AN EXPLORATION OF THE DIFFUSION DYNAMICS OF OPEN SOURCE SOFTWARE (OSS): AN AGENT-BASED COMPUTATIONAL ECONOMICS (ACE) APPROACH

by

Muhammad Adeel Zaffar

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Information Technology

Charlotte

2010

Approved by:

____
Dr. Ram L. Kumar

____
Dr. Kexin Zhao

____
Dr. Moutaz J. Khouja

____
Dr. Cem Saydam

____
Dr. Yogendra Kakad

ABSTRACT

MUHAMMAD ADEEL ZAFFAR. An exploration of the diffusion dynamics of open source software (OSS): an agent-based computational economics (ACE) approach. (Under the direction of DR. RAM L. KUMAR)

Despite the rising popularity of Open Source Software (OSS), there is limited understanding of the factors that affect the diffusion of OSS at the organizational level. Review of the literature suggests that previous empirical and analytical studies on this subject matter though valuable in their own respect, either did not address the full spectrum of critical factors in one model or did not investigate the impact of critical factors in enough detail leaving some gaps in the literature. In an effort to bridge these gaps, this dissertation develops a model to a) jointly investigate the effect of critical variables other than price on the diffusion dynamics of OSS, b) investigate the effects of social networks or inter-organizational relationships on the diffusion dynamics of OSS, c) propose a new software price discounting scheme and compare its effectiveness against traditional software price discounting schemes on the diffusion dynamics of OSS. An Agent-Based Computational Economics (ACE) approach is adopted to develop a comprehensive simulation model to investigate the aforementioned research problems. Although, desktop operating system software is used as an exemplar to investigate the diffusion of its open source and proprietary alternatives, the framework proposed in the dissertation is general enough to be applied in the investigation of diffusion of other kinds of software as well.

DEDICATION

This work is dedicated to my daughter, Khadijah Zaffar.

ACKNOWLEDGEMENTS

I would like to thank several people for their help and support over the past few years without which I would have had a hard time finishing this dissertation. Dr. Ram Kumar: my dissertation chair, for his constant support and guidance over the course of my degree. In the physical absence of immediate family members, he acted as a father figure and a friend when I needed more than just guidance in terms of research. Dr. Kexin Zhao: for her frank and objective reviews and the time she had to spend away from her family to provide excellent feedback and support. Dr. Moutaz Khouja: for his amazing personality that made it fun to do research with him. Dr. Cem Saydam: for his patience as I tried to find my way in the early part of the program and his constant support as I reached the end. Dr. Yogendra Kakad:, for his insightful comments that helped shape the dissertation and will undoubtedly help in future research. In addition to the aforementioned committee members, I would like to thank Dr. Mirsad Hadzikadic and Dr. Susan Winter, both of whom helped me grow as a researcher and continue to provide invaluable advice. I would also like to acknowledge Mr. Jonathan Halter for providing exceptional technical support with the use of the University’s cluster.

It is difficult to thank my parents, Afzal and Khalida Zaffar, in a few lines. They are one of God’s blessings and a life-time of gratitude would not compare to what they have done for me. Finally, I would like to thank my family: Mariam (my wife), for her love and patience as I went through the most challenging and tense parts of my degree; my brother (Dr. Muhammad Fareed Zaffar) whose presence in the US in the early years was crucial in keeping me on track and my sisters, Nausheen and Naveed, whose prayers and emotional support pulled me through the tough times of my program.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION AND LITERATURE REVIEW

1.1 Introduction

There is increased interest in the phenomenon of open innovation in general and open source software (OSS) in particular. A growing number of servers and databases are already running on OSS (Wheeler, 2005). Furthermore, an increasing number of organizations are either looking to move completely to open-source systems or they are making their existing systems compatible with OSS (Wheeler, 2005). OSS has been studied from various perspectives such as adoption and diffusion (Bonacorsi and Ross 2003), pricing (Kim et al, 2006), licensing (Tirole and Lerner, 2005), contribution (Lerner and Tirole, 2001), quality and release management (Michlmayr, 2005) etc.

This research explores the diffusion dynamics of OSS. It recognizes that OSS diffusion is a complex phenomenon and emphasizes the need to study it using multiple theoretical perspectives. OSS is an innovation and hence can be studied from the diffusion of innovation perspective (Rogers, 1995). OSS can also be viewed as a type of standard and hence can be examined using the growing body of research on standards (Zhu et al, 2006). Since OSS is a software product, characteristics of software products such as upgrades (Ngwenyama et al, 2007), licensing and support also influence its diffusion.

A review of the previous literature suggests that there is some understanding of the factors that affect adoption and diffusion of OSS (Bonaccorsi et al, 2006; Kim et al, 2006; Masanell and Ghemawat, 2006; Zhu et al., 2006). However, the manner in which these factors jointly influence the dynamics of OSS diffusion has not received adequate attention. Through a series of essays, this dissertation investigates the diffusion of OSS from various perspectives.

The first essay identifies and examines the interaction effects between key determinants of diffusion of OSS. While drawing from the literature on OSS, standard diffusion and innovation diffusion, an agent-based computational economics (ACE) approach is adopted to develop a simulation model of OSS diffusion. The model illustrates the effect of the following key, yet under researched, variables on the diffusion of OSS: i) network topology; ii) network density, iii) variability in the support cost for OSS; iv) interoperability costs between different software; v) frequency of upgrades of competing proprietary software (PS); and vi) initial proportion of OSS adopters. Specifically, we address the following research question: *How do key variables individually and collectively affect the diffusion dynamics of OSS?* To the best of our knowledge, this is one of the first studies to examine the effect of upgrades on the diffusion of two competing software. The agent-based computational economics approach (Tesfatsion and Judd, 2006) used in our model allows for significant agent (OSS or PS adopter) heterogeneity in terms of size, planning of upgrades, technical competence with OSS, and support costs of OSS and allows integration of both economic and social concepts in one model. The desktop operating system (OS) market is used as an exemplar since some empirical data regarding its cost components is available.

The second essay shifts the focus from the importance of intrinsic firm-level factors to inter-organizational relationships on diffusion dynamics of OSS. A social networking approach is adopted to investigate the effect of structural characteristics of a network of organizations on the diffusion of OSS. Previous research has demonstrated that network structure can affect dissemination of information, knowledge and other social processes. Our objective is to investigate the impact of network structure on the diffusion of OSS in a network of firms. More formally, we pose the following research questions: *a) What is the relative importance of various individual-level structural measures in explaining the rate of diffusion of OSS b) What is the relative importance of group-level structural measures in explaining the rate of diffusion of OSS? c) Which of the structural measures are most effective in explaining the rate of diffusion of OSS?* The model devised in the first essay is used to investigate these questions.

The findings of the second essay motivate the third essay which demonstrates the effectiveness of pricing PS based on knowledge about the social network of consumers, in influencing the diffusion of OSS. Traditionally, software vendors offer discounts to encourage sales based on usage, quantity, and/or location. We explore the question that *if the network structure of consumers is known, would it be more profitable for the vendor to offer network structure-based discounts than any other type of traditional discounts?* Again, the simulation model developed in the first essay and the findings from the second essay are jointly used to explore this research question. Table 1 provides a broad overview of the three essays.

The remainder of this document has been organized as follows: The next two sections in Chapter 1 provide an overview of the literature on Open Source Software (OSS) and Agent-Based Computational Economics (ACE) in the light of this dissertation.

Table 1: Overview of the dissertation

	Essay One: Determinants of Diffusion Dynamics of Open Source Software	Essay Two: A Social Network Analysis of Diffusion of Open Source Software	Essay Three: The Impact of Network-Aware Pricing versus Traditional Software Pricing Schemes on Diffusion of Open Source Software
Research Questions	How do key variables other than price individually and collectively affect the diffusion dynamics of OSS?	What is the relative importance of various individual-level structural measures in explaining the rate of diffusion of OSS? What is the relative importance of group-level structural measures in explaining the rate of diffusion of OSS? Which of the structural measures are most effective in explaining the rate of diffusion of OSS?	Is network-aware software pricing more effective than a traditional software pricing scheme?
Theoretical Foundation	Diffusion/diffusion of Innovations/Standards, Open Source Software, Agent-based Computational Economics	Social Network Analysis	Social Network Analysis and Software Pricing
Motivation	Call in previous research to explore “strategic variables other than price” to “better understand the drivers of adoption” (Masanell and Ghemawat, 2006: p. 1083) Lack of a comprehensive model/framework that collectively investigated the effect of critical variables on diffusion of OSS	The results from the first essay showed that network structure is an important variable. We wanted to investigate it in more detail using literature on social networks Given the growing emphasis on global collaboration in the business world and the interdependence between organizations while making technology decisions, in this paper, we propose an investigation of diffusion of OSS through social network analysis (SNA)	The results from the second essay showed that strategically located firms in a network can affect software diffusion. This information can be used in different ways by a software vendor. We decided to demonstrate its use by incorporating it in a simple price-discounting strategy to highlight the importance of location information.

