus, the Chinese software industry, is strongly focused on domestic products rather than providing export outsourcing services, like India (Annual report of science and technology development of China, 2002; Tschang, 2003). Besides the above four benefits, Linux was seen as a great opportunity to build China’s own software industry (Li, Lin and Xia, 2005; Ni, 2005 and 2006). What’s more, a conventional belief in China is that the open nature of OSS – the access to the source code, as well as the transparency of the development also enables the domestic education and training of IT professionals (Liang, 2006; Pan and Bonk, 2007; Zhang, Sun, Guan, 2006). However, most open source or Linux related Chinese literature is published in newspapers and IT magazines rather than academic journals. These Chinese authors seem to be animated by a partisan spirit and hype, as well as focused too much on the benefits of reaping OSS. Furthermore, the open source movement in China drew less attention from international academic community. There are only few English papers discuss the successful side of applying OSS in China, and no paper address the failures of applying and developing OSS in China and what lessons can be drawn from these failures and experience (Li, Lin and Xia, 2005; Pan and Bonk, 2007).

2.3 Technology Studies

2.3.1 The Social Shaping of Technology (SST)

SST developed in Europe and the United States in the last three decades addresses both the content of technologies and the process of innovation. It has been progressively used as a powerful conceptual and analytical guide to the study of technology, including information communication technology (ICT) (Sørensen and Williams 2002). SST has emerged from a critique of a deterministic approach which sees that technology development could be taken for granted to meet social and economic needs, and thus merely be concerned with the social impacts of technology (MacKenzie and Wajcman, 1999). In contrast to deterministic views, the SST school claims that technology does not develop simple compliance with a predestined inner technical logic or economic rationality. In every stage of generation and implementation, technology is confronted with various technical options and each of these options represents different interests and implications for certain social groups. In this respect, the options selected are dependent on not only technical considerations but also on social factors. The final choice in every stage of technology development determines the direction or path of the next round of technology development and/or innovation. A particular technology therefore is a social product which is the outcome of negotiation between technical, social, economic, and cultural elements, and the interplay between different social groups within the specific context (Williams and Edge, 1996; MacKenzie and Wajcman, 1999). The social shaping approach develops a range of valuable conceptual tools and analytical models to improve the understanding of the relationship between technology and society. This section reviews several approaches under the SST framework, i.e. social shaping of information and communication technology (SSICT), social learning and biography of artefacts (BoA), which are employed to build the research framework of this study.

Social Shaping of Information and Communication Technology (SSICT)

A growing body of research has addressed the social and economic factors shaping the development and use of contemporary ICTs, in particular the software technology. A rich range of studies have been conducted to capture the character of software systems and their complexity of development and innovation process, including the area of computer-aided production management, modification of software packages, and enterprise resources planning systems (Williams, 1995; Dutton, 1996; Clausen and Williams, 1997; Slack, Stewart and Williams, 1999; Pollock 2001; Pollock et al. 2003). The complex interaction between ICT supply and use proliferated throughout the process in ICT innovation, and has not been adequately recognised before (Williams, Stewart and Slack, 2005). Developing sophisticated software for the consumer market requires detailed knowledge of the preferences and practices of users. Take accounting software for example, where such programming requires not only IT professionals, but also close collaboration with accountants and experts in several fields. The reason is that the in-depth knowledge of the rules and requirements of accounting must be considered alongside the good computing.

The claims to universality of ICT, which suggested new technological offerings emerging from technology supply, would provide finished solutions that could be readily and widely applied to a full range of uses. However, the many different users may have different perceptions of ICT and its utility. For example, the main purpose of applying computer systems in everyday home life are word processing, education, entertainment and so forth, while applying computers in organisations is to simplify data processing, improve working efficiency, reduce costs etc. Further, applying computer systems in an organisation always faces difficulties since the structure, decision-making process and many other practices of various organisations differ greatly. Developing a generic system which takes into account the context and characteristics of all types of organisations is an impossible mission. As a result, the universality of ICT will always be challenged by the diversity and specificity of the social contexts of application and use of ICT (Williams, Stewart and Slack, 2005; Williams, 1999).

There is a dichotomy of ‘into’ and ‘out’ in software development strategy; on the one hand software supply attempts to design more ‘society’ into the software, but on the other, tries to design ‘society’ out of the product according to the trade-off between utility and price. The former refers to customised solutions which were designed for specific contexts of use and/or particular users, the latter is the discrete application or generic software that can be applied to ordinary use and/or the user in broader settings (Procter and Williams, 1996; Williams, Stewart and Slack, 2005).

Social learning

In the mid 1990s, scholars from technology studies proposed the concept of ‘social learning’ (e.g. Rip, Misa and Schot, 1995; SØrensen, 1996). The core idea of social learning emerged from intellectual traditions of economic history, such as ‘learning by doing’ (Arrow, 1962) and ‘1earning by using’ (Rosenberg, 1982), and evolutionary economics – ‘learning by interacting’(Andersen and Lundvall, 1988). Borrowing the early concept of social learning, together with the existing insights from social shaping of technology (MacKenzie and Wajcman, 1985; Fleck, 1988; Williams and Edge, 1996), appropriation of artefacts (Pacey, 1983), and domestication of technology (Silverstone, Hirsch and Morley, 1992; Lie and SØrensen, 1996), Williams et al (2005) enrich the social learning framework. They criticised earlier supply-centred perspectives in technology design and proposed an appropriation-centred view, as well as further elaborating the active role of the various players, including technology developers, users and other local players in the innovation process, by conducting an empirical research – the “Social Learning in Multimedia (SLIM)” project, funded under the European Commission’s Fourth Framework Programme. In the social learning perspective, the concept of ‘appropriation’ of technology is replaced by advanced domestication because the term appropriation has been used in various levels in technology studies, ranging from firm level, sector level to country level. Therefore, appropriation has no dominant definition and general analytical methodology. The term ‘domestication’ here denotes the process of exploring and building technical capabilities, attributing meanings, and developing practices with the technology, integrated within local social conditions (Williams, Stewart and Slack, 2005).

Unlike the traditional industrial technologies, ICTs penetrate not only all the fields of national economy but also the everyday life in the community and home. The user acceptance and demands of new ICTs have become more critical for technology developers since social practice is increasingly integrated into the application of ICTs. Following the social shaping perspective, social learning once again highlights the intricate interplay between the suppliers and users. The suppliers of new products start their R&D activity with limited information about users and their needs. The users and user needs emerge and change in different phases of product development, through an interaction process between suppliers and users (Williams, Stewart and Slack, 2005).

The social learning perspective also provides a framework with which to rethink public policy making. In the last several decades, technology policies adopted by countries have been influenced largely by the view of determinism. Earlier technology policy is based on a simple linear model and technology-driven views, which focused on the technology supply side only. It generally took the utility of finished technology for granted, which presumed that a technology or artefact is well defined and developed in advance as a feasible solution for meeting consumer needs. Thus the main task of policy is to help new products and services to build demonstrators with which the new technology could be more attractive for customers, and further help to cultivate a sufficient critical mass of users. A negative consequence is that the traditional technology policies overlooked the pivotal role of use and user.

User in social learning perspective

In the early 1980s, Fleck from SST School recognised the important role of users and introduced the concept of ‘innofusion’ to elaborate the innovation process from the studies of social shaping of industrial technologies (Fleck, 1988). The term innofusion criticises the traditional ‘linear models’, which suggest that innovation only takes place in the technology supply side, as if the well-developed artefacts are invented in the laboratory to fit users’ needs readily and could be diffused to users through the market. However, many users’ needs and requirements are not understood in advance, but discovered and incorporated during the processes of technological design, trial and exploration. Thus user organisations can be seen as a laboratory in which suppliers must collaborate with users to understand the functionality of technology by learning users’ actual requirements. The subsequent implementation and use of technology, therefore, contributes to technology design. The technology innovation is a multi-cycle process, whereby innovation continues during the stages of implementation and use.

The social learning perspective echoes the idea of innofusion that the users engage in technology design and innovation. The creation and evolution of representation (Vedel, 1994) of users and uses are a central concern identified in SLTI (Stewart and Hyysalo, 2008). In designing, the artefacts do not simply develop with particular functionality. Some explicit and tacit conceptions/understandings must be developed as well, about the kinds of use that may be attempted, about the identity of users and about the context of use (van Lieshout et al., 2001). Social learning perspective criticises the politicized views of prevalent social shaping perspective and constructivist studies of technology which highlight the interests and values of particular actors (normally the designer in the supply end) with more power in designing. The actual user is treated as a passive recipient who was configured (Woolgar, 1991; 1996) and whose features were inscribed (Akrich, 1992; Akrich and Latour 1992) into the artefact according to designers’ personal experiences, expertise, and presumptions (Cawson, Haddon and Miles, 1995). Research starting from this perspective may have a negative result in that this involvement is restrictive or out of line with the actual users because the users and their values might be misconstrued. These early accounts portray a linear procedure within which the related user involvement materialised into artefacts by the designers so that the users had little influence on the artefact/technology.

However, the social learning perspective acknowledges that system design, in particular the initial design, is based on incomplete knowledge, uncertainty and inconsistent views about potential/future users and their settings. Although the designer always seeks to prefigure the users and then constrain the way in which the artefact is used, the actual users are still flexible since new meanings may be endowed during the domestication of technology, i.e. using the product in ways not anticipated by designer (Berg, 1994). In this respect, the design is not a one-off act, but is part of an iterative series of activities, informed by early hypotheses about the user and use, further experimentation and testing in the process of innofusion, and feedbacks from appropriation of the technology (to future artefact design).

The empirical cases of the SLTI project reveal that the knowledge of user involvement is much more complex in practice. Williams, Stewart and Slack (2005) indicate that there are many difficulties for acquiring such knowledge, like the extent of supplier-user linkages, uncertainties within new product designing, timeliness of the knowledge of existing users, uncertainty about identity of future users, difficulty in assessing users in private space and others. In order to involve the user, the designer or developer may employ three main types of intellectual resources: direct user involvement, constructions of users and indirect evidence about users. Studies in the last decade develop many techniques (e.g. market research, user panel, and so on) by which developers are able to gain direct information about users (Akrich, 1995; Williams, Stewart and Slack, 2005). However, one serious pitfall of such involvement is the high cost making these methods impractical for newly established firms with limited financial budgets. In the case of absence of direct knowledge of users, many designers construct users by drawing upon the personal experience, expert knowledge, and evidence in existing product markets. The potential weaknesses of these two types of methods are salient as well, because the features of users and their contexts are easily neglected by experts (e.g. I-methodology described by Oudshoorn and Pinch (2003) and Akrich (1995)). Also, the information extrapolated from existing product markets may no longer be applicable in a context where a product is updated and changed rapidly.

In a nascent market which is fraught with the unpredictability of technological change, market organisation and user’s uptake, as well as lack of contact between potential users and suppliers, collecting knowledge and understanding the context of users is extraordinarily difficult for technology developers due to the asymmetric distribution of knowledge between them (Williams, Stewart and Slack, 2005). Stewart and Hyysalo (2008) define the term of ‘intermediary users’ or ‘intermediaries’ to refer to the organisations adopting a technology for their customers or employees. All the activities and functions of intermediary users are associated with knowledge creation, translation and dissemination (Bessant and Rush, 1995; Howells, 2006), in the sense which they are able to play the facilitating, configuring and brokering roles to bring end users into supply-side innovation efficiently.

Biography of Artefacts (BoA)

This study is also inspired by the concept of ‘Biography of Artefacts (BoA)’, which was developed recently by researchers in University of Edinburgh (Pollock and Williams, 2009; 2010): drawing upon the concepts of “social learning” and “innofusion” of the SST perspective, researchers were able to study the technology development and innovation temporally by mapping the main actors and their interactions. The BoA theory can be seen as the supplement of traditional SST theatrical tools, which avoids the pitfall of viewing design studies as a one-off job and focuses on the development and innovation process both temporally and longitudinally (Williams and Pollock, 2011).

There is a substantial body of work on information system study, most of which has been conducted by either impact study or implementation study. The impact study refers to the research into the social and economic impact of adopting enterprise information system or packaged software. The impact studies are based on a supplier’s perspective and probe how the technology offerings solve the problems encountered by enterprise in work practice; it sees design and implementation as separate individual phases and thus ignores the role of organisational users in the process of technology implementation and further innovation.

In the past, implementation study (especially the single site implementation study) was regarded as an efficient tool for studying packaged software solution. This method enables researchers to capture rich local pictures of interactions between software suppliers and users, and the outcome of immediate implementation. However, single site implementation studies focus narrowly on either a particular stage of technology design and/or implementation, or a single site where innovative project takes place, and therefore provide no means to examine how the packaged software solution evolved over time (Koch, 2005, William 1997).

By studying the ERP (Enterprise Resource Planning) system, a small community of scholars found that implementation of the enterprise-wide system – ERP not only engaged with IT artefact, but also enlisted social actors (suppliers, users, intermediaries or consultants) and social elements, such as the implementation procedures and practice in user organisations (Orlikowski and Iacono, 2001; Koch, 2005). In order to provide better understanding of ERP implementation, the research methodology shall move from prevalent single site studies to multi-local studies (Scott and Wagner, 2003; Kallinikos, 2004; Koch, 2007). What's more, implementation study is characterized as short-term or medium-term, depending on the temporal frame adopted by the researcher, which typically ends in several months or a couple of years after the technology implementation. As such, the implementation study cannot provide understanding of how complex organisational packages, like ERP are developed as generic packages, since this methodology does not consider of the 'innofusion' effect – new design and innovation process is conducted after the technology implementation, by drawing upon feedback from users. Pollock and Williams (2010) express dissatisfaction of this relatively short-term study: the timeframe of research needs to extend to a longer-term, because the consequences of technology implementation may be unfolded after a long time, even decades (also see Williams and Pollock, 2011).

Pollock and Williams (2009) develop the concept of BoA to describe the interactions among actors over different sites and time frames. Building on Fleck's 'innofusion' (Fleck, 1988) and 'spiral of innovation'(Fleck *et al.*, 1990), the BoA perspective also attempts to trace the 'accumulated history' of technology and describe how the historical context continues to influence the shaping of the technology. It draws attention to the way in which a technology extends from one place to another, in order to examine how it is adopted and refined according to the needs of each new place.

BoA perspective also redresses the limitations of the actor-centred framework, such as the early ANT theory (Williams and Pollock, 2011). From the viewpoint of ANT scholars, researchers should follow the actors so that studies could cross local sites and times (Callon and Law, 1982; Latour, 1987). Focusing on actors, however, has incurred criticism from some scholars, because this method concentrates on the actors who have direct relation to technical issues, thus overlooking other actors, as well as neglecting historical and institutional elements which sustain the technology development (Kallinikos, 2004). Once the researcher's view shifts from the 'local' to 'global' level, the BoA concept is useful in providing a more 'context view', ranging from local technology design/implementation to a broad 'macro-level' that includes many issues the local actors must take into account (Morrison, 2002; Williams and Pollock, 2011).

By analysing organisational software packages at different biographic stages, Pollock and his colleagues demonstrate how a generic system can emerge from local implementation and development (Pollock *et al.*, 2003, 2007). In the information system development field, the conventional insight of STS is that the expert system is designed and developed for specific work practice and site so that it is extremely difficult to transfer to other contexts (Webster and Williams, 1993; Finchman et al., 1994; Berg., 1997; McLaughlin et al. 1999). This contradiction between generic and specific does exist in reality, such as ERP systems. These systems are now called generic or global solutions because they have already travelled to various industries, sectors and countries. This phenomenon also draws attention by STS scholars, but they tend to focus on the effort of software localization: the way systems are transferred to new user settings. Pollock et al.

(2007, 2009) argue that current STS pays much less attention to creating generic solutions from a supplier's view. They also develop the notion of 'generification' to elaborate the process in which suppliers design solutions to work across context by means of employing a range of strategies to struggle and negotiate with user organisations. During the process of generification, a strategy called 'accumulating functionality' is adapted by system suppliers during the *birth* stage. In this stage, suppliers must establish close ties with individual user organisations, in order to match their local specific requirements and accumulate functionality of systems. As the functionality of a system is enriched and the user base grows, the biographical stage of a system moves forward to subsequent stage. Suppliers begin to consider designing generic solutions and shift from individual users to a more public design environment – a larger extended user community. Suppliers in this stage, use other strategies (such as management by community, management by content, management by social authority, segmenting the user base) to identify and smooth diversities, as well as capture collective requirements. The final stage of software package's biography is *the future* in which suppliers develop more than one template for implementing the same process so that the system can have additional flexibility to fit 'generic particulars'.

Such a generification process can also be found in the growth of contemporary complex information systems – e-infrastructures. E-infrastructure is considered as a global or generic solution offer for sharing information across the boundaries between organisational groups (Pollock and Williams, 2010). E-infrastructure emerges from an isolated system originally designed for a specific user group and use. To be an e-infrastructure, the isolated system inevitable experiences a generification process (Hyysalo, 2010). Some of the articles in a special issue published in *Journal of the Association for Information System on e-infrastructures* in 2009 (Edwards et al. 2009) probe how e-infrastructures cope with local diversities during the process of e-infrastructures growth. An underlying theme in all the articles is that any new infrastructure must reach an installed base or user base, as well as match the generic features of human habits and work practice (Edwards et al. 2009). One of the crucial moments of infrastructure consolidation and growth is that the infrastructure can be adapted to a broader context from the place where it was designed. This moment shows an evident generification process, and is full of negotiations between system builders and actual users: system builders must compromise or make appropriate trade-offs between a number of conflicting goals, particularly either between catering to specializations and meeting larger community needs, or between diversities and universalities.

In Pipek and Wulf's study on e-infrastructure, authors identify that one of the tensions in e-infrastructure development is the management of the user community and dealing with the relation of the particular and the generic (Pipek and Wulf, 2009). The long-term e-infrastructure – WATERS system is an appropriate example. In the birth biographic stage, the system was designed by individuals – computer scientists, but supplied service for other participants – hydrologists. In this case, all requirements of early users were accommodated into the system. With more and more intended users participating in the system development, however, it was impossible to design WATERS to match all hydrologists' individual needs. In order to support the community's long-term research tasks, the computer scientists had to search for similarities between all hydrologists' requirements and only the generic requirements were taken into account in the later development stage. In the future stage, with the environmental engineer community becoming intended users, WATERS would serve both communities. By conducting a survey in the new user community and recognizing the similarities of both communities' research requirements, the developers realized some research data and functions (templates) could be treated as generic features for both user communities. However, the result of the survey also showed that particular environmental engineers wanted access to different databases, which also required additional software technologies. In order to allow the flexibility in the search function, the system was tailored so that the particular researchers can find data according to specific measures or variables of interest.

Ure *et al.* (2009) review the experience of various eHealth projects (HealthGrids) development in the UK and similar development processes can also be identified in their work. The authors state that all the project teams had to deal with diversity – interests (normally competing) from researchers, clinicians and ontologists by adopting strategies, in order to develop data infrastructures in eHealth. They illustrate that the BIRN (Biomedical Informatics Research Network) project employed an open strategy – open access to encourage user participation and collect the knowledge about datasets' attributes and format in the birth stage. In order to balance the diverse preference of user groups and stable semantics of data infrastructure, the project team managed diversity in the later stage. Such compromise was made after the prototype has been demonstrated to clinicians at different sites. In the future stage of biography of BIRN, the project team separated a fairly stable core according to the generic features of various user communities' preference. Some particular attributes of data would be created in accordance with local uses and work practices of the specific domain.

The contribution of BoA to this study offers a guideline to follow, which identifies specific and particular moments in the evolving process of a software package. The more important the utility of BoA is, the more it identifies biographical expectations associated with the characteristics of software package, and further reveals how the technology suppliers design the software based on their understanding of customer (and of these biographical expectations). However, adopting the BoA framework – studying immediate actions within a long-term duration of technology's life cycle is problematic, in that it requires researchers to generate a trade-off between the breadth and depth of study (Pollock and Williams, 2009). The large-scale software technology, for instance, may be implemented in many loci with specific local features. If the researchers conduct very detailed short-term temporal study on every locus, the longitudinal study would become extremely expensive. The researcher, therefore, must choose particular moment(s) to conduct detailed qualitative study by setting convincing criteria, such as the accessibility of case and profound design/implementation fragment.

2.3.2 Infrastructure studies

The SST perspective and related theoretical tools, in particular the BoA framework, supply an analytical framework to understand the dynamics and contingencies during software and e-infrastructure development over time and at different loci. As discussed above, the ‘growing’ of e-infrastructure (Edwards *et al.*, 2007) faces the same issue as developing generic organizational software package – catering to the diversity of users’ requirements and consequently pursuing generification, in order to work across sites and contexts. The research object of this study – Linux, is a special kind of software with some features of infrastructure. The concepts of infrastructure studies can help the researcher understand the broader context in which Linux evolved, extract the characteristics of Linux, and identify the most important dynamics and tensions surrounding the evolution of Linux in China, participating actors and intricate interplays between them.

The word ‘infrastructure’ has been widely used in the last two decades, with reference to the technical structures or physical networks supporting the society, such as railroads and highways (Summerton, 1994), electronic power network (Hughes 1983), Internet (Edwards, *et al*, 2009), and so forth, as well as the services facilitating the economy (Edwards, 1998; Edwards, *et. al*, 2009; Fomin and Blechar, 2005).

The retrospective studies from a STS perspective show that an infrastructure is grown from isolated systems designed and controlled by a builder. Once the system is established locally, it may extend to other places, domains or contexts by intricate processes of transfer, adaptation and growth. In order to achieve such technology transformation, not only the technological elements, but also the social, organisational, institutional and political factors embodied in the system may be required to change. Assembling these disparate elements tightly together is a long-term, complex and large socio-technical project in which these elements are required to mutually adapt and adjust (Jackson, *et al.*, 2007). In this sense, the dynamic created during the process of infrastructure growth is defined as ‘reverse salients’ by Hughes (1983; 1987), which refers not only to intractable technical problems, but also to the legal, political, social, or cultural problems involved. The solution of reverse salients is required for the entire system to work or to grow. In the past studies on infrastructure development, the government and other national-level institutions played critical roles in identifying and shifting reverse salients. (Hanseth, 2002; Jackson *et al*, 2007).

The successful formation of infrastructure originates from competitions between isolated systems developed by builders in a technical regime. Apart from one system beating out all the others directly, the competition can also be resolved by the creation of 'gateway' technologies or social arrangements that permit all these systems to interoperate. The gateway, therefore, is seen as another dynamic which enable heterogeneous systems to link or merge into networks and internetwork (Edwards, *et al*, 2007).

Another important dynamic is path-dependency. Jackson and his collaborative authors (2007) point out that the cumulative nature of infrastructural development, together with the technical and social elements tightly tied to it, led to path dependencies (also see Hanseth, 2002). Once the solution/infrastructure is established, shifting to another alternative solution is costly and difficult. The path dependency here is related to the lock-in effect of choices among competing technologies. The most cited example is the QWERTY keyboard which is not the best keyboard layout but has remained in place for decades until today (David. 1985).

For achieving an effective alignment of coupled technical and social elements, researchers have gradually recognised the importance of managing unanticipated problems and tensions which arise during the course of infrastructure design and implementation, whether these are technical or human (Hanseth *et al.*, 2006, Ure e*t al*., 2009; Pollock and Williams, 2010). Although the tensions identified by actors are not the same, and might be varied on grounds of different infrastructures (Ribes and Finholt, 2009), there are always inevitable contradictions between infrastructure developers' intentions and complex and diverse uses (Ciborra *et al*., 2000; Lawrence, 2006).

Infrastructural software

In order to understand the nature of OS, such as Linux in this study, and conduct this research, a new concept, ‘infrastructural software’, is hereby introduced.

The computer, as a physical machine consisting of visible hardware, is useless without software. The software transforms the physical machine into a myriad of special purposes virtual machine. The OS, in this sense is the foundational component/software upon which a large parts of users’ activities are based. This research uses the term infrastructural software (which also appears frequently in Chinese government formal official documents) to refer to the OS. As the low-level software system, OS on the one hand, is the most basic component for computers/terminals. The OS integrates the heterogeneous elements for computers, which includes not only technical components but also organisational and social elements to implement the specific purposes of users. On the other hand, the OS enables the foundational functions for computers: computer system-intra-system data exchange and information communication, i.e. transferring data between memory and disks and rendering output onto a display device.

However, an OS is not necessarily an infrastructure. In the case of Chinese Linux development and adaptation, both embedded and platform Linux is examined. The embedded Linux is developed for embedded equipment with specific local user requirements and special hardware framework so that it requires a great deal of technical effort in customised development. The embedded OS must be tailored as compactly as possible: only the programmes and components that are associated with functionality and hardware are retained, and others must be peeled from OS as redundant. Due to differences of functions and hardware between various embedded equipments, like all other embedded OS, an existing embedded Linux is difficult to port to another embedded product. Any difference leads to expensive technical efforts, and even the OS has to be designed from scratch. As infrastructural software, the embedded Linux lacks flexibility to travel to other contexts as infrastructure.

On the contrary, the platform Linux was developed as infrastructure for Chinese government purposes. It is seen as the software system to support all IT applications within the whole Chinese society. The rationale is that the original Linux is developed as a general-purpose OS and supports all kinds of applications and functions of computers and more complex computing systems. In other words, platform Linux is a robust and common resource for all traditional computers used as the basis for e-infrastructures (e.g. Internet and e-Science) as well as other computer-based traditional infrastructures (e.g. traffic control system in road or rail system). The platform Linux therefore is supposed to port to another domain, user organisation or context easily.

2.3.3 Technology studies from a developing country perspective

Modern technology is so important for all countries that it is considered to bring about expeditious economic growth and good quality social welfare (Madanmohan, Kumar and Kumar, 2005). The capability of developing technologies is not homogenously distributed around the world. The production of advanced technology and knowledge is largely concentrated in the developed world so that international technology transfer is the most preferred route to attain access to advanced technologies and acquire technological capability by DCs with limited R&D resources, which shortens the technological gap between the developed and developing world (Lin, 2003; Putranto, Stewart, Moore, 2003). This point of view is demonstrated by successful experiences of newly industrialised countries in which technology transfer efficiently facilitates the increase of productivity and technological capabilities of local economies. Simultaneously, firms in the developed world are normally willing to transfer their advanced and latest technology to DCs to establish a quasi-monopolistic position with which to make significant returns (Liu, 1995). The long-term significance of technology transfer is the possibility of replicating knowledge and applying broadly, beyond the boundaries of the firm originally involved, in order for transferees to assimilate, adapt and improve upon the original technology from abroad (Shen, 1999).

The formal modes of technology transfer between two economic organisations generally include direct foreign investment, joint venture, direct sale, turnkey projects, license and patent authorisation, technical agreements, among others (Kumar *et al.*, 1999). These modes have been classified into three categories by Bell (1987): transfer of capital goods, technological, engineering and managerial services; transfer of the skills and know-how to operate and maintain the technology; and transfer of the knowledge and expertise for implementing technology development.

In respect of international technology transfer, dependency theorists propose that the control over the technology is reserved within the developed world so that DCs are locked into a dependent relationship with developed countries. Gabayo (1996) suggests that there are two types of technological dependency: mutual and asymmetric. The terms of mutual dependency refers to a country that plays a role as both recipient and supplier in technology transfer, that is, when a country imports advanced technology from abroad while pursuing its own technological development. The transferred technology is aimed at complementing local technology resources for further development. Against this backdrop, technology is supposed to modify and integrate into a local developed system, smoothly and efficiently. As a result, the country achieves internal technology development and thus has the possibility to export new technology. In the sense of asymmetric dependency, transferred technology is usually used as it is without any assimilation, adaptation and modification. This situation occurs when a country has insufficient technological capability to develop and localise the transferred technology. As might be expected, mutual dependency enhances independence between the transferee and transferor, as well as contributes to the transferee's technology development. In contrast, asymmetric dependency prevents the transferee from acquiring know-how and mastering transferred technology, and therefore strengthens the dependent relationship of transferee on transferor. The technological situation in most DCs takes the latter form of dependence. From a perspective of development studies, the biggest challenge of technology transfer exercises is to satisfy the need for foreign know-how while taking measures to minimise technology dependency. Developing indigenous technological capability is one of the pivotal conduits to the measures above and is also an essential condition for sustainable development (Huq, 2004; Paukatong and Paul, 2006).

This transfer is a complex process but not just a simple purchasing behaviour. Transferees generally have to devote substantial efforts to assimilate, adapt, and improve upon transferred technology. These behaviours are related to technology capability, which is a wide term and essentially comprises the ability to select, utilise, adapt, improve, and finally to further develop technology. Even mundane local activities like plant layout, quality control, maintenance, and process optimisation may call for substantial capability building when technologies are transferred. The same technology is often employed at widely differing levels of efficiency in different DCs. Many of these differences can be traced to varying levels of technological capabilities in the host DCs (Lall, 1993). Therefore, the effectiveness of technology transfer is subject to the level of indigenous capabilities.

A few typologies of technological capabilities are defined in light of the diverse perspectives and purposes of researchers (Desai, 1984; Lall, 1982; Lee, *et al.* 1988; Shen, 1999). This research considers that the two-level typology proposed by Shen is the most suitable categorization to study technological capability in DCs. A hierarchical relationship between technology transfer and technological capabilities is shown in Figure 1.

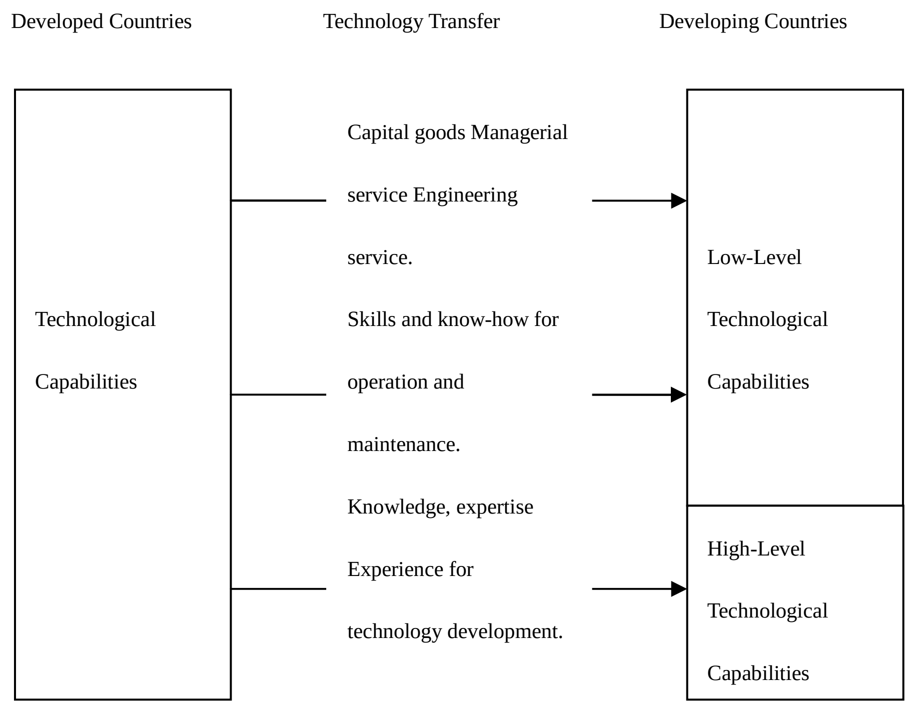
Developed Countries
Technology Transfer
Developing Countries
Technological Capabilities
Capital goods Managerial service Engineering service.
Skills and know-how for operation and maintenance.
Knowledge, expertise Experience for technology development.
Low-Level Technological Capabilities
High-Level Technological Capabilities

Figure 1: Relationship between technology transfer and indigenous technological capabilities

This typology divides technological capabilities into low- and high-level: the former refers to the capabilities which implement and manipulate a transferred technology, while the latter is in relation to further technology innovation (Shen, 1999). It provides a clear picture which echoes SST perspective and innofusion – foreign transferred technology adapted into local context, and then innovated further according to developer's/adapter's new understanding of local users' requirements and broader context in host DCs. The low-level technology encompasses identifying and purchasing suitable technology, as well as for operating, maintaining, repairing and adapting the technology for increased production and efficiency. What is essential is that DCs need capability (such as properly qualified manpower) to assess competing technologies on their own merits and to select those that are found to be most suitable under the circumstance (Huq, 1991). Lall (1982) indicates that the completed transfer of knowledge and skills required for fully understanding and controlling the technology cannot be achieved in a short time period because the tacit knowledge (e.g. R&D processes, technological efforts and experiences) cannot be transferred with a new technology even though it comes with a full set of blueprints and manuals. Consequently, the low-level technological capability is an essential condition for effective technology transfer. The technology transferee with low-level capability, however, would not be likely to move beyond the current technological level as defined by the transferor. Due to deficiency of high-level technological capability, it is liable to result in dependencies for DCs during the international technology transfer for the reason that future technology upgrading would rely heavily on the foreign supplier.

High-level technological capability enables the transferee to create new products, new processes, new designs, and even new technologies. It requires the technology transferee to participate, involve, and learn the technology development actively. A set of learning mechanisms consisting of in-house learning ability and the ability to exploit external mechanisms is the most crucial element for high-level technology capabilities (Biggs, *et al*, 1988; Desai, 1984). To archive efficient technology transfer independent of the developed world, DCs should not focus merely on efficient assimilation of technology in the short-term, but on the ability to create and innovate based on transferred technology (Liu, 1995). The capability of innovating technology and creating technological knowledge is crucial for DCs to shorten the gap between them and developed countries, through technology transfer.

Ample evidence has established a strong link between international technology transfer and building indigenous technological capabilities building. On the one hand, international technology transfer from the developed world can be an important source of building indigenous technological capability in DCs; on the other, sufficient indigenous technological capability is perceived as a prerequisite for successful technology transfer (Cusumano and Elenkov, 1994; Kumar, 1999; Kim, 1999; Lall, 2000; Madanmonhan, 2003). Successful technology transfer and technological capability building not only result from in-house technical efforts, like duplicative imitation through reverse engineering and corporate R&D, but are also influenced by external socio-economic factors. A sufficient local knowledge base is of importance for both technology transfer and local technological capabilities building. Training of local manpower is necessary for understanding and assimilating transferred knowledge, so a comprehensive educational system is required (Grieve, 2004; Kim, 1999; Reddy and Zhao, 1990). Another conduit to accumulate a knowledge base is achieving effective basic and applied research programmes in universities and public research institutes (Kim, 1999; Madu, 1989). The universities and research institutes in host DCs, therefore, are important actors for technology transfer and indigenous technological capabilities building. The government of DCs is another crucial actor (Gallagher, 2006; Liu, 1999; Madu, 1989). From the experience of Newly Industrialised Countries in East Asia, like South Korea, Kim (1999) suggests that the South Korean government has reformed their university education from teaching-oriented to research-oriented, for cultivating specialists in various scientific and engineering areas. Furthermore, the South Korean government took the initiatives and measures to deepen national R&D capabilities, such as establishing government research institutes, as well as encouraging universities to conduct basic research and recruit researchers by the means of financial support. Besides the leading role in improving the system of domestic education and training (also see Hobday and Rush, 2007; Gallagher, 2006; Madu, 1989), the government's selective intervention should focus on retaining a liberal economic environment, protecting domestic infant industry, controlling foreign direct investment, and providing infrastructural institutions and service (Grieve, 2004; Huq, 2004; Lall, 1993; Noland and Pack, 2003). Being aware of the active role of government, and universities and research institutes, as well as the linkages of technology transferees (normally firms) with the above main actors within the local economy, are critical for enhancing capabilities (Hobday and Rush, 2007; Kumar, *et al.*, 1999; Lall, 1993; Wei, 1995).

The technology transfer literature criticises many social and economic analytical approaches that treat all types of technology as an abstract entity, detached from any political and economic drivers and social contexts, irrespective of their individual natures, and pays little attention to the way that it is used in a specific socio-technical context and circumstance. Different technologies interact with human society in different ways and at different socio-technical interfaces (Shen and Duan, 2007). For example, OS software is an infrastructural technology, which unlike many discrete technologies has strong network effects (Bresnahan, 2001; Brynjolfsson and Kemeter, 1996; Cowan, 1992). It requires a critical mass of users and interoperability or other network effects to survive and to sustain its position in the market. In other words, the more users that use the software, the more application software will be developed to port on this infrastructure and the more valuable it becomes to each user. To switch from one OS to another is costly, not just because of the cost of the software itself, but the cost of the entire system consisting of many other technologies, such as middleware and application software, and human resources who are maintaining the system. These lock-in effects (for instance, interoperability benefits) may well turn the initial choices of users into *de facto* standards creating barriers to entry for newcomers. A technology as such is a socio-technical ensemble (Bijker, 1995) and moreover a living thing, the further development of which is a marriage between the technology and the specific socio-technical environment.

From the perspective of SST, Shen (1999) defines the term ‘reshaping of technology’ to portray the adaptation and further elaboration of technology development on site of the transferee/adaptor in a DC and ‘indigenous technological capabilities’ gained by the transferee which may well be different from those of the technology supplier/developer. Technology transfer is not a simple spatial movement process; instead technology needs to be adapted to fit into the local context. The social value and technical presumptions of technology embodied when it was developed in a developed country in the West may not fit the socio-technical environment in a transferee country in the developing world. The adaptation of an infrastructural technology like OS software is also a marriage between the technology adapter of a special site and the global developer. The adapter’s self-assigned binding with the developer community may not have the consent or even the awareness of the developer site in the early stages. However, it may change alongside the continuous development of the infrastructural technology, depending on the weight of the network branch that the adapter is creating and taking under control.

In general, the government role is pivotal to adapt an infrastructural technology like OS software, if it is considered of public interest. It is truer for a DC, as there is often an asymmetry in technology capabilities between the adapter and the developer from developed countries. However, just how the government can best play an effective role in supporting indigenous technology development and creates an effective institutional structure, is still highly debated (Gallagher, 2006; Hobday and Rush, 2007; Huq, 2004; Lall, 1992; 2000; Madanmohan, 2003; Noland and Pack, 2003; Pack, 2000; Shen, 1999, 2005).

2.4. Summary

According to the above two sections of literature review on OSS and the area of technology studies, the gaps between existing theories and reality gradually become lucid. The first group of gaps is related to OSS. Firstly, the existing literature cannot provide sufficient understanding of the open source movement in the context of DCs. The reason is that a new research methodology is needed to examine the design and implementation process of OSS in the context of DCs. Secondly, there still is a debate on government intervention in OSS development. The second group of gaps is associated the field of technology studies. On the one hand, social learning perspective emerged from the research conducted in the EU countries. It needs to extend this approach to a context of DC and further explores how the actors interacted with each other in the DC's context. On the other hand, the software technology transfer and corresponding local technological capabilities building is under-researched.

2.4.1 Open Source Software

Limitations on existing OSS literature to DCs

It is noteworthy that the theories on the OSS reviewed in the beginning of this chapter, are generalised from the relevant research in developed countries. These theories may not provide adequate understanding of the open source movement in developing world. To the mainstream social scientific researches conducted in developed world, the open source movement is the process of OSS development and innovation in communities: these researches studied and provided understanding of the motivations of programmers participating OSS communities and contributing their ability to OSS development, the administrative structures of OSS development, and the licenses of OSS. However, the users in DCs are different because most of them have very limited technical knowledge, which is compounded by the obvious language barriers. This is why the great mass of open source projects are controlled and administrated by the communities in the developed world. The open source movement in DCs' context, hence, is not the ideology behind the open source movement (such as the process of OSS development and governance, and the structure of communities) but the uptake of OSS.

According to the open nature of OSS and its development process in developed world, many authors predict and reiterate the benefits to DCs of reaping OSS. The starting point of these authors is taking the use of OSS for granted – as if the OSS would be suitable for most of the needs of users in DCs. Like peer researchers in other countries, nearly all of the Chinese authors also had a naïve view of OSS (including Linux), which is that the Chinese users would automatically adopt it owing to its technical superiority, and then benefit from its implementation. It is obvious that these social scientists lack an efficient research framework to study OSS in DCs due to the disciplinary division: most of researchers were from business schools, investigating the economic and managerial performance of OSS implementation in DCs. The BoA researchers (Pollock and Williams, 2009) critique this kind of 'impact study' because it lacks critical concerns: focusing only on the benefits generated by using OSS; characterising as short-term studies which concentrated on the immediate performance of OSS implementation; 'black boxing' the long-term evolutional history of OSS as well as ignoring the broader social context where the OSS locally developed; overlooking the social and political elements which cannot be controlled by OSS developers and local adapters.

Take China as an example, where the uptake of Linux has not been a natural and smooth process. There were over ten indigenous Linux providers that were founded in a rather short period of time after 1998, when Linux was introduced to China with government backing. In the subsequent two years, however, many of these players left the stage because they encountered various problems, notably an appropriate model for business that could sustain the finance of the Linux industry. Other studies reveal that the suggested benefits to DCs are far from straightforward; for example, Shen (2005b) points out that the affordability of Linux became rather irrelevant in China due to local specific social settings, such as rampant software piracy (also see Kshetri, 2004 and Ni, 2006).

Therefore, these issues give rise to many empirical and conceptual questions. Empirically, who were the actors during the Chinese indigenous Linux development and how have they interacted with one another? What have been the main challenges during Linux development and how have they been tackled?

Conceptually, how does the local technological capability get established? How does the OSS adapt into the local context of DCs? How does the experience contribute to further technological development? What are the key challenges for DCs to fully reap the benefits of the OSS that were not envisaged initially?

Government intervention in OSS development

A debate win mainstream OSS literature regards government's support towards the OSS. Based on the studies conducted in developed world, the proponents of the open source movement (Raymond *et. al*, 2000), political scientists (Evan, 2002) and economists (Schmidt and Schntizer, 2003) share the opinion that the government should not provide support towards OSS. By contrast, Shapiro and Varian (2003) and Comio and Manent (2005) propose that a selective public support policy will enhance the social welfare (also see Part II, FLOSS FINAL REPORT, European Commission, 2002).

Governments in DCs, like Brazil, China, India and some countries in Southeast Asia, adopted many policy initiatives to favour local open source movement. They wished to acquire benefits by the means of involving domestic open source movement directly and aggressively. The authors who studied OSS in DCs acknowledge the pivotal role of government in local OSS development. They indicate that the governments in DCs must lead the introduction of OSS because of their crucial role as purchasers, technology drivers and policy-makers (Cook and Horobin, 2006; Li, Lin and Xia, 2004; Ni, 2005 and 2006). Camara and Fonseca (2007) further stress that OSS in developing nations needs strong and wise policies to be successful and furthermore, need government funding to be viable.

Ultimately, is government intervention necessary for the open source movement in DCs? If so, what kind of state intervention or measure is workable?

2.4.2 Technology studies

Social learning in DCs

Social learning stems from the empirical studies in the developed world, in particular EU countries. It has not extended to DCs so that a DC context has not yet been taken into account. A reality in DCs is that on the one hand, the local technology depends largely on foreign ICT products. On the other hand, the dominant position of foreign products is entrenched owing to the technology dependency and lock-in effect. Many governments in developing world have tried to break the monopoly generated by foreign products and further build their own indigenous ICT industries by adopting policy leverage tools. Therefore, the role of government in developing economies is normally more important in the social learning process. In addition, indigenous technology suppliers conventionally face more difficulties than their counterparts in developed world due to the lack of industrial bases, finance resources and commercial experiences. Social learning perspective is a state-specific tool so that its extending to the DC context may help enrich its theoretical framework.

Technology transfer in DCs

The traditional technology transfer perspective provides an important insight to understand technology innovation in DCs, and analyses the process of local technological capabilities building. However, it was developed to account for the innovations in manufacturing and industrial technologies, such as heavy chemicals, automobile, steel, and semi-conductor industries. There are sharp differences between industrial technology and ICTs, in particular the software technology. Unlike the users for most transferred industrial technologies are normally organisational users in a specific domain, the users of ICTs could be personal users in the mass market and organisational users in all application domains.

Computer software has been widely applied in all sectors, areas, organisations, and everyday home life. This context where software has been applied is broader, and the way to software is diverse than industrial technologies. The uncertainties of user acceptance and user demands of transferred software technology/knowledge are consequently far more salient because of the diversity of users' requirements and the extremely intricate interactions between suppliers and final users in the market. Furthermore, software also has a stronger effect of positive network externalities than industrial technologies: the spreading of software is subject to user base and its compatibility to complementary computer software and hardware.

It is clear that there is a gap between the traditional technology transfer and the process by which software technology/knowledge is accommodated in DCs. This gap prompts several on going questions: How can the software technology/knowledge adapt in the context of DCs? What is the process by which DCs can build their local technological capabilities? Who are the main actors in the process of building these capabilities, and what are the links between them?

Chapter 3 Research Design

3.1 Introduction

This study sets out from a tradition of technology studies, in particular the SST perspective. The SST framework has been developed as an interdisciplinary theoretical framework with which the black-box of Linux and its development process in China can be opened up by means of both conducting detailed temporal implementation studies and examining the evolutional history of Linux longitudinally. Instead of leaving technology as an exclusive preserve of scientists and engineers, SST seeks to address the intricate interactions between social and technical features within a given context. This study is based on the case study method of the social shaping approach, addresses the specific socio-technical system within which Linux development takes place, maps out the actors involved in China’s Linux development, explores the interplay amongst actors, and analyses the local institutional mechanism which supports and regulates Linux development.

Section 3.2 describes the related framework, methodology and design, as well as the choice of research strategy. In section 3.3, we present data collection methods that have been adopted, and both the techniques and processes of analysis deployed in this study. Section 3.4 discusses the experience of conducting data collection in China, and explores the ways to manage difficulties during the process of data collection.

3.2 Research Design

3.2.1 Framework and methodology

A large part of traditional mainstream literature about software development, implementation and use was informed from the perspective of computer science, business studies and economics. These studies led to fragmental understanding, on the grounds of limited knowledge base and disciplinary division. As a result, a new and effective methodology is needed for better understanding of the relationship between technology and society, in particular the influence of broader social structures, the behaviours of the main actors and the interactions between them (Pollock and Williams, 2008).