Since OSS remains a very broad area of investigation, it is only appropriate that its understanding and scope of investigation is laid out in the context of this research. Furthermore, since agent-based modeling has been adopted as the method for investigation, it is necessary that this choice is described in detail and justified at the outset as the most appropriate methodology given the context of this research. Chapters 2, 3 & 4 discuss the three essays. Each of these chapters provides a brief review of relevant literature to motivate the research questions. This is followed by a description of the planned experiments and analyses. At this point, the first essay is complete. Therefore, Chapter 2 also provides a detailed analysis of results, discussion and contributions of the study to both research and practice. Chapter 5 concludes the dissertation proposal with an update on the progress in the second and third essays, and a timeline for completing the remaining work.

1.2 Literature Review

The following subsections provide a review of the relevant literature on OSS, diffusion of OSS and agent-based computational economics.

1.2.1 Open Source Software (OSS)

Open source software (OSS) is any piece of software whose source code is made publicly available under terms that follow the ‘Open Source Definition’ (Perens, 1999: pp. 171-188). Generally, such software is freely available online.

However, companies such as Red Hat and Ubuntu charge a fee for providing support and complementary services. There are certain aspects of OSS that are distinctly different from proprietary software. We have modeled some of these aspects in our paper. These include license costs, upgrade costs, timing and frequency of upgrades, support and required level of technical expertise (Economides and Katsamakas, 2006; Gray, 2005; Guth, 2007; Hissam et al, 2002; Kamhorst, 2002; Leading Edge Forum, 2004). License and Upgrade costs: OSS adopters face zero or low license/upgrade (Kim et al, 2006; Masanell and Ghemawat, 2006; Tirole and Lerner, 2005). For example, Ubuntu's desktop Linux distributions can be downloaded for free from Ubuntu's website. In this paper we assume zero license/upgrade costs for OSS. Timing of Upgrades: In the case of PS, upgrades tend to be vendor-driven, whereas in the case of OSS, upgrades are generally demand driven. As a result there is uncertainty regarding the timing of new releases in the case of PS, whereas OSS upgrades are released more frequently and on a regular basis (Michlmayr, 2005; Raghunathan et al, 2005; Tawileh and Rana, 2006; Zhao and Elbaum, 2003). Recently "Microsoft said it would return to a goal of releasing major OS [operating system] upgrades every four years, with at least one minor release between each major" (Keizer 2007). It is important to note that here we are only referring to major upgrades since minor upgrades or patches are frequently released by both proprietary and open source software vendors. Furthermore, with a PS upgrade, the vendor eventually withdraws support for the previous version, thus, in many cases, forcing the customer to upgrade to the latest version (Bowman, 2006) or support the software on its own. For example, over the last few years Microsoft has withdrawn support for Windows 98 and ME (Bowman, 2006). On the other hand, with an open source operating system such as Linux, there is little or no coercion from the vendor to upgrade to newer versions. To model these differences related to upgrades, we assume that i) the support cost for a PS customer increases if its version is two or more versions older than the vendor's current version. There is no such increase in support costs for OSS customers using older versions of OSS. ii) OSS upgrades are available more frequently than PS upgrades. Support: Practitioner literature indicates that there is uncertainty regarding the support costs of OSS.

However, there is guaranteed support for PS. For example, in addition to vendor support, firms using OSS have the option of supporting the software on their own based on their technical capabilities or seeking help from online OSS development communities. Hence, it is difficult to ascertain overall support costs for a typical OSS adopter. However, there is guaranteed support for PS. For example, Microsoft offers an initial period of free unlimited support followed by paid support per incident [Microsoft's website]. We incorporate this difference by charging variable support costs for OSS and fixed support costs for PS. Technical expertise: Given the nature of development and support of OSS, a certain level of technical expertise is required to use OSS. Customers who do not have sufficient level of technical expertise will rely more on external support which tends to be uncertain in the case of OSS (Kim et al, 2006; Leading Edge Forum, 2004; Lin, 2008). This is less of an issue in the case of PS. We incorporate this important aspect in our study by modeling the level of technical capability for OSS adopters which is tied to the support costs faced by the adopter.

Given these differences between OSS and PS, OSS has been studied from various perspectives such as diffusion (Bonaccorsi and Rossi, 2003), pricing (Kim et al, 2006; Mustonen, 2003), licensing (Tirole and Lerner, 2005), contribution (Lerner and Tirole, 2001; Xu, 2006), development (Chakravaty et al, 2007; Mockus et al, 2000; Norris, 2004; Ruffin and Ebert, 2004; Scacchi, 2002), trust (Hissam et al, 2002), knowledge sharing (Sowe et al, 2008), quality and release management (Michlmayr, 2005; Raghunathan et al, 2005; Tawileh and Rana, 2006; Zhao and Elbaum, 2003), and evolution of software (Kamhorst, 2002; Yu, 2008). However, in this study we limit our attention to the diffusion of OSS and focus on key findings in the literature on OSS diffusion in the following subsection.

1.2.2 Diffusion of OSS

Diffusion of innovations is a very broad area of investigation. Fichman (2000) and Westarp and Wendt (2000) provide detailed reviews. The body of research within this area can be whittled down to the investigation of "three basic research questions: RQ1: What determines the rate, pattern, and extent of diffusion of an innovation across a population of potential adopters? RQ2: What determines the general propensity of an organization to adopt and assimilate innovations over time? RQ3: What determines the propensity of an organization to adopt and assimilate a particular innovation?" (Fichman, 2000: pp. 106-107). We address both RQ1 and RQ3: the specific innovation is OSS with firms forming the population of potential adopters. According to Westarp and Wendt (2000), i) some diffusion models investigate direct impact of neighbors; whereas ii) some investigate the impact of social structures on a firm's decision to adopt a given innovation. Our proposed model investigates both the direct impacts of neighbors as well as social structures, on a firm's adoption decision regarding OSS.

Prior studies have developed empirical, analytical and simulation models to investigate the factors that affect OSS adoption and diffusion. Bonaccorsi and Rossi (2003) developed a simulation model to study adoption and diffusion of OSS. They simulated a network of N firms (agents). All firms were using proprietary software at the start of the simulation. The software adoption decision was based on the perceived intrinsic value of open source software, the network externality and coordination factors (based on other member-firms in the network). The study concluded that OSS diffusion depended on the initial distribution of intrinsic values assigned to the technology by the agents. Dalle and Jullien (2001) proposed that any firm would choose OSS over PS if its local and global benefits outweighed its idiosyncratic preferences. The concept of 'idiosyncratic preferences' is in some ways similar to the one that was later used by Bonaccorsi and Rossi as 'intrinsic value' (2003). Both the local and global benefits were considered to be a function of the number of participants in a firm's network (including firms using the same or different standards). Mustonen (2003) showed through mathematical modeling that under certain market conditions both proprietary and open source software could co-exist. However, the firm selling the proprietary software must carefully evaluate pricing strategies. Kim et al (2006) studied two types of consumer firms (high/low-type based on internal technical capability) and three different types of pricing schemes for OSS (commercial, dual licensing, and support) under different market conditions (monopoly and duopoly). Using mathematical modeling, they were able to demonstrate various feasible pricing strategies in both monopoly market and duopoly market for PS as well as OSS vendors.

Table 2 provides an overview of the main studies on diffusion of OSS. Due to the lack of space, some of the column headers had to be abbreviated. Here they are in order from left to right: perceived value/benefit, network effects (local and global), network density (measured as the number of connections with neighbors), price (license cost), switching cost, support cost, other costs (training costs, setup costs etc.), risk (of adopting

Table 2 : Determinants of diffusion studied in the literature

	Perceived Value	Network Effects	Network density	Price	Switch. Costs	Supp. Costs	Other Costs	Risk	Cons. Hetero.	Network Topology	PS Upg. Cycle	Firm Size	OSS Tech. Cap
Delre et al (2007)	✓	✓	✓						✓	✓			
Westarp and Wendt (2000)	✓	✓	✓	✓					✓				
Bonaccorsi et al (2006)	✓	✓		✓	✓		✓					✓	✓
Dalle and Jullien (2001)	✓	✓		✓					✓				
Kim et al (2006)	✓			✓	✓	✓							✓
Weitzel et al (2006)	✓	✓	✓	✓	✓	✓	✓			✓		✓	
Chatterjee & Eliashberg (1990)	✓	✓		✓				✓	✓				
Bonaccorsi and Rossi (2003)	✓	✓											
Sundararajan (2005)	✓	✓	✓						✓	✓			
Zhu et al (2006)	✓	✓			✓	✓						✓	✓
Masanell & Ghemawat (2006)	✓	✓		✓									
Economides & Katsamakas (2006)	✓	✓		✓	✓								
This paper	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

another software), consumer heterogeneity, network topology, length of PS upgrade cycle, firm size and technical capability (with respect to OSS). On a cursory level diffusion of OSS in an organizational network can be compared with the diffusion of a product or a virus or epidemic throughout a network. It can be argued that traditional diffusion or epidemic models can be applied to investigate OSS diffusion (see for example, Dodds and Watts, 2005). SIS (Susceptible, Infected, Susceptible) and SIR (Susceptible, Infected, Removed) are two common classifications of such models (see Delre et al, 2007 for a more detailed review). However, there are some important differences between the spreading of an epidemic or a virus and software or innovation diffusion which limit the applicability of the traditional models: a) unlike viruses or epidemics, software does not diffuse merely by virtue of a connection with other firms that have adopted the same software, b) in software diffusion, firms may not be equally 'susceptible' to the adoption of a software. Furthermore, the susceptibility is dependent on a multitude of social and economic factors such as number of inter-organizational relationships, the importance of those individual relationships, the size of the organization in question, internal cost benefit analysis etc. Therefore, instead of simplifying the diffusion process at the micro level, it is more appropriate to incorporate the complexity of the individual nodes, their diverse inter-organizational relationships and their decision-making process. Ultimately, these factors collectively drive network-wide diffusion. Agent-based simulation models facilitate the modeling of such complex, heterogeneous behaviors at the micro (agent) level to investigate macro-level phenomena (Miller and Page, 2007).