The research frameworks of current social scientific studies on OSS can be divided into technology design/development studies and snapshot/impact studies. The former is conducted from a technology-supply perspective, which discusses the important issues in the innovation process of OSS e.g. the personal and commercial motivations involving OSS innovation, the governance structure of OSS projects and communities, the prevalent OSS licences and their impact on innovative work etc. This research framework overlooks the users in the mass markets, and therefore is influenced unduly by the vision of OSS communities and promoters. The latter has a taken-for-granted and over-optimistic view on OSS implementation and use – as if users would be amenable to switch to OSS from proprietary software and benefit from its adoption. As reviewed in section 2.2.6, many academic authors, especially those from DCs, claim that using OSS helped users to save procurement costs, escape the dependent relation from proprietary software companies, break the software monopoly, and so forth. As Pollock and Williams (2008) point out, the snapshot framework normally considers a rather short timeframe after the emerging technology (in fact, the full benefits and costs of adopting OSS will be unveiled after years, or even decades). There are also uncertainties during the implementation of OSS. The work and use practices embodied in the OSS may be very different with the proprietary software, which could lead to implementation failure. This is reflected by a reality that OSS still accounts for a small market share, though the procurement cost is much lower than proprietary software. Moreover, two prevalent research designs and frameworks are keen to concentrate on either technology design/development or the impact of technology implementation and, therefore, lack the alignment between technology design and implementation, as well as neglect the broader local context within which OSS is developed and used.

In order to provide better understanding of the open source phenomenon and OSS design and use, a new research methodology and framework is needed to remedy the shortcomings of prevalent frameworks. Therefore, we need theoretical tools and concepts which can address:

1) short-term dynamics surrounding OSS implementation in specific site and moment, in particular the collision between users' requirements and suppliers' product development strategies;

2) a long-term evolution in which OSS travels across the boundaries of organisations, industries and sectors; and

3) the broader local context encompassing institutional arrangements, economic and social elements.

This study builds a research framework by drawing upon the concepts of SST, SSICT, BoA, social learning and infrastructure perspectives. SST and SSICT provide critical accounts to remind us to avoid any deterministic view in technology innovation studies, as well as to couple the technology design with its implementation in tandem (Williams and Edge, 1996; MacKenzie and Wajcman, 1999). Like many other technologies, the development of Linux always faced many design choices, which not only happened in Linux communities but also took place in the laboratories of commercial Linux suppliers and in Linux implementation and use. Besides technical considerations, the social, economic and institutional factors are also elements for patterning Linux.

The BoA is an emerging research framework addressing the multi-site analysis, paying attention to technology design and the implementation cycle and focusing on the interaction between design and use. It also stresses the historical stages (in particular both the pre- and post- implementation stages) by integrating longitudinal and contextual views. This study employs BoA in a rather loose way so that attention is given to the technology design. Giving attention to design and development from a technology suppliers' perspective does not mean that we are going to ignore the OSS implementation and use. The understanding of OSS implementation and use could help us to explore suppliers' later design choice. By employing BoA perspective, the research framework of this study sees the OSS innovation taking place in ‘micro-, meso-, and macro-’ levels: the ‘macro’ level normally encompasses the issues (such as institutional arrangement, economic and market elements etc.) that could largely influence the OSS innovation, and could not be controlled or changed by suppliers and users; the ‘meso’ level, in this research, is seen as the socio-technical regime in which many actors, including OSS suppliers, government, public research institutes/universities and so forth interact with each other to build a technical knowledge base for OSS design and innovation; the ‘micro’ level, finally, is the place where the interaction and learning mechanism take place between OSS suppliers and users/intermediaries.

In relation to the objective of this research, a social learning perspective is adopted for investigating the mechanisms and processes by which OSS suppliers tried to learn through interactions with other social groups, in particular the local users in micro level. Social learning perspective acknowledges the importance of local sticky knowledge (von Hippel, 1994) from user end and draws attention to the role of intermediaries (Williams, et al, 2005) who are able to bridge the gap between OSS suppliers and end users by collecting and collating knowledge, and transforming it to OSS suppliers. Although social learning emphasizes the mutual shaping of technology and society, this study mainly focuses on the shaping of Linux in China’s local context.

As we defined Linux as infrastructural software in section 2.3.2, design and developing Linux for local Chinese users will face similar dynamics and challenges as developing other infrastructures. With the concepts of infrastructure studies, we are able to understand China’s local context and identify the dynamics and challenges surrounding localisation innovation of Linux.

Following the tradition of the SST perspective, a double methodological approach is adopted in this study: on the one hand, it seeks to explore the detailed account of technological development processes. It is made possible by traditional actor-centred approach at the micro-level – following the technology suppliers cross different loci and over time. The actor-centred perspective helps us to identify possible actors who interacted with technology suppliers in implementation sites during the long-term technology evolution. It also highlights the way in which technical elements shape and are shaped by social elements through interacting with each other. On the other hand, this methodological approach addresses the influence of broader structural and historical factors. It helps to understand the context in which the technology innovation takes place. It further emphasises the role of government and the institutional structure of a national innovation system for technological development.

This research adopts case study as the research strategy. Empirically, a case study approach is most suitable for this research. There are many existing materials about Linux development in China, such as data about Linux sales, and the revenues and profits generated by Linux providers etc, as well as government official documents (reports and policies). However, they are static data or results in a certain period of time, so none of these materials show what has happened during such a time period, and therefore cannot answer the research questions. Instead, a case study approach is able to help to reveal the Chinese Linux development process in which we can find the main actors and their interactions. Theoretically, case study is a popular research method in the area of STS. Many prevailing conceptual perspectives and theoretical frameworks, such as the SST perspective, social learning, BoA, etc are developed and enriched by employing detailed case studies.

The pattern of technology is largely influenced not only by current but also by historical events and contexts (Pollock and Williams, 2010). Following the social shaping studies tradition and the notion of biography, we trace the 'accumulated history' of a special kind of OSS – Linux, focusing on the changes of suppliers' strategies and other actors' actions inside various sites over time. In this sense, setting up a time scale for the case study is important. Linux was introduced by suppliers in 1998 inspired by the Chinese government. Both embedded Linux and platform Linux were adapted and delivered to the Chinese market in 1999. Figure 2 was developed to show the time scale of the evolutional history of Chinese indigenous Linux.

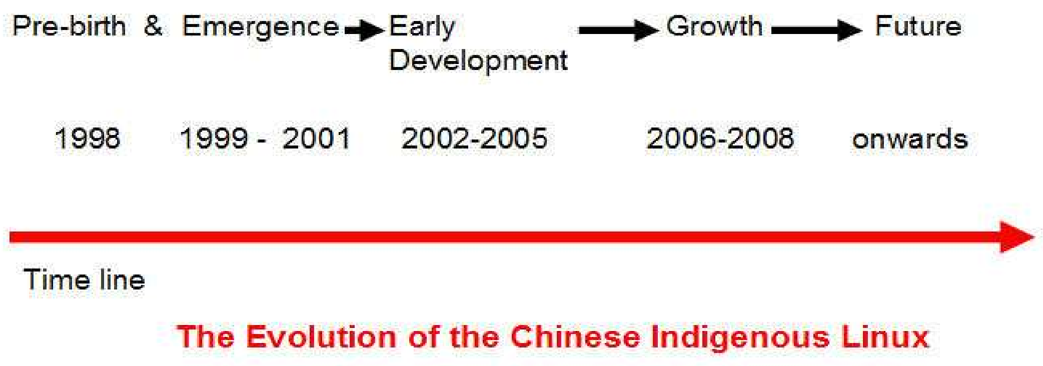

Figure 2: The evolution of the Chinese indigenous Linux

3.2.2 The selection of case study

I decided to conduct the case study on a firm level because firms are the final nodes for Linux entering into the software market, as well as being the places where most R&D activities were carried out. Furthermore, Linux providers are the key actors who have direct links to all other main actors, such as academia, users, government, etc.

Initially, I planned to examine all powerful Linux providers in China's software market so that I could conduct a comprehensive case study on Chinese Linux. These mainstream Linux providers are Novell and Red Hat from overseas, and two domestic Linux suppliers: Red-Flag and CS2C. Novell and Red Hat are the most famous Linux firms worldwide, while CS2C and Red-Flag started Linux businesses nearly one year after Linux was introduced to China in 1998. CS2C and Red-Flag are part of the very few indigenous survivors in the fiercely competitive market, and both became profitable from the end of 2004. According to the data from IDC (International Data Corporation) and CCID (China Centre for Information Industry Development), CS2C and Red-Flag are the only two indigenous Linux companies among the five largest Linux providers in China's market, and their Linux products accounted for over 40% of market share from 2006.

I had hoped to make explicit comparisons between domestic and foreign providers on their Linux design and development. However, the strategy of comparison between cases has been shifted towards a single in-depth case study after several pilot interviews. This decision was made largely due to the feasible accessibility of Chinese suppliers and poor availability of access to foreign providers. Both CS2C and Red-Flag were glad to offer me the opportunity to conduct field work and collect data. Another factor is that the main business of foreign Linux providers, such as Red Hat and Novell (SuSe) was focused on Linux sales in China before 2005. They offered very limited services, such as system consulting, solution providing and technical support to Chinese users. As a result, their products were not well adapted in the Chinese market because they lacked ties to local user groups.

In accordance with the existing literature, I initially intended to conduct case studies with two units: Linux desktop version and Linux server version. However, after pilot interviews, I regrouped the research units. Linux is a widely used OS in the fields of desktop PC, server, and embedded systems in China, and even in the rest of the world. In light of the traditional Chinese category method, Linux desktop and server versions fall into the category of platform Linux, while the embedded Linux is another independent group. I adopted this category directly since it covered all application areas of Linux, in particular the embedded area which has been neglected in most social scientific research of Linux.

Based on the later two research units, I chose CS2C as the research object in platform Linux and Red-Flag in embedded Linux. Both CS2C and Red-Flag were engaged in developing Chinese platform Linux from the very beginning. However, they later changed their initial technological directions because of the fierce market competition from both foreign proprietary OS providers and indigenous Linux developers. CS2C developed some notable platform Linux solutions which were not only frequently reported by Chinese publications, but also received attention from overseas. For example, CS2C designed and developed Happy Family PC solution for Chinese rural users, which was the first customised Linux-based desktop solution in the world. Therefore, this solution received acknowledgement from the World Bank because it is regarded as a feasible solution for solving the global digital divide. Even though Red-Flag still kept delivering platform Linux, the supplier also showed stronger technical and market power in the embedded field. Red-Flag is currently the largest indigenous embedded Linux provider, and its products almost dominate some embedded application fields, such as the Chinese lottery machine industry.

3.3 Data Collection and Process

According to the methodology of this study, which is informed by SST framework and related theoretical tools – social learning and BoA, both historical and contemporary perspectives are taken into account in this case study, which meant the longitudinal and contextual data has been collected. The time period of the case study is from 1998, when Linux was introduced to China's software market, to 2008. During this 10-year evolving process, the detailed temporal implementation projects are also studied. As described in 2.3.1, not all implementation projects would be examined in detail. This study only examines the important projects and moments – milestone projects which had significance to the next generation/version of Linux.

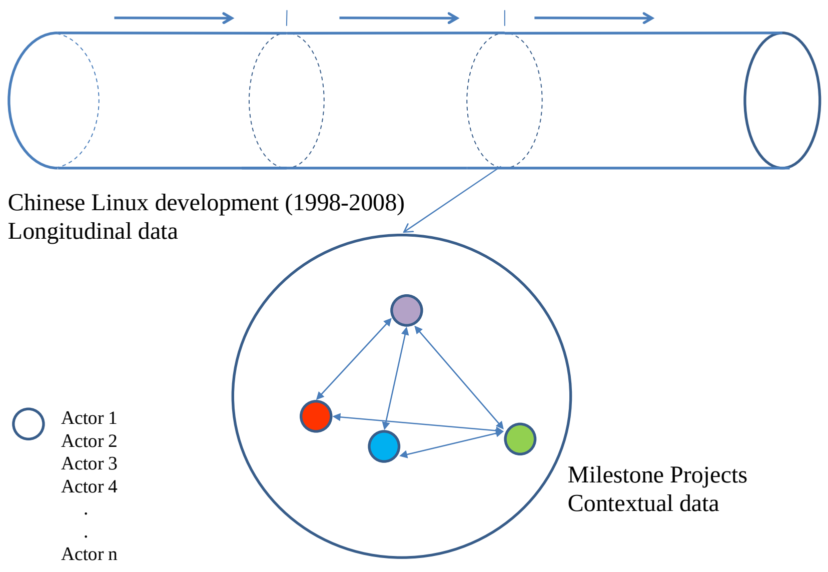

Figure 3: Longitudinal data and contextual data

3.3.1 Data source

The questionnaire, interview and content analysis of documents are the most common data-gathering methods in social science studies (Blaikie, 2000). But the questionnaire method will not be employed in this research design. On the one hand, unlike common social science studies, this research is focused on the specific case of Chinese Linux development by employing an interdisciplinary framework. The case is not a new social or business phenomenon so detailed statistical data has been already analysed by Linux suppliers and other third-party organisations, e.g. consulting companies and related government departments. All the data, such as sales volume of Linux, sales growth/decrease rate, the percentage area of Linux applications (personal users, server-side application, embedded system and so forth) form the basis for research. Additionally, the method of conducting a questionnaire is a costly and time consuming process and therefore it was considered too expensive for this study. Furthermore, the statistical data obtained directly from Linux suppliers or other third-party organisations is likely to be more reliable and precise than the data gathered by a PhD student.

On the recommendation of Yin (1994) on the use of multiple sources of evidence, I chose the triangulation approach: the primary sources are interview, complemented with documentation.

Primary data

In light of the specific characteristics of this research design and the nature of the interview questions, I set up two steps in the interview process: the first step was conducting open-ended interviews with the top management of both CS2C and Red-Flag, in order to grasp the bigger picture of Chinese Linux development. On that basis, I was able to explore the milestone projects, which helped identify the interviewees for the second step of personal interviews. For the interviews, I adopted the semi-structured interview technique recommended by Easterby (1991), whereby the researcher is able to build a rapport with the interviewees and reduce their reluctance to respond openly. In Chinese tradition, the manner of semi-structured interview avoids a rigid question/answer process, and expresses respect to interviewees. For the interviewers, the semi-structured method allows the researcher to manage the interviews more easily. It acts as a reminder to the interviewer and interviewee to stay within the topic, since it provides a structure which researchers can follow, and offers flexibility to expand the interview content as necessary.

In total, 44 interviews were conducted and each of them lasted approximately 60-90 minutes. The 18 interviewees who were selected from the first step interviews are from software industry, various user groups, policy research institutes, academic community and policy-making groups.

Most interviewees were interviewed more than once. Generally speaking, both Chinese platform Linux and embedded Linux development are complicated processes with several detailed cases to be examined. It is difficult to collect adequate data through a single interview owing to time constraints. Another reason for the limited number of interviewees is that the indigenous Linux providers, including Red-Flag and CS2C, are typical SMEs (Small and Medium Enterprises) so they have limited staff for each department. Take the manager in the embedded Linux department of Red-Flag for example, six interviews took place. His responsibilities are managing the development team consisting of less than five software engineers, coordinating the relationship between the embedded Linux department and the company, communicating with customers, and even participating in the product R&D and technical tasks directly, if necessary. The manager is the staff member with the longest experience in the embedded department because the IT industry is characterised by high personnel mobility. Due to the better salaries, software engineers are likely to transfer to MNEs once they have accumulated enough experience through one or two projects. The manager, therefore, is the only interviewee who can provide answers to the interview questions regarding both the detailed development history of Red-Flag embedded products and the processes of specific milestone projects.

Interview design

Interviews from China's software industry

Interviewee	Related details	Interviews
VP (Vice President) 1 of Red-Flag	Strategies and development of embedded product.	2
	The cooperation between Red-Flag and ISCAS	1
VP (Vice President) 2 of Red-Flag	The government policy.	1
Manager in embedded department of Red-Flag	Detailed development story of embedded system	2
	SinoData SN2000 and SN3000 cases	3
	The cooperation between Red-Flag and ISCAS	1
President of CS2C	General development story of Linux platform product, especially in early stage.	2
	Government policy and support	1
Support and Server Director of CS2C	The product line of CS2C	1
Senior Product Manager in R&D centre of CS2C	Detailed development story of Linux :	
	platform product	1
	Sailing project	1
	Haier project	1
	China Construction Bank (CCB) project	2
Director of CS2C & BUAA (Beijing	The cooperation between CS2C and BUAA	1
	Government policy, in particular the	1

University of Aeronautics & Astronautics) Joint Laboratory, Professor in Software School in BUAA	government funded research project	
President of Red Office (Chinese open source office productivity suite developer)	Government policy to open source software and Linux	2
Staff of Open Source Technology Centre, Intel	The role of Chinese Linux in Intel's Open Source ecosystem.	2
Total: 9		25

Table 1: Interviews from China's software industry

Interviews from user communities

Interviewee	Related details	Interviews
User 1: CTO, SinoData (embedded Linux user)	Why did SinoData choose Red-Flag embedded Linux Development stories of SN2000 and SN3000 lottery terminals.	1 1
User 2: Deputy Head of Beijing Science & Technology Commission (platform Linux user) Director of Beijing Software Industry Productivity Centre	Beijing Government Software Procurement Sailing Project Sailing Project	1 1

(platform Linux user)		
User 3: Manager of PC department, Haier (platform Linux user)	Happy PC desktop Linux-based software system	2
Total: 4		6

Table 2: Interviews from user communities

Interviews from policy researchers in government affiliated institute

Interviewee	Related details	Interviews
Prof. Guangnan Ni (fellow of China academy of engineering). Participating all Linux related policy-making	China's government policy to Linux The limitation of existed policies Future programme for supporting Chinese Linux	1 1 1
Dr. Wang, Associate professor in CASTED (Chinese Academy of Science and Technology for Development)	China's government policy to Linux Evaluation of past and current policies	2 2
Mrs. Xu Professor in CASTED	Government software procurement policy to Chinese Linux	1
Department Director of IPC (Intellectual Property Centre)	China's government policy to Linux	2
Director of Guangdong Linux Centre	The China's national policies Local (Guangdong province) policies The implementation of the policies	1 1 1

Total: 5		13

Table 3: Interviews from policy researchers in government affiliated institute

The detailed interview content

According to the development history of Linux in China, I set pre-birth and birth stage, early development stage, and growth stage for both platform Linux and embedded Linux.

The platform Linux of CS2C

Development of platform Linux in China	Interview Guideline
Pre-birth and emergence 1999-2001	What was the context and motive in which CS2C entered the Linux field? What was interpretation of CS2C on Linux? How did CS2C assume the use and users and then formulate its concept of Linux? How did CS2C interact with other social groups, such as user, government, complementary software providers, as well as academia, to develop generic version?
Early development 2002- 2005	What were the reasons that lead to the forking of desktop and server version from the original general Linux package? How did CS2C interact with other social groups in developing Linux desktop and server version respectively? How did the customised project – Sailing project feed back to its generic solution?
Growth and future 2006-onwards	What is the context within which CS2C currently develops Linux? How did the customised project – Haier Happy Family project and CCB project feed back to its generic solution? How did CS2C

	interact with other social groups? What's the strategy of CS2C in desktop and server product?

Table 4: Interview content for platform Linux

The embedded Linux of Red-Flag

Development of embedded Linux in China	Interview Guideline
Pre-birth and emergence 1999-2001	What was the context in which Red-Flag entered the embedded Linux field? How did Red-Flag formulate their concept of embedded Linux product? What did Red-Flag learn from other social groups within their product development?
Early development 2002-2005	To cultivate product, how did Red-Flag interact with other social groups? What did Red-Flag learn within the customised project – SinoData SN3000 lottery sale terminal?
Growth and future 2006-onwards	How did CS2C interact with other social groups? What's the strategy of Red-Flag currently and in the future?

Table 5: Interview content for embedded Linux

The Chinese government policy to Linux

In the case study of Chinese Linux development, the role of government and the administrative measures to support indigenous Linux are changed in accordance with different stages of the Chinese platform Linux development trajectory.

Development of platform Linux in China	Interview Guideline
Pre-birth and emergence 1999-2001	What was the role of government at this stage? And what kind of measures did government

	adopt?
Early development 2002-2005	Did the government change their role? If so, why, and what new administrative measures did the government employ?
Growth and future 2006-present	What are the strategies and policies of government at this stage? In particular, what are the new tools and measures adopted by the government? Are past government policies and measures relevant for Chinese Linux development? What should the government do in the future?

Table 6: Interview content for government policy

Secondary data

Content analysis of documentation will be adopted in this research as secondary data sources. Documentation is not only used for setting the context for interview, but also employed as evidence to validate the interview through a triangulation of data sources (Remenyi, 2000; Yin, 1994). The documentation consists of the functional descriptions of Linux products, reports on user requirement analysis from suppliers, paper-based records of meetings between suppliers and users, internal email between suppliers and users, government policy documents, articles on third-party newspapers, magazines, academic journals, Internet content etc.

In order to examine the social-technical network of Linux development, the user is the key actor to be accessed. However, adopting Linux, in particular employing platform Linux, is still a controversial issue for the users who have already adopted Linux. The Linux suppliers therefore implied that they were unwilling to facilitate access to their all clients. In order to tackle this problem of accessibility, on the one hand, I tried to link these users via other conduits, such as private networks and academic links; on the other hand, some internal official certificated documentation was employed as supplements to interview data. There are a great number of Linux implementation projects financially supported by various Chinese central and local R&D funds. Some of these projects, such as the Sailing project and the CCB project, are identified as milestones for China's Linux desktop and Linux server respectively. However, not all target individuals of the user organisations can be reached smoothly due to a variety of reasons, such as the high staff mobility of the IT industry, the political sensitivity, commercial secrecy etc. Therefore, the detailed official reports submitted by user organisations to the financial providers – government and/or public R&D funds, are a very important source to remedy the lack of accessibility to user organisation in person. Some chapters of the reports of both the Sailing project and the CCB project in different stages were obtained from relevant government agencies. These documents elaborate the projects from a user perspective. Table 7 gives a big picture of the links between research questions and the source by which we are able to collect data.

Research Questions	Sub-Question	Data Source
1. How has Linux adapted in China's broader contexts?	(1) Which struggles and tensions did the Chinese supplier face through the development of Chinese Linux? Which strategies did they use to enable them to deal with these problems?	1. Personal interviews with Linux suppliers and local user groups. 2. Documentation: functional descriptions of Linux products, user requirement analysis and reports from Linux suppliers, records of meetings between suppliers and users, internal email between suppliers and users, and internal government reports submitted by users.
	(2) How did the Chinese	1. Personal interviews with

	socio-economic context and institutional arrangements influence the way in which the Chinese Linux was adapted and developed?	Linux suppliers, user groups, academic researchers, and policy researchers. 2. Documentation: academic papers, articles on newspapers, magazines, and Internet content.
2. What is the role of Chinese government in Linux adaptation and development?	(1) Are the government interventions necessary for Chinese Linux adaptation and development?	1. Personal interviews with Linux providers, user groups, staff from other IT companies, and policy researchers. 2. Documentation: academic papers and internal government reports submitted by users.
	(2) What are the political measures adapted by Chinese government?	1. Personal interviews with Linux providers, user groups, staff from other IT companies, and policy researchers. 2. Documentation: government policy documents, academic papers, articles in newspapers, magazines, and the Internet.
3. What are the wider implications that other DCs		1. Personal interviews with Linux suppliers, policy

can draw from China's experience in relation to adapting an infrastructural OSS into their local environment?		researchers, and member of policy-making team. 2. Documentation: academic papers and articles on news paper, magazine, and Internet.

Table 7: Research questions and data source

3.3.2 Data process

The interviews were either voice-recorded or handwritten and eventually transcribed. All the interview data from different respondents was presented in historical sequence. I did not accept all that the interviewees described as the truth without taking their background into account. For example, the CS2C and Red-Flag treated each other as main competitors in the platform Linux market. The interview data from the staff of both companies had to be interpreted against the interviewees' background, in particular the content regarding a rival. Furthermore, nearly all the interviewees did not like to clearly state the drawbacks or the negative sides of government policies; instead they tended to imply this. At time interviewees' responses were unclear and it required interpretation of their responses to understand their viewpoint. Combining the interview/primary data with documentation and other secondary data enables us to construct the qualitative historical development of Chinese Linux.

A triangulation of interviews and documentation was employed. The documentation generally serves to increase the interview reliability by triangulating the data sources (Yin, 1994). However, the documentation retrieved from public media, in particular the Internet, cannot be regarded as wholly reliable data without checking its validity. As a result, another checking process is used in this research. Some of the interviewees were given drafts of my working papers which summarised the data from both interviews and documentation for feedback if they were relevant to their area of expertise.

Some working papers in Chinese which are organised by both previous interviewing data and the data collected from public media were sent to the interviewees who have relevant experience and expertise to verify the authenticity.

3.4 Conducting Field Work in China

3.4.1 Accessibility

To all social scientific studies which employ case studies, the accessibility of the case is of primary importance. "*The best case should be the one from which we can learn the most and we can spend the most time with.*" (Denzin and Lincoln, 2003). Accessibility in this case study means both accessing the objective organisations and interviewing individuals.

One of the difficulties that can impede the research is political sensitivity. As the Chinese government adopts administrative leverage tools to manage all aspects of social life in China, the role of government is an inevitable feature for social scientific studies. In this case study, the government plays a very important role in the development and spread of indigenous Linux because of national interests, such as breaking the monopoly generated by foreign companies, protecting domestic software industry, etc. The topic of policy and measure towards Linux will be sensitive to researchers, in particular those who come from abroad. In addition, Beijing hosted the 29th Olympic Games in the middle of 2008. The Chinese government treated the year of 2007 and 2008 as an extremely sensitive period. In the subsequent year of 2009, four staff in the Shanghai office of the Rio Tinto Group were accused of bribery and espionage. These two political affairs led to my request of accessing government officials to conduct interviews, to be denied outright.

Another difficulty is business sensitivity. Normally, some respondents, in particular interviewees in for-profit organisations, have no interest in social scientific studies which are regarded as leading no direct profit growth in China. Having the same foreign competitor, the relations between major indigenous Linux suppliers, like CS2C and Red-Flag, should be co-operators on marketing level. However, the reality is that these two Linux providers treat each other purely as competitors in the domestic market, especially in the government procurement segment, so interviewees from Red-Flag and CS2C were unwilling to answer some questions in detail in order to retain commercial confidentiality.

3.4.2 Managing the difficulties

As described above, conducting field work in China may produce problems of accessibility for researchers coming from foreign countries due to different political systems, cultures and context.

An ‘encircling’ strategy was adopted to cope with the political sensitivity. The S&T policy researchers in affiliated research institutes of government agencies, such as the IPC (Intellectual Property Centre,) and CASTED (Chinese Academy of Science and Technology for Development), were chosen as the alternatives to individual government officials. The S&T policy-making process in China is a collective decision-making process, which involves not only the government officials but also the top national experts in their research fields. Prof. Guangnan Ni (FCAE Fellow of China Academy of Engineering) in the Institute of Computer Technology of CAS, is a participant who is fully involved in all of the OSS related policy-making processes. As a fellow academic researcher, Prof. Ni was happy to be interviewed..

Regarding business sensitivity, establishing trust is given priority in China (Fukuyama, 1995). Based on my experience, there are three steps to achieve trust. The first step is establishing contact via authentic conduits. Personal links are of huge importance for generating contact and even further links via existing connections (Yang, 1995). Drawing upon my supervisor’s private links, I established contact with CS2C and IPC directly. With the assistance of my supervisor, I obtained opportunities to contact Red-Flag via the ISCAS (the largest shareholder of Red-Flag) and CASTED via the Institute of Science, Technology & Society, Tsinghua University. The second step is generating the interviewee’s interest.

Before entering field work in mid 2007, I completed several working papers regarding every aspect of Chinese Linux development based on literature review and previous research. One paper has been published in an international conference (Zhou, *et.al.*, 2007). As a result, I was able to connect quickly with every interviewee and could even present my papers to the interviewees, according to their expertise. The final, but not least step was to establish a rapport with interviewees (Peng, 1999). There is an old Chinese saying: “*Learn to be a right man before doing right things.*” In order to achieve that, I shared not only my Linux related knowledge which was mainly accumulated from English literature, but also the knowledge and experience beyond this research, such as personal experience in Europe and my hobbies, with interviewees.

Chapter 4 The Qualitative Historical Development of Red-Flag Embedded Linux

4.1 Introduction

This case opens the process of Chinese embedded Linux system evolution within the context of China. The focus of the chapter is to understand the strategies employed by Chinese embedded Linux technology supplier – Red-Flag, in order to deal with the tensions between diverse user organisations' requirements, the intention of developing a generic solution, and the sources of knowledge and skills for building up the generic features of the solution.

This chapter is organised according to the evolutional stages of Red-Flag embedded Linux. It begins by discussing the generation of the embedded product market in China and the technical differences between embedded computer systems and general-purpose computer systems. The technical features of embedded computer systems are the clue to demonstrating the requirement of establishing close ties between embedded system suppliers and their clients, which also is the basis for understanding the complex interplay between suppliers and clients.

Sections 4.3 describes the emergence of Red-Flag and the earliest evolutional stage of embedded Linux product: the pre-birth and emergence stage. In this stage, the development strategy adopted by Red-Flag catered to all the user requirements in whole Linux-based solution design, in order to accumulate knowledge of user applications in different domains. At technological level, supplier made the first attempt to collect generic features of embedded Linux design and development, though this attempt was seen as unfeasible later. As a result, the product strategy shifted from developing embedded solution to providing an embedded Linux customising service and an EDK.

Section 4.4 elaborates the events that happened in the early development stage of

Red-Flag embedded Linux, including the financial support from the Chinese government and a milestone project – design embedded Linux for SinoData SN-3000 lottery sales terminal. In this stage, a shift on supplier's development strategy was made from catering to all user requirements to catering to the 'generic' user requirements on Linux. The latter is characterised as the requirements which have market potential, i.e. functions can be applied across the boundaries between different domains, as well as some updated hardware.

Section 4.5 portrays the development process of Red-Flag embedded Linux in the growth and future stage, in which the supplier faced many tensions that forced supplier to shift to new strategies on both customer management and product development. Red-Flag in this stage, divided customers into two categories: leading customers and normal customers. The former are the customers who wanted to work with supplier and spend money on new function development. The latter are the users who wanted to develop embedded solution by their own and were willing to spend limited money for their embedded project. In order to match the needs of the latter customers, Red-Flag develop a generic solution – 'DevsPartner' which enables users to tailor embedded Linux based on mainstream hardware framework and functionalities, and also to develop generic applications, such as video playing, due-screen displaying, etc. If the knowledge and function templates in leading users' customised projects were seen to have a generic feature or potential market, they would be uploaded to DevsPartner.

The chapter concludes with a summary and brief discussion in section 4.6.

4.2 The General Background

4.2.1 The influence of government policy

The birth of the Chinese indigenous Linux occurred in a context in which a large potential market was constructed by the Chinese government.

The Chinese government showed great interest in Linux since a wave of enthusiasm swept across the world in 1998. The first government organised seminar on Linux was held on 19 August 1998, with participants from the Ministry of Science and Technology, the State Information Centre, CAS (Chinese Academy of Science), and some indigenous software companies. It provided a formal place for actors of government, academia and industry to express their understanding on Linux and GPL (General Public License). Six months later, another official seminar named “Linux and China’s software industry development” was organised by the Ministry of Science and Technology. Participants from industry and academia analysed the underlying negative impacts and threats of market dominance caused by foreign OS companies on China’s economy, technology, software industry and national information security. They further appealed to develop a Chinese indigenous OS based on Linux. Linux hence was regarded as an alternative to foreign proprietary OS.

In the seminar on “China’s opportunities provided by Linux development” held in April 1999, Mr. Guanhua Xu, the Vice-Minister of Ministry of Science and Technology announced that Linux-based indigenous OS was listed on the state software development special plan. The first government official document on which indigenous Linux was listed is “The Guide for Current Preferential Key Sectors of High-Tech Industrialisation” issued by The State Development Planning Commission and the Ministry of Technology and Science in July 1999.

“…..According to local and general specific situations, the governments and administrative departments at all levels shall support the High-Tech Industrialisation projects which were listed on the guide by the means of making preferential policy and investment……”

Later on, the Ministry of Information Industry restated the government’s strong supportive attitude to indigenous Linux development and deployment in two other seminars organised at the end of 1999 and 2000.

Due to the fact that the applied areas of Linux were generally divided into embedded/micro computers and conventional computers (i.e. PC and server), Chinese policy documents stated explicitly on Linux OS for both types of computers. “The Guide for Current Preferential Key Sectors of High-Tech Industrialization" listed embedded OS, in particular indigenous embedded Linux as a key sector among all 138 sectors within the guide. By acknowledging the high coupling of embedded OS and specific embedded product/equipment, the Chinese government also issued preferential policies for supporting IC (Integrated Circuit) product design and manufacture. On 24th June, 2000, "Several Incentives for the Development of the Software Industry and IC Industry" (2000, No.18 Document) was promulgated by the State Council. Apart from the clauses on supporting embedded Linux, several important issues associated with IC industry development were also identified in the No.18 Document. For instance, the actual income tax of the enterprise which is engaged in IC design and manufacture would be reduced to 3% from 17% by means of a tax rebate and refund from 2000-2010. From the first year of profitability, the newly established IC enterprise would be exempt from income tax in the first two years and additionally be allowed a 50% reduction in the 3rd to 5th years. This document also suggested that government would support R&D activity of IC enterprises through financial subsidy. Another important issue was the high priority attributed to domestic products in government and state-owned enterprise procurement.

4.2.2 Technology specificities of computer systems and their OS

In order to present the evolutional stories of Red-Flag embedded Linux and CS2C platform Linux (as described in Chapter 5) explicitly, this section is set to clarify the technical differences between embedded computer systems and conventional computer systems, which helps to understand the design and implementation of these two types of Linux OS.

Early computers were expensive and used for numerical computation as large-scale equipment since their invention in 1946. They were generally deposited in special computer rooms until the emergence of the microprocessor in the 1970s. Microcomputers with microprocessors then were rapidly applied to various fields of life due to small size, low cost and dramatic rise in processing power and functionality. Some microcomputers were embedded within programmable controllers to implement the functions of intelligent control. In contrast to traditional general-purpose computer systems, they were called embedded computer systems.

An embedded computer system is a special-purpose computer system dedicated to one or a few specific tasks. It is usually embedded as part of a device. Embedded systems span all the aspects of modern life and are widely applied to the fields of manufacturing industry, telecommunication systems, transportation systems,consumer electronics etc.

Both the hardware and software of embedded systems must be optimally designed, in order to avoid redundancy. The hardware configuration is subject to the performance of whole embedded system. Unlike the genera-purpose computer system, the high processing speed and power of the processor are not the principal factors for embedded systems. The power consumption, size, cost of an embedded processor and the requirements of specific applications must be also taken into account. An embedded system is usually run with limited computer hardware resources, for instance limited memory, small or non-existent keyboard and/or screen.

An embedded software system is generally burned into the memory chip. Even though the processing speed of processor and memory volume are continuously facilitated by semiconductor technology development, the hardware resources are still regarded highly scarce in most cases. Therefore the optimal embedded system calls for high quality source code because shorter code length of programmes leads to higher running speeds.

The technological requirements and technology development direction of embedded computer systems are distinct from conventional general-purpose computer systems. The technological requirements of general-purpose computer systems are characterised by high speed and high volume numerical computation, its technology direction is developing higher bus speed and larger storage volume. While the technological requirement of the embedded computer system is the high intellectual control capability to objectives, like embedded equipments, its technology development direction is providing higher reliability and performance.

In the literal sense, an embedded OS is an OS for an embedded computer system with dedicated and specific applications. General-purpose OS is designed for general-purpose computers to implement multi tasks and cannot match the rigorous constraints of embedded system thanks to different technological requirements. The embedded OS must be optimizing designed to seek an extremely compact size according to the specific applications and hardware framework of an embedded product/equipment. It is used for programmable microcontrollers with limited system resources so that its kernel is smaller than general-purpose OS.

In contrast to general-purpose OS, some embedded OS also have real-time performance constraint to meet. In the case of multiple tasks, embedded computer systems must guarantee the timeliness and validity of tasks in accordance with their implementing priority. They require specialized scheduling algorithms and optimal development of OS to respond to particular events.

4.3 The Pre-birth and Emergence of Red-Flag Embedded Linux (1999-2001)

4.3.1 Red-Flag Software Co. Ltd

CAS is the core unit to building the Chinese indigenous computer industry. Facing the economic and technological blockades generated by Western countries in the first 30 years of the establishment of P.R. China (1949-1979), CAS developed indigenous computer equipment and OS independently, and made significant contributions to the Chinese "Two Bombs and One Satellite[5]" project and to the national economy. Due to the fact that the

[5] During 1964-1970, China successfully exploded its first atomic bomb, hydrogen bomb and launched first satellite sequentially. 'Two Bombs and One Satellite' project is regarded as the core of cutting-edge science and technology for national defence by China's Communist Party Central Committee, which built China's own

Chinese market was almost monopolised by foreign OS products since China's reform and opening-up in 1979, the Institute of Software in CAS (ISCAS) has conducted further research on OS, as well as participated in most domestic OS related research projects founded by Chinese national and local R&D funds from the national "6th Five-Year Plan" to "9th Five-Year Plan" (1980-2000). In the 1990s, ISCAS developed lots of large-scale application computer systems based on UNIX; additionally, it achieved successes in the research on information security, parallel applications in distributed/clustering environment, and embedded electronics and communications products by applying Linux. Drawing upon the research findings of Chinese character processing, information security and so on, the ISCAS released Red-Flag Linux distribution in 1999.

Red-Flag Software Co. Ltd is a wholly independent commercial software enterprise with a strong government background. It was formally established by ISCAS and NewMargin Venture Capital in June 2000. The CCIDVS (China Centre for Information

scientific research capabilities. (http://www.cpcchina.org/images/2010-09/26/content_11348027.htm)

Industry Development Venture Capital) became the third shareholder of Red-Flag in 2001. Apart from ISCAS, the other two shareholders have an extremely strong government background. The NewMargin Venture Capital represents the interests of Shanghai Municipal Government, the National Development and Reform Commission, and the Ministry of Commerce. CCIDVC is a direct affiliate of the China Centre for Information Industry Development which is the commercial arm of the Ministry of Information Industry.

4.3.2 The background for birth of Red-Flag embedded Linux

Potential market segments

"*We believed that the embedded system would have broad market prospects in China, and Linux would be definitely suitable for an embedded system due to its technical features. What's more, the high customisable nature of Linux makes it valued more for embedded systems than for other application fields.*"

-Vice president of Red-Flag

Early knob-based analogue control systems no longer matched the demands of the Chinese industrial control field. High system complexity and intelligent equipment required more applications to be loaded and more peripherals to be connected. As the cost of microprocessors and microcontrollers fell, the 3C (Computer, Communication and Consumer Goods) product developers in the consumer market, started to design new intelligent and numerical customer terminals which had the features of smaller size, easier control and higher extendibility to various applications. An embedded system with high reliability and stability is the principle basis for above intelligent control equipment as well as consumer electronics and appliances. In contrast, the China's PC and server markets were almost dominated by a few foreign proprietary OS products; but the embedded system market was rife with dozens of OS. In this respect, there was a good market opportunity for Linux.

Technology accumulation

The accumulated technological experience in OS R&D is another main reason for Red-Flag developing embedded Linux. As the vice president of Red-Flag explained:

“*ISCAS provided a strong technology base for us. Drawing on the past research findings of open application systems development, and the experience of Chinese OS R&D, we were able to conduct embedded Linux R&D easily and rapidly.*”

Linux is a monolithic kernel, where the modules of scheduling, file system, networking, memory management, device drivers and so forth are integrated directly with the kernel and run in kernel space. The monolithic kernel architecture can improve system response time and CPU utilisation rate due to the use of minimal time and resource for generating communications between internal integrated components. However the large amount of code of monolithic kernel is not suitable for embedded systems which normally have small memories and limited hardware resources. By contrast, microkernel is a computer kernel which provides the basic mechanisms needed for implementing an OS only, such as scheduling, address space support and inter-process communication. The design of microkernel reduces the amount of code running in kernel space because the device drivers, protocol stacks, file systems and user interface are contained in user space. Many engineers believe that running application code in user space is more reliable and easier to debug so the development process is easier and the code is more portable. The microkernel consequently is characterized by smaller size and is more suitable for embedded systems than a monolithic kernel. In the mid-1990s, before the establishment of Red-Flag, ISCAS conducted a series of research on modifying original Linux kernel to the microkernel architecture, in order to make it suitable for embedded applications.

The embedded systems were initially applied in national defence and military fields in China. ISCAS developed lots of embedded systems for military electronic equipment, such as weapon control systems, radars, communication equipments, as well as aeronautics and space equipment. Some of above military embedded systems were based on Linux. Moreover, ISCAS also undertook basic research on embedded Linux for the purpose of industrial control, although none of above products eventually entered the market due to national security reasons, e.g. the national defence, the research findings and experiences are valuable for Red-Flag embedded Linux.

4.3.3 The emergence of Red-Flag embedded Linux

Along with the blooming of global Internet application at the end of the 1990s, embedded systems were not confined to industrial control application, but spread to consumer electronics and portable devices. Owing to its low cost, together with the ability to be modified easily, Linux became a major alternative to the proprietary OS in the field of embedded application.

Red-Flag made a detailed evaluation on popular embedded Linux products initially. There was a big technological gap between foreign suppliers and Red-Flag. Purchasing mature components and technologies was perceived as the most economical solution for Red-Flag due to the financial limitation and time pressure. The manager in the embedded department recalled,

"*In order to bridge the technological gap, and save the costs of R&D and time, the strategy of our embedded OS was that we would not develop the same technologies which have been available on the market*"

For reasons such as safety and usability, some embedded applications have real-time computing requirements. Linux, however, is not a real-time OS although it has an extremely rapid response time for many applications. Extending Linux to be an OS was given top priority in Red-Flag Embedded Linux development. There were two types of real-time embedded Linux solutions: one was modifying the kernel directly, the other approach was applying a third-party patch – a Loadable Kernel Module.

After a feasibility evaluation according to existing technological capabilities of both

Red-Flag and ISCAS, Red-Flag estimated that the time to modify the original Linux kernel to a real-time OS would take a minimum of 12 months. The delivery of Red-Flag Embedded Linux would lag far behind the schedule, and the company would lose many market opportunities. Red-Flag therefore, signed a six-year contract where it obtained the patents of a real-time Linux kernel and a real-time SDK (software development kit) from RedSonic, which is a real-time embedded Linux solution provider in the United States.

Early embedded systems were for industrial control or industrial automation purposes, which have a simple or no user interactive interface and device. However, the GUI (graphic use interface) and user operation content gradually became important as numerous embedded systems have been employed on consumer electronics, such as PDAs (personal digital assistant), mp3 players, and smart phones. The X-Windows system which is normally adopted by general-purpose Linux to implement GUI and user-input control functions, is unsuitable for embedded Linux on account of its large size. Red-Flag thus chose the MicroWindows as the windowing system for its embedded Linux.

The rise of the World Wide Web makes web browser an essential component for user/network terminals. Red-Flag introduced ViewML to its embedded Linux at the end of 1999, which is an open source web browser for embedded Linux.

The Chinese character input from the numeric keypad was another new technological challenge for Red-Flag. Unlike traditional QWERTY keyboards with 26 English-character keys, some mobile devices, such as most cell phones only have numeric keypads. Therefore, a Chinese input system which enables users to input Chinese characters on number keys is necessary. In 1999, Red-Flag established a technology alliance with Qcode Information Technology Ltd., which is located in Hong Kong, to acquire the Q9 input system. The Q9 input system complies with National Chinese Character Set standard GB18030-2000, National Chinese Character Component Standard GB13000-1, and National Chinese Character Input Standard GB/T-18031-2000. It provides Pinyin input, structural input and all other popular input methods in the Chinese speaking world for numeric keypads.

Purchasing strategy accelerated the product delivery time. Red-Flag divided the potential market into two segments: industrial control field and consumer electronics market. Red-Flag identified that the digital set-top-boxes and mobile terminals would be promising in the consumer electronics market. Therefore Red-Flag launched its first version of an embedded Linux solution for the digital set-top-boxes, PDAs and smart phones respectively, based on its own assumptions regarding possible users and uses. The first version of Red-Flag Embedded Linux which was delivered at the end of 1999 included a compact Linux kernel, LibC (C standard library), and various application software.

4.3.4 Early generation (2000-2001)

Entering the embedded system market was not a smooth process for Red-Flag since it lacked detailed domain knowledge of potential customers and end-users. As a special OS serving one or more dedicated tasks, embedded OS cannot be designed generically as standard packages, but must be customised with regards to specific user requirements and working environment. Small size is normally a primary concern in most applications so modifying, tailoring and testing Linux kernel to fit user needs were key steps for embedded OS development.

The first version of embedded Linux developed by Red-Flag was more like a conceptual product or internal prototype. The embedded OS is always coupled with specific applications and hardware. Lack of direct user involvement was a barrier for developing Red-Flag embedded OS in the beginning.

At first, Red-Flag sought to explore customised contracts with every embedded product and solution provider it could. This strategy was employed for two reasons. Firstly, Red-Flag could get close to the user organisations and thus understand their requirements and end-user experience. This enabled Red-Flag to compile a catalogue of users’ needs and accumulate development experience in accordance with this catalogue. Secondly, the embedded OS has a close relationship to hardware. The compilation, assembly and debug methods of embedded OS are subject to the hardware architecture. In order to enhance embedded OS’s compatibility to hardware, a supplier must acquire the specification and technology index of the hardware. However, hardware providers only allowed their customers to access this core technological information. As a result, theoretically, the more contracts Red-Flag got, the more chance it obtained the technological knowledge of the hardware. Furthermore, the desktop and server departments of Red-Flag were in the red at that time since the foreign proprietary OS had dominated the Chinese general-purpose OS market. The embedded department had to desperately generate revenue for company survival.

Fortunately, the prominent reputation of ISCAS helped Red-Flag explore market opportunities. In this stage, Red-Flag needed to develop its embedded Linux from scratch based on different application requirements. Every contract was a unique project: some embedded Linux were developed for PDA, some were tailored for digital set-top-boxes, and some for products. Diverse specific applications were associated with different hardware architecture and devices, and therefore led to a big distinction in the complexity of embedded Linux. Peripheral requirements were also varied, for instance, the user input devices ranged from QWERTY keyboard, numerical pad, remote control etc.

Unlike general-purpose OS, embedded OS cannot be delivered to users as a generic OS or be downloaded directly from the Internet. As a result, a successful reference site – existing embedded product or equipment is a pivotal factor for exploring market opportunities for embedded OS suppliers. The early contracts helped Red-Flag to build up its product image and reputation.

Two Early Projects – T&W and Shengli Projects

The first customer who signed a contract with the Red-Flag embedded department was Shenzhen T&W Electronics. T&W Electronics is a broadband terminal equipment supplier in China. T&W Electronics intended to develop a new model of digital set-top-box, which would not only have the traditional functionalities of audio and video decode, but also provide a function of Internet access.