1.2.3 Agent-Based Computational Economics (ACE)

Agent-based computational economics (ACE) is a relatively new and growing area of economics (Tesfatsion and Judd, 2006). At its core lies the notion of agent-based modeling and simulation. This section introduces ACE and discusses its use to study diffusion dynamics of OSS. ACE can be applied to a problem by defining a set of agents with related attributes, behaviors and fitness function; the simulation environment and the overall performance-measuring objectives of the environment. Depending on the nature of the system being modeled, there can be many types of agents (cells, species, individuals, firms, nations etc) and each type of agent can behave differently. These agents could act independently, collaboratively or competitively. Over time, as a result of repeated interactions, aggregate behavior is likely to emerge that was not originally programmed in the system (Waldrop, 1992).

The open source market exhibits similar characteristics: consumer firms, proprietary software vendors, open source support-providing firms (all acting as agents), working towards their individual goals (profit maximization and/or sustainability in the market) while taking different actions (adopting different standards, pricing strategies etc.). In this research, each agent represents a consumer firm. All agents have a set of attributes (such as whether the firm uses OSS, its license, support, training costs etc.). Each agent can be influenced by the behavior of other agents to different degrees. The agents have to choose between upgrading their existing software and switching to the alternative software based on their objective or fitness function, which measures the average net annual cost savings over a planning horizon. Any meaningful behavior exhibited by the system arises from the collective behavior of the group of agents.

This research makes use of simulation modeling to investigate the diffusion dynamics of OSS. There are several reasons for choosing this methodology given the context of the research. First, in experimental contexts where significant amount of empirical research is lacking, simulations can illuminate or eliminate avenues for future research. To that extent, the actual numbers used in the simulations are not as important as the framework and resulting insights. This is an established notion in simulation based studies especially when simulations are used to "stimulate discussion" on a particular topic. Second, we find simulations being used in the literature on adoption and diffusion of OSS, standards and other phenomena (Bonaccorsi and Rossi, 2003; Dalle and Jullien, 2001; Delre et al, 2007; Mustonen, 2003; Wheeler, 2005). Third, agent-based simulations, which we intend to use, facilitate the development of more sophisticated models that are not limited by considerations of mathematical tractability. Such models allow joint investigation of a combination of social as well as economic factors (Tesfatsion and Judd, 2006). Hence, agent-based modeling facilitates crossing boundaries (economic vs. social modeling) in diffusion research. Literature also suggests that agent-based modeling may be most suited for modeling diffusion compared to other methodologies such as statistical models, analytical models and qualitative studies (Huang and Kapur, 2007). Finally, increasing computing power has made these computationally expensive simulations feasible (Srbljinović and Skunca, 2003)

Simulation models, like empirical or analytical models have to be validated. Lazer and Friedman (2007) state four key criteria for assessing simulation based studies: a) "verisimilitude" or face validity i.e. the behavior of the model should closely follow reality. This is a well established practice in simulation based studies where some intuitive results or results that corroborate theory or reality are used to build confidence in the model (see for example, Dutta and Roy, 2005; Manzoni and Angehrn, 1997; Ahituv et al, 1998; Abdel-Hamid 1992; Jones et al, 2006; Kwon et al, 2007). b) robustness, i.e. the results should hold in the face of trivial changes to the model c) replicable, i.e. other researchers should be able to completely replicate the results of this study and, d) "non-obvious non-trivial results" i.e. the model should allow the researchers to make new and insightful observations. In each of the three essays, the model and the results will be assessed based on these criteria. There is an additional important point that needs to be emphasized with reference to the validation of agent-based models in particular. As mentioned earlier, the whole concept of agent-based modeling or agent-based computational economics revolves primarily around the behavior of the individual agents. Therefore, once the modeled behavior of an individual agent can be justified based on both theoretical and practical grounds that can lend implicit support to the validity of the model and possible generalizability of the results and behavior of the model.

CHAPTER 2: DETERMINANTS OF DIFFUSION DYNAMICS OF OPEN SOURCE SOFTWARE

2.1 Introduction

There is increased interest in the phenomenon of open innovation in general and open source software (OSS) in particular. A growing number of organizations are either looking to move completely to open-source systems or they are making their existing systems compatible with OSS (Wheeler, 2005). However, despite this rising popularity, there is limited understanding of the factors that affect the diffusion of OSS at the organizational level (Masanell and Ghemawat, 2006). Prior studies in this area have made a valuable contribution by identifying some factors. However, they either did not address a broad range of critical factors simultaneously in one model or did not investigate the impact of critical factors in enough detail leaving some gaps in the literature. This is particularly evidenced by a recent call for more research to identify “strategic variables other than price [to] better understand the drivers of adoption” and diffusion of OSS (Masanell and Ghemawat, 2006: p. 1083). This study recognizes that OSS diffusion is a complex phenomenon and emphasizes the need to study it using multiple theoretical perspectives. OSS is an innovation and hence can be studied from the diffusion of innovation perspective (Rogers, 1995). OSS can also be viewed as a type of standard and hence can be examined using the growing body of research on standards (Zhu et al, 2006). Since OSS is a software product, characteristics of software products such as upgrades (Ngwenyama, 2007) and support also influence its diffusion.

We address the gaps in the literature by developing an integrated framework that simultaneously investigates a heterogeneous set of social and economic factors on the diffusion dynamics of OSS using an Agent Based Computational Economics (ACE) approach (Tesfatsion and Judd, 2006). We then apply that framework to illustrate how critical factors other than price affect the diffusion dynamics of OSS.

The proposed model illustrates the effect of the following key, yet under researched, variables on the diffusion of OSS: i) network topology; ii) network density; iii) variability in the support cost for OSS; iv) interoperability costs between different software; v) frequency of upgrades for competing proprietary software (PS); and vi) initial proportion of OSS adopters. Specifically, we address the following research question: How do key variables other than price individually and collectively affect the diffusion dynamics of OSS?

The desktop operating system (OS) market is used as an exemplar in this study since some empirical data regarding its cost components are available. Our results demonstrate that a) interoperability costs, variability of OSS support costs, and duration of PS upgrade cycle are major determinants of OSS diffusion; b) there are interaction effects between network topology, network density and interoperability costs, which strongly influence the diffusion dynamics of OSS; c) vendors should consider several strategic variables besides price such as interoperability costs, upgrade cycle, network topology and network density that significantly impact OSS diffusion; d) the proposed framework can be used as a building block to further investigate complex competitive dynamics in software markets in general and OSS markets in particular.

This chapter is organized as follows: the next section motivates the research problem. This is followed by the details of the proposed model, description of results and discussion. Finally, conclusions and ideas for future research are discussed along with the limitations of this research.

2.2 Literature Review

For a detailed review of the literature in the context of this essay, please refer to the literature review section in Chapter 1. In this section, an assessment of the literature is being provided in an attempt to motivate the research question.

There are three key reasons that have motivated the development of our model. First, as mentioned earlier, there has been a call in previous research to explore "strategic variables other than price" to "better understand the drivers of adoption" particularly in the context of Windows and Linux (Masanell and Ghemawat, 2006: p. 1083). We model the effect of the duration of the PS upgrade cycle and threat of withdrawal of support from the PS vendor, which have been missing in the prior literature, on the diffusion dynamics of OSS. This is important because firms consider the timing and frequency of releases offered by vendors when making their own software upgrade decisions (Ngwenyama et al, 2007). Second, the factors in Table 2 identified have never been investigated simultaneously in one study. We propose a comprehensive model that aims to study the individual as well as interaction effects of these factors on the diffusion dynamics of OSS. The factors identified in Table 2, such as network topology, network density and interoperability costs have been studied in prior research and that research serves as a theoretical basis for our model. However, given the interactive nature of these critical factors in the context of software adoption decisions, a model is needed that allows simultaneous investigation of these factors. Third, we believe that even the factors that were included in some of the previous studies were not studied in depth or received inadequate attention. For example, the changing roles of factors such as network topology over time, the issue of variability or uncertainty regarding OSS support costs and its impact on diffusion of OSS have not received adequate attention. Therefore, we identify six key, yet under researched, factors and study the effect of a range of values of these factors on the diffusion dynamics of OSS.

Therefore, we identify six key, yet under researched, factors and study the effect of a range of values of these factors on the diffusion dynamics of OSS. These factors include, network topology, network density, OSS support costs, interoperability costs, frequency of PS upgrade cycle and initial proportion of OSS adopters. The following subsections provide a more detailed literature review of each of these key variables.

2.2.1 Network Topology and Network Density

The effect of network topology and/or network density on diffusion of innovations has been studied from various perspectives (Delre et al, 2007; Fichman 2000; Harkola and Greeve, 1995; Lin, 2008; Westarp and Wendt, 2000). We examine OSS diffusion under three types of network topologies based on the previous literature: random, clustered and small world. These topologies exhibit different degrees of cliquishness (Watts and Strogatz, 1998). Cliquishness is the degree to which a node’s neighbors are each others’ neighbors. The random network exhibits the lowest degree of cliquishness, followed by small-world and clustered networks. The clustered network has cliques which are highly interconnected with each other. In the small world network not all cliques are highly interconnected with each other. Network density is modeled in our paper as the size of the local neighborhood or number of immediate neighbors. We used the Watts and Strogatz algorithm (Watts and Strogatz, 1998) to simulate the three network topologies, which have been used to study diffusion of fashions (Delre et al, 2007) and effect of network topologies and network densities on contribution to OSS projects (Singh, 2007). Key studies investigating the impact of network topology and network density on the process of diffusion are discussed below.

Westarp and Wendt (2000) demonstrated through simulations that network topology does have an impact on purchase decisions made by consumers regarding available software. However, in contrast to our model, their study did not take into account a) license, setup, and support costs, b) the heterogeneity of firms in terms of their size and interaction with neighbors which ensures that network effect benefits are not the same for all consumers and they vary in a non-linear fashion. Delre and colleagues (2007) studied the effect of social factors and word-of-mouth processes on the consumer decision-making process. In their model, the adoption decision was affected by "external marketing effort" and social pressures imposed on and by the consumers in their neighborhoods. However, in contrast to our model, their study has several limitations. First, the utility derived from adoption does not take into account the individual cost components of the software over a unique planning horizon for each firm. Second, the decision of each firm is being directly influenced by neighbors' neighbors, which is less realistic in the case of software adoption where the decision is being directly influenced by the immediate group of neighbors. Harkola and Greve (1995) compared the effect of cohesion and structural equivalence on the diffusion of technology innovations in an empirical study. They concluded that the effect of structural characteristics on diffusion varied based on the different densities of the network. However, we believe that their approach is different from ours in two important respects. First, they study diffusion of innovation at the level of individuals whereas we study it at the organizational level. Second, they do not consider the possibility that other strategic factors such as interoperability costs may interact with the effect of network characteristics on the diffusion of innovation.

In summation it can be said that there is some understanding of the effect of network topology and network density on the process of diffusion. However, as a result of some shortcomings and/or different contexts of research, insights from previous models cannot be directly applied to the investigation of diffusion dynamics of OSS.

We examine OSS diffusion under three types of network topologies based on the previous literature: random, clustered and small world. Network density is modeled as the size of the local neighborhood or number of immediate neighbors – higher number of immediate neighbors results in an denser network. Watts and Strogatz algorithm (Watts and Strogatz, 1998) was employed to simulate the three network topologies, which have been used to study diffusion of fashions (Delre et al, 2007) and effect of network topologies and network densities on contribution to OSS projects (Singh 2007). In our network of firms, two firms have a link between them if they conduct transactions with each other. These transactions at the basic level could represent an exchange of documents, reports, data, or any other kind of electronic information between neighboring firms. On each link, the firms conduct a certain number of these transactions and as a result, the link is undirected. If the neighboring firms are using incompatible software, they will both incur interoperability costs per transaction. These interoperability costs will in part affect the decision of each firm in upgrading its existing software or switching to the alternative software. In the clustered network, each firm's neighbors are likely to be each others' neighbors as well. This is highly unlikely in a random network where by virtue of having random connections, a firm's neighbors may have no relationship with each other. In the small-world network, both kinds of firms exist, i.e., some neighbors are well connected with each other and some are not connected to each other.

2.2.2 OSS Support Costs

The impact of support costs on diffusion has been recognized in the previous literature (see Table 2). However, this impact needs to be investigated further in the context of OSS since practitioner reports suggest that there is considerable uncertainty regarding the magnitude of OSS support costs (Leading Edge Forum, 2004). This is understandable since not all firms have sufficient level of technical expertise to support non-vendor-backed software such as OSS or even a vendor-backed OSS that may not be compatible with other software. Firms with programmers that participate in OSS development could have significantly lower support costs when compared to other firms. Therefore, not only do we consider the possibility of uncertainty regarding OSS support costs, we weigh it with respect to a firm's technical capability in managing OSS (Kim et al. 2006).

2.2.3 Interoperability Costs

Previous literature has indicated that interoperability issues play a significant role in standard adoption (Chen and Forman, 2006; Katsamakas and Xin, 2005; Wilkins et al, 2004). In this paper we assume that when neighboring firms conduct transactions with each other (i.e. exchange data), they incur overhead costs per transaction if they are using different systems. These costs are aggregated over a volume of transactions to compute interoperability costs incurred by each firm with each one of its neighbors.

Switching platforms (operating systems) can have several implications on a firm's existing portfolio of applications (Gray, 2005). In the current desktop operation system market, Microsoft has a clear dominance. Thus, even supporters of Linux concede that the pool of compatible applications for Linux is smaller compared to the pool of compatible applications for Windows. This difference may diminish over time. However, right now firms can expect to incur some interoperability costs to communicate with partners using a different platform. These interoperability costs could take the shape of additional effort (labor hours) or software required to ensure interoperability of applications. None of the previous studies on diffusion of OSS have specifically addressed this issue.

2.2.4 Duration of PS Upgrade Cycle and the Threat of Withdrawal of Support

To the best of our knowledge, availability and timing of technology upgrades in the context of adoption and diffusion of OSS have not been studied. In this research, we only consider major upgrades. Minor upgrades and patches are frequently released for proprietary as well as open source software. However, it is when firms are making major technology upgrade decisions that they may decide to 'jump ship' (McAllister, 2006). If firms neither upgrade nor switch to a different software, they anticipate that soon the support for the existing version will be withdrawn and they will have to reconsider their decision (Bowman, 2006). Availability of hardware upgrades also influences the decision of firms (McMillan, 2004). Furthermore, there has been extensive research on the release cycles of open source software which points to the coordination issues in release management (Michlmayr, 2005). Although release management is beyond the scope of this model, it suggests that in the absence of vendors, OSS upgrades tend to be demand-driven whereas PS upgrades are vendor-driven. This is understandable since with open source software, consumers have the option to initiate and/or become involved in the development of a desired upgrade. On the other hand, with proprietary software, consumers have to either wait for the vendor to release the upgrade, have the upgrade custom-made by the vendor, or purchase the required change through a third party.

The timing of upgrades from the consumer perspective can affect the diffusion dynamics as well. Mukherji et al (2006) state that "unlike other types of investment decisions, firms would benefit from a long term "plan" for investment in IT upgrades" (p. 1685). Furthermore, it is important that long-term costs are taken into account during upgrade decisions since choice of an operating system is more like a platform decision that affects hardware, existing application portfolio, staffing/training issues etc. (Gray, 2005). The duration of this "long-term" may vary for firms depending on their size and industry. Hence, firms will face different annual upgrade costs and that must be factored into the decision-making process. As discussed earlier, in the context of the desktop OS market, OSS and PS vendors have different upgrade frequencies and PS vendors withdraw support for earlier versions once upgrades are released.

2.2.5 Initial Proportion of OSS Adopters

Literature on innovation diffusion suggests that the mass of current adopters can affect the non-adopters and vice versa (Markus, 1990). The concept of "critical mass can be defined as the minimum amount of some resource (people, money, etc.) needed before another condition or product explodes into existence" (Dick, 2004: p. 235). This implies that the size of the existing base of adopters of an innovation can strategically impact the future diffusion process.

In the desktop operating system market, Microsoft has by far been the most dominant player, becoming the de facto standard. Based on prior studies, we consider two different proportions for the initial number of OSS adopters in a population of consumers (10% in (Wheeler, 2005) and 30% in (Dalle and Jullien, 2001)).

2.3 Research Methodology

This research adopts an agent-based computational modeling approach towards the investigation of the research problem. Please refer to Chapter 1 for an overview of this modeling approach. This section provides a detailed description of the proposed simulation model and simulation parameters.