As recorded by Red-Flag, T&W Electronics initially required Red-Flag to provide a whole product solution which included both the embedded software solution (i.e. Linux and application software) and hardware structure design. With the whole solution, T&W Electronics was able to directly transfer the concept of the broadband digital set-top-box into the product. As a software provider, Red-Flag, however, claimed that they had no expertise or experience in hardware structure design. After a series of negotiations, T&W Electronics and Red-Flag reached a consensus that Red-Flag only developed a total system solution for T&W Electronics because T&W Electronics had no capability in developing embedded software.

Unlike the applications developed for PCs or servers, embedded applications employ a unique development mode called host/target approach. In this approach, the applications which were written on a PC or workstation were to be executed on another computer. The former computer system is the host, the latter where the programme ran is called the target. If the host and target were two different types of machines, a special compile approach ‘cross-compile’ would be called for. The reality is that most programmes are developed on an X86 architecture computer, while the target is normally non-x86 architecture. In order to execute embedded software including OS and applications on target, a new compile environment has to be set up on host and a correct compiler must be selected according to the target hardware platform. Similarly, a debugger is located on the host. It can control the execution of the programme on the remote system and retrieve information of its implementation over a serial port or network. Setting up this embedded application development and running environment, as well as further completing the debugging and testing tasks, required special and professional skills.

However, Red-Flag immediately found that porting its conceptual embedded Linux to the set-top-box directly was an impossible mission. The features of the conceptual embedded Linux were rife with the supplier’s own assumptions and it only had basic functionalities such as GUI, digital audio and video decode. As new hardware structure was employed and a new function of Internet access was introduced, technically the embedded Linux had to be re-designed and re-tailored to match the specific product. Three months later, a brand new Red-Flag embedded Linux-based solution was delivered to T&W Electronics.

It is noteworthy that Red-Flag received strong technical support from ISCAS in this project, in particular the embedded applications development. For example, the EPG (electronic program guide) which is a key component of user controlling system was cultivated by the research staff of ISCAS. A microweb server was introduced into the software solution with which the end-users were able to access LAN and Internet on GUI. The design and development of EPG was organised as an academic paper and published in the "*Journal of Computer Science and Technology*" (August, 2001) by the authors from ISCAS.

Soon after, Red-Flag won another contract from Nanning Shengli Science & Technology. The situation faced by Red-Flag was the same as in the T&W Electronics project. Red-Flag was asked to provide a whole software solution for a PDA product. Because of the specific functions of business, entertainment and wireless network connection of this PDA, Red-Flag similarly was unable to apply its conceptual version of PDA embedded Linux. Additionally, this project could not draw upon the development experience of the previous T&W project since these two projects were largely distinct in hardware architecture and functionalities. Red-Flag, therefore, had to develop this embedded Linux starting from scratch.

From the Shengli PDA project, Red-Flag was aware that mobile terminals require better image quality than traditional embedded equipment in human-computer interaction. However, the existing windowing system that Red-Flag employed – MicroWindows was unable to provide GUI with high image quality. This was perceived as a limitation to the extendibility of Red-Flag embedded Linux. In order to increase developing flexibility, Red-Flag signed a cooperation agreement with Trolltech in July 2001. Trolltech is a leading cross-platform GUI system provider in Norway and its product – Qt/Embedded provided both a high quality GUI and a framework for graphic interface application development. The cooperation between Red-Flag and Trolltech was restricted not only to the technology field, but also expanded into the business area. In accordance with the agreement, firstly these two companies would develop Red-Flag embedded Linux for mobile terminals and other embedded equipment based on Qt/Embedded GUI system; secondly, Red-Flag would become the only business agent for Qt/Embedded products in China; thirdly the two companies would establish a Qt/Embedded technique centre in China and Red-Flag would provide technical support and training for Chinese users for using Qt/Embedded.

Nevertheless, the size of Qt/Embedded is too big for some embedded Linux applications (varying from 700KB to 7MB according to the specific application), in particular when the function of Chinese character display and input was loaded. Against this backdrop, it normally ran on high-end embedded products due to its high system resource consumption. To explore the low-end market, Red-Flag ported MiniGUI, which is one of the earliest and most famous OSS in China, to its embedded Linux in the same year. As a LGPL licensed OSS controlled by a Chinese community, MiniGUI supports various

Chinese character sets and popular Chinese character input methods naturally. MiniGUI is suitable for the embedded equipment which had very limited hardware resources and low requirement in image quality because the size of MiniGUI is only around 500KB after compiling.

SinoData Lottery Sales Terminal

Developing embedded Linux for SinoData[6] lottery sale terminal was a key project for Red-Flag and generated a considerable amount of income (over £800,000). This case study is mainly focused on SinoData SN-2000 and SN-3000 lottery sale terminals. This section will describe the design process of the SN-2000 project and the interactions between Red-Flag and SinoData. The details of SN-3000 project will be elaborated in section 4.4. It is also noteworthy that the feedback from SinoData and accumulated technological

[6] SinoData is a lottery solution provider located in City of Shenzhen, which currently occupies the largest share in China's lottery solution market.

experience had a profound impact on Red-Flag embedded Linux development.

Introduction to China’s welfare lottery industry

Along with the establishment of China’s Socialist Market Economy System, China also started to reform traditional central-controlled social welfare administration in the mid 1980s. The China Social Welfare Lottery Committee for Raising Funds and China National Welfare Lottery Issuance and Administration Centre were both founded in 1987. The national welfare lottery market was divided according to the provincial administrative areas. To regulate the local lottery issuance, the provincial Welfare Lottery Issuance and Administration Centre was founded in every province, autonomous region and municipality.

The bloom of China’s lottery market was facilitated by computer, IT and network technology development. In order to avoid lottery fraud, a computer managed lottery sales system, consisting of a data centre and thousands of sales terminals, was employed in 1995. Technically, there were three evolutional stages of this system: offline, semi-hotline and hotline.

Offline refers to the lack of network between the lottery sales terminals and the data centre. The sales data of every terminal was sent to the data centre by floppy disk or other external storage devices daily, at the close of business.

Semi-hotline lottery sales system indicated a system in which the sales data of each terminal can be uploaded to the data centre via a network. However, the data transmission is not real-time, and is done after business hours every day. Compared to the offline system, network replaces the movable and portable storage devices. The semi-hotline system provides more security than offline does.

The hotline lottery sales system is the most advanced system in which the terminals and data centre are connected all the time. The data for every transaction is sent to and stored at the data centre, as well as feedback to the terminal via a real-time mechanism. The security risk is efficiently covered because no data is stored on the local terminal. Employing the hotline technology, the sales period of every draw is extended as long as possible. Applying the hotline system consequently is seen as the way to significantly increase sales because according to research conducted by the China National Welfare Lottery Issuance and Administration Centre, the sales made on the day when the lottery takes place accounts for at least 50% of the total sales of every draw.

The first hotline computer lottery sales system was implemented in Lotto. Lotto is the most popular lottery product in the world, and was introduced by Shanghai Welfare Lottery Issuance and Administration Centre in 1999. The Lotto enables the gambler to select ‘n’ numbers from ‘m’ options actively. The new Lotto was accepted by Chinese lottery gamblers rapidly in 2000. Following the steps of Shanghai, an increasing number of similar centres in other provinces decided to apply the hotline sales system for issuing local Lottos.

SinoData SN-2000 V2.0 project

As a special financial tool, welfare lottery is strictly under the control of the Chinese government: its issuance and sale are exclusively controlled by China Welfare Lottery Issuance and Administration Centre which is a public organisation authorised by the Ministry of Civil Affairs. Instead of issuing to the whole country, as most European countries do, the issuance and sale of Chinese welfare lottery is organised in province-level administrative divisions. Therefore, government procurement is the only measure for purchasing computer managed lottery sales systems. Government procurement is in line with the ‘winner takes all’ principle, which means the whole lottery sales system in one area including the data centre and sales terminals, is generally provided by one firm.

Jiangshu Province Welfare Lottery Issuance and Administration Centre decided to issue a Lotto named “Jiangshu Wave” and upgraded its computer managed lottery sales system to the hotline system at the beginning of 2001. According to the bidding document, the new sales terminals must meet three basic requirements: 1. support Ethernet communication; 2. have real-time processing capability; 3. have a GUI (Graphic User Interface).

SinoData SN-2000 lottery sales terminal, which was based on DOS OS, matched none of the above basic requirements. Therefore SinoData had to develop a new sales terminal - SN-2000 V2.0, in order to enter the bidding.

Making comparisons among embedded Linux, Windows CE, embedded UNIX and other OS, SinoData chose Red-Flag embedded Linux as the OS for its SN-2000 V2.0. In contrast to the foreign proprietary OS, Red-Flag embedded Linux had the lowest price for not only the OS itself but also providing technical support for application software development and hardware design. Government policy was another reason for making this decision. According to the No.18 Document mentioned in section 4.2.1, the indigenous software products, in particular the OS, had top priority in government procurement. SinoData believed that employing an indigenous OS would help its SN-2000 V2.0 to win the bidding.

Every province-level Welfare Lottery Issuance and Administration Centre normally develops several new lottery products for every year so the embedded Linux for SN-2000 V2.0 was required to design with high extendibility for future applications. The products, even the same model of product developed for different provinces, to some extent are distinct. Take Lotto for example, Lotto in a given province required gamblers to select 5 numbers from 15 options, while Lotto in another province may require that 7 numbers are selected from 36 options. SinoData thus expected that their SN-2000 V2.0 would be a generic solution, instead designed only for “Jiangshu Wave” Lotto. In this respect, different application software is able to be loaded on the same terminal according to market demands.

Comparing T&W Electronic and Nanning Shengli projects, a uniqueness of the SN-2000 V2.0 project is that SinoData only asked Red-Flag to develop embedded Linux for its sales terminals. SinoData wanted to develop the lottery application software by using its own team because it is familiar with lottery domain knowledge and intended to apply for a software patent for its lottery application products. As the manager of the Red-Flag embedded department recalled:

“*The lottery application software is the core competence of SinoData so that the manager of it wanted to keep it as a trade secret. Therefore SinoData required us to provide an EDK (Embedded Development Kit) for its application development.*”

Under the circumstances of selling different types of lottery in a rather short period of time, real-time processing capability is necessary for sales terminals. The real-time Linux kernel and a real-time SDK (Software Development Kit) bought from RedSonic, were incorporated into the OS of SN-2000 V2.0 and a EDK (Embedded Development Kit) provided to SinoData.

Red-Flag built the EDK based on the GNU toolchain that played a vital role in the development of applications for embedded systems. In computer science terms, a toolchain is a collection of computer programmes/tools for creating programmes/software. A typical toolchain consists of a text editor for editing source code, a compiler for translating source code to machine executable programmes, and a debugger for testing and debugging the programmes and source code. The EDK supplied by Red-Flag had two modifications in contrast to the original GNU toolchain. On the one hand, in order to reduce the complexity in application development process, a simulator was developed with which the system running environment in target computers can be simulated in hosts so that all the development stages are able to complete in hosts. On the other hand, taking the programmers' habit into account, the interface and command of C++ compiler of EDK are analogous to Borland C++ compiler which is the most popular compiler, so the software engineers would be familiar with Red-Flag's EDK in a short time.

As well as providing the EDK to SinoData, Red-Flag was also required to provide service – technology support for application software development. This service was pivotal for this project since it was the first time that SinoData applied Linux. Red-Flag helped the SinoData's software engineers solve all the Linux related problems, and also offered two EKD training programmes for them.

New product strategy

In early stage, the manager in the Red-Flag embedded department understood that it is difficult to develop a generic solution for embedded systems because of their own unique technical features. As a result, Red-Flag wanted to cultivate a 'generic project development method' by drawing upon what had been learned from users and past projects. As the vice president of Red-Flag explained:

"At the beginning, we thought that there may be a lot of similarities among different embedded solutions, even though these embedded equipments are adopted in diverse fields or have distinct users and uses. For example, every solution shares the same Linux kernel and file system; and some features and functions of solutions may be simplified to a yes/no choice, i.e. having a network feature or not, having user interaction or not, having audio function or not, having video function or not, having GUI or traditional TUI (Textual User Interface) etc. Once the above features and functions were identified, all solutions normally go through the same process: tailoring Linux kernel, developing application software, testing and debugging. "

Later, in particular after the SN-2000 V2.0 project, Red-Flag found that every embedded system is a unique project. Even the tiny uniqueness of an embedded system may generate large differences during the design and development processes. Take GUI design for example, GUI system was employed in light of the hardware resource and customer requirements in graphic quality. However, the technical features of a variety of GUI systems are not the same. Although both MiniGUI and Qt/Embedded are GUI systems, they are built on distinct graphics engines. MiniGUI is based on SVGALib and LibGGI, while Qt/Embedded is built on framebuffer. Employing MiniGUI or Qt/Embedded is subject to the graphics engines supported by tailored Linux kernel because supporting different graphics engines required Linux kernel to be tailored in a distinct way. To minimise the kernel size, Red-Flag cut the redundant parts off from the original Linux kernel on 'trial and error'. It was obvious that the GUI design would create high complexity for embedded solutions development. Moreover, there are over 100 types of CPUs and chipsets used for embedded systems. Hardware architecture is another constraint to tailor, compile, and debug Linux kernel. All these steps are distinct under different hardware conditions. In theory, there are thousands of ways to tailor Linux kernel under the condition of two different GUI systems and 100 CPU and chipset architectures. If other hardware devices and applications software were taken into account, the complexity of Linux kernel development would increase greatly.

In order to provide a sophisticated embedded system, Red-Flag had to be familiar with the domain knowledge of various industries. However, learning domain knowledge from customers is time and energy consuming because embedded equipment has been applied widely in many fields and industries since the end of 1990s. Red-Flag had no means to reach the customers in every industry, and also had no financial support to conduct such a big market survey.

Maximising profit is another reason for Red-Flag changing the original market strategy of providing an embedded system. After adopting a catching-up strategy, there was no obvious technical gap between Red-Flag and foreign embedded Linux providers in 2001. The vice president of Red-Flag pointed out:

"*Having the best human resource in software engineering and research, together with purchasing and studying the most advanced embedded Linux technologies from international outstanding companies, the Red-Flag embedded Linux can been regarded as one of the most competitive products in the whole world.*"

The local customers in the fields of traditional electrical appliance manufacturing and consumer electronics production generally lacked technological capability in embedded system integration. They required Red-Flag to provide a whole embedded system instead of only an embedded OS. However, the Chinese indigenous product generally meant low price from the perspective of the Chinese customer so that the price of Red-Flag embedded system is only one-fifth of the price of foreign products. There was minimum profit in some early contracts for Red-Flag because learning the domain knowledge from customers accounted for two-thirds of project time and a large part of the cost. The costs of human resource and the time consumed on embedded system design and development are much less than those used in this learning process.

After the SN-2000 V2.0 project, Red-Flag perceived that the company strategy should be shifted in both technological level and product level. The idea of cultivating the "generic project development method" was abandoned because there is no way to implement it until Red-Flag had a large development team and accumulated sufficient experience from hundreds of projects. Learning from the SN-2000 V2.0 project, Red-Flag shifted its product strategy from providing the embedded Linux-based solution to supplying embedded Linux and EDK. In mid 2001, Red-Flag launched ControlLinux and EDK 1.0. The former is a general term used for Red-Flag embedded Linux product, which required to be customised, while the latter is based on the EDK developed for SinoData. The Red-Flag EDK 1.0 provides a friendly platform with which the users are able to develop embedded applications according to their actual requirements and knowledge. This new product strategy meant that Red-Flag was thus freed the costs of studying domain knowledge.

4.4 Early Development Stage (2002-2005)

4.4.1 The recession of Linux industry

Red-Flag and the whole global Linux industry were experiencing a downturn from 2002 to 2004. Red-Flag's embedded department was the only profitable department before 2002. The staff in the embedded department accounted for two-thirds of the staff of the whole company in 2001. However, the size of the embedded department rapidly condensed to two staff in 2002, as contracts had shrunk.

A representative contract at this stage is the Samsung MITS SCH-i519 project. The Samsung MITS SCH-i519 was a shining product in China, released in October 2003. It was the first smart phone based on Linux, supporting CDMA network and providing photographic function in China. In this project, Red-Flag did not limit the role of Linux development, but provided technology support for Samsung in applications development, and further offered training programmes for other third-party software providers who were willing to develop applications for SCH-i519. Red-Flag was eager to take part in some general embedded application software design and development (for instance the games) on account of the giant position of Samsung in the global market. As recalled by the manager in the Red-Flag embedded department:

"*Samsung is a world-class mobile phone provider so Red-Flag would definitely attract attention from other embedded equipments manufacturers through this collaboration. It was a precious opportunity to build up our product image and reputation in the embedded field. Therefore we had to try our best to participate in this project, even in embedded applications software development.*"

However, the SCH-i519 project didn't help the Red-Flag embedded department much because the progress of Chinese infrastructure construction was far behind the embedded intelligent terminals development at that time. In this stage, the T&W broadband digital set-top-box faced a vital problem, which was low connection rate. The VOD (Video on Demand) function required not only special video and audio computer servers, but also the video and audio contents (such as classical music and movies) stored on these servers. A dilemma was that the broadband service providers could not afford the cost of servers and contents unless more users bought these services; and the lack of video and audio content was the main barrier for users purchasing these services. By the same token, sales of Shengli PDA and Samsung MITS SCH-i519 were far lower than expected. The connection fee at the beginning of wireless internet service was quite high, which directly led to low sales of mobile terminals.

Simultaneously, most embedded product manufacturers were disposed to develop embedded systems themselves. Lower price was their primary motivation to deploy Linux. The vice president of Red-Flag pointed out:

"S*ource code of original Linux and tailored embedded Linux is available on the Internet freely, and source code of some embedded applications can be downloaded easily as well. Therefore they believed that developing an embedded system can be done simply by recruiting one or two software engineers to modify the available source code.*"

Furthermore, instead of cooperating with professional Linux enterprises, small-sized embedded product providers chose Linux workshops as their partners. The workshops are normally established by full-time college students. Unlike firms, these workshops can utilise the equipment and computers in the university’s laboratory, and in addition, have no cost of company operations. Therefore the prices of their products are definitely much lower than branded products.

Although facing the downturn, Red-Flag did not stop its R&D research on embedded Linux and EDK. Their research focused mainly on improving the real-time performance of Linux kernel and enhancing the technical competency of EDK. Collaborating with ISCAS, Red-Flag succeeded in applying for around 4 million RMB (equal to £400,000) funding from the government for its embedded technology R&D. The government grants helped the embedded department keep technology competency and product moving on, but more importantly provided financial support for running the department during 2002-2003.

Both the top management of Red-Flag and the embedded department believed that the embedded contracts would come back in the future. The lack of qualified engineers was an important challenge faced by Chinese embedded products providers. As the vice president of Red-Flag recalled:

“*By our estimates, the number of Chinese engineers who had capability in embedded Linux and solution design was less than 500 in 2004.*”

Owing to the high complexity in embedded Linux kernel tailoring, the embedded product providers who tried to develop embedded Linux using their own engineers found that time consumption and pecuniary costs were much higher than their assumptions. They therefore began to consider cooperating with third-party professional Linux enterprises, such as Red-Flag.

4.4.2 Financial support from government

In the Chinese central government official document of “The Guide for Current Preferential Key Sectors of High-Tech Industrialisation” and the No.18 Document, both Linux and embedded system were listed as key fields which were strongly supported by the government. In order to reinforce policies specified in No. 18 Document, the State Council promulgated “the Action Plan for the Rejuvenation of the Software Industry (2002-2005)”

(2002, No. 47 Document) to re-emphasise the priority of Linux and embedded system in Chinese software industry building.

Four government founded projects on embedded Linux research were undertaken collaboratively by ISCAS and Red-Flag. The first research project was “The Development Mode and Algorithm Research on Real-Time system” which is a key project sponsored by National Natural Science Foundation of China (NFSC) under the Grant 69896250-3. This project made comparative research on popular real-time Linux kernels, and further improved real-time performance by modifying original Linux kernel in a new way. The researchers and engineers developed a new Linux kernel which is suitable for large-scale real-time application through this project. From 2003-2004, ISCAS and Red-Flag succeeded in applying another two research projects: “Research on Hybrid Task Scheduling Mechanism in Real-Time Systems” and “An Investigation into the Optimal Task Scheduling Feasibility Analysis and Software Implementation for Real-Time Systems”. They were founded by 863 Programme (National High-Tech Research and Development Plan of China) under Grant 2001AA113201 and NFSC under Grant 60373053 respectively. These two projects focused on the algorithm of task scheduling for the circumstances when multi real-time tasks, and/or real-time and non-real-time tasks co-existed. Moreover, these two research projects also looked into the way to modify Linux kernel, in order to realise the new hybrid task scheduling mechanism. Soon after, the current vice president of Red-Flag, former Professor of Institute joined another 863 Programme sponsored research “Component-based Embedded OS and Development Environment” (Grant 2004AA1Z2050). This research largely enhanced the portability of embedded Linux, and therefore facilitated the application of embedded Linux systems in the market.

In China’s S&T research system, the research findings and patents generated from government founded projects belong to the government agencies. In order to facilitate the research findings and patents entering into the market rapidly, these government agencies therefore normally commission the organisations who undertook the project to manage the commercialisation of the research findings. As a result, the research findings of the four projects mentioned above were acquired by Red-Flag directly, and were used for enhancing the firm’s technology competency.

4.4.3 SinoData SN-3000 lottery sales terminal

Background

The lottery industry in China is characterised by high profit, which resulted in the emergence of illegal lotteries. There are two types of illegal lottery: illegal local government lottery and private lottery. The former refers to the lottery which is issued and sold by local government rather than the China Welfare Lottery Issuance and Administration Centre or its affiliated branches. The latter is the lottery issued and sold privately by an individual person or organisation. In order to regulate the lottery market, Chinese central government formulated administrative regulations and adopted measures to strengthen the control of the national lottery industry; additionally, they required every province-level Welfare Lottery Issuance and Administration Centre to develop more new lottery products in order to attract gamblers.

In 2005, local Welfare Lottery Issuance and Administration Centres started to develop a new lottery – Keno. Keno is one of the popular lotteries in the global lottery market, and is normally sold in places of public entertainment, such as bars and pubs, coffee shops etc. Keno’s draw takes place frequently (minimum every three minutes), and the prize of every draw is small. The sales terminals for Keno normally have strong multi-media functions, for example enabling the lottery gambler to watch cartoons and hear pleasant music when the draw takes place. But the multimedia function was not taken into account in designing the last version of Chinese sales terminals. Therefore, lots of local Welfare Lottery Issuance and Administration Centres decided to purchase new lottery sales terminals.

SinoData was ranked the second largest lottery solution provider in 2004. Its SN-2000 V2.0 terminal was seen as a successful product, which helped SinoData win contracts in six provinces, and over 18,000 terminals were sold. The high technology capability of Red-Flag and the good experience with the past cooperation were two key reasons that SinoData chose Red-Flag as a collaborator in its SN-3000 development.

The tensions between Red-Flag and its customers

The customers of Red-Flag embedded Linux were embedded systems integrators and/or embedded solution providers. They always required high flexibility and functionality so that the specific local applications can be loaded easily. However, embedded OS providers generally concentrated on how to optimise and tailor a compact Linux kernel because their embedded Linux often faced conditions of limited hardware resources. The manager in the embedded department of Red-Flag commented:

"*In the stage of embedded solution design, the hardware is the first step to be considered in light of budget. The second step is designing the concept of embedded applications based on the local requirements of the system. The last step is making a decision about which OS should be selected. Hence, embedded OS providers generally have little space to negotiate with their customers because the hardware and applications have already been fixed by embedded product providers.*"

One of the tensions in SN-3000 stemmed from the realisation of the cartoon playing function. Considering the existing lottery sales terminals' lack of multimedia function, playing cartoons became a principal requirement for the sales terminal procurement. The SN-3000 had to match this requirement as a basic function otherwise SinoData would lose the qualification to bid, which also meant Red-Flag would lose the contract. However, the procurement bidding document did not clearly define the term of cartoon playing, such as content, format, and display mode of cartoon.

SinoData decided to implement the cartoon playing function by employing Flash technology. Flash is a popular multimedia authoring tool with which the designer and/or developer is able to create multimedia documents, including the content of text, graphics, audio, animation, video etc. Compared to traditional animation designed using a graphic library of advance programming language (such as C++ and Java), Flash animation offers better graphic quality and playing fluency. Furthermore, the flash animation design requires lesser computer skills, hence the learning curve for a beginner is short.

The initial idea of SinoData was to integrate the operating interface/window of lottery sales and the window for cartoon playing into one screen. These two windows were supposed to be easily switched by sales clerks. In accordance with SinoData's initial idea on display requirements, Red-Flag perceived that an upgrading of Linux kernel from version 2.4 to 2.6 was necessary. Although the Linux kernel 2.6 was released at the end of 2003 and had already been applied to many platform Linux products, i.e. desktop Linux and server Linux, the kernel of Red-Flag embedded Linux was still version 2.4 because it could match all customers' requirements before the SN-3000 project. With the rapid development of IT technology and media technology, the Linux 2.4 kernel was too old to play flash media files. The Video4Linux system in 2.6 kernel was modified and upgraded so that the new kernel would support the newest media file playing. Upgrading embedded Linux kernel to the newest version was not a smooth job for Red-Flag. Its software engineers had to amend previous methods in kernel tailoring and modify the EDK as well, since there was a range of new mechanisms adopted in Linux 2.6 kernel.

After Red-Flag upgraded its Linux kernel, SinoData changed its initial idea because it conducted a market survey. It found that operating lottery sales and playing the cartoons of the lottery draw on the same screen would lead to conflict because the lottery sales had to be stopped whilst the cartoon played. The conflict was incompatible, and would damage lottery sales directly. Consequently, SinoData modified the hardware specification of the SN-3000 to have two independent screens: one for playing lottery draw animations; the other for lottery selling.

Dual-screen display was not an emerging technology; some display chip design and manufacturing firms had succeeded in developing this technology from hardware design, i.e. integrating two display units into one chip. However, implementing dual-screen display from a software perspective was a challenge for Red-Flag. In traditional PC architecture, the interactive mode of computer and operator is based on a single screen. Additionally, traditional window systems were never concerned with the requirement of dual-screen display. Another challenge was that the operator’s actions on each screen must be caught precisely otherwise operation distortion would be caused and further break the whole system down. As the manager in the embedded department of Red-Flag described:

“*On our internal project estimation meeting, we perceived that it was a technical challenge for us. There was no way to acquire existing technology from open source communities, and even to purchase solution from other software companies. Moreover, this challenge had to be conquered in at most two months otherwise the whole project for SN-3000 would lagged behind the schedule so that SN-3000 would miss the time to attend bidding.*”

In order to help SinoData win the procurement bid, Red-Flag made a decision to develop dual-screen display technology from a software perspective even though the R&D cost would significantly exceed the initial budget. Red-Flag believed the benefits generated from this technology development would be much more than the pecuniary cost.

“*In our cooperation with SinoData, the R&D cost is met by SinoData. If SN-3000 won the procurement bid, SinoData would pay us in accordance with the amount of installations. Although the actual R&D investment would exceed the cost that SinoData promised to pay Red-Flag if we developed the technology for dual-screen display, we still intended to develop this technology because we would be the first company to do so in the embedded application field. Therefore, our reputation could be greatly enhanced. We believed this technology could help us expand market opportunities because its application could be ported to other embedded products.*”

The embedded engineers of Red-Flag spent five weeks developing the dual-screen display technology. A mechanism named “device abstract intermediate layer” was inserted between the display hardware driver and graphic engine. With this mechanism, the dual-screen display function could be achieved on SN-3000.

The prototype of embedded Linux was delivered and ported to the hardware platform of SN-3000 in three months. Nevertheless, at the following testing stage, Red-Flag found that the CPU of the prototype SN-3000 was not able to accommodate the software system. The task of playing Flash cartoon took up to 72% CPU resource and the application programme for lottery selling consumed 17% so the CPU resource reserved as system redundancy for coping with system emergency was not enough. As a result, the CPU was always overloaded which caused the system to be down during the system test. The most direct and easiest solution to this problem suggested by Red-Flag was to upgrade the CPU. Understanding that this upgrading would generate additional costs for SN-3000 and lead to a higher final price which was a barrier for the procurement bid, Red-Flag had two suggestions for SinoData: either upgrading the CPU or abandoning Flash cartoons, instead developing a cartoon programme by utilising C graphic library. If SinoData refused to upgrade the CPU, the cartoon programme would enable SN-3000 to match the cartoon playing requirement in procurement documents, even though the graphic quality and flexibility of C cartoon programmes were not as good as Flash animation. Making a careful comparison between the two solutions suggested by Red-Flag, SinoData eventually substituted VIA PD 10000 CPU for the original one.

Soon after, a new function of SN-3000 was suggested by SinoData. It noticed that the screen used for animation playing was suspended most of time because the average frequency of every draw is 10-12 times per hour. SinoData hoped that the SN-3000 could play video advertisements in the spare time. This value-added function suggested a different technical requirement for Red-Flag. As pointed out by the manager of the embedded department of Red-Flag:

"*Technically, the lottery draw animation is a Flash file developed by SinoData or local China Welfare Lottery Issuance and Administration Centre. But the file format of advertisements is video file. The implementation measure and technical logic behind them are completely different.*"

Although SinoData acknowledged that VIA PD10000 CPU had poor video decoding performance, they still required Red-Flag to realise this function from a software perspective because SinoData were unwilling to upgrade their CPU again. However, Red-Flag insisted that SinoData should upgrade the CPU to VIA C3 which integrated a MPEG2 hardware decoding system. Red-Flag claimed that the hardware decoding system would provide better video quality than any software decoding system would.

"*We believed that there was a tendency in semi-conductor technology development, for the video and audio system to be integrated into CPU, in particular the low-end CPU, like the CPU for embedded systems. We are able to tailor our embedded Linux to be compatible with a software decoding system, but this Linux kernel and technical approach will be discarded in two years at most on account of rapid development of semi-conductor technology. It is unworthy to be investing in this requirement from SinoData.*"

-Manager of the embedded department, Red-Flag

Due both to Red-Flag's solid position in the Chinese embedded industry and the positive cooperation experience from SN-2000 V2.0, SinoData upgraded their CPU from VIA PD10000 to VIA C3 again, as suggested by Red-Flag.

Another problem emerged as the MPEG2 hardware video decoding function of VIA C3 had not been activated for Linux. Generally, there were two solutions: one was that VIA develop a Linux patch for its C3 which enabled the Red-Flag Linux to invoke the MPEG2 decoding system of the CPU; the other was that VIA provide technical documentation and a development kit to Red-Flag so that Red-Flag is able to activate the decoding function. In reality, however, VIA would help in adopting neither of above solutions because Red-Flag is not its customer. The requirements of Red-Flag arrived at VIA finally through SinoData, because SinoData is one of VIA's largest customers. This is also the routine way that Red-Flag connected to hardware providers in customised projects. However, VIA is located in Taiwan. There are differences between China's and Taiwan's intellectual property related laws, therefore VIA did not want to provide detailed technical product documentation to Red-Flag. As a result, Red-Flag supplied the embedded Linux developed for SN-3000 to VIA, and the tasks of verification and activation were completed by VIA alone.

Managing the tensions

It took around eight months to design and develop an embedded Linux for SN-3000. The actual investment for customised service was about 194,000 RMB (£19,000), which exceeded the original fee paid by SinoData (SinoData paid 100,000 RMB (£10,000) and the remaining was invested by Red-Flag). Four formal face-to-face meetings involving Red-Flag and SinoData were held during these eight months, and 476 emails were sent and received by the manager of the embedded department of Red-Flag.

In the SN-3000 project, Red-Flag perceived that like all the Chinese indigenous embedded system integrators and solution providers, SinoData had two problems which resulted in low efficiency and slowed down progress in embedded Linux development. As the manager of the embedded department of Red-Flag recalled:

"*Weak project management capability and technology capability are the vital weaknesses of SinoData. These two problems not only damaged the project schedule, but also led to various additional costs.* "

Weak project management capability

SinoData did not have a project management team, and therefore lacked efficient inter-organisational communication and intra-organisational coordination mechanism. The embedded department's manager of Red-Flag explained:

"*All of the software engineers in Red-Flag have at least a Master's degree from ISCAS which is the top software educational and research institute in China. Except the top management and financial department, all middle level managerial staff and department managers are promoted from software engineers. Our development progress and software project management therefore is strictly according to the principle of software engineering. From our viewpoint, the project management of SinoData was in chaos because it didn't even have a project management team. I had to always connect with the CTO of SinoData directly because I could not find the right person to discuss the detailed content of the project.*"

The software department of SinoData had initially arranged to connect with the Red-Flag embedded department. Owing to the lack of software development technology on/for Linux, all the problems arising during the testing process would be fed back to Red-Flag immediately by the SinoData development team. However, the problems associated with hardware and devices were too complex to be solved merely by the communication between Red-Flag and SinoData's software department. Take the hardware compatibility, for example, small default components such as resistance and capacitance did not weld onto the Chinese indigenous hardware and devices in order to reduce cost. Although these hardware and devices had passed strict testing in DOS, Windows and UNIX running environments, they could often not run on Linux. The lack of a small default component could lead to a loss of signal or cause problems in signal synchronisation on Linux. The best way to solve these problems was for SinoData to replace these hardware components. Nonetheless, the SinoData software department could not coordinate with the hardware department in solving this problem because they were at the same administrative level. The manager of the embedded department of Red-Flag pointed out:

"*The SinoData hardware department rebuked that the software department had no right to require them to replace the hardware. They further claimed that the hardware they purchased had no problem so the software department and Red-Flag ought to shut up and enhance the system compatibility.*"

This kind of internal conflict was reported to SinoData's CTO by Red-Flag because the CTO have sufficient authority to coordinate both hardware and software problems. In order to make communication more efficient between Red-Flag and SinoData, as well as organise the project related departments more optimally, Red-Flag strongly suggested SinoData set up a project management team for SN-3000 project.

"*The relationship between SinoData and us is co-dependence on SN-3000 project. Our contract was not a one-off contract, instead it was a strategic relationship. The profit Red-Flag generated from the embedded Linux is subject to the sales amount of the SN-3000 terminals. Thus we must help SinoData to enhance its project management capability.*"

-Manager of Red-Flag’s embedded department

Weak technology capability

“*Uncertainty generally resulted from incomplete evaluation on technical specification. However, in the stage of user requirement analysis, our customer normally wasn’t clear about what kind of embedded OS they wanted, and they couldn’t even define their embedded product clearly sometimes.* ”

-Manager of Red-Flag’s embedded department

SinoData had no staff to deal with the specific duty of user requirement analysis. From a supplier’s perspective, the user requirement document provided by the customer was quite brief. The customers generally described the hardware platform and the function of its embedded solution or product ambiguously because they did not know how to describe these requirements from a technical perspective. Therefore, Red-Flag had to provide a technical consultant service which helped its customer to refine the technical specificity of embedded product.

In the SN-3000 project, playing animation is a necessary function. FPS (frames per second) is one of the most important indexes for animation and video playing, which is the measurement of the frequency (rate) at which an imaging device produces unique consecutive images. Nonetheless, SinoData did not state the requirement on FPS for animation playing function because they did not know if FPS was suitable for SN-3000. This original function requirement proposed by Welfare Lottery Issuance and Administration Centre was forwarded to Red-Flag directly without any detailed technical description. To tailor embedded Linux for SN-3000, Red-Flag must understand the technical index of all functions of the whole system, which SinoData should be clear about. Thus, Red-Flag helped SinoData to define the FPS of animation and gave suggestions in animation design. For instance, there is a free version of Adobe Flash player for Linux. However, this free version Flash player is too old to play the Flash files developed on a newer version authoring tool. Red-Flag, therefore suggested SinoData design and develop the lottery draw animation on a lower version tool to avoid such problem.

Red-Flag insisted that SinoData must upgrade the CPU for SN-3000 twice owing to SinoData’s inaccurate evaluation on CPU processing capability. The CPU in prototype SN-3000 is not able to accommodate the task of Flash animation playing because it consumes too much CPU resource. Later, SinoData proposed a new function of video playing for SN-3000. To acquire satisfactory video playing quality, Red-Flag persuaded SinoData to upgrade the CPU again. As the manager in the embedded department of Red-Flag recalled:

“*Based on the contract, application software development and hardware platform design are not our tasks. However, SinoData lacked related technology capability so we had to provide such a consultancy service to them. There were many additional tasks generated for us when SinoData changed its ideas. Every time SinoData upgraded the CPU for SN-3000, we had to re-tailor, re-compile, and re-test the embedded Linux according to the new CPU.*”

Apart from receiving direct technical support for applications development, SinoData also sent their applications development team to Red-Flag to study Qt/Embedded programming and multi-threaded programming during the SN-3000 project.

The SinoData SN-3000 lottery sales terminal won the government procurement bids in 12 province-level administrative areas, such as Jiangshu, Shandong, Liaoning, Hainan, Tianjin and Shenzhen. Furthermore, the SN-3000 was exported to some Southeast Asian countries. According to the statistical data of China’s lottery industry, SinoData was ranked as the largest lottery solution provider in China from 2006. With the successful sales achievement, Red-Flag eventually generated over 1 million RMB (£100,000) profit on SinoData’s SN-3000 project. This project was regarded as a basis for building Red-Flag’s potent reputation in embedded field. Red-Flag became the largest indigenous embedded Linux provider soon after. Technically, cultivating embedded Linux for SN-3000 not only helped Red-Flag to boost technical accumulation and technology advancement, but also enabled SinoData to innovate new applications, and further created new value for SinoData.

4.5 Growth and Future (2006-onwards)

4.5.1 The bottlenecks

In 2006, the top management of Red-Flag recognised that there were several bottlenecks slowing down the embedded department development. The vice president of Red-Flag indicated:

"*After cooperating with SinoData in the SN-3000 project, we started to review the development history of our embedded department and its products at the beginning of 2006. Several obstacles for he embedded department future's development were identified. These technical and commercial obstacles originated for both us and the external market.* "

Technology upgrade

The first bottleneck is that the Red-Flag embedded Linux has lagged since the Linux 2.6 kernel was released at the end of 2003. The kernel tailoring and system developing experience of Red-Flag was mainly based on the previous Linux 2.4 kernel. Red-Flag had tried to adopt a new kernel to develop their embedded Linux in the SN-3000 project in order to fit MEPG2 playing requirements. However, Red-Flag's existing experience only addressed the video playing section. There were still many new mechanisms that were introduced into the 2.6 kernel. A new modular mechanism enables easy portability as every component of kernel has been defined independently so that components can be modified or deleted separately, without affecting others. Apart from the modification on the video playing mechanism mentioned previously, a new ALSA (advanced Linux sound architecture) mechanism offers better audio playing performance for the 2.6 kernel. Furthermore, new kernel provides better compatibility with hardware and devices, in particular the newly released products like storage media with larger size, 64 bit CPU, USB 2.0 transport protocol and so forth.

Since employing a new scheduling algorithm, the response time of task implementation in Linux 2.6 kernel has been remarkably reduced. In this respect, the new kernel, therefore, is treated as a real-time OS kernel to the extent that it fits into the real-time constraints of all civilian applications without any third-party patch or additional modifications on kernel. On account of this new feature, Red-Flag did not plan to extend the contract with RedSonic regarding the patent of real-time Linux kernel technology and real-time SDK, which were bought by Red-Flag in 2001, in order to remedy the real-time deficiency of previous Linux 2.4 kernel.

As a result, Red-Flag perceived that the Linux 2.6 should be introduced to its embedded Linux in order to keep its leading technology capability in China's embedded industry and save costs on buying patent as well. The new Red-Flag embedded Linux had better compatibility, real-time performance and multimedia functions. However, it was obvious that Red-Flag still had to conduct research on Linux 2.6 kernel in advance, and subsequently modified its EDK to fit the new mechanisms in Linux 2.6 kernel. It was not a smooth process. As the manager of the embedded department of Red-Flag recalled:

"*Making a decision to upgrade the kernel of our embedded Linux is very simple. But the implementation process is time and energy consuming. What I needed was the additional financial support and human resource since my department can't stop ongoing project*s."

Human resource

There is a unique phenomenon in Red-Flag, which is that every product department has two development teams. One team consists of full-time staff, while another is a temporary development team comprising of the researchers and post-graduate students of ISCAS. The embedded department has five staff, among them three are software engineers. ISCAS has a special embedded computing laboratory with several researchers and students to help Red-Flag.

However, a dilemma emerged: there was a lack of human resource to explore more market opportunities. On the one hand, more and more embedded solution providers who had attempted to develop embedded Linux by themselves or by cooperating with Linux workshops, began to actively explore cooperation possibilities with Red-Flag, as Red-Flag gradually built its reputation in China's embedded industry. Most of these solution providers' previous R&D projects were not successful because embedded OS development requires specific professional skills, but there was no professional training programme on embedded Linux technology in China. The Linux workshops, which were established by full-time college students, normally dissolved with the students' graduation. That is to say, the after sale service cannot be guaranteed. On the other hand, since the Chinese government gradually adopted more measures to support indigenous Linux from various aspects, rather than focusing mainly on Linux R&D, the direct financial support for Linux R&D was reduced significantly from 2006. Red-Flag was aware that recruiting employees to enlarge its development team could generate heavy financial burdens in the future. Moreover, the size of the temporary development team is subject to the number of research projects and students of ISCAS so it would not be enlarged significantly.

It is common sense that the more projects the embedded department undertook, the more profit they made. Nevertheless, lack of human resource was definitely a barrier limiting the embedded department in project acquisition. Red-Flag always had to make a necessary trade-off between the number of on-going customised projects and the capability of conducting projects.

Additional costs

In customised embedded projects, the customers' requirements generally were too vague for Red-Flag to understand easily. The reason is that customers did not clearly know what kind of embedded system they required. Therefore Red-Flag normally started the project by providing technical consultancy services from which the customers are able to make out their exact requirements from a technical perspective, and Red-Flag is able to ascertain the necessary information. Playing a role of consultant service provider, Red-Flag proposed suggestions on both hardware selection and application software development. In addition, these customers' requirements were not consistent. Customers always like to change their requirements according to new market surveys. This may happen before or during the design and development process, and even during the system testing stage. All these consultancy service and requirements changes resulted in additional costs for Red-Flag. The manager of the embedded department of Red-Flag claimed:

"*In most embedded projects, we are OS provider, technical consultant service provider, and project management leader. All the services are free except OS development. We have to understand the detailed requirements from our customers because the OS is unable to develop without this necessary information. The project management of the whole embedded solution should be the task of our customers. If our customers made clear their own requirements at the beginning, and had sufficient capability in managing the whole project, at least half of the time and pecuniary costs could be saved. We therefore would have time to take more projects and generate more profits.*"

Developing technology and experience accumulation

Red-Flag employed two distinct product strategies for their embedded Linux at the stage of pre-birth and emergence, and the stage of early development respectively. At the beginning, Red-Flag provided the whole embedded system solutions for its customers, which included embedded Linux and embedded applications. But Red-Flag found that every embedded solution has its specific application context so they had to study domain knowledge from customers. In order to save time and energy, Red-Flag switched to a new strategy of providing customised embedded Linux and an EDK. The EDK enabled the customers to develop embedded applications based on their requirements and domain knowledge.

Both earlier strategies of Red-Flag were characterised by customisation. Pivotal factors for customised projects are developing experience, and technological capability accumulation. The accumulated experience and technology contributed greatly towards subsequent projects. However, as Almeida and Kogut (1999) suggested, most experience and technological knowledge was difficult to codify, being tacitly held by participants in project development.

Red-Flag noticed that such experience and capabilities which stemmed from past projects cannot be cumulated and materialized efficiently because of the high mobility of software engineers. The vice president of Red-Flag pointed out:

"*Our junior software engineers are likely to move to foreign software companies after participating in two or three projects, since those companies offer more attractive salaries and better positions. As a result, the experience and knowledge accumulated from past projects would be lost as experienced engineers leave. The human capital flight became an obstacle for the development of every product department*. "

4.5.2 A solution: generic embedded Linux developing toolkit

All of the four bottlenecks mentioned previously stemmed from the earlier embedded product strategies of Red-Flag, when providing customised products was treated as the only business for the embedded department. The embedded department thus started to look at the possibility of developing a generic Linux solution. As the manager in the embedded department of Red-Flag recalled:

"*The turnover and net profit of our department sustained a high increase rate every year. The ratio of net income per employee of our department is ranked top place in the whole company. However, I realised that our turnover and profit would not jump rapidly with the limited resources, unless there is a generic version for distribution in the software market place.*"

Later, in a feasibility report of generic embedded Linux-based solution, the manager wrote down the following text:

"*From 2000-2005, our customers were embedded solution providers. They are experts and have sufficient professional domain knowledge of their own industries. They like to develop applications by themselves since the IP (Intellectual Property) issue has been paid* *more and more attention in China. However, our customers were unable to develop embedded Linux for their solutions because they lacked knowledge in Linux architecture and related Linux technology. Therefore, they needed specialised customised service from us.*"

Based on this detailed feasibility report, Red-Flag repositioned its place in the embedded market as a service provider rather than a software provider in 2005. The embedded department abstracted the Linux kernel tailoring method and experience from past projects and blended them into a generic tool kit named 'DevsPartner', with which every customer is able to tailor Linux kernel, develop embedded Linux, and develop embedded applications according to the specification of the customer's embedded solution. DevsPartner is seen as a saviour for solving the shortage of human resources in customised project development. In addition, there are many small embedded contracts with limited budget in the market. Red-Flag had to ignore these small projects in the past because they could not generate profit for Red-Flag. The emergence of DevsPartner would help Red-Flag to enter this market niche. DevsPartner has two main functions: Linux auto-generating, and the provision of IDE (Integrated Development Environment).

The function of Linux auto-generating

Embedded Linux development not only requires expertise in software technology, but also involves professional hardware knowledge. Unlike the computer hardware for PC or server, the hardware adopted in embedded equipment is generally designed for special-purpose and employed in specific areas. The embedded system developers must clearly understand all the parameters of the hardware because they are required to set up the file system and tune the parameters manually in most cases. For example, embedded product normally has special display hardware and peripherals. The developer therefore must set up the display resolution ratio and refresh rate manually by linking the specifications of the display peripheral. Any error may damage the hardware directly. In addition, the OS for general-purpose can be developed as a platform-independent system which could support nearly all mainstream CPUs. However, the types and numbers of CPU which are applied in embedded equipment are far more than the CPUs adopted in PCs and servers, so there is no mainstream CPU for the embedded product. Each type of CPU has a unique instruction set, which makes them incompatible with each other. The developer therefore is required to consider the CPU platform in embedded Linux development. Concerning the IP issue, not all hardware manufacturers supply detailed technical information in the manuals. Where technical information or the means to acquire it is lacking, the development experience is indispensable.

In contrast to the proprietary OS, one of the advantages of Linux is enabling third-party developers to tailor and modify its kernel and components. But there is no authoritative document and validation tool to explain the inter-dependability and relativity between the components of kernel so the trial and error method is the only way to start tailoring and modifying, although this method consumes much time and energy. In order to develop compact Linux kernel for the embedded system, the libraries which are unrelated to the functions would be removed from the kernel. However, this work has to be implemented gradually: the library is removed one by one, and each step must be followed by stability testing. The reason is that the tacit logic relations existed among the libraries: some libraries must be retained since they are tacitly linked to the libraries which are necessary for functions. In this situation, the developers' experience is important as well.

Understanding the technical logic and index is the basis for Linux tailoring and developing and Red-Flag extracted this knowledge from its past projects. Except for the hardware platform's specific requirements, Red-Flag found that there are similar functions between previous projects although their customers have come from disparate industries. Red-Flag believed that there is a trend whereby future embedded products would have more and more similar functions. In the SN-3000 project, the functions of dual-screen display and video advertisement playing are able to easily port to another sales terminal. In fact, Red-Flag has already ported these functions to POS (Point Of Sale) terminals in convenience stores. In the past, the embedded solution providers limited their products to basic functions for the reason of cost control. For example, the set-top-box product had the basic functions of video signal controlling and decoding, as well as interacting with end-user by remote control, in contrast with the sales terminal which only requires the function of sales data processing. Along with the development of computer hardware technology and semi-conductor technology, today's embedded CPUs have several times more process capability than the CPUs in the past, integrate more and more functions on a single chip, and provide support for more devices and peripherals. Another change resulting from technology advances is that the price of CPU and storage media has plummeted. In this respect, the embedded solution providers noticed that their products must integrate value-added functions and provide wide extendibility in order to survive in fierce market competition.

Technically, the development processes of every customised embedded project of Red-Flag are the same: analysing the whole embedded system from hardware and application software perspectives, designing OS (Linux), and developing testing programmes to test the response time and system resource. In developing DevsPartner, Red-Flag firstly re-composed the development process into several pieces; secondly, it categorised these pieces according to two conditions of hardware platform and functionality requirement; and finally, it assigned the testing programme for every possible combination.

In order to enable junior developers to complete the process of embedded Linux tailoring, developing, compiling and finally testing, Red-Flag designed a 'Linux OS auto-generating wizard' based on a graphic window interface. There are two sub-menus grouped in the main menu in accordance with the two conditions of hardware and system function. In the sub-menu of the hardware platform, the developers are able to select different options according to the CPU, mainboard chip set, display device, network device and other hardware or devices of the embedded product. While in another sub-menu of function definition, options such as animation playing, video and audio playing etc can be chosen in light of the functionality and software features of the embedded product. After receiving the auto-generated instruction from the developer, a Linux-based software solution will be generated as expected.

The function of providing IDE (Integrated Development Environment)

The Red-Flag EDK 1.0 is a set of tools used for embedded applications development. On the basis of EDK 1.0, Red-Flag developed the function of providing IDE for its DevsPartner. Since Red-Flag made a decision to stop purchasing the real-time Linux patent from RedSonic, the mechanism for real-time realisation in EDK 1.0 is modified on the grounds of new Linux 2.6 kernel. However, the purpose for developing DevsPartner is not for providing a new EDK, but to supply a tool with which the customers are able to deal with all the processes of embedded applications development easily, without or with little technical support from Red-Flag. The function of providing IDE has four new features, shown as follow:

1. Supplying a means of establishing customisable cross-compiling environment for embedded application development. The differences between various CPUs and hardware architectures have been taken into account. Thus this feature enables the users of DevsPartner to establish a reliable cross-compiling platform easily by just clicking mouse buttons under the step-by-step instructions. Developers can also import/export existed cross-compiling environments between different computer machines.

2. Providing separated wizards for various embedded applications development, such as for the development of graphic interface, hardware driver, and library. Programme templates (source code of some applications) are supplied as well, which are abstracted from past development experience accumulated in the whole embedded department. The aim of this feature is to save on cost and energy for applications development.

3. Providing a strong support environment. DevsPartner integrated a lot of compile and test libraries with high use frequency in embedded system development. These libraries normally are employed in the processes of compiling and testing. A detailed technical documentation is prepared for developers as well.

4. Supplying testing environments not only for embedded applications themselves, but also for the whole embedded system. The process of establishing a testing environment generally is done manually so it requires sophisticated skills. There are three steps: the first is to write a piece of programme named ‘boot loader’ to enable the target machine to self-boot; the second step is to set up a communication module with which the host computer is able to connect to the target the machine through Ethernet or serial line; the final step is to upload testing and debugging tools to target machine from the host. These steps usually take a couple of days to be completed. Furthermore, in the case of the embedded product which applies to automation controlling, there is no display function and screen; the developers are required to write additional programme to control the communication mechanism for the breaking testing programme and feeding the error information back to the host as a series of numbers. Therefore understanding this information requires adequate development experience.

With DevsPartner, the developers do not need to set up a testing environment manually, but simply connect the host machine and target computer by Ethernet or serial line. A testing channel will be invoked and the pre-installed testing tools will be uploaded to the target machine automatically according to the previous selections of developers. All the numerical error information will be fed back to the host, attached with corresponding text explanations.

4.5.3 A new market and product strategy

Market strategy

Red-Flag DevsPartner 1.0 was delivered to the market at the beginning of 2007. It was regarded as a leading innovation in China’s embedded industry and pioneered a new business model for Red-Flag since it is able to be sold on the software market without contradicting GPL license. As the vice president of Red-Flag indicated:

“*DevsPartner is not purely an embedded OS, instead it is an integrated platform for*

Linux-based embedded system development. We have the IP on Linux auto-generation and IDE although a Linux kernel is supplied as part of DevsPartner. Therefore we are able to generate profit from selling the license of DevsPartner. Red-Flag also provides corresponding professional training for our DevsPartner customers."

With DevsPartner, Red-Flag's embedded department started to divide its customers into two categories: leading customers and normal customers. The former are the embedded product providers who have a leading position in the market so that they normally have a stable turnover. The leading customers have both plenty of domain knowledge, and strong commercial and technological capabilities. In this respect, their leading position always pushes them to explore new market opportunities by proposing new concepts and ideas of applications. Red-Flag is required to provide customised services (technical consultant service, customised embedded Linux and Linux technical support) for its leading customers because their projects are generally both large-scale and novel. From the perspective of Red-Flag, establishing long-term cooperative relations with leading customers is necessary. On the one hand, providing customised service is the best way to build profit; on the other hand, developing customised projects for leading customers is an efficient conduit to gaining knowledge and accumulating technology. The normal customers refer to the small and medium sized embedded product providers. Their projects are generally small-scale with limited budget. The strategy of Red-Flag is that it helps normal customers to build sufficient technological capabilities to utilise DevsPartner, by offering training programmes.

Product strategy of DevsPartner

Red-Flag adopted a special product strategy for DevsPartner. The manager in the embedded department of Red-Flag explained:

"*In order to comply with market strategy, our product strategy for DevsPartner is that we only support the mainstream hardware and functions we selected and summarised from past projects.*"

There are thousands of hardware and devices which available for embedded equipment and products. For Red-Flag, it is too costly to support all the hardware and devices, so the supporting list of DevsPartner only covers the mainstream hardware which must be easily purchased in the domestic market. On the other hand, nearly all the customers have their particular requirements for the functionality. It is also impossible to fulfil all these local requirements through DevsPartner. Furthermore, in order to keep the long-term cooperation with leading customers, Red-Flag believed that the hardware and function supporting list should not be actively expanded otherwise some leading customers would instead develop embedded system using DevsPartner.

In case hardware and function requirements of normal customers are not in the supporting list of DevsPartner, Red-Flag would persuade them to change their original designs to comply with what DevsPartner provided. As the manager in the embedded department of Red-Flag pointed out:

"*For example, some small- and medium-sized embedded product providers wanted to apply some vintage hardware components in their products due to financial constraint. However, compatibility with old hardware is a significant problem to Linux because the downward compatible issue is ignored by the international Linux community which controlled the development direction of Linux kernel. Sometimes, it is even impossible to help our customers to solve this kind of problem by modified Linux kernel because the necessary technical information of the vintage hardware is no longer available. Therefore, we will advise them to adopt the hardware which DevsPartner supported.*"

However, this strategy does not mean that Red-Flag would not enhance the functionality of DevsPartner. On the contrary, Red-Flag has high regard for cooperation with leading customers. Leading customers are always willing to develop new industrial applications or deploy new designs on hardware platforms. Due to their leading position in the industry, the normal customers will generally follow their steps. Once a hardware or function is perceived as a promising design and can be accepted as a mainstream industrial application, it would be integrated into the DevsPartner.

4.6 Summary

In this chapter, we presented the historical development of Red-Flag embedded Linux which was divided into three stages: pre-birth and birth stage, early development stage, and growth and future stage. Red-Flag adopted a different strategy for each stage.

In first stage, Red-Flag focused on customised embedded Linux-based solutions provision. This strategy was adopted because their clients lacked sufficient technological capabilities to develop embedded systems. The benefits of offering embedded solutions were significant. On the one hand, Red-Flag could acquire detailed domain knowledge from its clients, which also helped it to accumulate developing experience for further innovation in embedded Linux. And on the other hand, early customised projects were important for establishing an industrial reputation.

Simultaneously, Red-Fag had intentions to build a 'generic' way in order to develop embedded Linux-based solution by drawing upon the past development experience. However, such a generic way of development proven to be infeasible due to the diversity of user requirements, particular applications and various hardware components which are closely associated with Linux developing.

In the second stage, Red-Flag switched its strategy from developing embedded solutions to providing embedded Linux and EDK. The process of collecting and learning domain knowledge of customers was regarded as too costly for Red-Flag because it had been time and energy consuming. With more successful projects, Red-Flag came to be in a more powerful position to negotiate with the clients from their own technical perspectives. Once the particular user requirements on functionalities or hardware framework were regarded that they could be applied across the boundaries between embedded products or application areas, Red-Flag would develop them.

In the last stage, Red-Flag developed a generic embedded system development suite – DevsPartner, with which the users are able to tailor its Linux and applications. The customers of Red-Flag, therefore, were divided into two groups: normal customers and leading customers. The former are the target users of DevsPartner. The latter play a leading role in its industry and generally have capabilities in conducting innovative R&D in new application and functionality. Red-Flag hoped to establish close links to leading customers and provide customised projects so that the developing experience based on selective new functionalities could become new elements for DevsPartner further innovation.

The government actions were crucial during the first and second stages. Firstly, the central government encouraged ISCAS to establish a commercial entity－Red-Flag. Secondly, the government financed plenty of research projects for embedded Linux R&D. Thirdly, the government explored a large potential market for embedded Linux by promoting Chinese IC industry development.

The support from academic entity－ISCAS was another critical factor for Red-Flag embedded Linux development and innovation. The institute not only allowed Red-Flag to utilise research findings and results from previous research projects, but also provided human resource to support Red-flag conducting customised projects.

Chapter 5 The Qualitative Historical Development of CS2C Linux Distributions

5.1 Introduction

Designing and developing an OS for a general-purpose computer system is largely different with embedded OS. The general-purpose OS is supposed to be applied in various fields with complementary application software. In this respect, the platform Linux was treated as an infrastructural software for the informatisation of the whole Chinese society by Chinese policy-makers. In contrast to embedded Linux, the adaptation process of platform Linux would face more challenges in practice, such as the compatibility with third party complementary application software, end-user lock-in effect etc.

Chapter 5 seeks to explore the local adaptation process of foreign transferred general-purpose OS – Chinese platform Linux supplied by CS2C, and investigate the historical events that happened in the evolutional process. This chapter demonstrates how CS2C managed to deal with local specific requirement and built generic Linux solutions for various user groups.

Guided by BoA concepts, this chapter is organised in accordance with the biographical history of CS2C platform Linux. Section 5.2 introduces the general background of the platform Linux supplier and the emergence of CS2C (CS&S) platform Linux. In the birth stage, CS&S developed their early platform Linux products based on a foreign transferred Linux – Red Hat Linux 7.2. The development strategy is focusing on the most salient local user requirement – enhancing the Chinese language performance, in order to eliminate the language barrier for Chinese users.

Section 5.3 presents the development story of CS2C platform Linux in the next biographical stage – early development stage. In this stage, government users became the first user group for CS2C platform Linux. By obtaining feedback from users, CS&S

categorised users into two groups: desktop users and server users. The development strategy adopted by CS&S accommodates particular local user requirements, for instance, developing desktop Linux according to some features of MS-Window, as well as developing DWDAS solution with which government server users are able to construct websites for e-government projects.

Section 5.4 describes the development process of CS2C platform Linux in growth and future stage. As more user groups participated in platform Linux design, the supplier designed generic platform Linux solutions for different user groups with particular requirements respectively. The development strategy employed by CS2C is ‘Standard and Variants’ in this stage. This strategy is applied to both desktop and server Linux: supplier developed standard versions of both types of Linux as generic OS for all the personal and organisational users in the mass market. Furthermore, CS2C also developed some variant version solutions which included both Linux and application modules for specific user groups, in order to deal with the diversity of the user groups’ work and use practices.

The chapter concludes with a summary and brief discussion in section 5.5.

5.2 The Pre-birth and Emergence of CS2C Linux (1999-2001)

5.2.1 The general background

The history of CS&S in OS development

As the earliest Chinese enterprise engaging in OS development, CS&S (China National Computer Software& Technology Service Corporation) has been successful in developing China’s first generation of self-controlled secure OS, named the COSIX 1.X series during China’s “8th Five-Year Plan” (1990-1995).

Being aware that Brazil has developed a self-owned OS in 1989, Mr.Peiyan Zeng, the vice minister of Ministry of the Mechanical and Electronic Industry and the director of State

Planning Commission, organised several official seminars regarding the necessity and possibility of developing an independent intellectual property OS. The participants reached a consensus that China should develop an indigenous OS due to the requirements of national information security and also for computer industry development. Therefore, the Ministry of Mechanical and Electronic Industry launched a UNIX based OS development project financed by Electronic Industry Development Fund. China Computer Technical Service Corporation and China Software Technique Corporation (the two corporations who merged as CS&S in 1990), the key holding enterprises of Ministry of Mechanical and Electronic Industry, undertook COSIX OS development projects. Two years later, the project was listed as a key project in the "8th Five-Year Plan".

The State Planning Commission, which is the administrative agency for compiling and auditing the State Five-Year Plan, adopted a flexible strategy on the intellectual property of software during the development stage. In order to achieve independent intellectual property, the initial development strategy of COSIX complied with international standards, but developed an OS separately with existing counterparts. Under such a strategy, although CS&S had purchased the source code of UNIX System 5 Release 4 from AT&T, the developers and engineers were forbidden to access the source code during the design and coding process. COSIX 1.0 was delivered in 1993 and passed the government evaluation organised by the Ministry of Electronic Industry (split off from Ministry of Mechanical and Electronic Industry in 1993) and CAS (Chinese Academy of Science). However, repetitive development of matured technology is inevitable because the issue of independent intellectual property was heavily stressed. Furthermore, COSIX 1.0 had serious compatibility problems because it had been developed within an isolated environment.

In the second stage, the State Planning Commission switched COSIX development strategy from the whole independent intellectual property to focusing on local application, such as the application of Chinese character, security, hardware adaptability, and so forth. A forking of COSIX products emerged in 1995: COSIX 1.1 was the successor of COSIX 1.0 that focused mainly on Chinese applications, such as a simplified Chinese character set standard, Chinese character card and corresponding Chinese input methods. COSIX 2.0 was a completely new OS development based on freeBSD. Although COSIX 1.1 and COSIX 2.0 were considered as great successes in the technical realm, hardware adaptability still had fatal limitations with the rapidly developing computer hardware industry. As a result, the expectation of commercialising COSIX 1.1 and COSIX 2.0 was not realised.

The commercialisation of COSIX series OS was thus given top priority in the following "9th Five-Year Plan" which was started in 1996. The State Planning Commission granted a two-year key project named "indigenous open system software platform" to continue supporting COSIX development. The following COSIX 1.3 and COSIX 2.1 made significant progress on Chinese GUI and hardware adaptability. CS&S began to explore pushing their COSIX products onto the domestic market, which was also the first market promotion behaviour for indigenous OS in China. Software compatibility, however, arose as a new obstacle for the application of COSIX. Without sufficient users, the application software providers were reluctant to provide technical support to COSIX. As the former vice president of CS&S and current president of CS2C reflected:

"COSIX series products were designed and developed for server applications so that we had tried to seek cooperation with Oracle, one of the largest database software providers in the world. But Oracle expressed that they had no interest in COSIX due to its limited market potential. "

Finally, COSIX OS were applied by only a few industry applications, for example, in the telecommunication sector, iron and steel industry, and military decision systems, as these users were giants in China's domestic market and had strong enough commercial or technical capability to solve software compatibility and hardware adaptability issues.

COSIX business of CS&S in transition

CS&S was established in 1990 by the merging of China Computer Technical Service Corporation, with China Software Technique Corporation. Being a wholly state-owned high-tech software company, CS&S operated under the direct leadership of the State

Ministry of Mechanical and Electronic Industry and later, in 1993, the Ministry of Electronic Industry.

On 14 November 1993, the third plenary session of the 14th CPC (Communist Party of China) Central Committee issued "Decision of the Central Committee of the Communist Party of China on Some Issues Concerning the Establishment of the Socialist Market Economy". The transformation of state-owned enterprises' operating mechanisms and the establishment of modern enterprise systems were identified as important conditions for building a socialist market economic structure. CS&S was listed as a pioneer enterprise for this reform. Seventeen business divisions were gradually restructured as Limited Liability Companies until 1997. With the restructuring of Chinese central government and the dismissal of Ministry of Electronic Industry, CS&S became a holding enterprise of China Electronics Corporation (CEC) which was constructed by several enterprises affiliated with Ministry of Electronic Industry in 1989.

In 1998, the "9th Five-Year Plan" discontinued support and funding of COSIX series OS development. Prior to this, all the OS related projects undertaken by the division of foundation software in CS&S were initiated and financially supported by government agencies. These projects excluded commercialisation and marketing, dealing only with the technical issues. In this respect, CS&S was more like a business research institute than a commercial organisation. With the reform of state-owned enterprises, CS&S became a wholly independently operated commercial software enterprise although it was still state-owned. The market issue was therefore soon put on the agenda. The former general manager of the division of foundation software in CS&S and current president of CS2C recalled:

"Indeed, the development of an OS is a technical challenge, but the competition from the foreign software giants in the market was much more difficult for us since we had less experience in OS commercialisation because our past efforts were not successful."

From lessons learned from the commercialisation of the COSIX OS in the "9th Five-Year Plan", the division of foundation software realised that support from hardware and complementary application software providers was essential for an OS. However, China’s computer and software industries were in the start-stage at that time so both the hardware and software markets were mainly dominated by foreign companies before 2000. To explore market opportunity, CS&S decided to adopt an open attitude to foreign counterparts. A new policy in indigenous OS development was importing foreign technologies and then building own self-dependent products based on them. The new COSIX OS would be divided into COSIX 64 and COSIX Linux, developed for high-end industrial users and for SMEs and personal users, respectively.

CS&S signed a cooperative contract with Compaq. According to the contract, Compaq would open the Kernel and key technologies of their 64-bit OS - Tru64 UNIX to CS&S. Based on support from Compaq, CS&S was able to develop COSIX 64 according to Chinese domestic hardware and software source, as well as the requirements of local users. The features of COSIX 64 were inherited from Tru64 UNIX, which meant that over 5600 complementary software applications for Tru64 UNIX could be ported directly to COSIX 64. As an OS developed for industrial application, COSIX 64 is designed for the Alpha computer, which is good for high performance and scientific computation. Along with the performance of X86 architecture computers improving dramatically, the share of Alpha computers in the high-end computer market has been completely captured by X86 computers in recent years. Although COSIX 64 was not a successful OS, CS&S gained fresh knowledge in user requirement analysis, project management, quality management, and market promotion from this cooperation with Compaq.

5.2.2 The emergence of COSIX Linux

The birth of COSIX Linux OS

As the largest state-owned software enterprise, CS&S had solid relations with several agencies of both Chinese central and local government. These government agencies are administrative departments of CS&S, from when the company reformed, financial sponsors for research projects, and in particular the customers of CS&S. CS&S attended all the seminars mentioned in section 1.1 of this chapter as the representative of China’s software industry. It acutely grasped that indigenous Linux would be supported strongly by the Chinese government. Therefore, if CS&S decided to develop indigenous Linux, it would easily obtain a variety of support due to its state ownership and past cooperation experience with government agencies. Moreover, government policies would generate a huge potential market, in particular the government procurement market for Chinese Linux. The director of division of foundation software in CS&S and current president of CS2C indicated:

“Unlike all the OS developed by us before, Linux is a “real” general-purpose OS because it is able to be ported to most of the mainstream hardware architectures, but not for specific computers architectures, like COSIX 64 for Alpha computer only. The international Linux communities moreover, would provide endless technical supports for us.”

CS&S released COSIX Linux 1.0 on the basis of Red Hat Linux 7.2 on 28 September 1999. It can be seen as the Chinese version of Red Hat Linux 7.2 for the reason that the only differences were in the application of Chinese characters. Staff of the division of foundation software in CS&S and current product manager of CS2C recalled:

“We wrapped a Chinese application layer to the Red Hat Linux 7.2, which included a simplified Chinese user interface, Chinese input method and corresponding font library. The development of COSIX Linux 1.0, in fact is a rebranding process of Red Hat Linux 7.2.”

The Linux kernel and components were developed by western programmers and hackers so that they didn’t originally support Chinese display and applications. The foreign Linux distribution providers did not have the capability to develop Chinese Linux as well. Accordingly, Chinese application is a natural market niche for domestic Linux suppliers. CS&S bought the Intelligent ABC Chinese input method from Beijing Yangning Chinese Information Processing Ltd, which is a very powerful and popular input method in China. It has been integrated within MS Windows, IBM AIX, and Apple Mac OS as a basic Chinese input method so that most Chinese computer users are familiar with Intelligent ABC. In the meantime, CS&S signed a contract with Founder Electronics to obtain authorisation for employing its Chinese font within COSIX Linux 1.0. Founder Electronics is the top Chinese font libraries provider in China and its products have been picked by mainstream Chinese word processing software, including the Microsoft Office Productivity Suite.

In line with the initial product strategy of COSIX Linux, 'to match the requirements of all kinds of users', CS&S designed and developed COSIX Linux as a huge omnipotent aggregation of various functions so that it could be utilised by encompassing a variety of industrial and PC users. CS&S believed that Linux on the one hand provided most convenience to users: for example, a user option for a computer which is connected to the Internet to switch its role from server to PC easily without the need to re-install a desktop OS, and vice versa. On the other hand, only one R&D team and customer service team were required, reducing costs.

The development strategy of COSIX Linux

Chinese application was the essential task for all domestic Linux providers. A fierce debate soon arose regarding two technical measures for realising the Chinese application. One of the technical measures is called plug-in ‘Chinesisation[7]’. The idea behind it is to construct a Chinese platform through a kernel-independent plug-in programme with which users can read and input Chinese characters. The other measure is kernel Chinesisation, which modified the Linux kernel directly and loaded Chinese font library to the kernel. The former is considered an orthodox way to realise Chinese application since several Chinese plug-in platforms have been popular solutions for MS-Windows, UNIX and Solaris before Linux introduction to China in 1998. The latter is a novel approach to achieve Chinese application with open Linux kernel.

[7] This word is created and accepted by the Chinese software industry, which refers to the way to implement Chinese application in foreign software.

CS&S adopted a traditional plug-in Chinesisation mechanism in their COSIX Linux 1.0 because they thought that modifying the kernel may generate a negative effect on system stability. However, unlike those OS with matured GUI (like MS-Windows), lots of Linux operations must be done on console interface (command line interface). A problem for COSIX Linux 1.0 is that the performance of a Chinese plug-in programme on console interface was not good as it was on GUI. As a result, CS&S adopted kernel Chinesisation measures to their COSIX Linux 2.0 which was introduced to the market in June 2000. A so-called Unicon mechanism was integrated to the 2.0 version, which includes a kernel patch used for loading a Chinese font module to kernel. With Unicon, the kernel could support double-byte character sets so that the East Asian characters, such as Chinese, Japanese, and Korean, can be displayed and inputted on the console interface.

Owing to historical and regional reasons, Chinese has more than one coding standard. These standards are normally incompatible with each other. For example, Traditional Chinese character (BIG5 character set) cannot be displayed correctly on a Simplified Chinese (GBK character set) OS. An improvement of COSIX Linux 2.0 provided support to both GBK and BIG5 by a character code converter.

COSIX Linux 3.0 was delivered in 2001. Apart from integrating the new 2.4 Linux kernel, CS&S continued to enhance the performance of Chinese application. COSIX Linux 3.0 provided Chinese descriptions for all system tools (commands) on console interface, which enabled users to understand the way to manipulate the system and the parameters to configure the system.

5.3 Early Development Stage (2002-2005)

Animated with an enthusiastic ‘open’ spirit, like other Chinese Linux providers, CS&S distributed their install disk of COSIX Linux for free. As the former manager of the division of foundation software in CS&S and current president of CS2C explained:

“At that time, all the Chinese Linux providers seemed never to have taken revenue and profit into account. We drew upon all the public channels to distribute our Linux product

without any charge. For instance, we participated at most of Linux and open source events in China, which were extremely popular from 1999 to 2000. The more of our Linux install disk we sent out at Linux events, the more successful we felt. It seems that the Linux business in China should not be evaluated by profit at that crazy time."

However, after the collapse of NASDAQ and the Internet bubble in 2000, most Chinese Linux suppliers, including Blue-Point which was listed on NASDAQ, announced quitting the Linux related business. A reality recognised by the surviving Chinese Linux companies was that Linux is a business for them and they must generate profit from developing Linux to sustain company operations.

Even though CS&S greatly enhanced the Chinese performance for their COSIX Linux distributions, these three early versions of COSIX Linux can still be seen as Chinese versions of Red Hat Linux 7.2. CS&S did understand that getting close to end-users, studying their requirements, and improving their products based on the knowledge from end-users is the most important conduit to further developing their COSIX Linux. But there was no real end-user in China because Chinese users had been locked in foreign OS. The former manager of division of foundation software in CS&S and current president of CS2C recalled:

"From a perspective of marketing, CS&S spent most of time distributing Linux install disk but did not track the users who were willing to sample Linux. We realised very late that the some of the distributed Linux disks were never installed, and the uninstallation rate of Linux for the rest of the disks was extremely high, so we lost the opportunity to cultivate the user market."

Studying and imitating international Linux distributions was thus regarded as an efficient way for COSIX Linux development. Although COSIX Linux distributions were developed based on Red Hat Linux, CS&S still kept downloading and analysing other Linux distributions from the Internet, because every Linux distribution has its own features and development strategies from the perspectives of both technical and application issues, e.g. some stressed security features, some focused on usability, and some emphasised server performance. A former staff member in division of foundation software in CS&S and current product manager of CS2C indicated:

"The development strategy of Linux kernel and components was controlled and decided by international communities. Additionally, the foreign Linux providers participated actively in the communities, and their products had more end-users than ours. In our opinion, international mainstream Linux distribution represented the newest and hottest technical and application issues. We downloaded the newest versions Linux, read the handbooks, and analysed newly-added features. In order to understand the user experience, our developers furthermore acted as users by installing and operating those Linux distributions."

Understanding that domestic users and foreign users were distinct in their user habits, levels of computer skills and attitude to open source, CS&S introduced COSIX Linux firstly to their staff, and then to university students through campus events. However, these users were either IT specialists or students in engineering subjects, who were greatly different from end-users in the mass market with less IT skills. Therefore, their feedback was not what CS&S desired. COSIX Linux 1.0 - 3.0 can be seen as experimental products from today's viewpoint because the gulf between local end-user and COSIX Linux are still distant. This situation did not change until the Sailing Project which was initiated and founded by the Beijing government. It was the first time CS&S got close to user organisations and systematically studied the requirements of end-users in the mass market.

5.3.1 The Sailing Project

Beijing government procurement on indigenous Linux

On 24 June 2000, "Several Incentives for the Development of the Software Industry and IC Industry" (the No.18 Document) was promulgated by the State Council. Two important issues identified in the No. 18 Document related to Chinese indigenous Linux. One was the document reaffirmed that state funds for science and technology would support the R&D activity of key software technology. The indigenous OS, such as Chinese Linux, was included on the list. The government procurement measure was another issue. Indigenous software should have priority in government procurement whilst they had the same price/performance ratio to foreign counterparts.

A dilemma was faced by indigenous Linux at that time: on the one hand the supplier could not improve the Linux product without users and their feedback; on the other hand, the current indigenous Linux would not have real users without improvement. The government procurement policy was seen as the only remedy to generate a user market at that time. As the Deputy Head of Beijing Science & Technology Commission pointed out:

"Government procurement policy shows the government's determination to support indigenous Linux for both Chinese software enterprises and users."

The People's Government of the Beijing Municipality was the first Chinese government agency to add indigenous Linux to their procurement list at the end of 2001. Initially, the indigenous software encompassing Linux and office suite only accounted for a small part of their procurement plan: most of the software products for procurement were from Microsoft.

In the first round of negotiation, Microsoft expressed clearly that they wanted to sell MS-Windows XP OS bound with MS-Office suite, otherwise the single price of the OS would be quite high. In order to prevent piracy, MS-Windows XP had to be activated on the Internet and Microsoft had a set of mechanisms to monitor any computer that installed Windows XP. According to the security feature of the government intranet, the Beijing government proposed that they wanted to buy MS-Windows 2000 instead of the newest introduced MS-Windows XP OS because they believed that online activation might generate security risks. In the second round of negotiation, Microsoft denied the proposal of the Beijing government. Microsoft only agreed to sell MS-Windows XP to the customer, as well as the online activation measure, and price was not negotiable. As the Deputy Head of Beijing Science & Technology Commission recalled:

"The lower price is one of the reasons that we bought indigenous Linux. Linux has no licence fee so its price is much lower than proprietary OS. Another reason that we decided to

abandon Microsoft's products was their arrogance, which was derived from their monopoly in China's software market. "

Finally, all Microsoft's products were excluded from this procurement and all software purchased by the Beijing government was indigenous Linux and office suite: 2,801 copies of Linux were bought on 28 December 2001, among them were 1,401 copies of COSIX Linux.

The Sailing Project

The users in the Beijing government complained that there were big gaps between indigenous Linux and MS-Windows in usability and functionality. As the director of Beijing Software Industry Productivity Centre recalled:

"The users in Beijing government are not PC specialists or Linux enthusiasts; they want an easy-to-use OS like MS-Windows."

Therefore, Beijing Software Industry Productivity Centre, which is an affiliation of Beijing Science & Technology Commission, approved and launched the Sailing Project on 18 January 2001. The goals of this project were to solve the usability problem for Linux products purchased by the Beijing government, and further promote the adaptation of Linux in office applications. This project also had strong support from Ministry of Information Industry, Ministry of Science and Technology, and the State Council Information Office.

Over half of the purchased Linux was COSIX Linux, so CS&S was one of the most important participants among 18 organisations within the Sailing Project. The two main tasks of CS&S were simplifying the Linux installation process and improving its usability, as well as solving the document printing problem.

Simplifying Linux installation and improving usability

The installation of MS-Windows series products is just a matter of clicking a mouse for users. The system installation is completed in a graphic interface by mouse control. The hardware installation and configuration is set aside until after the OS installation is completed. To the user familiar with the MS-Windows installation process, the installation of COSIX Linux is much more complex and requires users to master sophisticated IT skills. The installation of COSIX Linux was carried out in command-line interface, and hardware installation and configuration had to be done during the installation process. Therefore the user was asked to input commands by typing the keyboard to perform specific installation tasks and hardware installation. A former staff member in the division of foundation software in CS&S and current product manager of CS2C pointed out:

"Like its predecessor – Red Hat 7.2 Linux, the COSIX Linux has 14 steps in installation and each step requires users to type commands from the keyboard. Furthermore, some operations for installation are very strange to Chinese users who are familiar with MS-Windows. Take hard disk partition task for example, Chinese users normally want to format the specific partition where Linux is installed, and keep the data in another partitions. However, the default for Linux formatting function is to format all the hard disk partitions."

CS&S finally realised that most Chinese users abandoned Linux before the installation was completed. It therefore made great efforts to simplify the Linux installation. The new system installation task for COSIX Linux employed a visual graphic interface, and every installation step had detailed Chinese descriptions and instructions. The main tasks to users during system installation were hard disk partition and network configuration, the rest of the tasks could be done automatically with default setting. The hardware installation and configuration was removed from the system installation process. In this respect, the COSIX Linux installation was seen to be as easy as MS-Windows.

MS-Windows series OS products are compatible with nearly all the popular hardware devices. They are able to identify the model of specific hardware automatically, as well as provide instruction to users so that they are able to install the correct drivers. The hardware installation and subsequent configuration tasks requires the least, or even no, IT expertise. The user just has to select the "NEXT" button by clicking the mouse. In contrast, installing hardware in Linux environment is extremely difficult. Due to the lack of a hardware identification tool, the user has to clearly understand the type and specification of the hardware before installing the drivers. The user is then required to edit text files and type commands in a console interface in order to complete the driver installation. To implement and optimise the hardware performance, the subsequent task was hardware configuration. The configuration tools in traditional Linux are not collected as a tool-kit and can also only be implemented on a command-line interface. To a user in the mass market with less IT skills, configuring COSIX Linux is very hard.

CS&S developed a control panel, which is a big tool-kit integrating hardware detector and system configuration tools based on a graphical interface. Take internet configuration for example, with a graphical network management tool, not only is the user able to obtain the general information for driver installation, but also the traditional operation of typing English configuration commands is transferred to a GUI based operation so that it can be completed by clicking the mouse, as in MS-Windows. The Linux user thus can experience the convenience which was only provided by MS-Windows in the past.

Being aware that the main tasks of users in Beijing government were office affaires, and knowing that users were hoping that the COSIX Linux would be improved and modified according to MS-Windows system operation style, CS&S improved the file explorer components. The new graphic interface based file explorer recognised and loaded Windows partitions automatically, provided file and print sharing with computers installed with MS-Windows in the network neighbour, and provided a “My File” folder to store files etc.

Document printing

MS-Windows series OS products integrated a powerful printer management system. The printing function is assimilated into a GDI (Graphic Device Interface) module so that the information regarding the printer can be obtained by invoking the GDI module.

Linux is a UNIX-like OS so its printing system was originated from UNIX. However, a standard printing API was lacking from UNIX systems, since different UNIX had unique printing systems for historical reasons. Each UNIX system therefore required specific printer drivers. This is why lots of printer providers stopped developing drivers for UNIX systems. Similarly, the UNIX's deficiencies regarding a lack of standard printing API and printer drivers were inherited to Linux.

Text file printing, in particular public document printing, was one of the most important requirements for government users. The official document for either publishing or internal exchange purposes has a rigid format. In contrast to MS-Windows, the document format controlling function of Linux is weaker; hence the format and layout of printed document are different to what the user sees on screen. What's more, the printer configuration task is too intricate to be completed by unsophisticated users. In order to solve this inbuilt problem, "improving printing management and configuration" was listed as a key sub-project of the Sailing Project.

The difficulty of printer configuration was greatly reduced by introducing a hardware detector, and substituting a graphical based printer configuration tool for the previous command-line tool. To solve the problem of the printing effect, the printer driver is regarded as a pivotal. However, Chinese printer providers, such as Legend (LENOVO) and Founder, did not develop drivers for Linux. As the current product manager of CS2C and former staff member in CS&S explained:

"A printer driver for Linux was the key for solving the printing problem. We believed that if the printer providers develop drivers for COSIX Linux, the next steps would be simple. But the situation was not what we would have ever thought……..."

CS&S reached the Chinese printer providers through the Beijing government by seeking their cooperation and support. At that time, nearly all of the Chinese branded printers were OEM products based on foreign products. The drivers were developed by foreign companies so that drivers obtained by Chinese printer providers were in binary code instead of source code. Another fatal problem was that the Chinese OEM printers and corresponding foreign printers were different in appearance due to the sale strategy of foreign companies. As a result, the Chinese suppliers did not know the type of corresponding foreign printers for their OEM products, and furthermore could not compile drivers for COSIX Linux due to there being no source code. For the reasons mentioned, the

Chinese printer suppliers could not offer any support or help to CS&S.

CS&S decided to adopt a ‘walking-around’ strategy which meant achieving their goal by using an indirect route. CS&S modified the Linux printing system mechanism, which enabled users to invoke the printers shared by network neighbours who used computers installed with an MS-Windows OS.

Sailing Desktop Linux and COSIX Linux 3.1 Desktop

Based on all the improvements from CS&S, Red Flag, School of Computer Science of BUAA (Beijing University of Aeronautics and Astronautics), and so forth, a new OS called Sailing Desktop Linux was delivered to the Beijing government on 15 June 2002. As the director of Beijing Software Industry Productivity Centre recalled:

“Compared to COSIX Linux 3.0 we bought before, the new Sailing Desktop Linux had improved greatly in usability and user-friendliness. We organised an audition for the new delivered Linux and the conclusion was that the Sailing Desktop Linux matched the majority of government OA (office automation) application requirements.”

The Sailing Desktop Linux was installed on the terminals of government OA (office automation) systems in various Beijing government agencies, such as Beijing Municipal Administration Commission and Beijing Discipline Inspection and Supervision Bureau.

In the same month, CS&S released COSIX Linux Desktop 3.1 OS. Unlike its predecessor, the COSIX Linux Desktop 3.1 focused only on the desktop applications, in particular OA applications, so that all the components and services for server applications were excluded. It had a specific potential end user – the unsophisticated user in China’s mass market.

The forking of COSIX Linux

The Sailing Project was the first opportunity for CS&S to reach the real end-users and understand their assumptions of Linux. The aim of the project was to enhance the performance and improve the usability of Linux to reach the OA requirement.

To be designed as a general-purpose OS, the earliest three versions of COSIX Linux 1.0-3.0 integrated the components for all potential applications, encompassing desktop and server applications. Based on the feedback from users in the Beijing government, CS&S recognised that the performance of COSIX Linux was poorer than MS-Windows in desktop application because the large size of COSIX Linux used much hard disk storage space and the services for server application wasted too much system resource. Accordingly, all the components and services for server application were removed from Sailing Desktop Linux.

After the Sailing Project, CS&S realised that the requirements and assumptions of personal users and server users of OS are quite distinct and that COSIX Linux should be developed as two product lines in accordance with the two types of user.

Firstly, the personal user paid attention to usability and user-friendliness of the OS, but the server user was more concerned with stability and performance. To achieve high-usability and user-friendliness for the former, developers had to introduce new features and functions, such as a friendly GUI (Graphic User Interface), convenient system management tools etc. However, these features may have risked the stability of the OS unless they were tested strictly by engineers, and in particular tested by the users in the mass market as black-box testing. Therefore, the preferences and requirements of personal users may be contradictory to those of server users.

Secondly, Sailing Project expressed that Chinese character processing and printing is one of the most important content factors for desktop application. But the printing function normally is not necessary for nearly most types of servers apart from the printing server.

Last but not least, the usage of PC and server determined diverse requirements for hardware adaptability and software compatibility. For hardware adaptability, the personal user normally requires that the OS is compatible to most external devices, such as printer, web camera, external storage media, and so on. In order to achieve stability and security, the server user, however, seldom allows external devices to connect to the server. In respect of software compatibility, the server user requires that the OS is coupled seamlessly with mainstream server application software, such as databases, web service software, mail service software, network administrative tools, and so on. Additionally, they require tuning tools to optimise the performance of the application software, whereas personal users generally hope that the OS is able to support as much application software as it can, in particular entertainment applications.

According to the understandings of user requirements for PC and server, CS&S released COSIX Linux products for both the PC user and server user respectively. A ‘newest and fastest strategy’ was adopted for COSIX Desktop Linux: the OS would use the newest released Linux kernel, components and functions. A more conservative strategy was employed on COSIX Server Linux development. In order to fulfil the requirements for providing uninterrupted service, COSIX Server Linux was inclined to employ stable kernel and components. Only when the new functions developed for desktop Linux had proved that they would not damage the reliability of the whole system, and could really help to improve the usability for server users, were they added to COSIX Server Linux.

As mentioned above, CS&S released COSIX Desktop Linux 3.1 in June 2002, and COSIX Server Linux 3.1 soon after.

5.3.2 New start for COSIX Linux

The market generated by government and the support from government

The Government On-Line Project was initiated by China Telecom, SETC (State Economic and Trade Commission) and the information administrative departments in more than 40 government ministries from 1999. Its core mission was to use information technology to help all levels of government to transform themselves from an administration-oriented to a service-oriented mission. According to the statistical data from CNNIC (China Internet Network Information Centre) on 2005, 90% of ministry and province levels and 80% of city and county levels of government set up official websites.

The “Law of the People's Republic of China on Government Procurement” was adopted by the Standing Committee of the National People's Congress on June 29, 2002. It constitutes the legal framework regulating public procurement behaviour in China. The 10th clause states clearly that government procurement shall target domestic commodities, engineering works and services, with only a few exceptions. Three months later, State Council promulgated “the Action Plan for the Rejuvenation of the Software Industry (2002-2005)” (2002, No. 47 Document) to re-emphasise the priority of Linux in government procurement and state science and technology funds support. According to the Ministry of Finance, the total amount of government software procurement had reached 100 billion RMB (£10 billion) in 2002, the procurement amount related to the Government On-Line Project accounted for 35 billion RMB (£3.5 billion).

Both the laws and regulations, and the Government On-Line Project generated a huge market for domestic IT enterprises. The strong government background and state-ownership of CS&S was one of the important reasons why their products found favour on the list of government procurement. Following the first software procurement in 2001, the Beijing government launched subsequent procurement behaviour in the three years of 2002, 2003 and 2004. The indigenous Linux products were not restricted to desktop versions as in 2001, but expanded to Server Linux Products. CS&S was one of the Linux suppliers for the Beijing government.

In order to spur and support R&D capability in indigenous software industry, some departments in China’s government issued their own financial plans; these most famous plans included North Software Base by State Development Planning Commission, National Software Base by national Torch Plan, and the National 863 Programme Accomplishment Industrialisation Base. Once an indigenous software enterprise was listed in one or more of the above plans, it would get financial support from the corresponding government department. After the Sailing Project, CS&S became a member of the above three support plans.

The strategy of desktop Linux development

CS&S always treated the government as its largest market. In the desktop segment, they sought to provide complete OA solutions for government users.

In order to facilitate Linux-based indigenous OS development in accordance with Chinese users' habits, and further develop a desktop OS to match the requirement of e-government, the Ministry of Science and Technology initiated a key software project "Desktop OS and Its Affiliated Environment" with funds from the 863 Programme in 2002. CS&S is one of the Chinese Linux suppliers to undertake this R&D project.

Compromising for users

Drawing upon the experience of participating in the Sailing Project, CS&S was aware that end-users had never taken the technical issues into account; what they cared about was the usability. Like all Linux providers, CS&S had to generate profit to survive in the domestic market. Understanding that Chinese users are unsophisticated users and their use habit has been locked-in to MS-Windows, the development strategy of COSIX Linux Desktop is 'accurately compatible with MS-Windows OS'. The term 'accurately compatible' means minimising the difference between Linux and MS-Windows on GUI and the way users operate the OS. Therefore, all end users who were familiar with MS-Windows were able to switch to Linux without starting a new learning curve. As the former manager of the division of foundation software in CS&S and current president of CS2C explained:

"The reality in China is that MS-Windows series of products almost dominated the desktop segment of the OS market and accounted for over 90% market share. The strong user lock-in effect and lower level of IT skills of end users determined that any OS wanting to enter this area cannot avoid compatibility to MS-Windows. Otherwise, the end users in the mass market would not accept it."

In order to enhance usability for Chinese users, CS&S found that the essential step should be to make the system functions visible to the user. As explained by the product manager of CS2C:

"From our point of view, the menus of MS-Windows are the entrances for system functions, which provided a conduit for users accessing these functions. However, the system functions of Linux are distributed in various locations and folders so that they are hard to find by users."

Referring to MS-Windows, CS&S redesigned the desk interface and reorganised the "start" menu and "control panel" for their COSIX Linux 3.1 and successor versions. Additionally, COSIX Linux added similar icons for the menus and functions that users accessed frequently in MS-Windows, such as "my computer", "my documents", "network neighbour", "file explorer", "control panel" and so forth. The users with experience in operating MS-Windows were thus able to find the functions and applications they were looking for easily on Linux GUI.

In addition, the pages of foreign websites were designed and developed in light of international standards. These web pages are browser and OS independent. No matter what kind of browser used, the web pages displayed on screen are the same. But the Chinese programmers are used to developing web pages using Microsoft's tools and technology. With the purpose of accelerating access speeds, a large number of the Chinese web pages were further optimised for Microsoft IE (Internet Explorer) browser. The layout and format of these web pages would be abnormal when accessed by browsers such as Mozilla and Firefox, on Linux. CS&S had to develop some patches for COSIX Linux to make sure that the Chinese web pages developed and optimised for Microsoft IE could be correctly displayed on COSIX Linux.

In line with the aim of and with funding from the 863 Programme, COSIX Linux Desktop 4.0 was delivered to the domestic market and passed the audition organised by Ministry of Science and Technology.

Criticism from international Linux communities

All the improvements regarding enhancements to the usability of Linux incurred heavy criticism from international Linux communities. The contributors in international Linux communities are sophisticated users and IT professionals who developed Linux to fulfil their own needs for function or enjoy technical challenges. Another motivation for their contribution to Linux is the signalling incentive which stems from a desire for peer recognition. Therefore, developing and modifying Linux for unsophisticated end users, such as developing easy-OS management and configuration tools, and developing friendly GUI, led to poor technical satisfaction and peer recognition. The reason for this is that developing such features only reflects the understanding of market and users, rather than showing programmers' talents in programming and technical capabilities.

The International Linux communities are flourishing and have significant influence on the whole Linux realm. Most of the foreign end users of Linux are Linux enthusiasts and contributors, who are favourable to open source software and accept all the features of Linux. They complained that any improvement made by Chinese developers should coincide only with international technical standards rather than trying to make Linux compatible with given proprietary OS or technologies. Consistency with proprietary elements would ultimately contradict the free nature of Linux finally.