In our diffusion model, agents are firms where each firm is using a proprietary or open source desktop operating system (for example, Microsoft XP and Red Hat Linux). Each firm has to decide whether to upgrade its existing software or switch to the alternative software. As is the case for desktop operating systems, we assume that the proprietary software dominates the network at the beginning of the simulation. Initially, firms using OSS represent a small percentage of the total population and are randomly distributed. Each firm will periodically evaluate its technology (hardware and/or software) based on its planning horizon. This notion of a planning horizon is based on Mukherji et al's statement that "in the case of frequent upgrades, it is important for firms to decide the frequency at which its technology must be replaced" (Mukherji et al, 2006 : p. 1685). Furthermore, they state that firms generally adopt a "long term 'plan' for investment in IT upgrades". Therefore, in our model, firms decide whether to upgrade the existing software or switch to the other software at the beginning of their planning horizon. This notion of a firm's technology upgrade frequency or planning horizon is closely tied to the technology vendor's release cycle. As an example, consider a firm that has a Windows operating system deployed on its desktops. If this firm chooses to upgrade its existing desktop operating system, it will only choose to do so after a newer version has been released. In other words, the desktop operating system upgrade frequency of the firm cannot be shorter than Microsoft's Windows release upgrade frequency. Therefore, we model the length of a firm's planning horizon (PH) to be greater than or equal to the upgrade cycle (UC) of its respective vendor. The longer the PH, the more reluctant a firm is to consider software changes due to reasons such as organizational inertia, risk aversion or lower innovativeness. To simulate that behavior we chose a range of values of PH for the firms and these values were distributed across the entire population in proportions similar to a S-shaped curve (i.e., 20% firms have a PH=UC, 30% have a PH=UC+1, 30% have a PH=UC+2, and 20% have a PH=UC+3). Furthermore, firms are connected in a network in which each link represents a business relationship. Connected or neighboring firms conduct business transactions with each other which could represent an exchange of documents, reports, data, or any other kind of electronic information. If neighboring firms are using incompatible software (or software with interoperability issues involved) they will incur interoperability costs per transaction. Each firm considers whether to adopt the other software or upgrade its existing software, at time t based on the following decision function:

$$\frac{C_{t+1}^{U}-C_{t+1}^{S}}{C_{t+1}^{U}} \geq (1-\alpha)\times(1-n) \quad \begin{cases} TRUE & \Rightarrow \text{Switch to a different software} \\ FALSE & \Rightarrow \text{Upgrade the existing software} \end{cases} \qquad (1)$$

where

C_{t+1}^{U} represents annual costs if the firm decides to upgrade its software

C_{t+1}^{S} represents annual costs if the firm decides to switch to the other software

α represents the *degree centrality* of a firm

n represents the proportion of a firms' neighbors who use the proposed new software.

The left-hand side (LHS) of Equation (1) represents expected annual cost savings or expected benefit from switching to the alternative software as against upgrading the existing software. These costs include license, setup, training, support and interoperability costs. . However, with new innovations such as OS, these costs could be uncertain [18, 21]. Hence, consistent with economics and innovation theory adopters could expect the value of the left-hand side to be different from zero depending on their risk preferences, and how they process uncertain information [6]. The right-hand-side (RHS) of Equation (1) represents a threshold unique to each firm. For example, if the left-hand-side for a firm A is evaluated to X then a firm such as Wal-Mart and a firm such as one of Wal-Mart's suppliers will value these savings differently and hence require LHS values to be greater than different thresholds (RHS values). Thresholds could represent risk preferences [6] and/or social influence [9], and are well-accepted in diffusion research. Furthermore, prior research also suggests that firms anticipate the decision of their neighbors when making their own technology decisions [48] and they may not place an equal level of emphasis on the decision of their neighbors. The RHS of our decision function combines economic and social perspectives to incorporate this multi-level heterogeneity by stating that a firm's valuation of its savings will be determined by a combination of factors: its level of social influence (or centrality, captured by α) and the decision of its neighbors (captured by n). A highly influential/central firm such as Wal-Mart will not worry too much about the decision of its neighbors (who have low centrality) when it makes technology adoption decisions. Therefore, for a firm such as Wal-Mart, (1–α) will become a low weight attached to the decision of its neighbors (1–n). However, it could have a nonzero threshold as discussed above. On the other hand, let's say for one of Wal-Mart's suppliers, with low-centrality, the decision of its (few) neighbors might be more important so a low α will result in a higher weight (1–α) attached to the decision of the neighbors. The added interesting element is that the software choice of neighbors also affects the LHS of the decision function. If more neighbors are using the other software, then holding centrality (α) constant, a firm is more likely to be pulled towards the choice of its neighbors when interoperability costs are high. Interoperability costs reduce expected cost savings and make a firm likely to conform to its neighbors. However, the firm is not bound to conform. As mentioned earlier, a highly central firm will attach low weight to the decision of its neighbors, which would mean a lower threshold or lower RHS which in turn would mean that even a small proportion of cost savings will encourage the firm to switch to the alternative even if all its neighbors continue to use a different software. Furthermore, in our simulations, the most central firm makes an adoption decision prior to its neighbors because it is less prone to social influence in making its own adoption decision. However, once it has made an adoption decision, it influences all its neighbors who make subsequent adoption decisions. For the same LHS and n, a more central firm (higher α or lower $1-\alpha$) is more likely to switch.

Going back to the simulation process, since firms upgrade to keep up with the latest technology/features and/or to maximize the utility they can derive from the existing software (Ngwenyama et al, 2007), we assume that if they do not switch, they will necessarily upgrade to the latest available version of their existing software. Firms upgrade under the assumption that upgrades offer quality advantages and that if firms do not find the other software (PS or OSS) viable, they will avail the quality improvements offered by the upgrades. This eagerness to avail quality improvements may not be the same for all firms. Hence, each firm has a different planning horizon.

A 3x3x3x3x2x2 study was designed in order to study six main variables. This study uses three different parameters each for the following: network typology (random, small world, clustered), network density (low, medium, high), OSS support costs (support costs slightly higher than PS on average with low variability; support costs slightly higher than PS on average with very high variability; support costs much higher than PS with very low variability), and interoperability costs (low, medium and high). Two different parameters are used for the length of the PS vendor's upgrade cycle (short and long) and the initial proportion of OSS firms (low and high). Fifty samples were drawn from respective distributions for each of the random variables, and the results were averaged. The simulation was run for 100 time periods, for each of the 324 combinations. Figure 1 illustrates the flow diagram for our model. The key parameter values used in the simulation are summarized in Table 3. Wherever possible, we used numbers obtained from the practitioner literature. The model was developed using NetLogo (Wilensky) – see Appendices C & D for details).

Simulations were extremely computationally intensive and were run on a Linux-based cluster which had 200 CPU cluster blade servers, Intel Xeon CPUs and gigabit Ethernet interconnections with 2TBs of dedicated network attached storage.

2.4 Results and Analysis

Two criteria were defined for diffusion: a) if the number of OSS adopters doubled, b) if the number of OSS adopters increased by 50% during the course of the simulation. All propositions remained the same under both conditions which indicates the robustness of the model. Actual numbers based on the first criterion of diffusion will only be reported here. Diffusion predominantly occurred with high initial proportion of OSS adopters and did not occur under very high OSS support costs. Out of 324 cases, diffusion occurred in 72 cases. Logistic regression was performed to test which of the six factors increased the likelihood of diffusion of OSS. The analysis revealed initial proportion of OSS adopters to be the most statistically significant factor in increasing the likelihood of diffusion of OSS, followed by interoperability costs, network density, high variability in support costs and network topology. The model correctly predicted about 86% of the diffusion cases and 96% of the non-diffusion cases. Table 4 shows a summary of results from the logistic regression.

Subsequent analysis focused on the cases in which diffusion did occur. Multiple linear regression revealed PS upgrade cycle to be statistically significant in predicting the diffusion of OSS as well. In order to determine the statistical significance within the levels of each factor, repeated measures tests were performed.