Apart from the critique on China making Linux compatible with MS-Windows strategy, international Linux communities always ignored the contribution made by Chinese Linux developers. The Chinese character display and input functions have not been taken into account by international Linux communities because the core leader(s) of Linux kernel and components projects are not Chinese, and further they have no means of dealing with Chinese characters. To adapt Linux in China, this task had to be done by Chinese commercial or individual developers. Although CS&S and other Chinese Linux suppliers dedicated themselves to improve the performance of Chinese applications, new problems and requirements ceaselessly arose with Chinese Linux development. Take solving the Chinese code problem for MP3 files as an example. A strong multi-media entertainment function is necessary for Linux to attract end users. Playing MP3 audio files is seen as a very basic function for PCs. MP3 files generally consist not only of audio code but also a piece of literal information for describing the audio, such as the name, the author and album of a given song. According to international MP3 coding standard, the literal information was coded in line with UTF-8 standard. That is the reason why only UTF-8 coding standard is supported by the mainstream international Linux distributions. However, the GB2312 and GBK Chinese character sets are national standards in China so that most of the literal information of MP3 files is coded to comply with these two sets in China. As a result, the information of MP3 cannot be displayed correctly on Linux. The Chinese Linux providers certainly should fix this problem by developing a plug-in programme for their Linux distributions.

Nonetheless, the programmers in international Linux communities do not care about foreign languages other than languages in the Latin phylum. The GB2312 and GBK are treated as the only regional language standards, so all the improvements on Chinese applications are excluded from Linux development, even though they are convenient for Chinese local users.

The strategy of server Linux development

In the desktop area, the technological advantages of Linux were invisible because of the unsophisticated feature of Chinese end users and their locked-in effects. On the contrary, many powerful server mechanisms of UNIX are inherited by Linux. Server has higher requirements in network performance and security than PC. Linux has perfect network compatibility which supports nearly all popular network protocols. As a UNIX-like OS, Linux has a more mature and securer network management mechanism, naturally inherited from UNIX. Most server application software is integrated into the mainstream Linux distribution, including the COSIX Linux 1.0 and its successors. Therefore, Linux can be used for server OS with very few accommodation tasks. As a former staff member in the division of foundation software in CS&S and current product manager of CS2C explained:

"Like the international mainstream Linux distributions, COSIX Linux, and later COSIX Server Linux integrated a powerful web server software - Apache. Furthermore, nearly all Linux distributions have the components providing various network services, like the FTP, E-mail, Telnet, and so on. We can say that once a Linux system is set up, the computer can be

used as an Internet Server. ”

In the Beijing government procurement projects launched in 2002, 80 copies of COSIX Linux were purchased as server OS. They were running on web servers, database servers and mail servers. After that, COSIX Server Linux 3.1 won the bidding of the Beijing government procurement again in 2003. Both COSIX Desktop Linux and Server Linux have been applied in the OA system of 44 agencies of the Beijing government since 2003.

Being the capital, the Beijing government has far greater financial and manpower resources than all other local governments in IT application. However, the government agencies in small cities or counties have a major task, under the Government On-Line Project, to establish official websites to publish information. Consequently, CS&S found that what the government user really wanted was not merely an OS, but a total solution to sustain official website development, operation and maintenance. The COSIX Server Linux was just an affiliation of this solution. However, CS&S soon felt that providing website solutions for individual government agencies was time and energy consuming. At the same time, the sale amount of COSIX Linux was limited. Therefore, CS&S decided to provide a generic website solution based on COSIX Linux. On the one hand, there were too many business opportunities to be captured by CS&S due to lack of manpower and time. A generic website solution would bring more contracts. On the other hand, one of the core businesses of CS&S is developing Linux, rather than providing a customised website. The more generic website solutions sold, the greater the sales that COSIX Linux generated.

Many governments made the majority of investment on computer hardware and devices in the initial stages of the Government On-Line Project, rather than focusing on website development and subsequent maintenance. Unlike website development, which is a one-off project, maintaining a website is a long-term job. In addition, customers may pay solution providers for developing a website, but they have to maintain and administrate the website themselves. To implement a basic function of information publishing for a government website, the administrator needed to develop or edit an HTML web page on a local computer, and then update it to the web server in a traditional administrative way. For instance, there are three steps for publishing new information. The first step is developing a new web page for the information; the second step is editing the index page to add a new hyperlink referring to the new page developed in the first step; the last is updating both index page and uploading the new page to the server where the website is stored. A sophisticated administrator is normally required for website maintenance.

However, not all levels of government agency have their own website administrators so a number of government websites were called dead sites, referring to a web site without any updating after development. Moreover, many government websites are separated from government intranet so the government staffs were unable to interact and provide services to people online. Instead of establishing online images, the final aim of the Government On-Line Project was to realise e-government and provide a convenient online service for people. All levels of government accordingly had a new requirement to upgrade their server system and set up a dynamic website.

In accordance with the dilemma of the higher technical requirements and unsophisticated staff, which government agencies faced regarding webpage development and further maintenance, CS&S developed a Linux-based solution named DWDAS (Dynamic Website Developing and Administrating System). The system broke down the whole website into several objects so that the user/developer was able to manage every element of each webpage separately, no matter how large and complex the webpage is. With DWDAS, all the tasks of website developing can be done by simply clicking the mouse. This solution included some templates for webpage designing as well. As a former staff member of the division of foundation software in CS&S and current product manager of CS2C recalled:

"Some of our government customers told us that they had been complaining because the design of their webpage was too awful to comply with the image of a government agency. For example, too many photos or flash files were loaded on the index webpage, which slowed down the access speed. To help customers to design appropriated web pages and further establish good online images, some good webpage templates were integrated within out DWDAS

drawing upon the experience of browsing hundreds of government websites in foreign countries. All the customers then had do wais just fill the content into the templates.”

The DWDAS contained an information publishing system, which was used to publish and/or edit government affairs information, laws and regulations. The three steps of information publishing in a traditional approach were no longer necessary in DWDAS. A user/administrator can publish public announcements and news, as well as edit the content of existing web pages by just typing words in a background management system. The complexity of website maintenance was greatly reduced so that normal government staffs were able to maintain websites in their spare time. As a former staff member of the division of foundation software in CS&S and current product manager of CS2C pointed out:

“The staff’s tasks in information publishing include author, auditor and administrator. The information publishing system of DWDAS provides a windows interface for them separately so that they can complete their task by inputting words or clicking the mouse under a certain authority.”

The website developed by DWDAS could be connected to the government intranet network or OA system. As a result, the website would be matched with specific functions of the particular government agency. The website therefore realised the interactions between people and government staff: on the one hand people are served by government staff without visiting their office in person; on the other hand, people are able to provide feedback to government agency.

Government websites generally have a great amount of published information so that in the past visitors always spent lots of time finding historical data. The DWDAS provides a searching engine component for websites, with which visitors are able to locate what they are interested in easily and quickly.

In summary, the technical requirements of website developer and administrator are significantly reduced by employing DWDAS. In term of website developing and constructing, the efficiency of an unsophisticated user of DWDAS would be higher than a professional with traditional website development tools, while the website maintenance tasks become quite easy for an unsophisticated user.

The Establishment of China Standard Software Co. Ltd. (CS2C)

Before 2004, Linux business only accounted for a small part of the whole of CS&S, and the division of foundation software had not generated enough income to break even the operational cost of Linux development. Most of CS&S's profit was made from application software development, industry solutions and related service provision. However, all of the above profitable businesses are related to MS-Windows OS products. As a result, CS&S realised that seeking further cooperation with the international software giants, like Microsoft would be an effective approach, leading to rapid development.

In November 2003, CS&S terminated its contradictory relation with Microsoft in the

OS area, and became one of the global strategic cooperation partners of Microsoft[8]. According to a memorandum of cooperation, CS&S would develop solutions for Chinese customers based on Microsoft .net platform and MS-Office. At the same time, Microsoft would supply their newest technology, provide training programmes for software engineers, and offer outsourcing contracts to CS&S.

In order to strengthen connections to Microsoft, the foundation software division was peeled off from CS&S, to establish CS2C collaborating with CETC (China Electronics Technology Group Corporation) and ECICT (East China Institute of Computer Technology, NO.32 Research Institute of CETC) in 2003.

[8] Microsoft CEO & President Steve Ballmer visited China in the June of 2002. The State Development & Planning Commission of the Peoples Republic of China (SDPC) and Microsoft signed a memorandum of understanding (MOU) to begin the largest joint Sino-Foreign software industry cooperation in China. Under the Memorandum Microsoft invested 6.2 billion RMB (£620 million) in China during 2003-2005. The Chinese government believed that signing of this MOU will have a significant impact in attracting foreign investments in China, expanding the export of software and related IT products and the import of advanced technologies and management experience from abroad. In order to implement this MOU, Microsoft signed a memorandum of cooperation with CS&S in 2003.

The whole COSIX Linux business of CS&S and an indigenous office suite business of ECICT comprise the core business and products of CS2C. CS2C would continue developing Desktop Linux and Server Linux distributions with the new brand - NeoShine, as well as solutions based on NeoShine Linux distributions. The ECICT office suite was seen as a beneficial supplement for the new NeoShine Linux business by integrating as an OA desktop solution for government users.

5.4. Growth and Future (2006- onwards)

5.4.1 CS2C desktop Linux

Happy Family Project

Some Chinese computer manufacturers, such as LENOVO and Acer, released a type of double-mode PC in 2002. The core idea of a double-mode PC is that users can operate the PC like an electronic appliance, which means some functions can be implemented without starting the OS. With double-mode PCs, unsophisticated users, even the users with no skills in operating a PC, are able to enjoy the entertainment functions of a PC.

As the largest in China and the fourth largest white electronic appliance manufacturer in the world, Haier made a decision to enter the PC business in 2004. Facing extremely fierce market competition in China's PC market, Haier and its cooperative partner – Intel, who was promoting GAPP (Government Assisted PC Program) in China, identified that China's rural area is the next large potential market for the PC business. Therefore they established a Rural PC Laboratory to conduct the R&D on PCs for Chinese rural users in 2005.

After designing and constructing the hardware structure for the rural PC, named Happy Family PC, the next step was software solution designing and development. However, Haier specialise in electronic appliance and consumer electronics designing and manufacturing, while it has neither a software development team nor experience for software solution development. Haier consequently decided to outsource the software solution project to a professional solution provider at the very start. Haier spent two months in selecting a software solution provider and eventually signed an outsourcing contract with CS2C. The project manager of Haier Happy Family PC explained:

"There were three reasons for selecting CS2C as the software solution provider for our Happy Family PC. The first reason is the lower price of Linux. As the PC product focused on Chinese rural users, the price is the first problem that had to be taken into account. In China, the cost of computer hardware is very low so that the license fee for proprietary software including OS, office suite and entertainment, generally accounts for half of the total cost of a PC. Deploying open source software would definitely reduce the cost of Happy Family PC significantly. The second, users in rural areas have not been locked in to MS-Windows as Chinese urban users have. As a result, using Linux as the OS would not contradict their use habit, which is non-existent. The last reason is that CS2C can customise their Linux products to match the unsophisticated features and requirements for functionality of Chinese rural users. "

CS2C had become one of the four global strategic partners of Intel in the Linux area in April 2004. Before the new CPU and chipset products were delivered to the market, Intel would provide them to CS2C for testing purposes. In order to guarantee their hardware is compatible with CS2C Linux perfectly, Intel also provided a large quantity of technical support as well. The Happy Family PC is based on Intel hardware platform so that CS2C face no problem in hardware adaptation.

Before sending user requirements to CS2C, Haier first conducted a very professional market investigation in two villages. As the project manager of Haier Happy Family PC recalled:

"By contrast to our presumptions, the price of the PC is not the essential factor for rural users who are willing to buy a PC. What they really care about is how to use it and what they can obtain by using it."

According to the market research data, Haier found that the requirements of rural people vary according to different ages. For example, the elders are interested in medical treatment related information, while the adults care about their businesses, and children are eager for educational resources. Haier thought that Happy Family PC should fulfil all the requirements of the rural users at different ages.

Besides the traditional multimedia module, Haier believed that the software system of Happy Family should include an education module, an agricultural information module and a module to provide medical treatment knowledge and information. Moreover, the system should also have a friendly GUI and be easy to use.

In the user requirements provided to CS2C, every detail was addressed. However, not all detailed requirements were able to be implemented by CS2C. As the product manager of CS2C pointed out:

"We acknowledged that detailed user requirements were very helpful since we didn't need to spend time in understanding their presumption and/or help them refine their ideas. However, some of their requirements were too detailed to be matched exactly as they are described. With respect to these requirements, we have negotiated with Haier many times. That's the reason why the final software system of Happy Family PC is different from the original requirements from Haier in some ways."

Users in Chinese rural area are characterised by few or even no IT skills so one of the purposes of the software system of Happy Family PC is to enable the rural users to operate a PC easily. In the original user requirement, Haier hoped that the GUI could be represented in a cartoon style. The GUI should look like a room and the function modules be represented by certain objects. For example, a desk referred to the education module. When a cartoon puppet controlled by the user moved to the desk, then the education module is invoked. But CS2C could not realise this requirement as Haier wished, because it was not good at animation development. As a result, the function modules were designed as different icons on the GUI. Users are able to invoke the application by clicking these icons. Although the final solution is not exactly what Haier described, it was finally accepted by Haier because the purpose of easy-usability was realised in CS2C's solution.

Regarding the entertainment module, Haier definitely hoped that all the mainstream media format files would be able to be played on Happy Family PC. Owing to the intellectual propriety issue, CS2C provided two options to Haier. One option was that only the media files without intellectual property should be supported by Happy Family PC. If the user demands to play media files with intellectual property, they would have to buy responding players and decoding libraries themselves. The other option was that Haier bought the players and decoding libraries so that CS2C could meet the requirement of playing all mainstream media format files. Haier finally picked the former option.

Furthermore, some information, like the agricultural knowledge, education knowledge, medical treatment knowledge, was supposed to be stored on local PCs so that users were able to read the information by accessing application modules. To CS2C, the only approach to obtaining the information was to collaborate with professional information providers. In addition, the size of the information largely exceeded the size of Happy Family PC's hard disk. Therefore, only information that is considered to be accessed most frequently was stored on the hard disk. As the product manager of CS2C described:

"We acknowledged that the software solution would be perfect if Haier's requirement for collected knowledge and information could be met. But Haier had to accept our suggestions due to the financial budget. "

The final soft solution for Happy Family PC

The software solution for Happy Family PC included two GUIs: one is a traditional interface of CS2C NeoShine Desktop Linux; the other is a navigation interface for the rural users with few or no skills in operating a PC and accessing the Internet. With a navigation interface, the user has no need to input the web address constituted by English words, but just clicks the icon of function modules and then the local and online resource will show on screen.

Four function modules are designed and integrated into the software solution of Happy Family PC.

The first one is Agricultural Information Platform. This module is actually the search engine for the database of a website which is an affiliation of the Chinese Academy of Agricultural Science. With this module, the user is able to obtain necessary information regarding pest prevention and control, planting and breeding, and the Chinese agricultural products market.

The second module is Distance Education Platform. The situation in China is that education in rural areas lagged far behind the development of education in urban regions. In light of the market research conducted by Haier, most samples indicated that child education was one of the most important reasons for buying a PC. With this module, children in rural area can access the same educational resources as children in urban regions. In order to realise this module, CS2C cooperated with K12, which is one of the most famous websites providing education resources at primary and high school level.

The third is the Medical Treatment Platform. This module is seen as a remedy for backward rural medical conditions. Users, in particular the elder user, can study health knowledge by accessing the Internet. In some areas, rural users can accept an online service provided by hospitals.

The last module is the Multimedia Entertainment Centre. Users can play media files, such as film, music, and photographs stored on hard disk, disc and other storage medium directly without running Linux. At the same time, the screen can be used as a TV set to play cable TV programmes through the integrated TV card.

This CS2C NeoShine Desktop Linux-based software solution achieved remarkable market success along with Haier’s Happy Family. Over 400,000 copies of the software system were sold with Haier Happy Family PC in 2006. They also drew the attention of the World Bank in an exhibition organised by the Union Nation at the end of 2005. The World Bank soon initiated a worldwide promotion for rural PC, and Haier’s Happy Family and its software solution was cited as a successful example for other developing countries.

New product strategy on Desktop product

Learning from the Happy Family Project

The business model of generating profit has long been a difficult problem for Chinese Linux providers. Despite strong Chinese government support through public procurement and financial subsidies, indigenous Linux only fulfilled the basic requirements of OA. However, there are barriers for widely deploying Linux for government OA. In order to achieve specific working purposes within administrative systems via the Internet or government Intranet, some special software packages are developed by, and are required to be installed under the instruction of, superior administrative agencies. For example, the Ministry of Finance required its subordinates to install customised accounting software with which the standard accounting data could be transformed into an encrypted format and transferred smoothly between the ministry and its subordinates. But nearly all of these specific software packages were developed for MS-Windows. The suppliers of these application packages normally distribute nationwide, and most of them are small-sized enterprises. Porting this specific application software to Linux platform was not only time and energy consuming, but also led to additional pecuniary costs. This problem exceeded the responsibilities and services which Chinese Linux enterprises were able to undertake. For this reason, MS-Windows products are still the most prevalent OS products in the Chinese government OA system.

Furthermore, lacking complementary application software in other desktop application fields, there is still a wide gulf between Linux and MS-Windows. This is the most important reason for low installation rates and high uninstallation rates of desktop Linux. As the president of CS2C pointed out:

"Besides direct support from government, Linux also benefited from government support in developing indigenous office productivity suites. The usability and functionality of

indigenous office suites are very similar to MS-Office suite. That's the reason why Linux can fulfil the basic requirements of OA. However, both the number and quality of Linux application software in other areas is less than the applications for MS-Windows. This shortage of Linux application software could be remedied efficiently, if we focused on the business of providing customised desktop Linux or the solution based on it for users with a specific requirement in functionality or systems operation."

Learning from the Happy Family project, CS2C realised that a customised strategy would exert the open features and customisable advantages of Linux to its maximum. Not all customers required a versatile software system and OS; they may have just limited requirements and purposes for a PC. In this respect, only open source software, like Linux, can be developed according to the exact idea of developers or users. Thus, CS2C began to explore customised products based on traditional desktop Linux distribution.

The production of customised Linux desktop

According to the strategy in developing a customised desktop Linux product, CS2C released a Linux-based Hotel PC solution in cooperation with the Great Wall Group. Both noticed that there was a big gap between hotel information management and the actual requirements of the customers in hotels. On the one hand, market competition for the hotel industry became more and more fierce; therefore, hotels had a requirement to deploy IT in their business processes to reduce operation costs. On the other hand, the customers, including tourists and business travellers, had a requirement for an information service. The Hotel PC solution was installed on the PC in every room and contained specific modules regarding local delicacies, local tourist attractions and scenic spots, shopping areas, weather conditions etc. Furthermore, the traditional entertainment module cultivated in Haier's Happy Family project, such as video on demand and online games, have not been ignored. Around 500 copies of Hotel PC solution were pre-installed on the Great Wall computers in 2006 and 2007.

At the same time, CS2C signed another contract with Bluestar to provide a Linux-based solution for Bluestar Elder PC. Taking the features of elder users into account, besides the traditional entertainment module and the feature of easy usability for the elder user without IT skills in operating a PC, two new functions of Hardwiring Input and Magic Magnifying were added into the solution for Elder PC.

5.4.2 CS2C server Linux

The first phase of CCB Project

CCB (China Construction Bank) established on 1 October 1954, is one of the four state-owned commercial banks in China. At the beginning of the 1990s, CCB began to apply IT within the core business and construct a network transition clearing system for the whole bank. Apart from the network for the whole bank, regional transaction networks have been established in 210 cities since 1997.

The UNIX OS was widely deployed in the information systems of China's finance and accountancy sectors. Like other state-owned commercial banks, CCB also installed SCO UNIX as the OS for PC and Server computer at all branch levels. Along with the rapid development of the IT industry, some drawbacks of SCO UNIX were perceived. Take the hardware adaptability issue for example, the users of SCO UNIX faced a problem in that a lot of newly released hardware cannot be recognised by the OS because SCO stopped upgrading its UNIX products. The problem is the same with the compatibility of newly delivered software. The task of expanding new business is limited by these technical drawbacks so CCBSB (China Construction Bank Shanxi Branch) was determined to replace their existing UNIX based business system with a new financial solution.

Based on past lessons in choosing a proprietary OS, CCBSB made the decision of choosing indigenous Linux for their business system. As the director of the Department of Science and Technology CCB Headquarters commented:

"On the one hand, Linux can be customised according to the specific financial applications. As a result, the components and modules that were not related to our applications can be removed off from Linux. The system thus would have higher stability and

performance. In addition, Linux is supported by all the international mainstream software and hardware suppliers in the server application field so that we don't need to worry about the compatibility problem. On the other hand, we believed that Linux communities would provide continuous support to Linux innovation and development. Choosing foreign proprietary software indicated that a dependent relation to foreign companies is generated. We have been suffering from applying SCO UNIX not only because of the high cost of its products and services but also the SCO company stopped providing technical support and upgrading its products. Finally, financial security is another important concern that we must take into account."

The project of replacing SCO UNIX with indigenous Linux in the bank counter system of CCBSB obtained a financial subsidy of 30 million RMB (£3 million) from the key software application project in the 863 Programme, namely "The Key Demonstration of Indigenous Linux OS Applied in Financial Areas" in July 2003.

Rather than deploying traditional PC-Server architecture in the counter system, the CCBSB adopted Terminal-Server architecture. It meant that the terminals on counters were only input/output devices of one or more servers. In contrast to the term of PC, the terminal has no independent hardware and software. Through an Ethernet network or series port cable, the terminal can invoke a user interface for the counter operator from front banking server, with which all the operations are completed on the server directly.

In line with strict security requirements, the task of switching application software from SCO UNIX to Linux was to be completed by CCBSB. According to the contract with CCBSB, CS2C was required to provide server Linux for the counter system, and Linux related technical support, such as solving hardware adaptability and providing complete system testing. In order to accomplish application software switching, CCBSB also asked CS2C to supply the necessary technical knowledge for software development under the Linux running environment. There was no detailed user requirement provided to CS2C, but a brief description of the application mode and structure of the counter system.

CS2C spent five weeks developing the customised Server Linux for CCBSB, which contains a running environment and development environment. The former was a tailored compact minimal-sized Server Linux for running the software system, while the latter contained more software packages and libraries, which are used for application software development and porting. The user is able to take options regarding these at the beginning of system installation. The product manager of CS2C explained:

"In light of our experience in software development, more software packages lead to less stability of the system. Therefore, we tailored a minimal OS for the daily operations of CCBSB. For instance, all the GUI related libraries and components are cut off because we were aware that their terminals are character-based interface. In addition, they have a printing requirement so that a complete printing system is retained. Other packages and libraries, like editor, grammar checker, complier, testing tools, C Language library and so forth are used for application software development, but have no relation to application running, so they are kept only in the development environment."

Hardware adaptability task

After delivering the Server Linux package, the next crucial task was hardware adaptability. CCBSB were unwilling to change or upgrade any existing hardware devices so the hardware adaptability would be the most time-consuming task for CS2C. There are some special hardware devices designed and developed for financial application purposes. However, they only provided drivers for the SCO UNIX. CS2C therefore had to make them compatible with Linux.

The adaptability task for a special printer, called a bankbook and form printer, was a good example. Unlike most of the banks in Western countries, Chinese banks do not provide bank statements periodically for their customers. All the historical transactions are printed on a bankbook which is kept by the customer. In addition, all the customers served at the bank counter are required to fill in special formatted tables on a slip and then sign their signature. Therefore, the printing function of a counter terminal is limited to bankbook or table printing only. However, the printing mechanisms of Linux and UNIX are distinct. In the previous SCO UNIX based counter system of CCBSB, the format and layout of bankbook and bank slip were preset by an independent software package, while in Linux environment, this printing function is controlled by the printer driver. As a result, CS2C had to solve this problem to avoid printing chaos.

Another problem faced by CS2C was to enhance the adaptability of Linux to vintage hardware. The CCB was one of the earliest Chinese enterprises to apply IT to its business and operations so some of the hardware was outdated. However, Linux kernel did not provide compatibility to these vintage hardware devices. In order to save costs, CCBSB hoped to keep these old hardware devices, therefore CS2C had to modify the Linux kernel or develop plug-in programmes for old hardware adaptation. As the product manager of CS2C pointed out:

"Every terminal is linked to the front banking server through a series port cable. During the system testing process, we found that counter terminals and the server could not communicate with each other. We accordingly checked the OS, application software, hardware, and the cable respectively and carefully. We eventually realised the reason was that the international technical standard of series ports had been updated so that the existing cable could not establish communication on the updated Linux. The easiest solution was substituting these cables with new ones. But CCBSB had thousands of old cables so they asked us to solve this problem by making modifications to our Linux system. "

There are some problems which are unable to be solved by the means of modifying Linux. For example, a standard server mainboard normally has two series ports, while the front banking server needed to link more than two terminals. Thus a special device called a multi-user card was employed, which provided additional series ports for the server. However, there were four suppliers of CCBSB's multi-user card according to different procurement time periods and none of them had developed drivers for Linux. In addition, technological advances meant that the series port connection had been substituted by Ethernet since it offered much faster data transforming speed. The multi-user card for series port cable connection was therefore about to quit the market at that time and only one supplier still manufactured this product. Facing this problem, CCBSB had to purchase additional multi-user cards from this provider to replace the products manufactured by the other three suppliers. CS2C was required to develop the drivers for Linux based on the source code supplied by this supplier.

Porting application software

Porting application software refers to switching the application software running on the previous SCO UNIX platform to Linux. Besides the application software developed for financial applications there is also a good supply of supporting software, such as database and middleware. Substituting Linux for SCO UNIX required not only additional development for application software porting, but also a requirement to solve the compatibility problem of Linux and the supporting software. As the project manager of CS2C stated:

"Porting application software should have been a very easy task for CCBSB. Theoretically, they just needed to recompile the source code of the application software in Linux environment. Practically, however, the porting process was not smooth. We had to provide technical support to ensure success of the porting process and enhance the capability of Linux to the supporting software."

Being a UNIX-like system, not all of the mechanisms of Linux are the same as SCO UNIX. For example, thousands of grammatical errors were found in the source code of the application software on the Linux environment. As a result, this source code could not be compiled to machine executable programme/software. In order to solve this problem, CS2C modified its grammar checker of Linux based on the features of UNIX.

CS2C-BAA Joint Laboratory

The CCB Project was the first that applied indigenous Linux in a huge and complex application system. As a result, the project management was in chaos owing to the lack of relative experience in coordinating such a complex project where several actors were involved. As discussed above, the compatibility problem was salient in this project. Neither the project managers of CCBSB nor the software engineers of CS2C were able to work out a solution efficiently in the case of compatibility problems. The resultant measure was that the project manager assembled the technicians of all providers to a meeting to investigate the reasons leading to the problems and to figure out the remedy. As the product manager of CS2C explained:

"*Before the compatibility problems were solved, we had to stop our ongoing jobs. However, not all the factors that caused the problem were associated directly with Linux. For instance, a problem might have resulted from a capacitor on the mainboard. But CS2C was required to take part in all the meetings because technical support for switching to Linux was our responsibility. If we knew the factors that caused the problem in advance, it would not be necessary to stop all of our work.*"

Attending such meetings was perceived as time and energy consuming in some cases so that CS2C decided to establish a special technical support team to help the project manager of CCB and the engineers from software and hardware providers to precisely locate the roots of the compatibility problem in the later stage of the CCB Project.

In order to save time and pecuniary costs, and without affecting the schedule of the whole project, the strategy employed was to explore collaboration with external public R&D resources. The School of Computer Science in BUAA which has both a pre-eminent academic and industrial reputation in software system testing was chosen as a partner. The head of the school expressed a positive attitude to the proposal of this cooperation. As the professor of BUAA and R&D director of CS2C pointed out:

"*Cooperating with industry in R&D is our pressing task because the government R&D funds only support the R&D projects with specific commercial potential. In other words, product commercialisation has become the key criterion for successful R&D project application since 2000. As a result, establishing cooperation links with CS2C could not only help us to apply for projects from government R&D funds but also provide a conduit for commercialising our R&D findings.* "

A temporary team, which consisted of engineers from CS2C and researchers and postgraduate students from BUAA, was set up rapidly. Besides offering support to CCB project development, they would also supply continued support to further upgrades of the CCBSB system. Several months later, the temporary team became the formal department of CS2C, named the CS2C-BAA Joint Testing Laboratory.

CS2C found that most of its customers had no clear idea on their potential computer servers, especially from the technical angle. Some of the customers were willing to buy a complete computer server from computer manufacturers, while others tended to buy hardware components to build a server. The hardware adaptability, therefore, became one of the inevitable issues to applying Linux, in particular for those building their own servers. What's more, various Linux distributions are different to some extent although they are based on the same kernel. As a result, the hardware and software products which claim to support one Linux perfectly may not be smoothly ported to another Linux without local tuning. The task of the joint laboratory is to provide decision support from a technical angle for the customers of CS2C – testing in advance the compatibility and reliability of the application software and hardware which the customer wants to purchase.

Developing administrative tools

In the last round of system testing, CCBSB suggested that some administrative tools were necessary for enhancing the system maintainability. For example, in the case of system abnormality, there should be some monitoring and testing tools to assist the system engineer analyse the problem, as well as providing the solution. The product manager of CS2C recalled:

"We have noticed that usability is a very important feature for the user. Even though we developed some administrative tools for our product referring to the MS-Windows, we still have no idea of the meaning of usability to actual users and what we should do for users. The requirement of CCBSB to develop administrative tools, therefore, was seen as a new idea for us to enhance usability. We grasped the idea that these tools could be ported to all server Linux products."

A set of log management and analysis tool were developed in accordance with CCBSB's requirements and integrated into other Server Linux products, COSIX, later. The information of system running would be recorded as system logs so that the system engineer is able to check the historical data in the case of system abnormality.

The whole project was accomplished in October 2005. CS2C solved and assisted CCBSB to directly solve thousands of technical problems. The CS2C NeoShine Server Linux completely replaced the SCO UNIX of bank counter systems in over 700 local branches of CCBSB. Inspired by the CCBSC experience, another two province level branches – CCBHB (China Construction Bank Hainan Branch) and CCBSB (China Construction Bank Shanxi Branch), as well as a city level branch, CCBQB (China Construction Bank Qingdao Branch) applied NeoShine Server Linux in their bank counter systems in 2006. Since 2006, the successful experience has been promoted to the whole CCB system.

The second phase of CCB Project

In order to improve the efficiency of business processes, CCB Headquarters launched the DCC (Data Centre Consolidation) project also which obtained financial support from the 863 Programme. The investment for implementing the DCC project from the 863 Programme reached 40 million RMB (£4 million).

In the existing multi-level architecture, every local branch is equipped with a front banking server to provide service to all the terminals within the same local branch. For the cross-province transactions, for which data must be processed in the highest servers located in CCB’s Headquarters, all the data has to transfer upward from the front banking servers to the highest level server, in off duty time. Then the data processed in the highest level servers will be sent to the appointed front banking server. As a result, a real-time service for customers is not available.

According to the vision of the DCC project, the previous front banking system in local branches will be upgraded and all the front banking servers in local branches will be integrated into a province level branch. The terminal in the local branch, therefore, invokes the service from the cluster directly at province level. The efficiency of transaction data exchanging is significantly improved by this way. Take cross-province money transfer within the CCB system for example, the transaction generally took at least 24 hours to complete. After implementing the DCC project, real-time cross-province money transfer can be achieved. As the director of Department of Science and Technology CCB Headquarters said:

"*Drawing upon the DCC project, the efficiency of our transactions has been improved greatly. In this respect, we are able to offer a better and more rapid service for our clients.*"

The DCC project resulted in new technical requirements to CS2C NeoShine Server Linux. Firstly, higher terminal management capabilities were called for. The maximum number of terminals which Linux can be linked to was 256. However, CCB hoped their server could manage more terminals in the new data integrative system. In addition, higher stability was required as well. In the past, if any abnormity occurred with a server in a local branch, at most 8 terminals connected to this server would be affected. In the new system, however, any crash of a server may paralyse hundreds of terminals in a region or city, rather than a local branch. CS2C recommended an '8+2 architecture' to CCB for establishing server clusters at province branch level. The 8 represented eight primary servers for handling daily transaction from terminals in a given region, while the 2 indicated two backup servers, which would become primary servers to recover all data and services when failures were detected on one primary server.

Secondly, the providers of application software and database software were changed in the new system. For this reason, CS2C had to make efforts to improve software compatibility again. CCB headquarters decided to pick Oracle database to replace the Informax 7.2 database. As mainstream database software, Oracle database is perfectly compatible with all Linux distributions. In order to completely avoid financial risk, CCB required CS2C to show the certificate issued by Oracle to confirm that CS2C NeoShine Server Linux is entirely compatible with Oracle database products. However, Oracle headquarters stopped issuing such testing certificates for all Linux distributions because the cost for Linux testing generally reached millions of USD per year. Drawing upon private contacts, CS2C finally obtained a testing report issued by Oracle China. As the product manager recalled:

"*A college classmate of mine was the senior staff in Oracle China. Through him, CS2C asked Oracle China to run a system testing for our Linux and issue a report to certify that our Linux is completely compatible with Oracle database software.*"

Thirdly, according to the regulations issued by the central bank of China – People's Bank of China and internal controlling and management regulations of CCB, CCB headquarters plan to develop a counter monitoring system for their new counter front banking system. CS2C not only provided strong technical support for the system development, but also modified Linux to provide a service for the monitoring system. With the special service, all the actions of keyboard and mouse operated on the terminal could be captured and stored in an area by employing the integrated log management tool.

The new integrated front banking system based on CS2C NeoShine Server Linux was first established in CCBLB (China Construction Bank Liaoning Branch) in September 2007. By the end of 2007, the new system has been ported to CCBAB (China Construction Bank Anhui Branch), CCBQB (China Construction Bank Qinhai Branch) and CCBHB (China Construction Bank Henan Branch). The plan of CCB headquarters was that all the CCB province level branches would employ the new system from 2008 and this plan was to be accomplished by 2010.

The Strategy of CS2C NeoShine Server Linux

At this stage, CS2C divided their Linux server product customers into two categories: industrial customer and normal user. The former refers to large-sized leading industrial enterprises. Due to the specific applications and strict operation standards, CS2C is normally required to provide a customised server Linux as the core business system for these enterprises. The latter category of customer is the user who will apply NeoShine Server Linux on its server computer to implement some ordinary tasks, such as email server, web server etc.

The industry solution based on NeoShine Server Linux

Chinese indigenous Linux providers have been in a weak situation since their establishment because they were faced with high pressured competition from foreign giants in the market. These Linux providers had to try their best to fulfil all the requirements of the large-sized industrial customers. As the project manager of CS2C explained:

"It is difficult to persuade industrial customers to change their initial idea in China. Unlike the international software giants, Chinese OS suppliers are not strong enough to negotiate with industrial customers."

Drawing upon the successful experience of the two-phase CCB project, CS2C recognised that industrial applications may provide a new market opportunity for its Linux business. On the one hand, the industrial application will help CS2C to improve usability and enhance the compatibility of their server Linux. As low-level platform software, an OS has to not only be adapted to the special hardware device, but also be compatible with the application software. Once a Linux-based industrial application project succeeds, CS2C is able to port their server Linux and development experience to other customers in the same industry easily and directly. Relying on its successful experience in the financial area, CS2C obtained contracts from China Universal Asset Management and Shanghai Stock Exchange in 2007 and 2008, respectively. On the other hand, the development experience and some user requirements can be fed back to the CS2C NeoShine Linux server products. For example, the CCB project helped CS2C to collect and understand the user requirements, in particular the maintainability requirements. Therefore the corresponding improvements made in the CCB project, such as log management and analysing tools, were integrated into all the CS2C NeoShine Server Linux products. Last but not least, industrial customers normally have huge budgets so their contracts would generate much profit for CS2C.

The CS2C NeoShine Functional Linux Servers

Apart from the server Linux-based industrial solutions, CS2C also developed some functional Linux server products. They integrated specific function modules to help users to easily and quickly construct application servers, such as web servers, mail servers, database servers, file servers, proxy servers etc.

There are commercial and technical reasons for CS2C adopting the strategy of developing functional server Linux products. Commercially, NeoShine Server Linux is a standard distribution originating from international Linux communities. According to the GPL, with which Linux is licensed, any person is able to obtain Linux freely so that it is impossible to generate enough profit from selling it. As the product manager of CS2C explained:

"The GPL states that Linux and its source code are available freely so that the market access of Linux business is extremely low for profit-oriented organisations. Providing services is one of the business modes in the West, but only very few Chinese users are willing to pay for the Linux service. This situation forced us to make value-added developments to Linux in accordance with market segments."

Technically, NeoShine Linux server comprises Linux kernel and open source components. It is normally hard to reach the actual requirements of professional applications completely without system configuration and performance tuning tasks. At the same time, it has low maintainability and usability since these issues are generally ignored by the International Linux communities. In this respect, Linux suppliers have to provide improvements to remedy the maintainability and usability problem.

The product line of NeoShine Functional Server Linux consists of Linux Email Server, Linux Database Server, and Linux Enterprises Application Server.

The idea of Linux Email Server came from the Government On-line Project presented in section 2. At the early stage, the government focused mainly on website development so that government staff generally used a public email service provided by portal websites, or rented an email server from an ASP (Application Service Provider). As email gradually became perceived as an ordinary and efficient way to generate communication between government agencies, as well as between government agency and people, the task of establishing an independent mail server was put on the agenda by all levels of government.

However, they faced two main challenges during the establishment of an email server. One is associated with the email server application software. For the administrator, it was hard to choose the email server application software from various options on the market when taking the compatibility with Linux and system stability into account. Moreover, system configuration and maintenance are highly complex, which calls for adequate experience. The other challenge was that the email server, in particular for the government, required extremely high security therefore a powerful firewall was necessary. Acknowledging these two challenges from previous government users, together with recognising that the email server would be adopted widely in various industries and enterprises, CS2C developed the NeoShine Linux Email Server to help the administrators of potential customers to establish an email server rapidly and easily. The email server application software was integrated into the NeoShine Linux Email Server. Not only was the compatibility of Linux and email server software optimised, but also the complexity in system configuration and maintenance was reduced significantly, since many steps were designed to complete automatically. It has a management tool named Webadmin with which the administrator is able to complete the main system maintenance task - email backup on a web page. Regarding email server system security, a powerful firewall has also been integrated into NeoShine Linux Email Server. The firewall has two main functions. One is providing rapid speed in email text and attachment anti-virus scanning; the other is that this firewall is able to detect and deny service attacks from hackers.

The NeoShine Linux Database Server is perfectly compatible with all the popular foreign and domestic database software. CS2C firstly submitted its server Linux distribution to the mainstream database software providers for testing purposes and ran the system testing in the joint testing laboratory established with BUAA. Database configuration and tuning are extremely complicated tasks which require abundant expertise and corresponding experience in light of different database software. CS2C abstracted their expertise and experience and materialised them as tools in its NeoShine Linux Database Server. With this administration assistant tool package, the administrator could either choose an automatic configuration option according to preset parameters or pick the manual mode, where most of the tasks of database configuration and tuning are possible to complete using the graphic instructions provided by the NeoShine Linux Database Server.

The NeoShine Linux Enterprises Server is designed for implementing routine business operations. It integrated the administration tools of NeoShine Linux Database Server since database software is one of most important components for the business system of every enterprise. Drawing upon the feedback from the CCB projects, CS2C also developed additional tools for server system performance and stability analysis, as well as maintenance. The tools were designed to not only record the historical data as a system log and show current data of the system resource usage, such as the resource of CPU, hard disk, network and memory, but also to provide an analysis mechanism to help administrators to inspect and examine the reasons for low system performance and system crashes based on the system log. Furthermore, over 100 system check points were setup in the system automatically. Additionally, the information and data of system check points will be recorded as a system healthy log. The system can send this system healthy log back to CS2C periodically, according to the customer’s subscription. The support team of CS2C will therefore be able to help the customer to analyse and avoid potential system errors.

5.5 Summary

In this case, we present the process of platform Linux adapting in China’s distinctive context. Its innovation process reflected suppliers learning process in seeking and identifying users, as well as understanding Chinese user preferences and requirements.

At very beginning, in order to develop Chinese indigenous Linux, CS&S simply translated the original English version of Red Hat Linux into Chinese. However, international Linux distributions were designed for keen users who normally are the enthusiasts of OSS. The reality in China was that there was no market for Linux because the users in China’s mass market have been locked in commercial proprietary OS, such as MS-Windows, UNIX etc. The Linux providers, therefore, had no way of understanding the features of Chinese users and then collecting their requirements. In order to achieve the goal of being dependent of foreign countries, the Chinese government arterially created a market for Chinese indigenous Linux by the means of government procurement.

In the second stage－early development stage, CS&S had the opportunity to reach its first local user－the Beijing government through the Sailing Project. From the feedback of the Beijing government, CS&S realized that the personal/office users are different from organisational users in many aspects, for example a friendly GUI, document printing etc so it divided its Linux distribution into PC version and server version.

At the final stage, CS2C switched its strategy on PC Linux from focusing on government office users to concentrating on a niche market. According to the features of users in local niche markets who had little or no IT skills, few requirements on computer system upgrading and limited functionality, CS2C began to explore the opportunities to cooperate with PC providers, such as Haier and Bluestar, to provide a Linux-based desktop solution. On the server end, drawing upon the feedback from the CCB project, CS2C delivered functional Linux servers for organisational/business/industrial users. These server Linux distributions had corresponding management tools for specific server application, which had greatly simplified the tasks of installing, tuning and maintaining server systems.

This study also showed that the government interventions on platform Linux were more powerful than on embedded Linux since its development faced more challenges. Chapter 6 will examine the role of the Chinese government during Chinese Linux development.

Chapter 6 Chinese Government Policies towards Linux

6.1 Introduction

OSS (Open Source Software) has attracted attention from not only the programmer and academic communities, but also from governments. In respect of government intervention in the software market, the free-market advocates, proponents with a libertarian background of the open source movement, and most of the literature, in particular economic literature, is sceptical about justifying forms of government intervention in the software market (Jeff Taylor; Raymond et al., 2000; Evan, 2000; Schmidt and Schnitzer, 2003; Comio and Manent, 2005). By contrast, a significant phenomenon is that dozens of governments and municipalities from all continents have passed or proposed explicit policies fostering the open source movement and encouraging the deployment of OSS in public administration. The US government agencies, like National Security Administration, Department of Energy, Federal Aviation Administration, and US military have been the customers of Linux from the early 2000s. The French government contracted with Mandrakesoft to increase the use of Linux in mid 2002. In May of 2003, Munich city government switched its entire network to Linux from MS-Windows. For the purpose of cutting expense, the European Union has required that OSS should be systematically considered in public procurement, since the spring of 2003. Moreover, many DCs, such as China and Brazil have become the most cited successful cases of adopting policies to promote OSS effectively as well.

The Chinese government started efforts to develop OS since 1960s. Due to the traditional planning economic system, the government have been playing a critical role in all the indigenous OS R&D projects. These projects were organised and financially supported by the government, administratively. All the OS products had the explicit purpose of scientific research, and were developed for specific hardware architecture; therefore the commercialisation issue has never been taken into account. At the beginning of 1990s, CS&S which is a state-owned software solution provider began to develop OS for users in the mass market. However, this commercial activity was greatly influenced by a concept of 'Completely Independent Intellectual Property Rights'. As a result, the programmers had to start R&D activity from scratch: writing every line of code from the beginning rather than utilising existing foreign software technologies. Once the product was delivered, it already lagged behind market requirements due to the longevity of R&D. This is the reason why China's software market, especially the OS segment has been dominated almost completely by foreign proprietary companies. Before Linux entered China's market, MS-Windows products from Microsoft were installed in over 95 percent of Chinese desktops; the high-end OS market was taken by foreign UNIX and MS-Windows as well. With the number of computers increasing rapidly and IT technology gradually penetrating into all sectors of the national economy by the mid 1990s, the Chinese government recognised the importance of self-controlled OS, not only to the indigenous software industry but also to the whole national economy.

The Chinese government showed great enthusiasm and interest for Linux since the wave of enthusiasm swept across the globe in 1998. In a traditional Chinese way, the Ministry of Information Industry and Ministry of Science and Technology organised four retreats regarding Linux future development in China during 1998-1999. Besides the policy makers, representatives from domestic software industry and academic fields were invited to participate. In contrast to their foreign peers, the uniform voice from Chinese academia was that Linux offered a powerful opportunity for China to develop a self-controlled mainstream OS, establishing a complete software industry, and further catching up in the global software technology race by the means of access to mature and sophisticated Linux source code.

6.2 Chinese Government Policies toward OSS

In 1999, indigenous Linux was formally listed on "The Guide for Current Preferential

Key Sectors of High-Tech Industrialisation” announced by both the State Development Planning Commission and the Ministry of Technology and Science. This was the first official policy on which the indigenous OS products, including Linux and its integrated application environment were listed explicitly (see chapter 5.2.1). Soon after that, the Ministry of Information Industry stated that Linux would be given government support. The official document and announcement from Chinese central government were the indicatives of government’s attitude to support Linux and the decision on developing a self-controlled OS based on Linux.

The State Council promulgated “Several Incentives for the Development of the Software Industry and IC Industry” (2000, No.18 Document) and “the Action Plan for the Rejuvenation of the Software Industry (2002-2005)” (2002, No. 47 Document) in 2000 and 2002 respectively. Both polices stressed that National R&D funding would strongly support key foundation software R&D activity. The OS was listed first in the catalogue of key foundation software.

The Ministry of Information Industry in addition stated that the development of Chinese OS products is the most important R&D task for China’s information industry in their issued “Outline of IT Industry in “10th Five-Year Plan” (2001-2005)”. Another crucial government agency related to indigenous OS R&D is the Ministry of Science and Technology. The Ministry, together with the National Development and Reform Commission, the Ministry of Commerce, the Ministry of Information Industry, and the National Standardisation Administration, issued “Implementation Notion on Further Improving the Capability of Technological Innovation for Domestic Software Enterprises” in 2004. The aim of this policy was to promote and improve the domestic capability of technological innovation in the fields of foundational software, such as OS and large scale database management system. According to this document, the “863 Programme” (National High-Tech Research and Development Programme of China) controlled by Ministry of Science and Technology will set up an independent Key Software Project to support foundational software R&D. In order to gradually establish an independent software industry, supporting application software for Linux was put on the agenda. The decision of supporting indigenous Chinese OS R&D was re-emphasised in “The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning (2006-2020)” which was released by the State Council. Even though Linux has not been mentioned clearly in all of above government policies, the reality is that Linux has obtained first priority in public procurement and state fund supporting. The important role of Linux in Chinese indigenous OS products was clearly specified in “High-Tech Industrialisation Programme in “11th Five-Year Plan” (2006-2010)” which promulgated by the National Development and Reform Commission in 2007.

According to the case study of Chinese Linux development, the role of government and the administrative measures to support indigenous Linux are changed in accordance with different stages of the Chinese Platform Linux development trajectory.

Time	Administrative Measures	Policies
The Pre-birth and Emergence of CS2C Linux (1999-2001)	Subsidy to Linux R&D	“The Guide for Current Preferential Key Sectors of High-Tech Industrialisation”
		“Several Incentives for the Development of the Software Industry and IC Industry”
		“Outline of IT Industry in “10th Five-Year Plan” (2001-2005) ”
	Public procurement	“Several Incentives for the Development of the Software Industry and IC Industry”
Early Development Stage (2002-2005)	Subsidy to Linux R&D	“the Action Plan for the Rejuvenation of the Software Industry (2002-2005)”

		“Implementation Notion on Further Improving the Capability of Technological Innovation for Domestic Software Enterprises”
	Public procurement	“the Action Plan for the Rejuvenation of the Software Industry (2002-2005)”
		“Law of the People's Republic of China on Public Procurement”
	Subsidy to adaptation	
	Standardisation	
Growth and Future (2006-Onward)	Strengthening software IP protection	“Relevant Issues about the Pre-sale installation of Official OS Software in Computers”
		“Issues on the Pre-sale Installation of Official OS Software in Computer and Office Device Purchased by Government”
	Linux education and training	“Notice about Establishing Linux Technology Training and Promoting Centre”
	Enhancing the compatibility with software and hardware	“The National Outline for Mid-Term and Long-Term Scientific and Technological

		Development Planning (2006-2020)”
	Subsidy to Linux R&D	“High-Tech Industrialisation Programme in “11th Five-Year Plan” (2006-2010)”
		“The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning(2006-2020)”
	Public procurement	“High-Tech Industrialisation Programme in “11th Five-Year Plan” (2006-2010)”
		“The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning(2006-2020)”
	Standardisation	“The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning(2006-2020)”

Table 8: Chinese government policies and administrative measures

6.2.1 Cultivating the domestic market

"Linux is superior to Windows technically, so that providing market incentive is the most important supportive measure for its success in China. The government therefore should purchase and apply indigenous software, including the indigenous Linux distributions."

-Prof. Guangnan Ni

Generating market by policies

For the segment of platform Linux, which includes Linux desktop version and Linux server version, the government is one of the most important customers[9]. In order to improve

[9] The government is the first user for Chinese platform Linux; government procurement accounted 55-60% for annual Volume of Sales of Chinese platform Linux during 2002-2005, and accounted around 35-40% for annual Volume of Sales of Chinese platform Linux during 2006-2009.

the efficiency of office activities and the decision making, all levels of government agencies trended to apply IT in their routine office tasks. The initial IT application in government agencies is OA, which is a general term including a wide range of applications of computer, communication and information technologies in office environments. It is the integration of office information functions, including word processing, data processing, graphics, desktop publishing and e-mail. Automation has changed not only the government office work environment, but also every concept and way of work, with the rapid development of network technology and increasing construction of information infrastructure. The meaning of automation therefore has expanded from internal office to external environments. The Chinese government began to focus on how to provide and improve government services, transactions and interactions with citizens, businesses, and other arms of government. China Telecom, SETC and information administrative departments in more than 40 government ministries and commissions initiated the Government On-Line Project in 1999. The market size of the government user was huge, with increasing annual investment.

Regarding the embedded Linux segment, the No.18 Document identified several issues for promoting the IC (Integrated Circuit) industry, e.g. IC design and manufacture firms will have a 14% discount during the period of 2000-2010, while the newly established IC enterprises would be exempted from income tax in the first two years and allowed a 50% reduction in the 3rd to 5th years. Furthermore, IC product R&D activities, such as product design and software design will obtain subsidy from the government. Instead of being a direct customer, the government has generated a huge potential market for embedded Linux by encouraging IC enterprise development.

Public procurement

The OS is characterised by network externalities so that a sufficient installation base is the pivotal factor for diffusion of OS (Bonaccorsi and Rossi, 2003; also see Bresnahan and Greenstein, 1999). Public procurement is a popular measure adopted by government in order to gain critical mass of users.

In accordance with No. 18 and No. 47 Documents, Chinese indigenous software, including indigenous Linux of course, has priority in public procurement.

"*The computing system which is invested in by government should adopted indigenous software system under the same performance/price ration.*"(No. 18 Document)

"*Regarding standard government software product and service procurement, indigenous software should have priority in public procurement.*" (No. 47 Document)

The "Law of the People's Republic of China on Public Procurement" passed by the Standing Committee of the National People's Congress in 2002, stated in the 10th clause:

"*Public procurement shall target domestic commodities, engineering works and services, except for a few exceptions.*"

Beijing municipality government was the first Chinese government agency to purchase 2,801 copies of indigenous Linux distribution, including COSIX Linux and Red-Flag Linux as their desktop OS in 2001. In the following two years, Beijing government not only continued to buy desktop Linux from indigenous firms, but also to gradually apply Linux to its various servers. Such an example is Pinggu District of Beijing, which has become a successful case in substituting currently-installed MS-Windows OS with indigenous Linux desktop distributions. There are 3,800 PCs out of 4,680 PCs in 188 agencies of Pinggu District which had installed indigenous Linux distribution up to 2005. Among them, 1,232 PCs had installed Linux and 2,568 PCs had installed dual OS, i.e. both Linux and MS-Windows (Wu, 2005).

Moreover, the ministries of Chinese central government began to purchase and apply indigenous Linux from the end of 1999. For instance, former State Economic & Trade Commission purchased hundreds of Red-Flag Linux for their OA system and PCs; CS2C Linux database server is adopted in the official document transmission system of Central Commission for Discipline Inspection of the CPC (Communist Party of China); CS2C Linux mail server won the contract from United Front Work Department of CPC Central Committee; and etc.

Following Beijing government and central government, all levels of local government started to purchase a large number of indigenous Linux copies for constructing their e-government project. There are hundreds of successful projects so far, for example, indigenous Linux has been applied in Guangdong Nanhai e-government project, Guangzhou Huangpu District e-government project, Zhongguancun Science Park e-government project, etc. In accordance with the market data from the largest Chinese IT consultant - CCID (China Centre for Information Industry Development) Consulting, the share of public procurement in China's domestic Linux market has always accounted for over 30%, and for at least 50% share in the Linux desktop segment since 2002, and China's domestic Linux market is significantly growing annually.

6.2.2 Subsidy

Subsidy for Linux R&D activity

"*The National Planning Commission, Ministry of Finance, Ministry of Science and Technology and Ministry of Information Industry should invest their technology development funds to foundational software R&D … The national funds for Science & Technology development will strongly support R&D activities of foundation and/or key software which included OS, database management system, information security system, embedded system …* " (No. 18 Document)

In order to guarantee sustaining financial support to software industry, No. 47 Document issued in 2002 declared:

"*A total of 4 billion RMB (£400 million) will be invested in the software industry during the period of 10th Five-Year Plan (2001-2005). The Information Technology Development Fund of Ministry of Information Industry, the 863 Programme, Key Technology R&D Programme, and The Innovation Fund of Medium or Small Science and Technology Enterprise of Ministry of Science and Technology, Industry Technology R&D Fund of National Planning Commission etc. will invest at least 3 billion RMB (£300 million) into software industry. The other 1 billion RMB (£100 million) will be provided by Chinese* *central government directly, from 2003 to 2005.*"

All the earliest versions of Chinese indigenous Linux distribution in fact were just foreign Linux distribution wrapped with a Chinese application package due to the lack of requirements of local users. From an application perspective, the Beijing government users found that Linux distributions purchased from CS&S and Red-Flag in the first procurement project were not mature enough for their OA requirement. There is a large difference between the operation styles of Linux and their previous installed OS products (e.g. MS-Windows). Furthermore, Linux had compatibility problems with existing installed applications software and hardware devices. Aiming to improve the performance and enhance the functionality of indigenous Linux to fit local government user's working environment, Beijing government initiated and funded the Sailing Project. The project greatly improved the usability and user-friendliness of Linux, as well as provided much better Chinese applications. In contrast to their original copies, the new versions of both Red-Flag Linux and CS&S Linux were evaluated as acceptable OS products for basic government office applications after the Sailing Project. This was the reason why the Sailing Project has become the most cited example of a successful Chinese government financial sponsored project.

With support from the 863 Key Software Project, both CS&S (CS2C) and Red-Flag completed the R&D projects on desktop OS in 2004. In light of user requirements and development experience accumulated from the past, the two Linux suppliers developed a windows-like GUI, improved the Chinese application performance, modified the adaptability to USB external devices and finally delivered their newest desktop Linux (COSIX Linux v4.2 and Red-Flag Linux Desktop v4.0) in the same year. In accordance with the "The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning (2006-2020)", the 863 Key Software Project was substituted by the "Core (core electronic component), High (high-end generic chip, like CPU), Foundation (foundation software, like operating system) Key Project" in 2008. As Prof. Guangnan Ni pointed out:

“*The strongest ever financial support to foundation software will be provided by the Core, High, Foundation Key Project. The expected official investment in foundation software R&D and application could reach 1 billion RMB (£100 million) every calendar year. Both CS2C and Red-Flag were the only two OS providers listed in the Core, High, Foundation Key Project.*”

Subsidy for Linux adoption

There is another type of subsidy which the government funds have paid to consumers for adopting indigenous Linux. As government users became the first group of consumers of indigenous Linux, some different levels of government, in particular the government in the

Chinese western poverty area[10], received subsidies to pay partial or all procurement expenditures from central government funds.

Knowledge and information is the most important resource for economic development nowadays. The production, transmission and acquisition of knowledge and information are significant to regions and countries (Hu, 2002). With the spread of the computer, the Internet is becoming the rapidest and most effective way to acquire and communicate information. The term ‘digital divide’ has become one of the most essential issues in the last 10 years, referring to the gap between people with effective access to digital and information technology and those with very limited or no access at all. Lack of financial budget is one of the reasons leading to the digital divide (Hu, Zhou, 2002). Furthermore, low cost is the issue mentioned by advocates of OSS so that applying OSS is a remedy for

[10] Most of Chinese mountain regions are in Western China (about 64.8% of the total mountain and 56.2% of the hilly land are in western region, as well as 72.8% of the western region is covered by the mountain and hilly land), which is an obstacle for developing local industry and agriculture in Western provinces of China. There is 62% of the total poor people are in Chinese western region, and this percentage kept increasing according to National Bureau of Statistics of China.

bridging the international and domestic digital divide without breaking intellectual property rights (Camara, and Fonseca, 2007; James, 2003; Ksheteri, 2004; May, 2006; Salvador, Sherry and Urrutia, 2005).

To bridge the Chinese domestic digital divide, the Ministry of Science, Ministry of Education, Ministry of Information Industry, Ministry of Agriculture, and other ministries and departments of central government launched the “Narrowing Digital Divide-Western Mission Project” in 2002. The aim of it is to develop appropriate IT products and apply them in the western provinces according to the local economic environment. Taking the depressed economy condition of the western provinces into account, the central government decided to provide financial support for this project through the 863 Programme. With the financial aid, Linux NCs (Network Computers) have been applied in nearly all the county level governments in the western region, including the Tibet district.

The fact that Linux has the third largest share of the global server market shows that it is competent for use in all kinds of server applications. Nevertheless, Linux was only installed on secondary systems in China, although a wide range of industry users have already applied it before. The reason that Chinese organisational/industrial users were unwilling to deploy Linux in their core business systems is that there was not any successful demonstration. As Dr. Ge Wang, researcher in CASTED indicated:

“*Even though the indigenous Linux has been evaluated as a qualified OS for servers, there are still no Chinese enterprises adopting it in core business systems. As a result, some state R&D funds have to supply financial aid to encourage Chinese users.*”

In order to build the confidence of applying Linux on core business system, the 863 programme has been sponsoring a demonstration project in the financial industry since 2003. The whole project is divided into two main stages. The main organisation to undertake the first stage was CCBSB (China Construction Bank Shangxi Branch) which planned to switch the OS on all the counter terminals in the branches of Shanxi province, from SCO UNIX to CS2C Linux Server. The first stage took 15 months to complete this switch: the OS installed on counter servers in over 700 local branches of CCBSB switched from SCO UNIX to

CS2C server Linux. According to data from CCBSB, the efficiency of business and transaction process increased 20% when adopting CS2C Linux and the estimated operation costs of CCBSB in 2005 was reduced by 50 million RMB (£5 million), which led to 200 million RMB (£20 million) profit increase compared to 2004. Inspired by the data from CCBSB, the CCB headquarters made a decision to promote CS2C Linux to a wider spread in the whole CCB system. In the subsequent year, two province level branches, CCBHB (China Construction Bank Hainan Branch) and CCBSB (China Construction Bank Shanxi Branch), as well as a city level branch, CCBQB (China Construction Bank Qingdao Branch), adopted CS2C Linux Server in their counter systems.

To provide better customer service as well as to reduce the maintenance cost, the CCB headquarters decided to conduct the second stage – DCC (Data Centre Consolidation) project at the end of 2004, planning to reconstruct the integrated front-banking system. A huge Linux installed server cluster at province level branch was substituted for the distributed front-banking servers in local branches. The aim of the second stage was "developing indigenous platform Linux-based solutions for the financial field". The result of conducting the DCC project would greatly enhance the capability of transaction data processes and provide a more rapid service for CCB customers. The 863 Programme granted another 40 million RMB (£4 million) to support the DCC project. The experimental DCC system was first installed in CCBLB (China Construction Bank Liaoning Branch) in September 2007. By the end of the same year, the new system had been ported to the CCBAB (China Construction Bank Anhui Branch), CCBQB (China Construction Bank Qinhai Branch) and CCBHB (China Construction Bank Henan Branch). The new system was gradually employed in the 38 province level branches of CCB from 2008 - 2010.

The demonstration effect of CCB projects finally reached the expectation of the Chinese government. Relaying the successful experience in the financial area, CS2C obtained contracts from China Universal Asset Management and Shanghai Stock Exchange in 2007 and 2008 respectively.

6.2.3 Enhancing the compatibility with software and hardware

The spread of a given OS is subject to the quantity and the quality of complementary applications, as well as the variety of hardware devices with which it is compatible. The economic term for this refers to indirect network externalities effect (Katz and Shapiro, 1985). In order to make indigenous Linux distribution usable for Chinese users, in particular office users, the Beijing Science & Technology Commission granted the Sailing Project. In phase I of the Sailing Project, the compatibility of indigenous Linux with hardware, such as printer devices, were greatly enhanced. The main task of the Sailing Project phase II was to solve the compatibility problem between indigenous office suite and MS-Office suite, and further optimise the performance of indigenous office suites, such as Red Office and WPS for indigenous Linux. The government realised that the application software is a pivotal factor for the spread of indigenous Linux so that the indigenous office suite was listed as foundational software soon after the Sailing Project. Take Red Office suite for example, it received a great deal of R&D funds from both central government and Beijing government. Its compatibility to indigenous Linux has always been stressed. As the CEO of Red Office pointed out:

"*In line with the software public procurement policy, both indigenous Linux and Red Office won a lot of government contracts. The compatibility of Red Office Suite to indigenous Linux therefore became an important evaluation criterion for all the 11 national and local R&D projects undertaken by our company from the 2001 (Sailing Project).* "

Before 2005, government investment was focused on office applications only. There is a gap between indigenous Linux and other complementary software and hardware.

"*The indigenous Linux should have link to not only office suite, but also other indigenous software because user groups other than just office users should be taken into account.*"

-Prof. Guangnan Ni

“*The low compatibility of indigenous Linux to application software will require users to spend much time searching appropriate application software for their Linux. If the Linux suppliers could recommend and/or provide solutions for users directly, more users may be attracted by indigenous Linux.*”

-Mrs. Ye Xu, researcher in CASTED

“*There are lots of Chinese computer manufacturers who wanted to choose Linux as their pre-installed OS due to the lower cost. However, most of them abandoned their original plan because of hardware adaptability problems.*”

-Dr. Ge Wang, researcher in CASTED

The Ministry of Science and Technology approved granting a key project of “10th Five-Year Plan” State Key Technologies R&D Programme “Indigenous Foundational Software Platform System Development and Application” in July, 2005. The goal of this project was to construct an open architecture based indigenous foundational software platform system by drawing upon the R&D results of past state technologies R&D programmes and the 863 Programme. The whole project was divided into 15 sub-projects, which were dedicated to the fields of development strategy, standard and testing, intellectual property, key technology, application demonstration, education and training, and so forth.

In light of market characteristics and users demands, CS2C cooperated with other software enterprises, as well as universities to undertake one of the sub-projects, “Research on Key Technology and Integrating Technology of Indigenous Foundational Software Platform” (under grant 2005BA112A02) as a leading role.

Three major tasks have been achieved in this project. Firstly, CS2C conducted R&D activity in hardware adaptability. After this, indigenous Linux distributions were compatible with 204 computer servers, 96 PCs, 79 printers, 7 scanners, 34 USB storage devices and 4 NCs. Secondly, CS2C carried out a series of R&D projects to improve the software compatibility of indigenous Linux, in particular with indigenous database software, mid-ware, and office productivity suites. The last task was integrating foundation software platforms for several defined application areas.

This project developed seven solutions which consisted of indigenous Linux desktop and server versions, indigenous branded Server and NC products, and indigenous complementary software. The solutions passed strict testing and achieved high reliability, performance, and functionality. The software interpretability which is long been a plague for Linux was solved effectively as well, which meant that all the components of the solution were interchangeable. This enables customers to pick a solution for an application system directly according to their working environment, and they no longer have to solve software compatibility and hardware adaptability problems. The seven solutions based on indigenous foundational software have been deployed successfully in the fields of e-government, agriculture, education and medical care and health.

6.2.4 Standardisation

Linux distributions differ in accordance to the specific intention of developers, though they are based on the same kernel. Some focus on security and stability, and others focus on the ease of use and friendliness.

As we know, the cost of switching to Linux from proprietary OS consisted of not only pecuniary costs but also user learning costs. Sometimes the latter is much higher than the former, particularly when the users are familiar with the GUI style of proprietary systems and the way to operate them. A problem arose in China since the GUIs, as well as Chinese display and input system of various indigenous Linux distributions, are not completely the same, which definitely increased the learning costs for end-users.

Today, users normally customise a variety of standard components into a solution to meet their specified working methods and requirements. The configurational thus is one of the salient characteristics of contemporary intricate technology, especially for ICT. Another dangerous problem is that indigenous Linux distributions are incompatible with each other. For instance, in order to attract more consumers, LENOVO developed special software named “LENOVO Image Studio”, with which the user is able to print pictures or documents with special display effects. In contrast to Red-Flag Linux which is able to invoke the software and its function completely; CS2C Linux, another popular indigenous system, can only implement the basic printing function. Consequently, a uniform standard for interoperability between all Chinese Linux distributions has become an essential issue (Ni, 2005).

At present, all the Chinese Linux vendors are SMEs with a weak position in the software market. Consequently, a *de facto* industrial standard is unlikely to form through the market competition mechanism in China. In order to regulate market orders, the Ministry of Information Industry originated an official Linux standard group to conduct the Linux standardisation project in 2004, which was financial supported by the 863 Programme several months later. The members of the group came from government agencies, academia, and industry. They were CESI (China Electronic Standardisation Institute) of Ministry of Information Industry, ISCAS, Red-Flag, CS&S, SUNWAH Hi-Tech, etc. (Gao and Xie, 2004). The main task of the group was to develop *de jure* standard through which to ensure the consistency in operation, development and API (Application Programming Interface) of indigenous Linux. The draft of "Linux Application Programming Interface (API) Standard", "Linux Desktop OS Technology Standard", "Linux Server OS Technology Standard", "Linux User Interface Standard" and "Embedded Linux Technology Standard" was submitted in the beginning of 2005.

6.2.5 Strengthening the software IP (Intellectual Property) protection

The higher license fee of proprietary OS is not a significant issue for China owing to rampant piracy. Ironically, the dominant position of foreign proprietary OS in China's software market is benefited by the spread of illegal copies. Jeff Raikes, who was Microsoft Business Group President, announced in 2007,

"*If you're going to be a software counterfeiter, then please copy and illegally use Microsoft products.*"

In order to create a good environment of software intellectual property protection, to maintain the order of the computer market and software market, to impel the independent innovation of software, and promote the sound, continuous and speedy development of the software industry of China; the Ministry of Information Industry, National Copyright Administration, Ministry of Commerce and Ministry of Finance announced the “Relevant Issues about the Pre-sale installation of Official OS Software in Computers” in April 2006, the symbol of the starting point of China’s software copyright movement. The notice regulated that all the computers manufactured within the territory of China must be installed with an official OS at the time of ex-factory. After four days, the “Issues on the Pre-sale Installation of Official OS Software in Computer and Office Device Purchased by Government” was issued by Ministry of Information Industry, National Copyright Administration, Ministry of Finance and Government Offices Administration of the State Council. The two mandatory policies have been also treated as the opportunity for promoting the deployment and spread of Linux in China.

The price of foreign proprietary OS is a significant proportion of a computer’s total cost, and the proportion has been gradually increased in recent year. The cost of indigenous Linux distributions is about 5 Yuan RMB (£0.5), which is far cheaper than the popular MS-Windows series OS products even purchased in bulk. Trying to bring down the PC price, some PC manufacturers accordingly choose indigenous Linux as the preinstalled OS thanks to no license fee. As Dr. Ge Wang in CASTED pointed out:

“*In the past, some indigenous PC manufacturers which focused on the domestic low-end market didn’t provide OS for their products because of the cost pressure. As a result, their customers generally installed illegal copies of Windows OS on their PCs. With the two compulsory policies, one the one hand the software piracy phenomenon could be prevented efficiently; on the other hand, these indigenous PC manufacturers are inclined to choose indigenous Linux distribution as their pre-install OS since the cost of Linux could be ignored finally.*”

Millions of indigenous Linux copies were installed on branded PCs in 2006. For example, all 23 types of DELL desktop and notebook pre-installed Red-Flag Linux. Another

Chinese mainstream Linux provider, CS2C, signed several OEM contracts with PC manufactures, like Founder, Great Wall, etc.

6.2.6 Linux education and training

To entrench its competitive position in China, Microsoft became vigorously involved in China’s educational projects. In the June of 2002, Microsoft established official linkage with top universities in China by launching the “Great Wall Plan”[11]. Microsoft played an important role in the curricula design, course teaching, and textbook compilation in the software schools of the top Chinese universities. Microsoft Research Asia, Microsoft's fundamental research arm in the Asia Pacific region, provides financial support for the construction of at least five computer fundamental courses, and/or Microsoft core and new

[11] According to MOU signed between The State Development & Planning Commission of the Peoples Republic of China and Microsoft, Microsoft promised to invest at least 200 million RMB (£20 million) to support Chinese software education system during 2003-2005.

technologies courses.

By contrast, Linux knowledge and expertise had been ignored in the Chinese educational system before 2005. The core contents in computer courses from primary schools, high schools to universities (except Computer Science and IT related majors) were all directly based on Windows systems or in Windows running environment (Zhang, Sun and Guan, 2006). For non-computer science major students, they were locked in the MS-Windows OS products and complementary applications. It was believed to be a significant barrier for the spread of Linux to unsophisticated users in the mass market. According to Prof. Guangnan Ni:

"*The syllabus in China's primary and middle school clearly requires that students must learn operating MS-Windows and MS-Office. As a result, the following computing education in universities had to be based on Microsoft products.*"

In addition, Linux expertise is generally listed as an optional course, rather than a compulsory course, for Computer Science and IT related majors in most Chinese universities. There is a lack of programmers who mastered core technologies and architecture of Linux (Zou, 2004). In accordance with gradually increasing demand for Linux systems, analysts estimated that the China market will be short of millions of Linux specialists and graduates in the next five years (2008-2013). In facing the disadvantages of backwardness of Linux development, solving the problem of Linux in the national educational system and professional training therefore become a pressing task for the Chinese government (Zhang, 2006).

"*The China's educational system ought to serve national innovation system. On the one hand it should cultivate users or potential users for indigenous Linux, which has been received a large quantity of state R&D supports; and on the other hand, it should provide adequate human resource for indigenous Linux innovation.*"

-Dr. Ge Wang

In July 2005, Ministry of Education and Ministry of Science and Technology issued the "Notice about Establishing Linux Technology Training and Promoting Centre". The document stated that the a National Linux Technology Training and Promoting Centre will be established in 40 key universities and the open source software content is required to be emphasised during IT education. Two important issues were identified: one is to provide short-term part-time training programmes for software engineers, and the other is to teach Linux expertise to college students at undergraduate and Master level. There is a trend that Linux expertise has gradually become a compulsory course, while proprietary software based technology has become the optional course. The ‘Zhejiang Province Model’ is the most successful example in Linux education. The Education Department of Zhejiang province helped Linux enthusiasts in local universities to establish LUPA (Leadership of Open Source University Promotion Alliance), which is the first Linux community initiated by college students in China. Besides providing different levels of Linux related modules to the college students no matter what major they are studied, Linux was listed as the content in NCRE (National Computer Rank Examination), which is one of the most useful qualifications for a starting career in all areas.

The closed nature of proprietary software limited the students to understand the mechanism and thinking behind the code. As a result, proprietary software based university education and training programmes will only strengthen the dependency on certain technology provided by firms. Introducing Linux expertise to the university education system provides updated code and research objects for the student majoring in a computer science related area, and further cultivates the capability of software innovation.

6.3 Negative Effects of Government Policies

6.3.1 Public procurement

There are two limitations on public procurement. Firstly, the policy maker lacked experience in issuing policies. The Ministry of Finance and Ministry of Information Industry issued “Implementation Measures for Public procurement of Software (Draft Version)” in 2005. The original aim was to regulate the public procurement behaviour. The software in the Chinese market was grouped into three categories in this official document: indigenous software, non-indigenous software with priority in procurement, and non-indigenous software. However, the document did not state the difference between the former two categories in procurement so this problem arose during the policy implementation. As the general manager of CS2C stated:

"*The document should state implicitly that the indigenous software should have first priority in public procurement if the performance of indigenous software reached the requirements of government users.*"

Prof. Guangnan Ni had more radical expression in this case:

"*This document broke the Law of the People's Republic of China on Public procurement since the law only defines two categories: indigenous and non-indigenous software. In all other countries, furthermore, there is no the term of 'non-indigenous software with priority.'*"

Several months later in the same year, the "Implementation Measures for Public procurement of Software (Draft Version)" was recalled.

Secondly, it is difficult to evaluate this measure practically because there is no official statistic data on the proportion of indigenous Linux in the whole OS for public procurement. While the foreign proprietary product still won lots of government contracts due to the lack of a solid system to monitor its implementation in public procurement. For example, the Beijing government who was the first user of indigenous Linux purchased MS-Windows XP in 2004.

6.3.2 Standardization

Two main problems existed on both membership constitution and the process of teamwork standardization. Firstly, there was no foreign complementary software company among the twenty six formal members. In light of the status quo of China's software industry and Linux development, Chinese users inevitably adopted foreign complementary software. As a result, the foreign providers of mainstream complementary software, like database providers, mid-ware providers should have been invited to participate in Linux standardization. The second problem is that the user group was excluded from the Linux standardization. Take embedded Linux standardization for example, the embedded system is a system designed for specific applications. There are thousands of technical differences resulting from the real requirements of function and consideration of cost control even amongst users from within the same industry. It is impossible to develop a sensible standard for embedded Linux merely based on the developing experience of Linux companies and some complementary software firms since these participants cannot take all the user requirements and application conditions into account. The embedded Linux standard should be established for various specific areas, such as for Set Top Box, lottery terminal, numeral control, etc. This can explain why there was no any feedback on the five drafts of the Linux standard submitted in 2005. The Core, High, Foundational Key Project did provide further financial support for the indigenous Linux standardization from 2009.

6.4 Summary

The western mainstream OSS literature held the opinion that the governments should limit their roles in supporting OSS over the traditional proprietary software (Evans, 2000; Schmidt and Schntizer, 2003; Comio and Manent, 2005). However, my research counters the dominant view of the established literature on OSS, which is that government support is an efficient way to facilitate the spread of infrastructural OSS – Linux. In China’s social context, especially at the time when Linux was first introduced, government support was extremely important. For example, the government provided up-front financial support to the Chinese Linux R&D in the earliest stage of the development, became the first user for indigenous Linux by the means of public procurement and played a role as match maker between R&D and industries and between industries for setting standards. With the government support, Linux has been adapted into Chinese environment and further developed to a great extent. The companies, such as Red-Flag and CS2S, have not only survived and but also flourished.

Chapter 7 Analysis

7.1 Introduction

The primary concern of this research is to understand the open source movement and OSS adaptation in DCs. Even though a large body of literature on open source movement and OSS has been published during the last decade, the majority stems from the context of the developed world, in particular the United States. The research objective of existing literature is the moral principles and ideology of OSS, including the emergence of OSS, the incentives of contributors, the governance structures of OSS communities, OSS licenses etc. This research on the social discourse of OSS provides a good understanding of the economic and political rationales embodied behind the open source movement in the developing world: the loosely connected OSS communities with their efficient governance structure; the strong support of companies which competed with the monopoly of proprietary software; the powerful intelligence and manpower resources sustaining OSS development and innovation. However, empirical investigation of OSS in DCs is lacking; the existing literature focuses mainly on the benefits acquired by DCs – as if the adaptation of OSS was taken for granted.

The aim of this chapter is to analyse the empirical research findings presented in chapters 4-6, link them to the theoretical framework and throw light on the central concern of this study. This research gradually unfolds the evolutional process of Chinese indigenous Linux and examines the implications for other DCs. My findings are in agreement with theories of STS, especially from a SST perspective, that Linux has to be adapted to the socio-technical network and the broader social and economic context in China.

Section 7.2 describes the biographies of Chinese embedded and platform Linux by discussing the development strategies adopted by two Chinese Linux providers, the social learning process in every evolutional stage, as well as the influence of the broader context in which Chinese Linux is designed and used. The biographies reveal that the adaptation of software like Linux is a process of 'generification'. Unlike the existing international technology transfer literature mainly addressing manufacturing and industrial technologies, understanding, categorising users/intermediaries and enrolling them into software design are crucial elements in local technological capabilities for software transfer.

Section 7.3 analyses the government role in Chinese Linux adaptation. The adaptation and further development of Linux as an infrastructural OSS required strong government intervention because only the government is able to bridge the gaps between the key actors and also coordinate their fragmented resources.

7.2 Biographies of the Chinese Linux

This section discusses how social elements shape the development of Chinese Linux, which resulted from different strategies adopted by the two Chinese Linux providers in both the international and domestic socio-technical context where they operated. From a SST perspective, this research looks into the interactions between different key players in the specific context of China and provides an analysis of how the two biographies of Chinese embedded Linux and platform Linux have been derived from suppliers' operations and strategies.

7.2.1 The biography of Chinese embedded Linux

According to the technical features, an embedded system normally has rigid physical constraints on system size, power consumption, computational capability, real-time performance, dedicated functionality, hardware framework, etc. In contrast to large-scale software packages which can be designed for sectors (such as ERP), embedded systems and their OS must be coupled tightly with specific equipments or products. They are normally catered for specific local contexts and work practices as customised software (packages). Each embedded OS, e.g. embedded Linux, is unlikely to be ported to another product without considerable effort and expense. Different products, and even the same product with a tiny difference in functionality or hardware component, may lead to the redesign and redevelopment of OS from scratch.

As shown by the empirical data we investigated in Chapter 4, Red-Flag explored its own ways to develop a 'generic' solution for embedded Linux-based system development. This solution enabled embedded product suppliers to tailor Linux based on their own needs and local settings. In order to design embedded Linux efficiently, Red-Flag always faced a tension between the need to cater for local user requirements and the need to develop global solutions driven by economic and financial performance. By presenting the qualitative evolutional story of Red-Flag embedded Linux, this study examines how the supplier tackled this tension. Figure 4 was developed to illustrate the biography of the Red-Flag embedded Linux system.

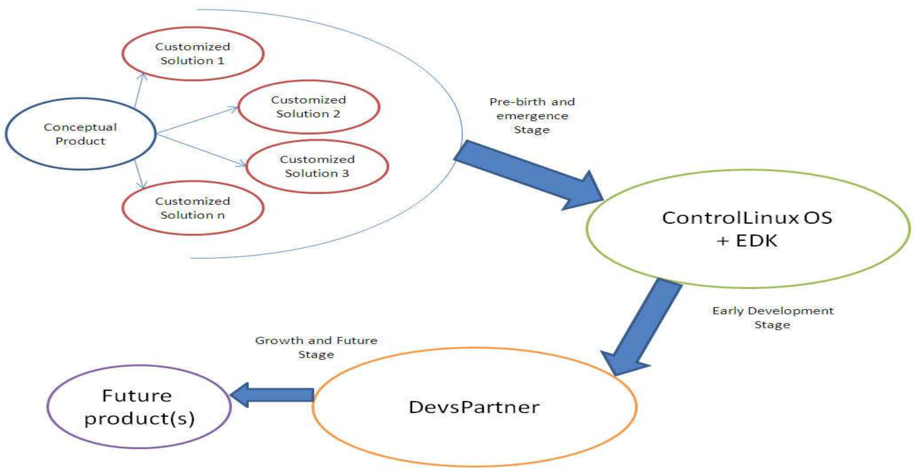

Figure 4: The biography of embedded Linux system

Drawing upon the BoA theory, we have traced the evolutional history of embedded Linux for a decade (1998-2008). By revealing the complex interplays between the technology supplier and other social groups, we have opened the black-boxes of both longitudinal history and particular moments or projects.

This study has investigated three biographic stages of Red-Flag embedded system: pre-birth and emergence, early development and growth and future stage. According to Figure 4, Red-Flag embedded Linux emerged from a conceptual product with no specific features to match any existing embedded product and equipment. It gradually grew to a 'genuine' embedded Linux-based system in order to meet some local needs. In the earliest stage, the knowledge of functionalities' development had been acquired by the supplier by catering for all local user requirements. Once Red-Flag embedded Linux further evolved to the "early development" stage, the supplier focused on embedded Linux provision and also developed EDK for users. The final evolutional stage of Red-Flag embedded Linux was defined by delivering a generic embedded system development solution. This generic solution is not an embedded system or OS which could be applied in all or a given sector or niche, rather it is a software package that enables the user to tailor embedded Linux and develop functions for embedded products.

Dynamics and Tensions

From the biography of Red-Flag embedded Linux, we find that its evolutional history reflected the process of generification – developing a generic solution. Thus, the supplier always faced tensions from developing generic solutions for various local contexts. As a result, Red-Flag had to deploy various development strategies to manage these tensions and make trade-offs between technical, social, and economic requirements.

Unlike designing large-scale organisational software packages, the supplier must not only concern the functionalities of the embedded system (e.g. the way the user operates the embedded system), but also take the embedded hardware into account, in order to design and development optimal compact-sized OS without redundancy. The traditional way to develop an embedded OS is that the suppliers (or organisational users) with strong R&D capabilities must design the OS for every individual embedded product or equipment. Reuse of the visible resource, like source code was no longer considered of high importance; whereas the invisible resource, for instance the tacit knowledge in designing and developing experience, became crucial for future projects. Due to the embedded systems being widely adopted in various working fields and everyday life, new functions were required to be added gradually to equipments or products. However, unlike the traditional computing (software) system that is capable of introducing new functionality by simply applying software packages without modifying the OS (like installing an ERP package to computer servers), embedded system generally requires redevelopment of the OS because existing embedded OS might not contain the necessary programmes and components for supporting a new application software.

By linking the empirical case of Red-Flag embedded Linux presented in Chapter 4, the supplier adopted two main development strategies in light of the technical and economic problems arising from the biography of embedded Linux. The early embedded Linux development strategy caters for all requirements of the individual user. Adopting this strategy could help Red-Flag to solve finance shortage problems, acquire domain knowledge of different applying fields and establish a convincing reference site in the field of embedded software. The later development strategy is that Red-Flag developed selected users' particular requirements only if these requirements were regarded as particularly generic requirements – having market potential (i.e. can be applied across application fields and areas) and being able to contribute to a generic package – DevsPartner in further development.

Although these two development strategies are similar to the process of generification revealed by the study of complex large-scale enterprise software package (Pollock et al., 2003), in practice this decision-making process was not smooth: Red-Flag had to consider technical, financial, economic and social factors and make a compromise between them. In order to solve the tension between developing a generic solution and catering for the local context, Red-Flag also adopted two different strategies. However, these two strategies focused on developing an experience rather than functionality, as would be in the case of complex large-scale package. The early strategy attempted to abstract generic knowledge in designing and developing embedded Linux-based system from the past customised projects. However, the supplier found that there were too many options in choosing hardware component and detailed functionality for customers' embedded products design. Any tiny differences between embedded products would lead to an enormous technical undertaking. The subsequent strategy of generification was to deliver a generic embedded system development suite – DevsPartner. By offering a training programme, small- and medium-sized clients are able to complete all the steps of developing a whole embedded system which included tailoring embedded Linux and developing application packages.

7.2.2 The biography of Chinese platform Linux

Platform Linux is a general-purpose OS which leaves great scope for further adaptation into new contexts from a technical point of view: it is supposed to support all of the mainstream software applications and hardware, while the physical constraints in system size, power consumption, CPU processing capability and so on are less stringent compared with embedded Linux. In this sense, platform Linux is originally designed and developed as a generic package for all kinds of users and local contexts.

Based on the case elaborated in Chapter 5, the main challenges for Chinese platform Linux suppliers are how to bridge the gap between the original Linux with generic technical features and the particular requirements of local users. This study took one of the famous Chinese Linux suppliers – CS2C as an example, and illustrated the way CS2C designed globally developed standard Linux to fit the Chinese local context and specific niches. In order to give a clear picture of the Chinese platform Linux evolutional process, Figure 5 was developed to show the biography of Chinese platform Linux.

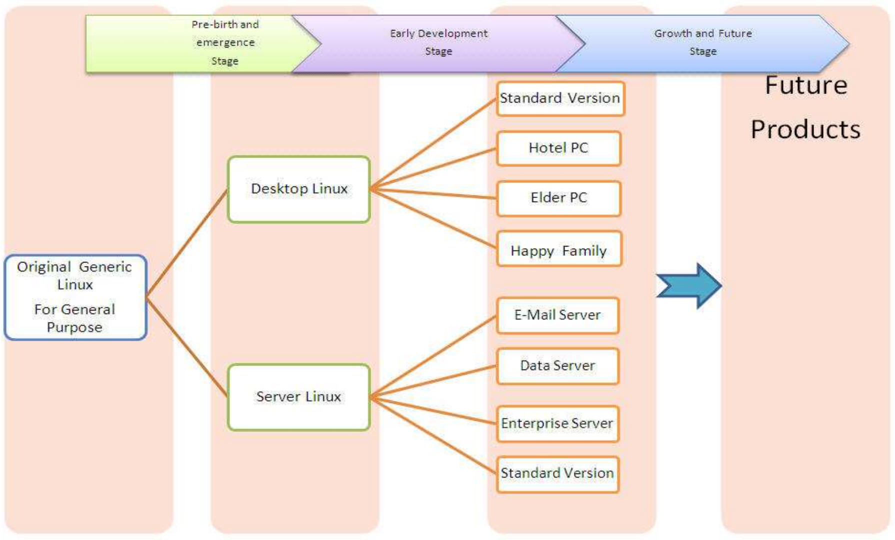

Figure 5: The biography of platform Linux

As shown in the diagram above, the Chinese platform Linux originated from an international distribution as a generic OS. It soon evolved into an early development stage, in which a general version was forked and delivered as desktop version and server version in light of mainstream application areas. The last stage investigated in this study is characterised by further differentiation, by which CS2C could cater desktop Linux and server Linux for different (personal) user communities and server application areas respectively.

Dynamics and Tensions

In order to provide explicit understanding of the dynamics and tensions which are rife in the process of Chinese platform Linux adaptation and innovation, this study developed the term ‘infrastructural software’. The literature review in 2.3.2 explains this term by linking the concepts and theories of infrastructure studies.

The infrastructure developers cannot entirely eliminate local diversity in work practice, but have to change their technology which was designed according to the original assumptions on work and use practice. To the term of infrastructural software, such as OS, the local work practice is the way the user operates the OS, as well as the way the user implements functions through the application software packages installed on the OS. In this premise, a successful OS must manage very well the user requirements in operating habit and interoperability between OS and application packages.

The Chinese platform Linux started from an assumption that the local context in China, where Linux is expected to be embodied and used, is the same as in developed countries, in particular the US and Western EU. Like all other Chinese Linux developers, CS2C simply translated a foreign Linux distribution into the Chinese language and released it to local mass market in the earliest development stage. However, this strategy was problematic due to the fact that China’s reality behind this assumption was far more complicated. OSS, including Linux, are designed from a ‘one world fits all’ view, the open access strategy allowing collective and collaborative knowledge acquisition, integration and achieving standards (Metcalfe, 2007; Feller *et al.*, 2007; Ure *et al.*, 2009). There were nearly no Chinese mass users, and only very few Chinese software engineers and scientists involved in the international Linux community, so the supplier had no means of acquiring the knowledge of local user practices. With the Chinese government becoming the first user group for Chinese platform Linux, a tension, thereby, was exposed between the standardised work practice represented by globally designed Linux and China's local-specific aim and practice.

The local practice is largely influenced by a common dynamic of infrastructure development – path-dependency effect. Path dependence (Arthur 1988) in technological change means that past events have a large impact on future development. Switching to Linux from the existing OS may involve various economic and political issues.

There are two forms of path-dependency related to use and supply ends. The user end path-dependency creates a user locked-in effect where a system or standard will be more attractive when it has built up a sufficient installed base before other competitive products do. From the perspective of economic studies, this phenomenon is referred to direct externalities, which elaborates the relationship between current user and future adoptions. Users can obtain more benefits from using a software product with an increasing number of other users purchasing and using the same software. Generally speaking, the larger the installed base the system or standard has, the higher switching cost the users pays. Apart from user habits in using the OS, path-dependency also strongly applies to the applications running on Linux: the layout of menus and the mode of operations of compatible applications are sometimes very different from applications running on MS-Windows; for example, once users are familiar with MS-Office suite which is only compatible with MS-Windows family, they will be reluctant to adopt other office suites to process routine work. In this sense, the user habits on application software also entrench the existing user deeply with MS-Windows. In making a shift to Linux, users have faced extreme difficulties due to the deep path-dependency effects. To PC users, the locked-in effect is reflected by the user's habits in using a given OS and operating the application software running on this OS.

To business or industrial users, switching to Linux implies pecuniary expenditure not only for purchasing a service from Linux providers but also for porting local specified applications to a Linux environment. The local specific applications were generally customised and developed for specific organisation tasks rather than being standard packages, e.g. the official document printing for government users. Because of the dominant position of proprietary OS products in China's software market, particularly MS-Windows, most of these customised applications were originally developed to be implemented in a MS-Windows environment. The reality is that not all existing customised application software could be modified to be compatible with Linux because some developers of these applications may not still be in business.

Another type of path-dependency effect shown on the supply end is that the future design decisions of technology are largely influenced by the decisions in the early design phase. In the case of Chinese platform Linux evolution, this form of path-dependency is heavily dependant on the development of complementary products. Like other OS products, Linux itself can be treated as a gateway to assemble heterogeneous devices and software packages together with the specific purpose computer. In order to integrate these technical components together, seamless interoperability is a sticking point. It is also known as the indirect network externalities, in economic terms. In order to integrate the technical heterogeneous components, the OS interacts with the application software and hardware devices via the API (Application Programming Interface) and drivers respectively. However, the incompatibility between different OS products, like Linux and MS-Windows having different APIs, gives rise to additional costs for the complementary product providers who are willing to develop products for both OS products. The complementary application software is a prominent example since it is directly in relation to the user application. The dominant market position of MS-Windows in China's market generates the *de facto* industry standards on which the quality and quantity of local Chinese application software for MS-Windows are based, to a much greater extent than for Linux. Developing complementary application software for Linux, which only accounts for a small market share, means that the providers will have difficulty being commercially successful. Only when the new-coming Linux accumulates a sufficient installed base/critical mass of users will the complementary product providers be willing to provide compatibility to it. The existence of complementary products in this sense, *vice versa*, is the key for achieving a critical mass of users.

In order to deal with the dynamics and tensions arising during the platform Linux adaptation and innovation, CS2C employed two distinct development strategies over time. The first strategy attempted to enrol local users in software design and collect their preference on Linux, due to the fact that there were no Chinese users in the mass market except a few enthusiasts in universities, who deployed Linux for academic research or for 'fun'. With more user groups participating in the design, CS2C developed platform Linux for all user requirements in order to enrich the features of functionality and easy-operability so that CS2C platform Linux could compete with prevalent MS-Windows in China's mass market. The later development strategy was 'Standard and Local Variants'. To the desktop OS market segment, CS2C sought to match the niche markets that were overlooked by foreign OS providers. In contrast to the personal users in the mass market, who had been locked-in by MS-Windows, personal users in niche markets normally had particular and specific requirements which required customised services and whole software solutions encompassing both Linux and basic application software. Apart from these Linux-based software solutions, CS2C also provided standard desktop Linux for Chinese mass users. To the server OS market segment, CS2C delivered several functional server Linux on grounds of the dedicated computer server application areas (with special optimisation tools to guarantee compatibility with mainstream server application packages) and also supplied standard server Linux for general server application. Adopting this strategy helped capture universal elements learned from both international communities and competitors and domestic users. On the one hand, CS2C therefore could design standard Linux to compete with other commercial OS in China's mass market by integrating updated core technical elements which have evolved and changed over time (i.e. new components and algorithms are frequently introduced to the Linux kernel by the international Linux community). On the other hand, it matched the purpose-specific work and use practice, generating profits for sustaining company operation by means of offering customised service.

It is noteworthy that the case study of Chinese platform Linux painted a picture in which software also experienced a process of generification. Unlike many generic software packages that have been studied (like ERP which were originated from an isolated system implemented in a single site, extended to multiple sites, and further travelled to other sectors or segments), the generification process of Chinese platform Linux represents a reversed orientation: it archived generification by differentiation. The original Linux is supposed to be a type of generic solution for all sites of computer and IT applications. By acknowledging the characteristics of Chinese local users and uses in order to compete with foreign OS products, CS2C designed and developed platform Linux as several sub-generic solutions. Some functions that catered for specific-purpose uses and practices were modularised, as well as reused in the next version of products.