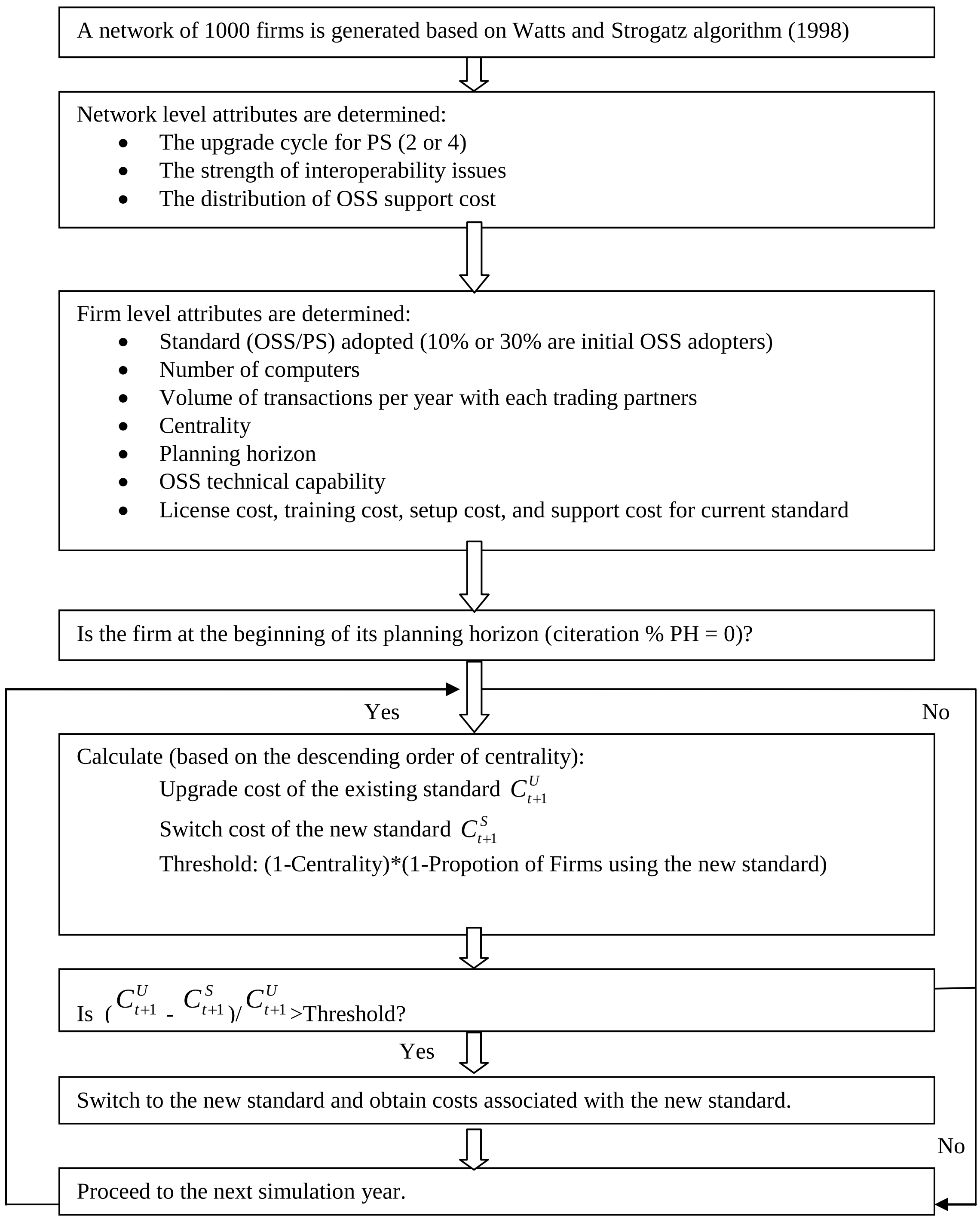


Figure 1 : Flow diagram for the basic simulation model

Table 3: Key Simulation Parameters

Attribute	Description
Network size	Number of firms in the network – 1000
Firm size	The number of computers each firm has is a proxy measure for firm size ~ U[100,500]
Initial number of OSS firms	Two different values were chosen : 10% (Wheeler 2005) and 30% (Dalle & Jullien 2001)
Network Topology	Three different network topologies were created using the Watts & Strogatz (1998) algorithm: random, small world and clustered.
Network density	This is measured by the average immediate number of neighbors of a firm. Three values were picked: 5%, 15%, 25% representing small, medium and large neighborhoods. 5% means that the local neighborhood of each firm will have 5% of the total firms in the network. This implies that larger neighborhood size will mean denser networks. With 3 network topologies and 3 sizes of neighborhoods, 9 different networks were created. The values for the neighborhood sizes were chosen to ensure that structurally the networks were different.
Upgrade cycle	This represents the duration between successive major upgrades offered by the PS or OSS vendors. We chose an upgrade cycle of 4 (long) and 2 (short) years for proprietary software (Keizer, 2007) and fixed the upgrade cycle for OSS to 2 years. Keeping in view the demand-driven aspect of OSS upgrades, we kept shorter upgrade cycles for OSS than for PS.
Planning horizon	PH indicates how often a firm conducts a major software upgrade. PH depends on the upgrade cycle of the software the firm is adopting. A range of planning horizons were assigned such that some firms had very short or very long planning horizons, majority had planning horizons in between these two extremes. The range of values were UC, UC+1, UC+2, UC+3 where UC represents the upgrade cycle of the firm's existing software (OSS or PS).
Volume of transactions	This represents the total number of transactions on each link in the network generated using a uniform random distribution U[100,500]. It was used to compute interoperability costs
Level of interoperability costs	If neighboring firms were using a different standard, we assumed that they incurred interoperability costs. The level of these interoperability costs was initially chosen to be 1, 3 and 5. However, once regression revealed the importance of this variable, a bigger range of values was tried for sensitivity analysis (0.1, 0.2, 0.3 … 0.8).
Current license Costs	$199, $0 per machine for PS and OSS respectively (Guth, 2007; Vaughan-Nicholas, 2006)
Training Costs	This was chosen to be $20, $30 per machine for PS and OSS respectively. A lower value for PS was chosen under the assumption that since PS already has a large installed base, new hires would be expected to be more familiar, hence easier to train, using PS than OSS.
Setup Costs	$325, $70 per machine for PS and OSS respectively (Vaughan-Nicholas, 2006)
Support Costs	Firms incur heterogeneous OSS support cost due to differences in degree of integration, customization, variability of OSS quality, lack of systematic version management and other factors (Kamhorst, 2002). Effective OSS support cost is determined by three normal distributions: N(60,15), N(60,60), N(250,50) depending on its mean value and variability. PS Support costs are kept fixed at $50 (Vaughan-Nicholas, 2006). Negative values were avoided by truncation to zero.
OSS technical capability	Firms' technical capability with respect to OSS are different (Kim et al, 2006) and are determined by a random variable drawn from N(0.3, 0.1)
Degree centrality	The more neighbors a firm has, the more powerful it is in influencing its partner's standard adoption decision and the more strongly it can be influenced by the decision of its neighbors.
Withdrawal of support	We model the threat of withdrawal of support by the PS vendor by doubling the support costs if the firm is 2 or more versions behind its vendor's current version.

Table 4: Beta weights, Wald statistic and Odds ratios from Logistic Regression

	B	S. E.	Wald	df	Sig.	Odds Ratio
Network Topology 1	-2.183	.972	5.038	1	.025*	.113
Network Topology 2	.000	.780	.000	1	1.000	1.000
Network Density	-48.441	10.439	21.531	1	.000**	.000
OSS Support Costs 1	-28.120	2498.950	.000	1	.991	.000
OSS Support Costs 2	3.754	1.032	13.232	1	.000**	42.685
Interoperability Costs	-2.961	.604	24.012	1	.000**	.052
PS Upgrade Cycle	.000	.335	.000	1	1.000	1.000
Initial Proportion of OSS adopters	53.979	10.986	24.141	1	.000**	2.772E23
Constant	.211	1.491	.020	1	.887	1.235

In most cases the diffusion curves were clearly separate, hence repeated measures tests, with number of OSS adopters as the dependent variable, were performed. However, when the diffusion curves from different parameter combinations overlapped or reached the same endpoint, the speed of diffusion was computed as follows and used as the dependent variable in the repeated measures tests:

$$\rho = \frac{1}{T}\left(\frac{\sum_{t=0}^{T} D(t)}{\sum_{t=0}^{T} f(t)}\right) \tag{2}$$

Where T represents the duration of the simulation (100 time periods in our model), “D(t) is the cumulative function of adopters at time t, and f(t) is the number of adopters at time t” (Delre et al, 2007, p. 193). ρ is useful when the diffusion paths to be compared reach the same endpoint (Delre et al, 2007). The following sections provide a detailed look into the main and interaction effects of these variables on OSS diffusion.

2.4.1 Main Effects

The analysis revealed that high interoperability costs favored the dominant standard by locking-in its users. Low interoperability costs, on the other hand, shifted the emphasis to other factors such as costs, network topology and network density thus reducing the possibility of lock-in (Figure 2). This meant that at the start of the simulation, high interoperability costs always prevented firms from switching to OSS. However, if due to other cost factors, OSS did manage to gain critical mass, the same interoperability costs hastened the diffusion of OSS throughout the network. This leads to the first proposition:

Proposition 1: When PS is the dominant software in the market, increasing interoperability costs reduces the diffusion of OSS

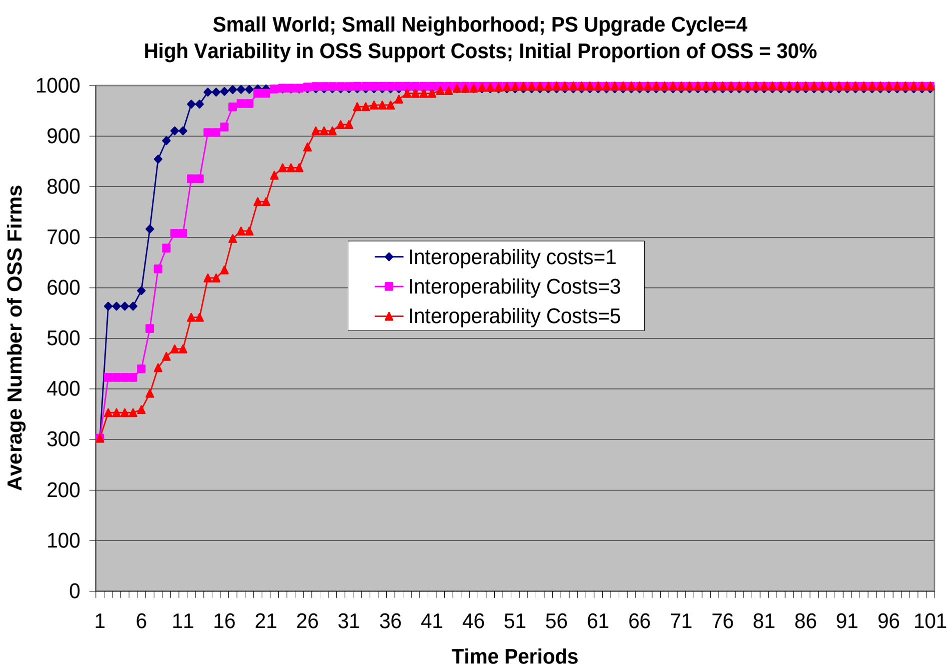


Figure 2. Decreasing interoperability costs favor diffusion of OSS

This is consistent with what we are seeing in the desktop OS market: interoperability issues are a major reason why many organizations are reluctant in adopting Linux or any other open source desktop operating system.

Similarly, shorter duration of the PS upgrade cycle tended to favor the diffusion of OSS (Figure 3) because a) it encouraged PS firms to consider upgrades/switches more frequently; b) firms upgrading PS faced the possibility of incurring higher one-time costs (such as setup and training costs), averaged per year, than OSS and this makes it attractive for some firms to switch to OSS. Hence, the second proposition

Proposition 2: Shorter PS upgrade cycles favor the diffusion of OSS

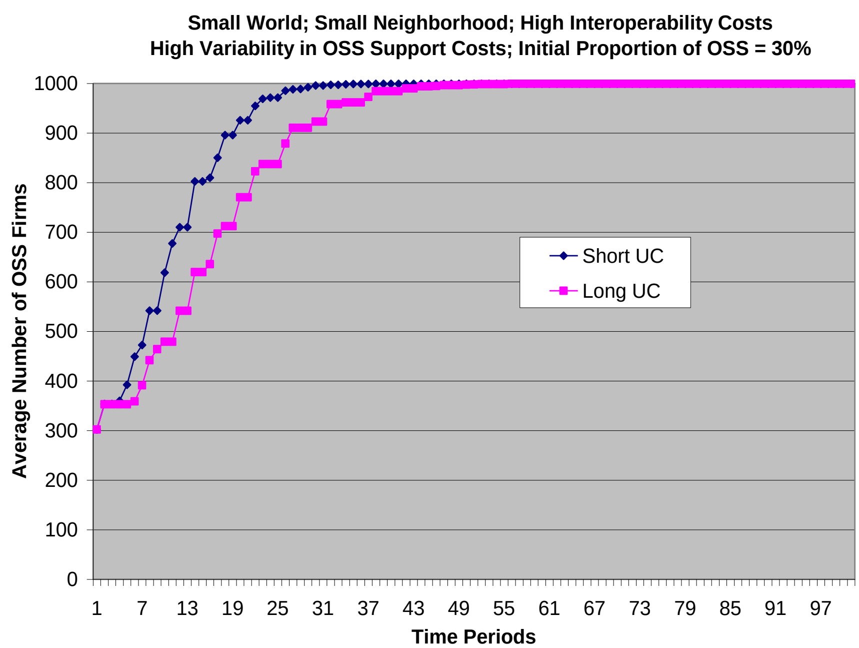


Figure 3. Rate of diffusion of OSS is faster when the PS upgrade cycle is reduced

Propositions 1 and 2 are somewhat intuitive and corroborate what we are seeing in reality. However, these results are valuable nonetheless as they establish face validity of the model (Lazer and Friedman 2007) and allow us to expound more interesting results with greater confidence in the validity of the model. This is an established practice in simulation based studies where some intuitive results or results that corroborate theory or reality are used to build confidence in the model (see for example, Dutta and Roy, 2005; Manzoni and Angehrn, 1997; Ahituv et al, 1998; Abdel-Hamid 1992; Jones et al, 2006; Kwon et al, 2007).

Significant diffusion of OSS occurred in the presence of high variability in support costs (N(60,60) in Table 3). This is not a readily intuitive result. Under high variability some firms incurred very low OSS support costs, thus their intrinsic value of OSS was large enough to motivate them to switch. When a sufficient mass of early OSS adopters was acquired, this further drove the OSS diffusion. It is worthwhile to note that if the proportion of OSS adopters was high (30%) to start with, then ceteris paribus, the critical mass was attained much faster than when the initial proportion was 10%. The low variability or very high magnitude of support costs (N(60,15), N(250,50)) was not able to make OSS seem more attractive than PS because most of the firms on average had very high support costs. Again, this result corroborates what is happening in reality. Despite having an established base of locked-in customers in the desktop market, Microsoft is slowly losing some of its market share to Linux.

On the one hand, there are firms (at the high end of this distribution) which feel that (either due to their lack of technical capability or lack of available compatible pool of applications) the support costs for OSS are too high. On the other hand, in addition to the low upfront costs, there are some firms which are facing very low OSS support costs (due to their technical capability or involvement in the open source online communities, or the way they are implementing these systems, flexibility in customizing and independent bug fixing and lock-in avoidance). These firms are finding it more attractive to adopt OSS than PS. Their adoption in turn influences other firms (Figure 4). Hence we state our third proposition:

Proposition 3: High variability in support costs favors the diffusion of OSS.

It is important to note that propositions 1, 2 and 3 were valid for all parameter conditions. The simulations revealed interaction effects between network topology, network density and interoperability costs that highlight the dynamics of diffusion. These are discussed in more detail in the following section.

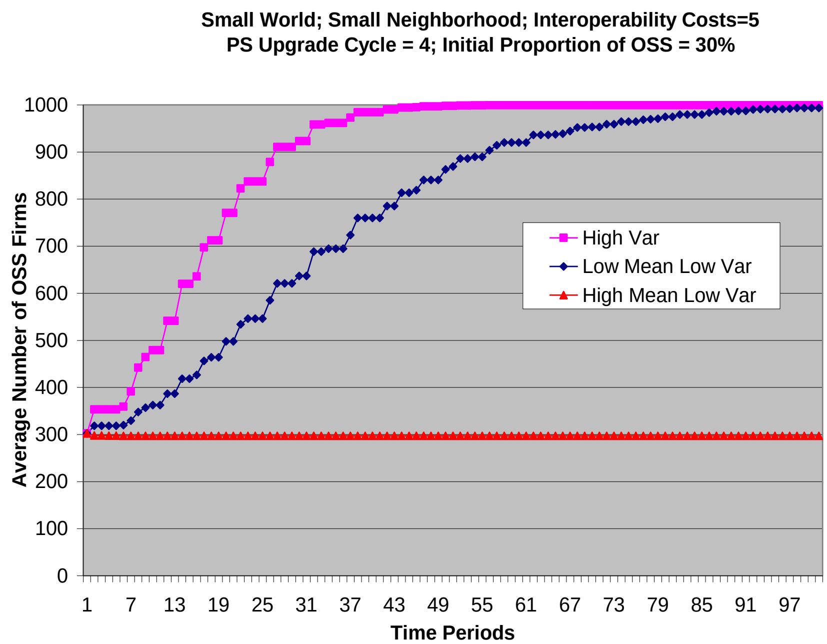


Figure 4. High variability in support costs favors the diffusion of OSS

2.4.2 Interaction Effects

Simulations indicate that the effect of varying network density is strongly linked to the strength of interoperability costs. When interoperability costs are high, less dense networks (small neighborhoods) encourage the diffusion of OSS the most (Figure 5), followed by denser networks (medium and large sized neighborhoods). As explained earlier in the context of the decision function, each firm simultaneously influences the decision of its neighboring firms and gets influenced by their decisions. For example, in high density networks, firms on average have more neighbors. This means that simultaneously, the decision of a firm a) influences many firms (its neighbors) and b) can be influenced by many firms. The level of this influence is affected by the strength of interoperability costs. If most of the neighbors of a firm have adopted a particular platform, they will likely influence the firm in question to adopt the same software in the case of high interoperability costs.

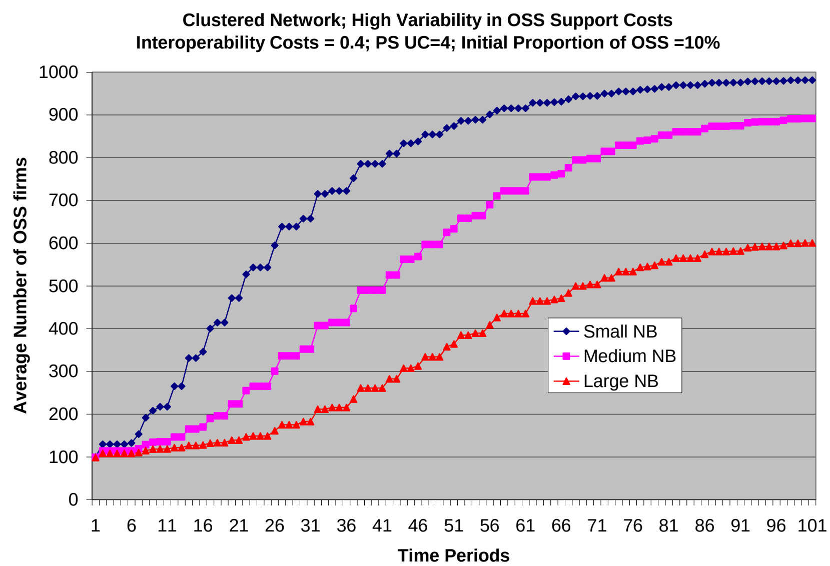


Figure 5. Rate of diffusion is fastest in small neighborhood (NB) or low density network with relatively high interoperability costs