7.2.3 Role of the user in Linux adaptation and further development

As discussed in 2.4.1, existing social studies of OSS in China focus mainly on the performance of employing OSS in a local context. These existing studies indeed point out that there are some challenges in employing OSS, as well as the importance of technical innovation. However, the process of generic OSS, like Linux evolution, remains understudied. This study investigates the innovation behaviours on the suppliers' side. Section 7.1.1 and 7.1.2 illustrate the ways by which Red-Flag and CS2C managed tensions between their intentions to design products and users' local needs in the course of developing generic Chinese Linux solutions, while in this section we will focus on suppliers' processes of learning from local users by linking the learning dynamics across the biographical stages.

Enrolling local users – user representation building

From a social learning perspective, some novel ways to use software are embodied into the context where they have been used rather than pre-designed by the software providers. In this respect, identifying, understanding, categorising and involving local users is essential to develop generic software packages.

An embedded system is a computing system designed to perform one or a small number of dedicated functions. It normally runs with limited computer hardware resources: little memory, small storage, special or non-existent keyboard and/or screen (Ganssle, 2000). Small size is a primary concern in most applications to keep hardware expenditure low, which requires the technology provider to rigidly modify and tailor the OS in accordance with the user's local needs. Owing to this customised feature, embedded systems therefore required any embedded Linux, like many other embedded OS products, to establish formal and tight supplier-user linkage. Through this linkage, the suppliers could build user representation easily. In the case of Red-Flag, the local customers' involvement in the embedded Linux design was necessary from the very beginning, since embedded Linux is not an independent product, but it has to tightly couple with the embedded equipment/product in the sense that it is geared to a specific user group.

However, this case also demonstrates that building user presentation is not a one-off job, but a process which is full of uncertainties. In the first two development stages, Red-Flag focused mainly on knowledge acquisition which is featured as conducting as many user implementation projects as possible.

At the very beginning, the representation of user and use was largely built based on supplier's own assumptions. Red-Flag adopted a catch-up strategy: the original Linux was not designed for embedded applications so Red-Flag had to make great efforts to modify the original Linux kernel and develop components for implementing embedded functionalities. Red-Flag bought several up-to-date technologies and components that were available in the international market, such as a real-time kernel patch, Chinese input methods, etc.

This conceptual embedded Linux, however, was soon found to be unsuitable for user needs. The pre-designed features and functionalities were immediately challenged by local users' embedded products. As a result, Red-Flag had to carry out mass-customisation which required the supplier to design and develop embedded Linux from scratch for each customised project. By experiencing this painful process, Red-Flag explored the possible generic features of their knowledge of designing embedded Linux. This first attempt of generification was generated from interaction with user organisations in early projects, which showed that Red-Flag built the user representation by enrolling local users directly. The first attempt was not successful due to the fact that Red-Flag realised it was impossible to accumulate a solid knowledge base for generating generic development, just as too many varieties of hardware framework led to the unpredictable complexity in embedded Linux design and development.

As Red-Flag embedded Linux evolved into the growth and future stage, the supplier changed its learning approach. Based on the extensive knowledge generated from related local adaptation projects in earlier stages, the supplier sought to continue the aim of developing generic solution because of the economies of scale. This required that Red-Flag to reconstruct its relationship to user organisations. Hyysalo's study on the biography of an ICT application indicates that the technology would have evolved into a stage in which the supplier sought standardisation by coordinating user's expectations (Hyysalo, 2006). In practice, Red-Flag's early passive way of learning was replaced by an active manner: taking the extensive knowledge acquired from past projects and the accumulation of power and reputation (successful reference site) as a basis, Red-Flag could be actively involved in embedded product design on the user side. This change enabled Red-Flag to develop a generic solution without passively accepting user local requirements which may challenge their standard solution. Red-Flag delivered a generic embedded Linux solution that is a complex package for generating embedded Linux and embedded applications. This package is an aggregation of developing knowledge on selected functions and features of past projects.

Platform Linux, on the other hand, was originally developed as a generic solution for the mass market so that the process of bringing users into product design may be more complex in practice than embedded Linux. From a social learning perspective, direct linkages between designer and user may be absent or rather weak in the case of mass market products (Williams, Slack and Stewart, 2005). The case of CS2C platform Linux development shows that the representation of local user and use, and the manners of local user engagement, vary according to the biographical stage of Chinese indigenous platform Linux.

At very beginning, the user representation in platform Linux design is achieved by an indirect manner – translating a foreign Linux copy to a Chinese version directly so that the representation of user is extrapolated from existing foreign users. Chinese local users in this sense were neglected in the design of Linux due to the fact that there was no local market for Chinese indigenous Linux – lacking keen users who are prepared to employ Linux. Furthermore, the Chinese government (like many other countries in both developing and developed world) had a rather over-politicised view on Linux, since the government hoped to develop an independent OS based on Linux to replace the foreign OS products so that China could rapidly be independent from foreign companies. As a result, the time constraint was also another reason for Chinese Linux developers neglecting local users. All features of Red Hat Linux which catered for foreign users were inherited by Chinese indigenous Linux. Nevertheless, users are extremely heterogeneous (Williams, Stewart and Slack, 2005): the foreign users of Red Hat Linux at that time were Linux enthusiasts, computing specialists, college students in science and engineering fields (particularly the computer science and electronic engineering subjects), with very different expectations and requirements to the users in the Chinese market. As a result, this version of Chinese Linux in reality cannot fit the needs of Chinese local users.

In the later stage – early development stage, CS2C built representation of user and use by both the direct manner which is based on a selected group of users – government user, and an indirect conduit according to the competitor – Microsoft. The government is the first user group for Chinese indigenous Linux so they are naturally the first local users involved in Linux design and development. The government users, in particular the office users, are heavily locked in the existing MS-Windows in the aspects of both system manipulation and application implementation. Based on the feedback gained from office users in Beijing government agencies, CS2C modified the way to operate Linux (such as simplifying the Linux installation process) and improved the functionalities of Linux (e.g. enhancing the official document printing function). Be aware of the reality that most of the potential users in China's mass market are familiar with MS-Windows, CS2C developed the GUI of its later Linux in light of MS-Windows.

The government user could be seen as the selected user with particular requirements in document processing and printing, like many other office users. However, one of the difficulties in building user representation is the 'representativeness' of a selected user group; this issue is in connection with the validity of information fed back from government users. Unlike discrete software packages, platform Linux is developed as an infrastructure so that there is a bilateral relation between user's uptake and complementary products. The empirical evidence shows that CS2C greatly improved user easy-operability of Chinese Linux, in the sense that local users who locked in MS-Windows are able to use Linux easily. However, the case also shows that the direct information about the government user of indigenous Linux cannot be a reliable basis for extrapolation. This problem is not reflected by Linux itself, instead by the context where user applies Chinese indigenous Linux.

In this way, the government created a market demand and offered the public sector as the first user/buyer. Yet, the case also demonstrates that the experience of designing Linux for the government was not successful for the whole mass market because CS2C did not fully understand how other users in the mass market use platform Linux.

In the last biographic stage, CS2C built the representation of user and use based on intermediaries. One of the crucial issues during Chinese platform Linux development is that compatibility between OS and complementary products. The functionalities of computer systems are subject directly to the hardware devices and application software. Platform

Linux, like many other OS products, plays as the bridge to integrate hardware and specific applications together to fit the needs of users. However, the diversity of users about their context and the way they used computer systems resulted in the complementary hardware and software becoming the bottleneck for growth of Linux as infrastructure to support the ‘informatisation’ of the whole of society in China. This phenomenon is also known as the indirect externalities effect in economic terms.

In order to solve this problem, in particular at the desktop end, CS2C engaged with intermediaries (PC companies) to develop Linux-based software solutions for niche markets. These intermediaries segmented the intended markets, interpreted the requirements of the end-users in the target market by conducting market research, and providing or feeding back this information to CS2C. In this respect, the CS2C is able to build user representations around the evidence of actual users via an indirect manner. At the server end, the government shifted its strategy from being a direct user to a combined strategy, providing subsidy to adoption of Linux, in order to attract organisational/business/enterprise users. In other words, CS2C could build user representation with government policy. From the feedback from both phases of the CCB project, CS2C began to understand that the system management tools would reduce the users’ work in configuring and administrating the software system for computer servers.

The case of the Chinese platform reveals the importance of local users and their representations in Chinese indigenous Linux design and development. It also shows that CS2C do not have sufficient capability to configure their target users, due to the existing user lock-in effect generated by MS-Windows, and solve the problem of compatibility between Linux and complementary products. In order to favour indigenous Linux as infrastructure, both the government and intermediaries played active and crucial roles.

Intermediary user

As computers were introduced across organisations and sectors, and geared to support an increasingly wide range of complex and variable organisational activities and everyday life (Friedma, 1998; William, Stewart and Slack, 2005), it was extremely difficult to collect knowledge and preferences of the end users on OS. Although the Chinese indigenous Linux providers acknowledged that the end user is the most important learning source for Linux design and development, they seemed to have no means for collecting information about end-users and enlisting them in the Linux design process, due to the absence of direct linkages.

According to the cases, the ‘users’ involved into both Chinese embedded and platform Linux design are not the direct end-users, but the intermediary users which are the organisations that adopted Linux as the OS of their embedded products (like SinoData lottery sales terminal) and computer products (e.g. Haier Happy Family PC), or for their employees (such as the Sailing project and CCB projects). The intermediary users configured not only the content of technology, but also the work practices and everyday uses. They are also the bridges gathering knowledge about end-users, translating and conveying the information collected from end-users into technical terms for Linux developers. In this respect, they can be seen as the ‘brokers’ who established ties between the everyday life usage, organisation practices, and the Linux suppliers’ innovation efforts.

In the case of Chinese embedded Linux, the customers of Red-Flag played an extremely pivotal role as brokers. The customised features of embedded Linux results in that there is no embedded OS that can be purchased separately in the mass market. The lottery sales terminal projects show that SinoData was the broker between embedded Linux provider and the end-users. According to the understanding of how end users wanted to use the lottery terminals, SinoData configured the content of embedded Linux by setting rules and regulations on the use of the lottery sales terminal.

In the case of Chinese platform Linux, it is a general-purpose OS so it is supposed to be developed as an infrastructure. However, there is no link between supplier and end-users because of, on the one hand, the user lock-in effect created by the existing dominance of MS-Windows, and on the other, the diversities of individual, organisational and industrial users and their complex contexts. In the desktop segment, the developer has to rely on brokers – PC companies. The brokers/intermediaries identified the end-users in local niche markets and collected and translated their needs in technical terms. With this knowledge, CS2C was able to understand the particular requirements of various user communities and design generic Linux-based desktop solutions for them. For instance, in order to develop Happy Family PC, Haier sent a research team to a village and conducted comprehensive surveys to collect the (generic) particular needs of rural users and to understand the application context in which the computer was to be used. Haier interpreted this raw information into detail which was provided to CS2C later. In the case of CS2C server Linux, the intermediary user, CCB, adopted Linux to replace outmoded SCO UNIX as OS of its core business system. CS2C had to work closely with CCB to design and develop server Linux because only CCB clearly understood the needs and operations of the end-users.

User in local technological capabilities building

The traditional technology transfer literature, such as Lall (1982), Huq (1991), Shen (1999) etc. argued that there is a need for indigenous technological capabilities in the adaptation process of foreign transferred technology. Their research, however, focused mainly on manufacturing and industrial technologies, whereby the understanding of indigenous technological capabilities building was associated with the technologies that had been studied, such as the ways to identify and select appropriate technologies and components, operate, maintain and repair machines, the in-house R&D capability for future technology development, etc. Furthermore, researchers paid much attention to the actors who were directly involved in the process of technology transfer, e.g. joint funds, technology transferors and transferees, R&D players, etc. The users of industrial technologies were studied as the technology transferees who tried to obtain a high-level of productivity, improve better product quality, manufacture new product, and so forth by applying advanced technologies. There is a big distinction between defining users in industrial and manufacturing technology, and software technology. The transferees of transferred software are indigenous software enterprises/providers whose goal is making a profit from selling software or providing related software service to local end-users. As a result, the role of the user who will use the software products as an independent player in software transfer is more complex and needs to be paid particular attention by researchers.

This study reveals that adapting globally designed Linux in China is a process of generification – developing generic Linux solutions for local users. The analysis of the two types of Chinese Linux's biographies shows that Linux suppliers adapted different development strategies to make the trade-off between universality and diversity, i.e. suppliers' intentions of developing universal solutions and users' specific work and use practices. Suppliers must enrol users/intermediaries into software design by building representations of use and user through a social learning process. Therefore, the supplier's capability of identifying, understanding, categorising of local users/intermediaries, as well as enlisting them in software design is a crucial element in technological capabilities building.

7.2.4 The broader context – challenges for adaptation

Another key issue which is to be discussed is the influence of the broader historical and institutional context in which Chinese indigenous Linux was embodied, developed and used. According to a SST perspective which is the main intellectual and theoretical base for this study, particular choices made by suppliers during technology development and innovation are associated with social, economic and political issues. This section seeks to explore how Chinese socio-economic factors and institutional arrangement shaped Chinese Linux by drawing upon the concepts of infrastructure studies.

Developing infrastructure is a large and complex socio-technical project, and uncertainties and challenges are inevitable and cannot be foreseen in advance. Drawing upon the term 'reverse salients' (Hughes, 1983), which refers to the most significant problems during the process of infrastructure growing, this study identified three dynamics for adapting and developing Linux in China:

The first dynamic is associated with social, organisational and legal issues.

Transferring Linux, whether internationally or nationally, to a new organisation, domain, or context is a mutual adjustment of not only technical, but also broad non-technical elements. As indicated in chapter 1, China is experiencing the transition from a traditional highly central planning economic system to a market economic system, which started in 1979. Under the Chinese traditional central planning economy, the industry and public R&D resources were separated with no institutional links between them and enterprises had little interest in R&D, since the output and selling of products had been planned and controlled by government. Simultaneously, universities and research institutes focused mainly on the research of their academic interests. Since all of their research projects were funded by (local and/or central) government, there were occasions in which the government played the role as coordinator for R&D projects.

Given the background mentioned above, the two main domestic Linux providers – Red-Flag and CS2C were strongly influenced by the Chinese traditional planning economy. Red-Flag was founded mainly by a traditional academic institute – ISCAS which is one of the earliest public institutes conducting academic research on OS in China. CS2C was split from the largest state-owned software company – CS&S, whose OS products were normally funded by government and characterised political or scientific purpose in the past. As a result, both suppliers lacked the knowledge of users and market, and had very little experience and capability to commercialise products. Moreover, CS2C as a firm had to concentrate largely on application research for surviving in the fierce market competition, so it lacked the capability in basic research, such as research in software algorithms, system testing and so forth.

The second dynamic faced by China in developing Linux is the gateway (interoperability). Many OS products are proprietary software so they are normally incompatible with other platforms for both hardware devices and applications. To the extent of interoperability between the heterogeneous OS (e.g. between Linux families and MS-Windows), it is very difficult to establish a gateway unless the owners of proprietary software agree to share their technological secrets. The notion of gateway in this case study, therefore, refers to the gateway between different Linux products (distributions). Technically, Linux products are developed by different providers based on the same Linux kernel. However, the components chosen for building Linux distributions may be distinct. For example, there are many options for Linux to choose a GUI, such as GNOME, KDE, FVWM95, etc. The layout and the way to operate these Linux distributions vary from one to another. This is one of the barriers for Linux spreading in China. What’s more, the application software designed or optimized for a given Linux distribution may not work perfectly on others. The interoperability consequently, is a key problem between Linux distributions.

The third dynamic is the effect of path-dependency. As we discussed in 7.1.2, the path-dependencies on both user and supplier ends stem from the local market environment.

In this respect, the selected development strategies employed by two Chinese indigenous Linux suppliers not only reflected their understanding of local work and use practices, but also worked to fit the Chinese specific context and dealt with the socio-economic dynamics surrounding the software development.

The case studies conducted by this research favour the government intervention in OSS development through a rich empirical data pool in chapters 4, 5, and 6. The growth and formation of infrastructures requires heavy government investment, encompassing both pecuniary and institutional. Besides, China’s ongoing transition of economy leads to the co-existence of both the traditional planning economy and market economy. As a result, government is still a key player in reconstructing socio-technical regimes, building network for R&D and commercialization, and regulating market order.

Although the cases of Chinese embedded Linux, platform Linux and government policies were presented separately in three chapters, a co-evolution process was discovered, whereby the macro environment, socio-technical regime and micro niches (software design and implementation) evolved. The macro environment is largely influenced by the government policies by which government intentions of developing self-controlled OS and promoting its deployment are embodied. The government intervention in the socio-technical regime and micro niches directly created and spurred the initial and potential market for indigenous Linux by linking the public research resources, suppliers and users. By this way, Red-Flag and CS2C could be seen as the selected technology developers to engage with co-created micro niches, due to their previous experience in Chinese OS development, accumulated technological capabilities and state-owned background.

With support from the government, the suppliers and Chinese public research resources, which were separated in the traditional planning economic system, built a new socio-technical regime by means of systematically interacting and cooperating during the transition. Facing fierce market competition from foreign commercial OS products, the establishment of a new socio-technical regime denoted that the suppliers are able to fit local users' needs and facilitate the growth of micro niches.

Drawing upon the experience of past implementation projects (micro niches), Red-Flag and CS2C recognised the features of local users, and their work and use practices through a social learning process. For the software suppliers, it is of primary importance to manage their customer relationships and adopt strategies to conduct software design collaboratively with users, which means to comply with the idea of 'innofusion'.

The case studies obviously show that the government's policy-making was not a static process, but reflected a learning process. The international relations with foreign governments and enterprises (e.g. the United States and Microsoft), as well as the lessons learned from the establishment and performance of socio-technical regime and micro niches, are the elements which were taken into account in the new policy-making process. In this respect, a new mutual co-evolution loop re-starts again.

7.3 The Role of the Chinese Government

As discussed in the literature review chapter, the computer is the infrastructure of traditional infrastructures and the most recent information infrastructure. Drawing upon the theory of 'virtual infrastructure' or 'second-order infrastructure' (Jackson, *et al.*, 2007), this research identified that the OS is the infrastructural software of the computer. The principle tasks of computers are about data: how to get it, how to share it, how to store it, and how to apply it to specific tasks. Infrastructural software implemented the most basic data management for computer machines.

From the perspective of infrastructural formation (Hanseth, 2002; Jackson, *et al*, 2007; Edwards, *et al*., 2007), infrastructural software is not genuine infrastructure naturally, although it has some features of infrastructure. In the empirical case of platform Linux development, Linux is just a software package as an isolated system (Hanseth, 2002; Jackson, *et al*, 2007; Edwards, *et al*, 2007; Edwards, *et al*, 2009) before growing genuine infrastructure (Edwards, *et al*., 2009). It must compete with other alternatives in the Chinese market, as well as on the global market. With the Chinese purpose of developing Linux as genuine infrastructure in China, it faced vast challenges of dynamics and intentions, such as path dependency effect. Government plays a powerful and active role in dealing with these challenges. The formation of infrastructure can be divided into three stages: local isolated system building; system transference to other contexts or domains; and system consolidation. The government role is explicitly important in the initial and final stages.

At the first stage, only the government has the capability to plan and manage large-scale and long-term infrastructure projects. From the perspective of economics, many modern infrastructures are like- or quasi-public goods so that they call for government support, in particular public investment. For example, the UK spent £275 million to promote E-Science from 2001 to 2006, and US National Science Foundation (NSF) established a cross-directorate Office of Cyber-Infrastructure, spending $175 million annually to promote new projects throughout the natural and social sciences. There is no doubt that governments would be willing to build an independent and affordable infrastructure. The development of Linux was financially supported by some states, in particular the DCs, like China, for the sake of reducing procurement cost and independency. The empirical data of this research also shows that the upfront finance supplied by government is a crucial resource for supporting Chinese Linux R&D.

Government role is also of central importance in the consolidation phase of infrastructure development. Graham and Marvin (2001) identified the term ‘modern infrastructural ideal’ to elaborate the phenomenon of infrastructure consolidation in the middle decades of the nineteenth century. Following this logic, many national governments not only invested in all or most of the infrastructures, but also provided relevant services, for example, the road and railroad system, electronic system and telephone network. In developed countries in Western Europe and US, some infrastructures were built privately but under public oversight by regulatory government agencies. Adapting and developing Linux as infrastructure calls for enormous investment and resource inputs. Only government has enough power to assemble and coordinate the fragmented resources along the supply chain.

In the specific cases of Chinese Linux development, the observations of government policies and measures are divided into two main categories. Firstly, supporting Linux R&D by pushing R&D to commercial entities (e.g. like Red-Flag and CS2C) and providing direct financial support to Linux R&D; and secondly encouraging user involvement into Linux design and development (such as government procurement and subsidies to Linux adoption), configuring the user (e.g. adopting anti piracy strategy and offering education and training for Linux), and issuing gateway – technical standards to enable different Linux distribution to be interoperable.

However, my case studies further point to the importance of the appropriateness of tools and means adopted by government in different development stages. Any misuse of these tools may result in problems. Take government procurement for example, due to lack of local users, the Chinese government had to become the first and only user for Linux desktop systems initially. This policy did help the companies to pull through in their early stage of operation before the Linux market was established, but it also led to serious problems. The indigenous Linux providers, in particular Red-Flag and CS2C (CS&S) treated each other as rivals in government contracts, rather than competing with foreign giants. This is also contrary to the government expectation of replacing foreign OS, particularly MS-Windows. Additionally, this policy did not really help the uptake of Linux in the mass market. Although the public sectors are obliged to buy and use Chinese Linux, the users didn’t actually use it. Later, the government learned to switch its sole role from market controller to market coordinator.

During the process of policy making, the Chinese government realized that the measures made according to the traditional ‘technology push’ principle, such as supporting technology R&D and artificially creating a market demand (government procurement) were insufficient for software adaptation and development, in particular the infrastructural software. The wide spread of Linux was subject to many other social and economic elements, and the interplays amongst other key players, i.e. the local user and public R&D resource.

7.3.1 Supporting Linux R&D

Supporting the R&D

Nearly all the research papers from the perspective of the developed world, particularly the United States are against the various government measures of subsidy. Schmidt and Schnitzer (2003) pointed out that the government should restrict itself to subsidizing basic research only because it has too many uncertainties to get financial incentive from commercial companies. The development of most software products, whether close or open source, is applied R&D which has well-defined potential users and predictable R&D outcomes. Therefore, software product R&D investment should be provided by the market.

From the Development Economics perspective, the market defectiveness of DCs is insuperable. Therefore, cultivating the enterprises is one of the main tasks for government in DCs besides establishing market systems and improving the efficiency of resource allocation (Wang, 2000). Software products, like other high-tech products, are characterized by high-investment, high-return and high-risk. As a new OS, the result of Linux adaptation and development has too many uncertainties – the costs and risks of Linux development may exceed expected benefits (Wang, 1997). The venture investment mechanism had not been established completely and/or matured at the start-up stage of Chinese Linux development. The Chinese government therefore issued investment and financing policies (see chapter 6) like the role of venture investment in developed countries, to supply indigenous Linux R&D.

Supporting the R&D could be seen as subsidies to Linux providers. To help suppliers facilitate delivering product to market and improve their products under the condition of limited financial budgets at the very beginning, the government funded a series of R&D projects to some indigenous Linux suppliers, such as Red-Flag and CS&S, which are the most successful indigenous suppliers today. For example, the earliest Chinese Linux distribution was developed under financial support from the government. Furthermore, the Sailing Project presented in chapter 5 and 863 key projects in desktop Linux development mentioned in chapter 6 are two further examples.

Providing incentive for cooperation between industry and public R&D resources

According to the cases of both embedded and platform Linux development demonstrated in Chapters 4 and 5, a problem faced by Chinese indigenous Linux suppliers, like Red-Flag and CS2C, is the shortage of human resource, in particular software engineers in their product development teams. In order to sustain and further enhance the R&D capability, the Linux suppliers had to borrow technical capability from public R&D resources, such as research institutes or universities.

Kim (1999) and Madu (1989) pointed out that the R&D research programmes conducted in universities and research institutes provide one of the conduits to accumulating a knowledge base which is the source for local technological capabilities building.

Nevertheless, the universities and research institutes were largely separated from industries in the Chinese traditional central planned economy. Against this backdrop, universities and research institutes had little interest in understanding market demands because all of the finance for conducting research was supplied by government, and R&D

research programmes were purely for academic or scientific purposes. Similarly industrial organisations, had no need to co-operate with academic organisations since the pattern, volume and technology of products were predominately and strictly controlled by the government but not the market, so that any development by the industrial organisation was in response to government directives. Nearly all of their research outcomes had been shelved. To facilitate commercialization of research findings and further enhance national competitiveness, the Chinese government initiated a top-down R&D system called Industry-University-Institute Strategic Cooperation at beginning of the China's economy transition (Bai, and *et. al*, 2007). According to government policy, a new evaluation mechanism was adopted in all institutes and universities, which treated government funded research projects with the same weight as academic publication. The evaluation mechanism is not only the criterion for staff promotion but also for extending employment contracts. Except for a few government funds which only focused on basic research, all levels of government funding regarded the commercialization of research findings and expected products quota as one of the most important factors for a successful application. This policy drove institutes and universities to actively establish tight links to industry.

The two cases reflect different types of co-operation between public R&D resources. Red-Flag is a spin off from ISCAS. There were two collaboration modes that have been elaborated in the cases. The first is that ISCAS provided an embedded computing laboratory including corresponding research staff and postgraduate students to Red Flag as a temporary development team. The T&W broadband digital set-top-box project is a detailed example for such collaboration. The other mode is that Red-Flag acquired research findings from the institute to enhance its technological competency. For example, ISCAS undertook many government funded R&D research projects aimed at improving the real-time performance of the Linux kernel. Some of these projects cooperated with Red-Flag because it provided a conduit for the commercialization of research findings.

Previously, CS2C was the Linux department of CS&S (China National Software & Service) – the largest state-owned software firm in China. The department was established as an independent firm dedicated to Chinese Linux related business in 2003, collaborating with another state-owned corporation – CETC (China Electronic Technology Group Corporation). CS2C and School of Computer Science in BUAA, combined their resources to establish a Joint Laboratory in software system testing during the CCB project. The cooperation between CS2C and BUAA continued after the project and the joint laboratory was firmly incorporated within the CS2C as a department. The head of joint laboratory-Professor Yuqing Lan from BUAA was employed as the R&D director of CS2C as well. Normally, there are 8-10 computing postgraduate research students from BUAA whose research was based on the CS2C Linux system. Apart from providing technical support for system testing during the CCB project, the laboratory also developed a series of testing tools by means of carrying out R&D research. These testing tools played important roles for later projects because system testing is a key step for all customised projects and new products R&D.

7.3.2 Linking Linux providers and users

Bridging the gap between Linux suppliers and local users

In Chinese traditional central planned economy, the commercialization of technology was never taken into account. The utilitarian view of science and technology development resulted in China's strategy of technology development being dominated by technology determinism for a long time (Shen and Williams, 2005; Williams and Edge, 1996). One consequence of such a utilitarian view is that most users and uses are neglected in the innovation process; their potential contributions are played down (Tait and Williams 1999). As a result, the linkage between technology development and users is rather weak.

Government procurement, as well as providing subsidies to Linux localization R&D, and user adaptation are two steps in linking the Linux suppliers and local users. Public procurement is the most direct measure in creating local users for Chinese indigenous Linux. With the help of public procurement, the Linux providers not only generate profit to sustain their business but also understand the features of Chinese users. According to the features understood about the users, the Chinese government also provided subsidies for Linux R&D with which the Linux providers are able to improve their Linux products to match the Chinese user preferences. To spread Linux more widely, the Chinese government also granted subsidies for Chinese Linux adaptation to attract industrial users to apply Chinese Linux.

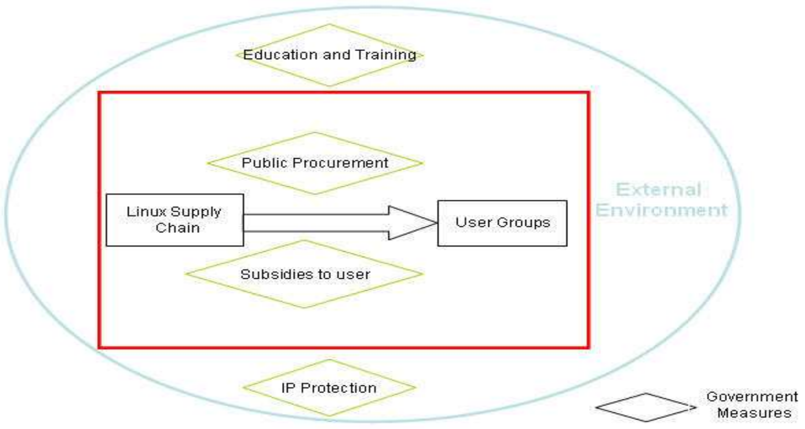

Figure 6: Government measures for linking suppliers and users

Public Procurement

From the case study of CS2C (chapter 5), the first implication of Chinese public procurement helped the indigenous Linux firms to acquire actual user requirements. The spread of Linux desktop in China is not a technology problem so it cannot be solved by Linux suppliers individually. To make Linux available for every Chinese user, the first step is to collect and understand adequate detailed user requirements. As late-comers in China's OS market, indigenous Linux providers had to treat survival as their principle tasks from the very beginning. However, they were unable to collect user preferences and requirements because there was no existing link between the companies and local users. Additionally they could not afford market surveys to understand local users due to lack of financial budgets. In summary, they had no understanding of their users' requirements. The reality in China was that Linux distribution cannot become widely used unless it is available and user-friendly to Chinese users, but it will be difficult to adapt to local user needs without advance knowledge of user preferences. Only the government can provide the remedy for this problem by creating markets and usage for indigenous Linux distribution. The Beijing government is the earliest local customer for indigenous Linux distribution. Apart from signalling its strong support from the Chinese government, the procurement behaviour created the first group of users who represented the personal users in China's mass market. The indigenous Linux distribution providers began to recognize that increasing user-friendliness, like improving Linux simplicity and ease of operation rather than Chinese application, was the only way to attract domestic mass users for broader success.

The second benefit of public procurement was helping to integrate Linux supply chain vertically. The Linux supply chain refers to the vertical system consisting of low-level hardware, Linux OS and high-level applications running on Linux. The relationship between the Linux supply chain and user is positively interdependent. On the one hand, Linux will not be chosen by the mass user unless it has sufficient applications and compatible hardware; on the other hand, the indigenous Linux providers are unable to persuade application software and hardware suppliers to develop applications and hardware drivers for their Linux distributions without a sufficient installation base. The government software procurement project has become the biggest market in China since the Government On-Line Project was launched in 1999. Once the government agencies made the decision to adopt Linux, the software and hardware suppliers would have strong incentives to develop applications and drivers for Linux distribution. For example, the government fund launched an 863 key project to organise domestic Linux developers, mainstream hardware devices and PC manufacturers, and application software developers to improve the interoperability between them and further integrate total solutions for industrial or government usage.

Subsidies to user

To spread Linux from a limited market (government market) to other sectors, the involvement of users from other domains is essential. Nonetheless, except for the government application, industrial or business users will be another large market for indigenous Linux. Furthermore, if the applications of indigenous Linux were limited in government applications only, the Linux providers would not have generated profits from market competition to sustain its operation, and would always have been dependent on government support. However, Chinese local industrial and business users were still reluctant to adopt Linux in their core business system although the indigenous Linux has been widely applied in the public sector. Adopting Linux in core business systems was regarded as highly risky, unless there is evidence to justify that indigenous Linux was secure and stable enough for business applications. Consequently, the Chinese government put the task of pulling industrial or business users to Linux applications on the agenda. The method picked by government is the subsidies to the users who are willing to apply indigenous Linux in its core business systems. The two-stage CCB project presented in chapter 6 is seen as the most successful reference of indigenous Linux application in China. Not only did the firms in the finance industry follow CCB to adopt Linux, but also industrial and business user in other fields such as postal service, train and surface transportation, telecommunication, energy and electricity, medical care and health industries, etc. applied Linux.

Configuring user

Linux education and training

From a STS perspective, including infrastructure studies, technology transfer and local technological capability building, knowledge base is one of pivotal factors. Against this backdrop, not only sophisticated software engineers but also normal users in the mass market are of importance for development and deployment of Chinese Linux.

However, Microsoft was already involved in the IT education system through its Great Wall Plan so Chinese IT and software engineers were locked-in to the technological regime of Microsoft. As a result, the Chinese software industry would depend on foreign technology. The measure of introducing Linux knowledge to both the Chinese higher education system and professional training system thus was an effective tool for solving this problem.

In order to bridge the digital divide, computer application education was carried out in some primary and secondary schools, and all universities throughout China. This universal education was mainly based on the products of Microsoft, like MS-Windows and MS-Office. This phenomenon resulted in the users in China's mass market being locked-in to MS-Windows. It is difficult to persuade existing users to transfer their OS from MS-Windows to Linux owing to solid user habit and high switching costs. In order to break the monopoly generated by foreign proprietary OS and to spread indigenous Linux, the Chinese government decided to cultivate potential users for Linux. The project conducted by Zhejiang province is considered as the most successful case in China. The education department in this province introduced Linux knowledge to NCRE (National Computer Rank Examination). The qualification certificate is one of the necessary conditions for starting a career in most government agencies and some multi-national companies in China. Therefore, more and more college students will study Linux actively. According to the statistics, there are over forty thousand college students in Zhejiang Province who passed Linux knowledge based NCRE up to the summer of 2009. Inspired by the experience from Zhejiang Province, Beijing, Shanghai, Guangdong Province, etc. also listed Linux knowledge as the content of their local NCRE.

Strengthen protection of software IP (Intellectual Property)

Compounded by low income and proprietary software's ever-increasing costs, a high proportion of software products in China are illegal copies. After entering the WTO, China has been obligated to provide adequate anti-piracy enforcement tools to remedy and prevent software piracy.

Two compulsory policies required all computers purchased by government agencies or sold in China territory to have pre-installed legal copies of an OS. Although the government did not specify that all PCs needed to install Linux, nevertheless, it effectively helped Linux in the market. The foreign and domestic PC manufacturers picked indigenous Linux distributions for their PC and server products due to its lower cost. What’s more, picking indigenous Linux is perceived as supporting domestic software companies and industry so that it will help the PC manufacturers to build a good business reputation in the Chinese market. The sales of Red-Flag and CS2C Linux have both reached millions of copies since 2006.

Bridging the gap between Linux suppliers and complementary technology providers

Like all other OS products, Linux is the platform for the whole computer software system. It is the bridge that links hardware devices and specific application software. In this respect, the success of Linux in China requires that it must have higher interoperability to complementary software and hardware devices. Establishing a mature Linux supply chain thus is treated as an important task by the Chinese government.

However, most Chinese local complementary software developers and hardware device manufacturers had only developed products for MS-Windows OS due to its dominant position in China’s market. Supporting Linux meant additional R&D costs for these complementary technology providers. In order to tackle this obstacle, Chinese local and central government used financial leverage to support a series of projects. For instance, in order to match the OA application requirements, the Beijing government supported not only indigenous Linux and office productivity suite development, but also improving the interoperability of Linux to office suite and hardware devices, such as printers through the Sailing Project. Another example is the “10th Five-Year Plan” State Key Technologies R&D Program – “Indigenous Foundational Software Platform System Development and Application” which is aimed at improving the compatibility of Linux with indigenous computers and application software, and further developing system solutions.

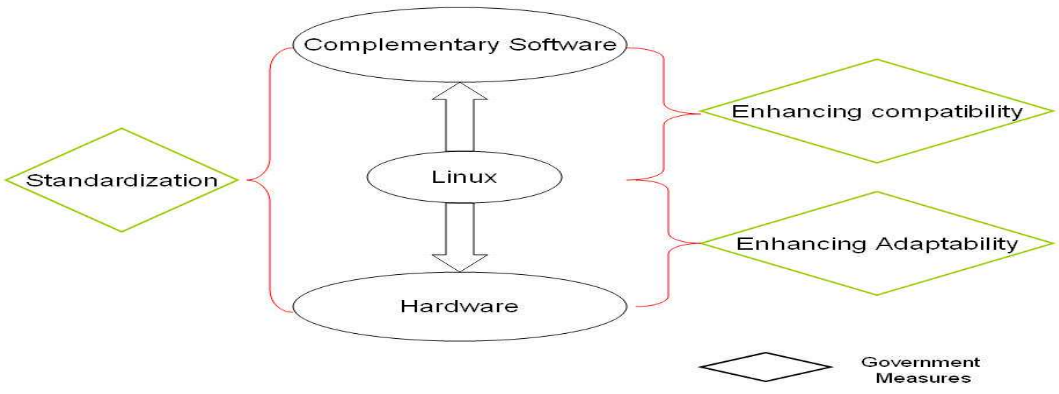

Figure 7: Government measures to build Linux supply chain

7.3.3 Establishing gateway– Interoperability

Tracking back to the growth history of existing infrastructure, the competition between the technical regimes behind the various isolated systems developed by respective builders was an inevitable issue. Other than one system beating out all the others, this competition can only be resolved by the creation of 'gateway' technologies or social arrangements that permit all these systems to interoperate.

The APIs (Application Programming Interface) of the various OS products are sharply distinct from each other so that they have different computer running environments. Both the hardware profiles and application software packages developed for these heterogeneous systems were incompatible. To be grown as genuine infrastructure, the infrastructural software, including Linux and other alternative OS products is to either eliminate other alternatives or have a gateway to permit compatibility. However, the reality is that they are competing in a global software market; there was neither a successful winner in the technical realm nor a gateway to enable them to be compatible with each other. The gateway which provides interoperability between heterogeneous OS products may not be developed in the future because users of computers are able to share and exchange data and information through an existing gateway – a set of standardized Internet protocols on application level.

As discussed in section 7.2, the interoperability of various Chinese indigenous Linux distribution and application is one of the main challenges for developing Linux as an infrastructure. The Chinese government organized indigenous Linux providers, complementary software developers, hardware device manufacturers, public research institutes and universities to formulate *de jure* Linux industry standard. Technically, with a standard, the hardware drivers and application software can be developed once and run on every indigenous Linux platform without any modification. Commercially, a uniform operating style and GUI for all Chinese Linux distribution could save learning cost and time for the users who are willing to switch to Linux.

7.3.4 Discussion

As infrastructural software, Linux was developed not solely by market force, but also by government initiative in China because of public interest. All computers have to install an OS to implement specific functions. Therefore information security and economic concerns of OS are important factors for the users. The indigenous Linux distribution was the solution for eliminating these two concerns. On the one hand, the security risks of an OS can be inspected and removed by users, in particular the government and industry users, since the source code of Linux is open; on the other hand, the users can save much of the cost because Linux distributions have no license fee. The economic advantage of Linux is seen as the remedy to bridge the domestic digital divide by the Chinese government.

Infrastructural software is characterized by strong network externalities. Katz and Shapiro (1985) propose in their paper that the network externalities give rise to demand-side economies of scale. Software products have much higher R&D costs but with lower and fixed marginal costs. In another words, once particular software is developed successfully, manufacture, distribution and delivery would be cheaper if they prevail in the market (Shen, 2005b). The average total cost of a software product will reduce as the production increases so that the software industry shows supply-side economies of scale as well. Both supply-side and demand-side economies of scale will result in natural monopoly. As discussed in the above sections of this chapter, the spread and development of Linux is subject to the quality and quantity of complementary hardware and application software. As a result, a supply chain which consisted of hardware manufacturers, Linux providers and application software suppliers is necessary for Linux. However, the monopoly generated by foreign OS suppliers will be the impediment for Linux supply chain building. From the perspective of development studies, breaking the monopoly generated by foreign OS providers, protecting the domestic software market, as well as encouraging market competition, are all important duties for the Chinese government.

Perfect and symmetric information is essential for optimal resource allocation. In reality, private producers of information have a vested interest in keeping it for their own consumption because the cost of information acquiring, transmitting and extracting is quite expensive. For this reason, the private market is unlikely to provide an adequate supply of information (Stiglitz, 1986). In China, Linux is limited to a small range of users because it was primarily designed and maintained by IT specialists and programmers individually, and thus only addresses the requirements and interests of these professional users. Furthermore, investment in market promotion for a new coming software product sometimes exceeds its direct R&D cost, in particular at the start-up stage of running a business. In contrast to commercial software companies, like Microsoft who spend hundreds of millions of dollars in advertising expenditure in China, Chinese indigenous Linux suppliers have less financial resources to advertise their Linux products to China's mass market. Chinese consumers in the mass market therefore were uninformed of the existence and the characteristics of Linux distribution. Comino and Manenti (2005) stress that the lack of awareness on the existence of OSS is a significant feature in the mass market segment. This phenomenon would lead to unfair market competition. The government preferential policies and measures, in particular the public procurement, enable Chinese users in the mass market to become acquainted with Linux by various conduits.

7.3.5 Rethinking policies from a Social Learning perspective

The policies and measures adopted by Chinese government to support Linux development as infrastructure could be categorized into two areas over time: the early stage is the phase in which Linux was established as an isolated system on China's market; the later stage is the competing phrase. The corresponding political leverages in the early stage were to protect the infant technology, which included the support of Linux R&D with enterprises by providing upfront finances and create an early market by government procurement. The measures employed latterly consisted of a broader series of policies being introduced in the course of developing Linux as an infrastructure from context specified local system. The detailed measures created and strengthened the links between Linux providers and other main social actors, as well as building a favourable external environment.

When the indigenous Linux providers initiated development of their Linux distributions, the government interventions were focused largely on technological issues – supporting Linux R&D. The policy-makers optimistically believed that indigenous Linux would be able to replace foreign OS products rapidly and grow as the infrastructure applied in all sectors of national economy, on account of its technical advantages in security, architecture, satiability, and so forth. The user in the domestic market will tend to apply indigenous Linux inspired by the signal of government procurement. However, the studies on the dynamics and tensions highlighted during the process of developing internationally transferred Linux as infrastructure in China uncovered that Linux cannot be adapted strictly from top-down planning and is highly dependent on social considerations. The original aim was unsuccessful as the MS-Windows OS products were still the most prevalent OS for the PC, and accounted for the largest share in the server market segment in China. This problem stems from the Chinese traditional 'technology push' policy which was largely influenced by the view of determinism (Williams, Stewart and Slack, 2005). The idea behind Chinese policy-making was affected by a linear model of innovation which was a one-way flow of technology innovation: developing from R&D behaviour to finished artefacts and diffusion to the consumer through the market. A negative consequence of such a model of policy-making was that the user and its practice were neglected in the innovation process (Tait and Williams, 1999).

It does not imply that earlier policies and measures failed; instead they were not enough to support infrastructure for development and growth. From a social learning perspective, uses and outcomes of Linux were not simply inscribed in the design of the product (on the supply side), but depended equally upon the way they were selectively taken up, adapted and incorporated within the practices of individuals and groups. Moreover, the Linux adaptation as infrastructure depended largely on the local socio-economic context and other actors, like a range of firms supplying complementary technologies, research organisations, government, and users. Developing infrastructural technology was a complex undertaking which required the policy-makers to take the social and economic factors, as well as the main actors and their active interplays into account. The empirical cases show that the technological advantages of Linux did not lead to market success naturally. Problems faced during building Linux as infrastructure included lack of complementary technologies and technological capability building mechanism, the existing user locked-in effect on foreign OS products, and rampant software piracy.

Chapter 8 Conclusion and Wider Implications

8.1 Introduction

This final chapter presents the overarching conclusions of this study, linking the academic debates and theoretical limitations reviewed in chapter 2 with the analytical content presented in chapter 7. It further examines how relevant China's experience in supporting indigenous Linux is to the main policy issues and concerns of other DCs.

This study criticises the research framework employed by social scientific research into OSS phenomenon in DCs. Mainstream OSS literature states that DCs are able to benefit from adopting and developing OSS, in particular Linux. However, this research finding is informed by a naïve methodology – for example the procurement and implementation of OSS in DCs is taken for granted, which is based on an assumption that globally designed OSS fits the local contexts of DCs and is ready for local deployment. Furthermore, the prevalent research model separates the design and implementation of OSS; it focuses instead on the economic and managerial performance of adopting OSS according to the technical advantages of OSS. This shortcoming results from disciplinary divisions – researchers who study OSS phenomenon in DCs from a traditional single disciplinary approach, such as economic analysis, law studies and so on – tend to treat development process of OSS as a 'black-box'. There is no enquiry into the technical realm – mirroring the reluctance of engineering and other technical disciplines – to engage seriously with the social dimensions of their works. The single disciplinary perspective adopts a rather de-contextualised snap-shot approach to study OSS in DCs, which puts the emphasis on a single site implementation of OSS at a particular moment. In order to mend the limitations brought about by the single disciplinary approach, a SST perspective is employed for this research. SST provides an interdisciplinary approach to understanding the complex social phenomenon associated with the design and implementation process of OSS.

To overcome the shortcoming of single site implementation and short-term research, this study particularly employs the theory of Biography of Artefacts (BoA) developed under SST perspective, which is an analytical tool for multi-site implementation and long-term life-cycle studies. Following the tradition of SST, this study also reveals the intricate interplays between actors and intermediaries in technology design, implementation and use by using a social learning perspective.

8.2 Conclusions

The objective of this research was not to focus on the ideologies behind the OSS as in much existing literature; it probed, instead, into the uptake of an infrastructural OSS – Linux in the context of China.

This research echoes the SST perspective, that the adaptation of Linux in China faces various context-specific challenges. The major suppliers in Linux adaptation are economically-oriented, which is very different from the philosophy and ideology of the international Linux community; the Chinese end-users are not OSS enthusiasts and therefore are not concerned about the spirits of the open source movement; the Chinese government sees Linux as an alternative OS to foreign proprietary products and the policy makers hoped to substitute foreign products with indigenous Linux. The studies on the biographies of both Chinese embedded Linux and platform Linux provide the evidences of how these actors with different purposes reach consensus in generic solutions design and development in a long-term process.

The findings of this study provide insights into two kinds of generic software package design for enriching the BoA perspective. Unlike large-scale software packages, such as ERP, PDM and e-infrastructures which have been studied by BoA scholars, embedded Linux and platform Linux have their own technological specificities. The DevsPartner is not a pure OS that could cross the boundaries between various embedded application areas, but a complex software solution for tailoring embedded Linux and developing embedded applications. Although the process of generificiation was similar to the ERP and PDM packages, through accumulating functionalities according to diverse local practices, standardising functionalities and delivering generic solutions based on standardised functionalities, the generic embedded Linux solution is achieved in a more abstract way. It takes place through extracting knowledge of developing functionalities and tailoring Linux on grounds of specific hardware frameworks, and further, materialising this expertise as the functions/templates of generic solution. The generification process of platform Linux represents a reversed orientation, which originated from a foreign transferred generic Linux and was achieved by differentiation afterwards. The generic platform Linux is defined and redefined due to the fact that suppliers gradually understand the local user and use through learning processes.

This study also provides understanding of software technology transfer. The traditional technology transfer literature that focused mainly on manufacturing-related elements, cannot explain the software technology transfer phenomenon owing to different technology specificities. For example the software, in particular the OS rather than limits to a given domain or area, can be applied in various fields and industries. Furthermore, its application is subject to the network externality effects. These phenomena indicate that software technology transfer had to take the broader local context (e.g. user diversities, complementary technologies, political power etc.) into account. My research findings demonstrate that the adaptation process of transferred software is a generification process. The transferred software was normally applied in a local context of one organisation or a group of mass users as the first step. The next step is that suppliers decided to design a more universal solution, with the purpose of matching the diverse requirements of all users (or larger groups of mass users) by reusing the accumulated knowledge and expertise. In this premise, the local end-user is of crucial importance to software adaptation. In order to build sufficient indigenous technological capabilities for adapting and further developing software technologies which are developed elsewhere, the following points should be taken into consideration:

1) a good understanding of the socio-technical interfaces – the use and implementation

of technology of the software in question;

2) the way to apply the software to local environments to meet user needs, and;

3) the abilities of collaborating with main players associated with the particular software, e.g. in the case of Linux, the core Linux community in the world, the complementary application providers and the intermediaries/end-users, etc.

In response to the issue that government intervention should be excluded from OSS development, this study confirms that the role of government is necessary for the adaptation and development of Linux in China. It augers that the government intervention is needed to compensate for some of the failures of Chinese market mechanisms during the transition, in particular to take responsibility for reducing some of the negative social consequences that may be produced by global market forces, like the monopoly generated by foreign companies and the lack of awareness of OSS.

We also articulate the idea of 'infrastructural software' to explain the social dynamics surrounding OS design and development. Adapting and developing Linux as infrastructural software in the contexts of DCs requires seamless networks to link technology providers, public R&D resources, complementary technology providers and users, as well as the integration and coordination of their resources. Furthermore, regulating the software market, by means of providing anti-privacy laws and strengthening the teaching of Linux knowledge in higher education institutes, is essential for Linux local adaptation as well. Additionally, in light of the BoA perspective, the social, organisational, and institutional factors in the macro-level of context in which Chinese indigenous Linux was embodied, developed and used, are difficult for technology suppliers and local users to change. Thus, only the government has commercial and political powers to guarantee the success of Linux adaptation. This study specially analyses the diverse policies and measures which were employed by the Chinese government to meet the expectation of developing Linux as infrastructural software. We argue that the public procurement and Linux standardisation are not completely efficient, which should be refined by policy-makers. With reference to suggestions for future policy-making, the Chinese government should stress the relations between global developers, such as Linux community and local developers, as well as organise local developers as one camp in order to compete with foreign proprietary software companies.

Drawing upon these research findings, the research may offer some suggestions for Chinese Linux players and wider implications for other DCs. Firstly, this research shows that, in such a globalised world, to ensure the success of Linux, seamless collaboration between actors along the supply chain is required. This collaboration is needed not only within the national boundaries of the network, but also in the international domain as the gap between Chinese actors and international community hinders the further development of Linux. Given the difference in ideologies between them, a good understanding of one another is important to overcome the barriers (e.g. lock-in effects) which are deeply entrenched and to enable a united front (having an affordable OS is particularly relevant to DCs). Secondly, Linux players in other DCs may learn lessons from Chinese experiences to speed their expansion of Linux in local markets. The governments in other DCs particularly can also look to the issues regarding different tools/means to support Linux adaptation activities in different stages, tied to their specific social context.

8.3 Reflections on the Study and Future Research

Apart from the analytical, theoretical and empirical value of this study, a few limitations are also needed to be mentioned in this section. These limitations are associated with research interests, data collection and the research framework. The reflections on limitations could make contributions to possible future research.

The primary concern to this study is to provide better understanding of the open source movement in DCs by conducting a case study in Chinese OSS adaptation and innovation. Following a SST tradition, we focused on suppliers' development strategies and innovation processes. The reasons for stressed OSS development mainly from the suppliers' perspective are that a detailed empirical investigation is still lacking, and current literature draws attention to the implementation and use of OSS in DCs. This kind of focus does not mean that the implementation of OSS is not of importance. We selected theoretical tools (such as the BoA and social learning under the SST framework) very carefully to build a research framework for this study.

This study selected two Chinese domestic Linux suppliers as research objects. The decision stemmed from the high accessibility of enterprises, through which we were able to collect the necessary data for investigating the Linux development process. However, the reality in China is that there are three main foreign Linux suppliers competing with indigenous enterprises in software market, and Linux products from these foreign suppliers account for around 50% of the local market share. Conducting case studies on indigenous Linux products using these terms of reference may incur limitations on the explanation of foreign Linux adaptation and development in China's context. Our case studies provide detailed data on indigenous Linux development and the socio-economic and institutional context of China. As a result, this study could be treated as the foundation for future research to test the local adaptation and development of foreign branded Linux.

As we indicated above, this study takes the implementation and use of Linux into account by adapting a theoretical tool – social learning. Accessing local users is still problematic because of various reasons, such as personnel mobility, business sensitivity and even political sensitivity (particularly for state-owned enterprises). Because of this, we have to employ an 'encircling' strategy to collect data on the practices of implementation from suppliers and/or from a secondary resource – reports or documents written by users. This strategy, however, may draw critiques from social scientific researchers, because these data cannot be seen to be completely reliable – data are likely to over stress the positive side of implementation and ignore the possible side effects.

This study examines two types of Linux application, embedded Linux and platform Linux, in order to supply comprehensive insights into Linux's adaptation and innovation. Nonetheless, there are many general themes that remain under-developed due to the fact that researchers must make trade-offs between depth and width. This is a common and inevitable challenge for researchers, especially for PhD students who have to balance time constraints and study scope. These themes (for instance, the bottlenecks described in the Red-Flag case, the relationship to the international Linux community in CS2C case, etc.) merit further investigation in future research.

8.4 Wider Implications

8.4.1 Policy implications for China's future Linux adaptation and development

This research shows that the Chinese government has played a very important role in the adaptation and development of Linux in China. According to a long-term government policy document on national S&T development –"The National Outline for Mid-Term and Long-Term Scientific and Technological Development Planning (2006-2020)", the Chinese government will continue providing substantial support to core software, including Linux development for the next ten years. There are two points which were overlooked by policy-makers in this document to which attention should be paid.

Firstly, the global and local perspectives are both important. Linux evolves as a complex assemblage of global and local elements emerging in a process in which the generic offering (emerging from the Linux community) is embodied in particular purposes and appropriated in the context of China. However, the policy for adapting Linux addresses much on a local level. Like most OSS, Linux is designed and developed at a global level, controlled by an international community. From a perspective of technology transfer and reshaping of technology, most of the OSS communities are located in developed nations so their contexts are very different from those of DCs. The open source movement in the eyes of scholars in the developed world is driven by the virtue of knowledge sharing and moral duty of knowledge dissemination. Thus, the social value and technical presumptions of OSS may not fit the socio-technical environment of the developing world. The programmers and companies in DCs, like China may have different incentives in adopting and developing

OSS. Lack of proficiency in English and heavy economic pressures are the barriers to Chinese programmers in joining the mainstream international OSS network. Without involving the international Linux community, the requirements from Chinese users are excluded from the design and development of global Linux.

The case studies also show that the process of adaptation and development of Linux in China have embodied the rationale of Chinese developers and users within the specific social, political and economic contexts. Thus the outcomes of Linux products are largely context specific. This is an obvious difference between the original spirit of hackers in the US, or the rationale of Linux development as an OSS project and the objectives of China. Discrepancies are more or less inevitable since the Chinese national S&T policies are interested greatly in local adoption – making globally developed Linux adapt to the Chinese context. In contrast, the main concern of the international Linux community is technical issues rather than its commercialisation and actual applications. To illustrate this, the indigenous Linux developers complained that their contributions on modifying Linux were excluded by the international communities. The foreign advocates and enthusiasts in international Linux community indicated that the Chinese developers should not modify Linux locally to mimic their main adversary – Microsoft. In this respect, the Chinese government should make a trade-off between the aspects of global features and local context, and provide incentive to encourage researchers and R&D engineers to participate the international Linux community.

Secondly, from the perspective of infrastructure studies, a gateway is extremely crucial for Linux. The two main Chinese indigenous Linux distributions are seen as heterogeneous systems to the extent to which not only the GUI and operations of different Chinese branded Linux distribution are distinct. But there are also interoperability problems between them (e.g. some hardware drivers and complementary application software developed for CS2C Linux would not be completely compatible with other Chinese Linux distribution). All these factors are the obstacles for Chinese users in applying indigenous Linux. Standard is one of the major gateways, through which the complementary product developers and users are capable of bridging these gaps. As discussed in the above section, the standardisation of Chinese indigenous Linux was not completely successful. A main reason is the fact that the two main Chinese indigenous Linux providers – Red-Flag and CS2C treat each other as rivals, rather than partners in domestic market. In order to resolve this divergence, the government should not limit its role on standardisation, but should encourage domestic Linux providers to cooperate further in conducting R&D projects. These projects should be mainly in connection with the user, such as developing friendly GUIs and improving easy operation in a standardised way, as well as consistently taking the interoperability of complementary products into account.

8.4.2 Implications for other DCs

This research highlights the importance of major players involved in Chinese Linux development, such as suppliers, the government, public R&D resources, and users, as well as the seamless linkages between them. However, the experience of Chinese Linux adaptation indicates that the international Linux community is also another key player which has been overlooked by Chinese players. The cultural and ideological differences between Chinese providers and international Linux community resulted in, on the one hand, a gap between Chinese players and the international Linux community. This phenomenon emphasised the dependency of Chinese players to international Linux community, especially to platform Linux. The reason is that the contributions made by Chinese Linux providers (although not entirely) were generally ignored by the community. On the other hand, the Chinese players may face context specific challenges (e.g. lock-in effects). To DCs, the key players may not have incentives to establish links to the international Linux community because they have their own motivations on Linux. For instance, while the local Linux suppliers focus mainly on Linux localisation and generating economic benefits, the local users care about the applications of Linux rather than technical issues. Meanwhile some governments could be concern about technology independency. In order to involve the core technology of Linux development, DCs governments should provide appropriate measures to encourage local suppliers and users to participate in the international Linux community.

From the viewpoints of many DCs, software industry might bring opportunities for leapfrogging because DCs have advantage of backwardness (Gerchenkron, 1962; Brezis, Krugman, and Tsiddon, 1993). As late-comers, DCs can draw on the lessons from the development experience of developed countries, and accelerate technology development by skipping some stages of development. A dangerous reality is that many DCs focused on the technical successes, while neglecting the broader social context in the developed world where the technology was designed and developed successfully. As a result, players in DCs have a narrow view on technology innovation – as if technical efforts, and economic and political demands are the only elements for technology development and innovation.

This empirical case of Chinese Linux uncovers a broader context which is also important to Linux adaptation and innovation. Nevertheless, suppliers and users have no means to change some social and economic elements. What government can do is to restructure some elements of the local context by issuing policies and employing administrative measures. They are the most pressing tasks for facilitating DCs' technological progress. Therefore, a lesson that can be drawn from this research for the governments in other DCs is that S&T policy making should not only focus on technical issues, but also take the construction of the external environment into account.

The Chinese government has acted as a director of change which is deeply rooted from the tradition of intervening between S&T and economy development. From a viewpoint of development studies, the software markets of DCs have been nearly monopolised by international giant players from the developed world. OS is a core software technology not only at the heart of the information technology industry but also important for the entire national economy. Without core software, technology would further deepen the dependency of DCs to the developed world. In this respect, the Chinese government has a specific policy of choosing Linux and directing its development as a substitution to foreign OS products. However, not all DCs have the same level of government focus as China, so they may only act as coordinators of change. In this premise, they can draw upon the experience of Chinese Linux development and adopt more liberal approaches to support Linux, such as offering technology training, supplying information to the market, and encouraging the cooperation of intra-organisations (e.g. R&D institutes and Linux firms).

Bibliography

Applewhite, A., 2003. Should government go open source? IEEE software 2003 Volume: 20, Issue: 4 88-91.

Arthur, W. B., 1994. Increasing Returns and Path Dependence in the Economy. University of Michigan Press, Ann Arbor.

Awazu, Y., Desouza, K., 2004. Open knowledge management: Lessons from the open source revolution. Journal of The American Society for Information Science and Technology, 55 (11): 1016-1019.

Bessant,J. , Rush,H. , 1995. Building bridges for innovation; the role of consultants in technology transfer, Research Policy, Vol:24, Pages:97-114

Bezroukov, N., 1999. A second look at the cathedral and the bazaar. First Monday, vol. 4, 1999. Retrieved from: http://www.firstmonday.org/issues/issue4_12/bezroukov.

Biggs, T., Shah, M., Srivastava, P., 1988. Technological Capabilities and Learning in African Enterprises. World Bank Technical Paper, No. 289, African Teaching Department Series, New York

Bijker, W., 1987. The social construction of bakelite: Toward a theory of invention. In The social construction of technological systems: New directions in the sociology and history of technology, edited by Wiebe Bijker, Thomas Hughes, and Trevor Pinch, 17-50. Cambridge, MA: MIT Press.

Bijker, W., 1995. Of bic~clesb, akelites, and bulbs: Toward a theory of sociotechnical change.Cambridge, MA: MIT Press.

Bitzer, J., 2004. Commercial versus open source software: the role of product heterogeneity in competition. Economic Systems 28 (2004) 369-381.

Bresnahan, T.F., Greenstein, S., 1999. Technological competition and the structure of the computer industry. Industrial Economics 47 (1), 1–40.

Bresnahan, T.F., 2001. Network effects and Microsoft, Stanford Institute for Economic Policy Research Discussion Paper No. 00-51, 2001.

Bokhari, S.H., Rehman, R., 1999. Linux and the developing world. IEEE Software Volume: 16, Issue: 1 58-64.

Bonaccorsi, A., Rossi, C., 2003. Why Open Source software can succeed. Research Policy 32 (2003) 1243-1258.

Brynjolfsson, E., Kemerer, C., 1996 Network externalities in microcomputer software: an econometric analysis of the spreadsheet market, Management Science 42 (12) (1996) 1627–1647.

Callon, M., Law., J. 1982. On Interests and their Transformation: Enrolment and Counter-Enrolment. Soc. Stud. of Sci. 12 615–25

Camara, G., Fonseca, F., 2007. Information policies and open source software in developing countries. Journal of the American Society for Information Science and Technology, 58 (1):121-132, 2007.

Campbell-Kelly, M., Garcia-Swartz, D.D., 2009. Pragmatism, not Ideology: Historical Perspectives on IBM’s Adoption of Open-Source Software. Information Economics and Policy 21 (2009) 229-244.

Cao, F., 2005. Research on Developing Strategy of Linux Software Industry and the Policy Supporting System. Science & Technology Industry of China. Feb, 2005

Capek, P.G., Frank, S.P., Gerdt, S., Shields, D., 2005. et al, A history of IBM’ s open-source involvement and strategy. IBM Systems Journal 44, 249－257.

Casadesus-Masanell, R., Ghemawat, P., 2006. Dynamic mixed duopoly: a model motivated by Linux vs. Windows. Management Science 52 (7), 1072–1084.

Ciborra, C. U., Braa, K., Cordella, A., Dahlbom, B., Failla, A., Hanseth, O., Hepsø, V., JLjungberg, J., Monteiro, E., Simon, K.A., 2001. From Control to Drift: The Dynamics of Corporate Information Infrastructures, Oxford: Oxford University Press

Colman, D., Nixson, F., 1994. Economics of Change in Less Developed Countries. 3rd edition. New York and London: Harvester Wheatsheaf.

Comino, S., Manenti, F.M., 2005. Government policies supporting open source software for the mass market. Review of Industrial Organisation 26 (2), 217–240.

Connell, C., 2002. Open Source project manage themselves? Dream on. Retrieved from: http://www.chc-3.com/pub/manage_themselves.htm

Cook, I., Horobin, G., 2006. Implementing eGovernment without promoting dependence: Open source software in developing countries in Southeast Asia. Public Administration and Development 26 (4), 279-289.

Cooley, R. E. (2004). The social construction of technology and information systems. In K. Grant (Ed.), Proceedings of the UK Academy of Information Systems Conference (pp. 121–129). Glasgow: Caledonian University.

Cowan, R., 1992. High technology and the economics of standardization”, Chap. 14, pp. 279-300 in Dierkes, Meinolf and Hoffmann, Ute (eds.) New Technology and the Outset: Social Forces in the Shaping of Technological Innovations, Frankfurt/NY: Campus/Westview. 1992.

Cusumano, M.A., Elenkov, D., 1994. Linking international technology transfer with strategy and management: a literature commentary. Research Policy, No. 23, pp.195-215

Da, D., 2005. Open Source: Transparent World. Information System Engineering. June, 2005.

Dalle, J.-M., David, P.A., 2003. The allocation of software development resources in ‘open source’ production. Discussion Paper of The Stanford Institute For Economic Policy Research.

David, P. 1985. Clio and the Economics of QWERTY. American Economic Review, volume 75 (1985), pp. 332–337

Deignan, M. P., 2001. CNET explores Apple's new Darwin kernel. CNet Reviews. Retrieved from http://reviews.cnet.com/4520-3673_7-020165-1.html.

Demil, B., Lecocq, X., 2006. Neither market nor hierarchy nor network: The emergence of bazaar governance. Organisation Studies. V 27 i10. 1447-1466

Dierckx, M.A.F., Stroeken, J.H.M., 1999. Information technology and innovation in small and medium-sized enterprises. Technological Forecasting and Social Change 60, 149-166.

Desai, A.V., 1984. India’s technological capability: an analysis of its achievements and limits. Research Policy 13, 303–310

Economides, N., 1996. The Economics of networks. International Journal of Industrial Organisation, 14 (1996) 673-699.

Economides, N., Katsamakas, E., 2006. Two-sided competition of proprietary vs. open source technology platforms and the implications for the software industry. Management Science 52 (7), 1057-1071.

Edwards, K., 2005. An economic perspective on software licenses-open source, maintainers and user-developers. Telematics and Informatics, 22 (2005) 111-133.

Edwards, P., 1998. Y2K: Millennial Reflections on Computers as Infrastructure. History & Technology 15:7-29

Edwards, P., Jackson, S., J., Bowker, G., C., Knobel, C.,P., 2007. Understanding Infrastructure: Dynamics, Tensions, and Design. Retrieved from:

http://cohesion.rice.edu/Conferences/Hewlett/emplibrary/UI_Final_Report.pdf

Edwards, P., Bowker, G..,C., Jackson, S., J., Williams, R., 2009. Introduction: An Agenda for Infrastructure Studies. Journal of Association for Information System Vol. 10: special issue, pp. 364-374

Evans, D.S., 2002. Politics and programming: Government preferences for promoting open source software. In Hahn, R.W., (ed.) Government policy towards open source software. AEI-Brookings Joint Center for Regulatory Studies. 34-48.

Evers, S., 2000. An Introduction to Open Source Software Development. Technische Universität Berlin, Fachbereich Informatik, Fachgebiet Formale Modelle, Logik und Programmierung (FLP).

Feller, J. Fitzgerald, B., Hissam, S.A. and Lakhani, K.R., 2007. Perspectives on free and open source software. Cambridge, MA: MIT Press

Fomin, V.V., Blechar, J., 2005. An analytic model for the prospective exploration of emerging infrastructures. In: Proceedings of the 4th International Conference on Standardisation and Innovation in Information Technology (SIIT), Geneva, Switzerland, 100-114. (September 21-23)

Franck, E., Jungwirth, C., 2003. Reconciling investors and donators: The governance structure of open source. Journal of Management and Governance 7, 401–421.

Franke, N., von Hippel, E., 2003. Satisfying heterogeneous user needs via innovation toolkits:the case of Apache security software. Research Policy, 2003,32: 1199-1215

Fukuyama, F., 1995. Trust: The social virtues and the creation of prosperity. New York: The Free Press

Gallagher, K.S., 2006. Limits to Leapfrogging in Energy Technologies? Evidence from Chinese Automobile Industry. Energy Policy 34 (2006), 383-394

Gao, L., Xie, Q., 2004. Standardization Status and Development Design of Linux. Information Technology and Standardization, Aug. 2004.

Gao, Li., 2005. Accelerate Development of Software Industry by Building and Ecosystem for Linux. Information Technology and Standardization, July. 2005.

Glott, R., Ghosh, R.A., 2005. FLOSSPOLS: Usage of and Attitudes towards Free / Libre and Open Source Software in European Governments. Retrieved from: http://flosspols.org/deliverables/D03HTML/FLOSSPOLS-D03%20local%20governments%20survey%20reportFINAL.html

Grieve, R.H., 2004. Appropriate Technology in a Globalizing World. International Journal of Technology Management and Sustainable Development, Vol. 3, No. 3, 173-187

Hanseth, O., Jacucci, E., Grisot M., Aanestad, M., 2006. Reflexive standardization: Side effects and complexity in Standard-Making, MISQ: Special Issue on Standardization, Vol. 30, Aug. 2006, pp. 563-581

Hauben, M., Hauben, R., 1996. Chapter 9: On the Early History and Impact of UNIX: Tools to Build the Tools for a New Millennium. In Hahn, Hauben, M., Hauben, R., (ed.) Netizens: On the History and Impact of Usenet and the Internet. Retrieved from:

http://www.columbia.edu/~hauben/netbook/

Hanseth, O., 2002. From systems and tools to networks and infrastructures --from design to cultivation. Towards a theory of ICT solutions and its design methodology implications. http://heim.ifi.uio.no/~oleha/Publications/Unpublished manuscript Downloaded Oct. 30

Hecker, F., 1999. Setting up shop: The business of open-source software. IEEE Software, Volume: 16, Issue: 1, 45-51.

Hertel, G., Niedner, S., Herrmann, S., 2003. Motivation of software developers in open source projects: An internet-based survey of contributors to the Linux Kernel. Research Policy 32 (7), 1159–1177.

Hicks,C., Pachamanova, D., 2007. Back-propagation of user innovations: The open source compatibility edge. Business Horizons (2007) 50, 315-324

Hobday, M., Rush, H., 2007. Upgrading the Technological Capabilities of Foreign Transnational Subsidiaries in Developing Countries: The Case of Electronics in Thailand. Research Policy, .36 (9), pp:1335-1356

Howells, J., 2006. Intermediation and the role of intermediaries in innovation, Research Policy, 35, pp. 715–728.

Hughes, T., P., 1983. Networks of Power: Electrification in Western Society, 1880–1930. (Baltimore, Md.: Johns Hopkins University Press, 1983)

Hughes, T.P. 1987, "The Evolution of Large Technological Systems" in The Social Construction of Technological Systems, eds. W.E. Bijker, T.P. Hughes & T.P. Pinch, The MIT Press, USA, pp. 51-82

Huq, M., 2004. Building technological capability in the context of globalization: opportunities and challenges facing developing countries. International Journal of Technology Management and Sustainable Development. Vol. 3, No. 3

Hyysalo, S., 2006. In search of useful theory of design-use relationship in health technology, Univeristy of Helsinki

Hyysalo, S (2010), Health Technology Development and Use, New York: Routledge.

James, J., 2003. Free software and the digital divide: opportunities and constraints for developing countries. Journal of Information Science, Vol. 29, No. 1, 25-33 (2003).

Jackson, S.J., Edwards, P.N., Bowker, G.C., Knobel., C.P., 2007. Understanding Infrastructure: History, Heuristics, and Cyberinfrastructure Policy First Monday 12, no. 6 (2007), Retrieved from: http://outreach.lib.uic.edu/www/issues/issue12_6/jackson/

Johnson, J.P., 2002. Open source software: Private provision of a public good. Journal of Economics Management Strategy, 2002 Vol. 11 Issue 4, 637-662.

Johnson, J.P., 2006. Collaboration, peer review and open source software. Information

Economics and Policy 18 (2006) 477-497.

Kallinikos, J. 2004. Farewell to Constructivism: Technology and Context-Embedded Action. C. Avgerou., C. Ciborra., F. Land. eds. The Social Study of Information and Communication Technology: Innovation, Actors, and Contexts. Oxford: Oxford University Press

Katz, M., Shapiro, C., 1985. Network Externalities, Competition, and Compatibility. The American Economic Review, Vol. 75, No. 3. (Jun., 1985), pp. 424-440.

Kim, L., 1999. Building technological capability for industrialization: Analytical frameworks and Korea's experience. Industrial and Corporate Change, Vol. 8, No.1, 111-136

Klein, H.K., Kleinman, D.L., 2002. The social construction of Technology: Structural considerations. Science, Technology, and Human Values, Vol. 27, No.1. (Winter, 2002), 28-52

Kogut, B., Metiu, A., 2001. Open source software development and distributed innovation. Oxford Review of Economic Policy, 17 (2), 248−264.

Koch, S., Schneider, G., 2002. Effort, cooperation and coordination in an open source software project: Gnome. Information Systems Journal 12 (1), 27–42.

Koch, C. 2005. Users? What users? Shaping Global Corporations and Generic Users with ERP. Proceedings of Workshop on User-driven IT Design and Quality Assurance, Royal Institute of Technology, Stockholm: 43–53

Koch, C. 2007. ERP – A Moving Target. Inter. Jour. of Bus. Infor. Sys. 2(4) 426–43

Ksheteri, N., 2004. Economics of Linux Adoption in Developing Countries. IEEE Software, 2004 vol:21 iss:1.

Kogut, B., Metiu, A., 2001. Open-Source Software Development and Distributed Innovation. Oxford Review of Economic Policy, Oxford University Press, vol. 17(2), pages 248-264, Summer.

Kuan, J., 2001. Open source software as consumer integration into production. Working Paper, Retrieved from: http://ssrn.com/ abstract=259648.

Kuk, G., 2006. Strategic interaction and knowledge sharing in the KDE developer mailing list. Management Science, Vol. 52, No., 7, 2006, 1031-1042.

Kumar, V., Kumar, U., Persaud, A., 1999. Building technological capability through importing technology: The case of Indonesian manufacturing industry. Journal of Technology Transfer 24: 81-96

Lall, S., 1982. Developing Countries as Exporters of Technology: A First Look at the Indian Experience. Macmillan, London

Lall, S., 1993. Promoting technology development: The role of technology transfer and indigenous efforts. Third World Quarterly, Vol 14. No.1, 1993

Lall, S., 2000. Technological Change and Industrialization in the Asian Newly Industrializing Economies: Achievements and Challenges. In L.Kim and R.R. Nelson, Technology, Leaning, and Innovation: Experiences of Newly Industrializing Economies, Cambridge: Cambridge University Press. Chapter 2

Latour, B. 1987. Science in Action, How to Follow Scientists and Engineers Through Society. Cambridge, MA: Harvard University Press

Lawrence, K. A., 2006. Walking the Tightrope: The Balancing Acts of a Large e-Research Project, Computer Supported Cooperative Work 15/4: 385-411

Lee, J., Bao, Z., Choi, D., 1988. Technology development process: a model for a developing country with a global perspective. R&D Management 18(3) 235–250

Leibovitch, E., 1999. The business case for Linux. IEEE Software, Volume: 16, Issue: 1, 40-44.

Lerner, J., Tirole, J., 2002. Some simple economics of open source. Journal of Industrial Economics, 50(2), 197−235.

Lerner, J., Tirole, J., 2005. The economics of technology sharing: Open source and beyond. Journal of Economic Perspectives, Vol. 19, No. 2, 99-120.

Li, M., Lin, Z., Xia, M., 2004. Leveraging the Open Source Software Movement for Development of China's Software Industry. The Massachusetts Institute of Technology Information Technologies and International Development Vol. 2, No. 2, Winter 2004, 45-63.

Liang, X., 2006. Investigation into Generalizing Open Source Software in the Colleges and Universities. Journal of Sichuan College of Education. Nov. 2006.

Lin, L., 2007. Impact of user skills and network effect on the competition between open source and proprietary software. Electronic Commerce Research and Application (2007), doi/10.1016/j.elerap.2007.01.003.

Lin, B.W., 2003. Technology Transfer as Technological Learning: A Source of Competitive Advantage for Firms with Limited R&D Resources. R&D Management 33, 3, 2003, pp327-341

Linus, W., 1998. Copyright does not exist. Retrieved from: http://www.w1npp.org/Events/2010/2010-FieldDay/ATV/30431604-Copyright-Does-Not-Exist.pdf

Liu, W., 1995. International technology transfer and development of technological capabilities: A theoretical framework. Technology in Society, Vol. 17, No. 1. 103-120

MacKenzie, D., 1990. Inventing accuracy: A historical sociology of nuclear missile guidance. Cambridge, MA: MIT Press

MacKenzie, D., Wajcman, J., 1999. The Social Shaping of Technology 2nd, Open University Press, P.vx

Madanmohan, T.R., Kumar, U., Kumar, V., 2004. Import-led technological capability: a comparative analysis of Indian and Indonesian manufacturing firms. Technovation 24 (2004) 979-993

Madu, C.N., 1989. Transferring technology to developing countries-Critical factors for success. Long Range Planning. Vol. 22, No. 4, 115-124

May, C., 2006. The FLOSS alternative: TRIPs, non-proprietary software and development. Knowledge,Technology, and Policy 18 (4), 142–163.

Metcalfe, R. (ed), 2007. JISC's Sustainability Study: A case study review of open source sustainability models. Available at: http://www.jisc.ac.uk/media/documents/programmes/distributed_elearning/sustainabilitystudy-1.0.pdf

Miller, J.S., 2002. Allchin’s folly: Exploding some myths about open source software. 20 Caradozo Arts & Entertainment Law Journal 491.

Morrison, A. 2002. Researching ICTs in Context’, InterMedia Report. 3/2002, University of Oslo

Mustonen, M., 2003. Copyleft-The economics of Linux and other open source software. Information Economics and Policy 15 (2003) 99-121.

Ni, G., 2005. Open Source Software and Government Procurement. China Government Procurement. July, 2005.

Ni, G., 2006. The Opportunities, Challenges and Duties of China's Open Source Software Development. China Information on Review. Sept. 2006.

Nikulainen, K., 2004. Open source software: Why is it here and will it stick around? SCRIPT-ed, Vol. 1, Issue 1, 2004. 137-159.

Noland, M., Pack, H., 2003. Industrial Policy in and Era of Globalization, Washing, DC: Institute for International Economics

Olsen, O.E., Engen, O.A., 2007. Technological change as a trade-off between social construction and technological paradigms. Technology in Society (2007), doi:10.1016/j.techsoc.2007.08.006

Orlikowski, W.J., Iacono., C., 2001. Research Commentary: Desperately Seeking he "IT" in IT Research – A Call to Theorizing the IT Artifact. Infor. Sys. Res. 12(2) 121-134

O'Mahony, S., 2003. Guarding the commons: How community managed software project protect their work. Research Policy, 32 1179-1198.

Pan, G., Bonk, C., 2007, The Emergence of Open-Source Software in China, The International Review of Research in Open and Distance Learning, Vol. 8, No.1 (2007).

Paukatong T., Paul H., 2006. Technology Transfer Induced Technological Dependency in an Electric Power Plant: Effects on Efficiency, Reliability and Flexibility. Global Journal of Flexible Systems Management, 2006, Vol 7

Peng, S., 1999. Mechanism of building trust: relation and legal measure. Sociological Studies, 2nd issue, 1999. pp53-66

Pinch, T., Bijker, W., 1987. The social construction of facts and artifacts: Or how the sociology of science and sociology of technology might benefit each other. In Bijker W., Hughes, T., and Pinch, T. (Eds.), The social construction of technological systems, 17-55. Cambridge, MA:MIT Press

Pipek, V., Wulf, V., 2009. Infrastructuring: Toward an Integrated Perspective on the Design and Use of Information Technology, Special Issue on e-Infrastructure, Vol. 10 (5) pp. 447-473.

Pollock, N., Williams, R., Procter, B., 2003. Fitting Standard Software Packages to Non-standard Organizations: The 'Biography' of an Enterprise-Wide system. Technology Analysis & Strategic Management 15(3): 317-332

Pollock, N. and Williams, R., 2009. Software and Organisations: The Biography of the Enterprise-Wide System or How SAP Conquered the World? , Routledge

Pollock, N., Williams, R., 2010. e-Infrastructures: How Do We Know and Understand Them? Strategic Ethnography and the Biography of Artefacts. Computer Supported Cooperative Work, Volume: 19, Issue: 6, pp. 521-556

Qian, L., 2005. Speed Up of Linux. Software World. June, 2005.

Raymond, E.S., 1999. The Cathedral & the Bazaar. O'Reilly. Retrieved from: http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/

Raymond, E.S., Band, J., Newman, N., Taylor, J., Lessig, L., 2000. Should Public Policy Promote Open-Source Software? Retrieved from: http://web.archive.org/web/20030802124831/www.prospect.org/controversy/open_source/newman-n-1.html.

Reddy, N.M., Zhao, L., 1990. International technology transfer: A review. Research Policy 19 125-307

Ribes, D., Finholt, T.A., 2009. The Long Now of Technology Infrastructure: Articulating Tensions in Development, Journal of the Association for Information Systems, Special Issue on e-Infrastructure, Vol. 10 (5) pp. 375-398

Roberts, J. A., Hann, I., Slaughter, S. A., 2006. Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. Management Science, 52(7), 984−999.

Salvador, T., Sherry, J.W., Urrutia, A.E., 2005. Less Cyber, More Cafe: Enhancing Existing Small Businesses Across the Digital Divide with ICTs, Information Technology for Development, :2005 vol:11 iss:1.

Sanders, J., 1998. Linux, Open Source and software's future. IEEE software 15 (5), 88-91

Schiff, A., 2002. The economics of open source software: A survey of the early literature. Review of Network Economics, Vol.1, Issue 1 66-74.

Schmidt, K.M., Schnitzer, M., 2003. Public subsidies for open source? Some economic policy issues. Harvard Journal of Law and Technology 16 (2), 473–505.

Scott, S.V. E. Wagner. 2003. Networks, Negotiations, and New Times: The Implementation of Enterprise Resource Planning into an Academic Administration. Inform. and Org. 13(4) 285–313

Shapiro, C., Varian, H. R. (2003), Linux Adoption in the Public Sector: An Economic Analysis. University of California at Berkeley working paper. Retrieved from: http://people.ischool.berkeley.edu/~hal/Papers/2004/linux-adoption-in-the-public-sector.pdf

Shen, X., 1999. The Chinese Road to High Technology. Macmillan Press

Shen, X., 2005a, A Dilemma for Developing Countries in Intellectual Property Strategy?- Lessons from a case study of software piracy and Microsoft in China, Science and Public Policy, Volume 32, Number 3, 1 June 2005, pp. 187-198(12)

Shen, X., 2005b, Developing Country Perspectives on Software: Intellectual Property and Open Source- A Case Study of Microsoft and Linux in China. Journal of IT Standards & Standardization Research, 3(1), 21-43, Jan-March 2005

Shi, J., Liu, X., 2005. Network Externalities, Software Business Models and the Market Structure of Operating System Markets: A Comparative Research of Windows and Linux. Finance and Trade Economics. April, 2005.

Stewart, J., & Hyysalo, S.,2008. Intermediaries, Users and Social Learning in Technological Innovation. International Journal of Innovation Management, 12(3 (Sept. 2008)), 295–325

Summerton, Jane, ed., 1994. Changing Large Technical Systems. Boulder, CO: Westview Press

Torvalds, L., 1999. The Linux Edge. In DiBona, C., Ockman, S., Stone, M. (Eds.), Open source: Voices from the open source revolution. O'Reilly.

Tschang, T., 2003. China's Software Industry and Its Implications for India. OECD Development Centre Working Papers 205, OECD, Development Centre. Retrieved from:

http://ideas.repec.org/p/oec/devaaa/205-en.html

Ure, J., Procter, R., Lin, Y., Hartswood, M., Anderson, S., Lloyd, S., Wardlaw, J., Gonzalez-Velez, H., Ho, K., 2009. The Development of Data Infrastructures for eHealth: A Socio-Technical Perspective, Journal of the Association for Information Systems, Special Issue on e-Infrastructure, Vol. 10 (5) pp. 415-429

Valimaki, M., Oksanen, V., 2005. The impact of free and open source licensing on operating system software market. Telematics and Informatics, 22 (2005) 97-110.

von Hippel, E., von Krough, G., 2003. Open source software and the private-collective innovation model: issues for organisation science. Organisation Science 14 (2), 209–233.

von Krough, G., von Hippel, E., 2006. The promise of research on open source software. Management Science, Vol. 52, No. 7, 2006, 975-985.

von Krough, G., Spaeth, S., 2007. The open source software phenomenon: Characteristics that promote research. Journal of Strategic Information Systems 16 (2007) 236-253.

von Krough, G., Spaeth, S., Lakhani, K.R., 2003. Community, joining, and specialization in open source software innovation: a case study. Research Policy 32 (2003) 1217-1241.

Wang, S., 2010. The introduction to indigenous operating system in last ten years. Retrieved from: http://news.newhua.com/news1/comments/2010/211/10211115036HJGE8F5H3I751C33D8B1KBJG8CJA6160K91835IF6CD5A.html

Waring, T., Maddocks, P., 2005. Open source software implementation in the UK public sector: Evidence from the field and implications for the future. International Journal of Information Management 25 (2005) 411-428.

West, J., 2003. How open is open enough? Melding proprietary and open source platform strategies. Research Policy 32 (2003), 1259-1285.

Williams, R., Edge, D., 1996. The Social Shaping of Technology. Research Policy, Vol.25, pp.865-99

Williams, R., Stewart, J., Slack, R., 2005. Social Learning in Technological Innovation. Edward Elgar Publishing Limited, UK

Williams, R., Pollock, N., 2011. Moving Beyond the Single Site Implementation Study: How (and Why) We Should Study the Biography of Packaged Enterprise Solutions. Information System Research (Forthcoming)

Wu, M.W., Lin, Y.D., 2001. Open source software development: An overview. IEEE Computer, Volume: 34, Issue: 6, 33-38.

Yang, C. F., 1995. Psychocultural foundations of informal groups: the issues of loyalty, sincerity, and trust. Paper presented at the 47th Annual Meeting of the Association of Asian Studies, April 6-9, 1995. Washington, D. C.

Zeitlyn, D. 2003, Gift economies in the development of open source software. Research Policy, 32(7), 1287−1292.

Zhang, L., 2006. Chinese Linux Communities Gradually Growing UP. Oct. 2006. www.CNETNews.com.cn

Zhang, Y., Sun, X., Guan, X., 2006. Research on Development of Linux Operating Systems in Higher Institute Education. Education Exploration. July, 2006

Zhao, C., 2003, OSS Movement of Redflag, Proceedings of Asian Open Source Symposium. Retrieved from www.asiaosc.org.

Zhou, Y., Shen, X., Fan, L., 2007. The Social and Economic Factors within ICT

Development in China: A Case Study of Linux Development in China. International Conference onWireless Communications, Networking and Mobile Computing, 2007. pp.6035 – 6038.

Appendix – Sample of The Transcript of A Interview

Date 25/4/2008

Interviewer: Yinhua Zhou

Interviewee: Mr. Naiping Han, President of CS2C

Interview goal: The general development story of CS2C/CS&S platform Linux, in particular the background of early products.

Time: 30 Mins

Interview guideline

1. Why did CS&S decided to develop Chinese Linux distribution in 1999? And what was the background at that time?
2. In recent ten years, CS2S made great achievements in developing Chinese Linux. Can you describe the development trajectory of CS2C/CS&S platform Linux?
3. Why did CS&S develop server Linux? Please tell the details and describe the background.

Transcript

Yinhua Zhou (Y):

Can you introduce the general history of CS2C/CS&S Linux?

Naiping Hang (H):

It is a long story. CS&S started conducting R&D in operating system (OS) for market in 1989, before I joining CS&S of course. Firstly, CS&S bought the license and all source code of UNIX system 5 Release 4 from AT&T. Secondly, developing team of CS&S wrote every line of code because the company stressed the intellectual property issue according to the available source code of UNIX. At present, I feel this idea is not an efficient way: CS&S could simply translate the UNIX in Chinese version and deliver to the end market. There are two reasons: on the on hand, CS&S have the full license to modify and redistribute the original UNIX from AT&T; on the other hand, the development of software and computer hardware product are extremely rapid in market, once the completely redeveloped CS&S UNIX was ready to release, the product had been outdated.

Y:

So far, we reviewed something about the CS&S in developing OS before making decision to develop Linux. I would like to know the story about CS&S Linux.

H:

Sure! The emergence of Linux provides a good opportunity not only for Chinese OS developers, but also for our country (China). Drawing upon the open source code of Linux, we would be able to develop Chinese version for domestic users easily, and therefore generated commercial earnings. To the country, it is a good way to control core software, such as OS, as well as catch up foreign countries in OS and software R&D.

Although CS&S heard about Linux in early 1994, we didn’t conduct any R&D on Linux until 1999. One of the reasons was that some multinational software and hardware giants, like IBM, Oracle, Sybase, Intel, Sun Microsystems, etc gradually provide compatible products for Linux. The other reason, and the more important reason was that our government also was interested in Linux.

Y:

On regular basis, the government will support Linux as well, right?

H:

You are right. Our government would support all the products, in particular the high tech products which they are interested in. The government normally would provide money for product R&D, as well as buy these products through public procurement.

Y:

That's an interesting point. Then I would like to know the details about government policies towards Linux later. Please continue the story from 1999.

H:

At the beginning of 1999, the related department in government and enterprises in software and computer industries had a dichotomy on whether we should develop Chinese indigenous Linux products. The voice from one side was that we could develop security plug-in running on MS-Windows in order to solve the security concerns. While the open nature of Linux might incur attach from hackers. Another side who are advocate of Linux claimed the backdoor and security vulnerabilities of Linux could be solved easily because the international Linux community has strong technological capability. But the MS-Windows OS is a 'black box' for users so that it's extremely hard to develop plug-in for it.

Y:

CS&S must be the advocate of Linux.

H:

Yes. As the biggest state-owned software company with over ten-year R&D experience on

OS, CS&S attended the Linux related seminars which were held by the government. We found that developing indigenous Chinese Linux not only was in line with the government future goal, but also helped the company to win government contracts and generate profit.

Y:

How about the market reaction to Chinese Linux product?

H:

Before 2001, all the domestic IT media treated Linux as the most promising product in domestic market. And if any IT media didn't mention Linux, then this media would be outdated. Domestic software companies also show over-enthusiasm on Linux products – as if with the delivery of Chinese Linux, the companies could survive, make money and expand. There were over ten indigenous Chinese products since 1999, such as Blue Point, Longshin, Xteam, CS&S, Red-Flag, etc. All the Chinese suppliers actively cooperated with every IT newspaper and magazines by the means of offering free Linux disc with paper-based IT media. The criterion by which people judged the success of Linux companies was how many free copies had been sent. With the collapses of internet bubble at the end of 2000, Chinese software industry and IT media were cooled down in 2001.

Y:

Can I say the first stage of Chinese Linux was a crazy stage?

H:

Yeah, you can say that.

Y:

So, what's the next?

H:

Beijing government Linux procurement is a key must be mentioned. Beijing government bought 6001 copies of Chinese Linux from CS&S and Red-Flag, among which, CS&S

accounted for 51% share. However, the users in Beijing government found that these Linux products were very difficult to use. For instance, document printing is the last but the most important step of government office procedure. But there was a compatibility problem between the existed printers and indigenous Linux. To make the Chinese Linux usable on government desktop computers, Beijing government initiated ‘Sailing Project’ in Jan, 2001.

Y:

Can you explain more details about compatibility problem between printer and Linux?

H:

Ok! Some prevalent domestic branded printers are not compatible well with indigenous Linux. However, these products were bought by different departments in Beijing government through government procurement in order to support domestic printer companies. We tried to contact the suppliers of these printers, but they could not provide any help about this case. The reason is that the suppliers didn’t have the source code of printer drivers since they bought the whole engineering solution (i.e. design and hardware framework of printer, as well as the driver) from foreign companies. It required extra cost for use to solve the problem. Therefore, Beijing government decided to invest our R&D.

Y:

Except for the printer problem, is there any other problems?

H:

Yes, we had to take the easy-operating into account. For example, the procedures of installing Linux were very complicated for users which required professional IT skills. However, end users normally don’t have these skills so that we tried to simplify the installing procedures. With a new GUI menu, users are able to install Linux and tune the hardware configuration by clicking ‘NEXT’ button and doing some simply choice.

Y:

Can I know more details about the Sailing Project?

H:

I'll arrange our senior manager in product department to meet you later, or perhaps next week. He also joined the Sailing project and knew everything about the whole project.

Y:

Thank you very much! Can you tell what was you learned from Sailing project?

H:

The most important lesson we drew from Sailing project is that we found users of desktop and server applications are distinct. Their requirements are different as well. For example, the desktop Linux could apply the newest Linux kernel, while the server Linux must apply old one due to computer server stresses system stability. Server Linux also had to deal with large volume date process and storage so that it requires RAID (this function has been popular on desktop computer in recent two years) or network storage functions. While the users of desktop Linux required high compatibility between OS and various peripherals, like printer and USB disk. At the same time, desktop Linux must support mainstream entertainment software as well. As a result, CS&S decided to develop Linux for desktop and server application respectively.

Y:

What's the strategies for desktop Linux development then?

H:

CS&S and later CS2C noticed that it is very difficult compete with MS-Windows OS in domestic market. It reflected by Linux has less application software, particularly the entertainment software that MS-Windows so that we decided to focus on the domestic users with special or specific requirements. We cooperated with domestic PC companies to develop customized desktop solution. PC companies normally used mainstream hardware so that there is no more compatibility problem. In addition, users with specific requirement would also require less software functions. This strategy is effective and our first contract was ‘Happy Family’ project. I’ll ask the senior manager in product department to provide details to you.

Y:

Ok. So how about the development strategy of server Linux?

H:

According to our experience in both product R&D and marketing, we proposed a concept of ‘functional server Linux’. These products are based on general server Linux, and also provide special mechanisms and tools to optimise the whole software system. Such as, Data Server Linux has extra component by which user do not need to tune the database software by complex manual procedures as before. All the tuning tasks could be done automatically when user tells the OS about specific name of the database software. The idea of functional server Linux was emerged from China Construction Bank project which is an on-going large scale government funded project. Again, I’ll also ask the product manager to accept your interview and provide details of what’s you need.

Y:

Thank you very much. Your information is extremely important for my PhD research.

H:

No worries. I’m going to meet somebody now so that I’ll see you later. If you need any help or assistance, please feel free to contact me.