In a denser network, where firms on average have more neighbors, such influence will increase. However, with low interoperability costs high density networks encourage the diffusion of OSS (Figure 6). This is because with low interoperability costs firms are less concerned about the software being used by their neighbors and in a high density network one firm's decision to switch can influence many firms and spread the 'news' quickly. Hence, we have the following propositions:

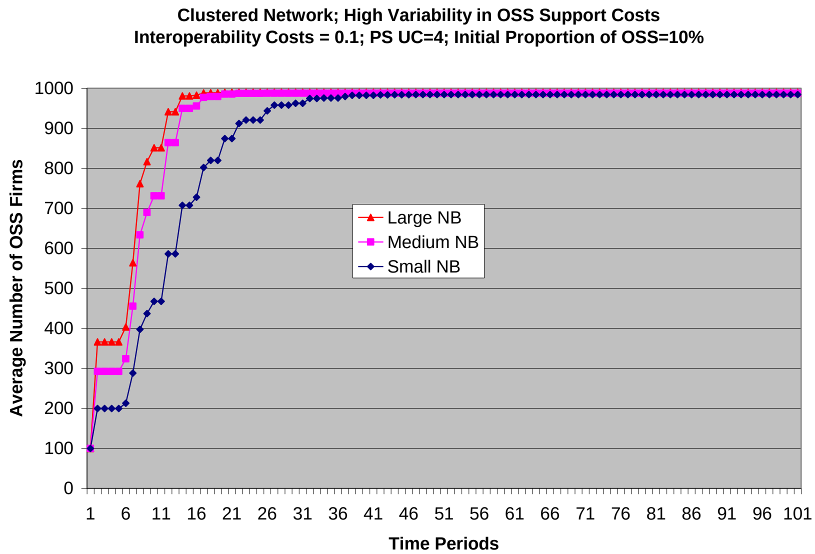


Figure 6. Rate of diffusion is fastest in large neighborhood (NB) or high density network with relatively low interoperability costs

Proposition 4a: With relatively high interoperability costs, diffusion of OSS is faster in low density networks and slower in high density networks.

Proposition 4b: With relatively low interoperability costs, diffusion of OSS is faster in high density networks and slower in low density networks.

In these and subsequent propositions, relatively low interoperability costs mean per interoperability costs <0.4 whereas relatively high interoperability costs are ≥ 0.4. It is important to note that the above mentioned effects of network density, a) do not force a firm to switch or upgrade an existing software, b) are not the only effects that a firm has to contend with. The cost components on the left-hand-side of the decision function (Equation 1) and the centrality of the firm will ultimately dictate the decision. These propositions demonstrate the dynamics of OSS diffusion. The exact values of interoperability costs for which these dynamics appear in our simulation are less important. Hence we use the terms 'relatively high' and 'relatively low' interoperability costs. As mentioned earlier, if the rate of OSS diffusion is too fast, it is hard to demonstrate the various effects of the key variables. Thus, the effect of network topology is more pronounced if the diffusion rate is slow.

Further analysis of the simulations showed that the effect of network topology varies with the effect of network density and interoperability costs. It was found that when interoperability costs are high in high density networks, there is a clear difference between the diffusion curves of the three network topologies (Figure 7). However, when the network density is low, the difference is less apparent. Here the difference between the diffusion curves is defined in terms of the absolute distance between the curves for the three network topologies. The explanation lies in the fact that the effect of network topology is stronger when the diffusion rate is slower. When interoperability costs are high in dense networks, the rate of diffusion is slow (Proposition 4a).

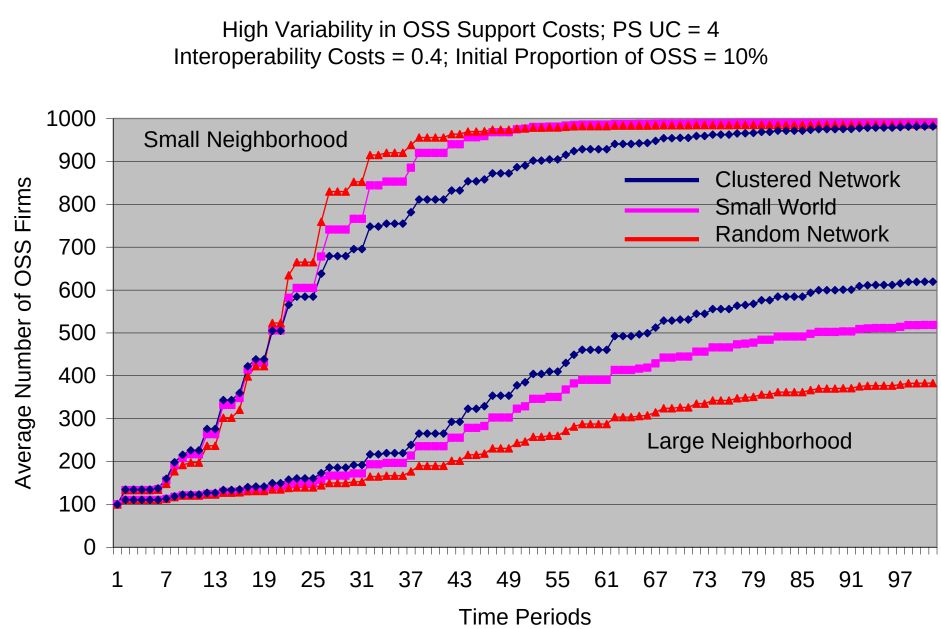


Figure 7. The difference between rates of diffusion across network topologies is more pronounced with relatively high interoperability costs as we increase the size of neighborhood or density of the network

Therefore, how firms are connected to each other (network topology) strongly influences the overall diffusion process. On the other hand, if rate of diffusion is fast, as is the case in high interoperability low density networks, then network topology makes less of a difference. Hence,

Proposition 5a: With relatively high interoperability costs, reducing network density reduces the effect of network topology on the diffusion of OSS

Similarly, proposition 4b indicates that in the presence of low interoperability costs, high density networks will facilitate faster diffusion of OSS than low density networks. In this case, reducing network density will dampen the rate of diffusion of OSS and enhance the effect of network topology (Figure 8). Hence,

Proposition 5b: With relatively low interoperability costs, reducing network density increases the effect of network topology on the diffusion of OSS.

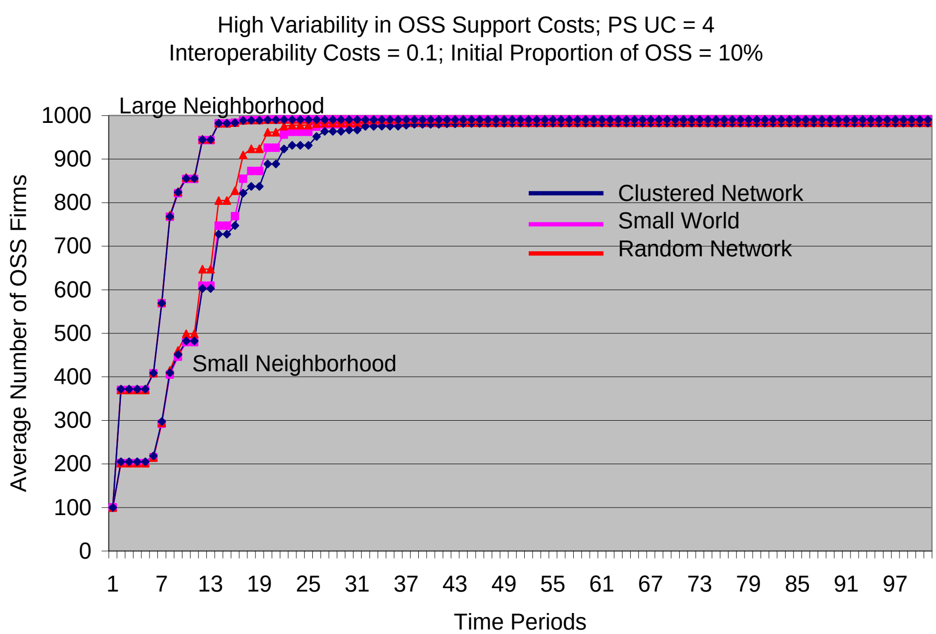


Figure 8. The difference between rates of diffusion across network topologies is more evident with relatively low interoperability costs, as we reduce the size of neighborhood

Propositions 4a and 4b demonstrate that the effect of interoperability costs on the speed of diffusion of OSS varies with the density of the network (two-way interaction); propositions 5a and 5b demonstrated that the effect of network topology on speed of OSS diffusion is closely tied to the effects of network density and interoperability costs (three-way interaction). These results automatically raise the question: given that network topology makes a difference, which network topology favors the diffusion of OSS?

Figure 8 shows that for the most part, regardless of the network density, rate of diffusion is fastest in a random network. However, in Figure 7, in low density networks, rate of diffusion is initially fastest in clustered networks, then in random networks and eventually in small world networks. In high density networks, rate of diffusion is considerably slower overall, but fastest in clustered networks followed by small world and random networks. These are very interesting results which have not been captured in earlier studies and warrant an explanation. Let us first consider the differences between the three network topologies. In a random network topology, the neighbors of a firm may not be well connected with each other whereas in a clustered network, the neighbors of one firm are very likely to be neighbors of each other as well. This means that the number of potential new adopters influenced in every time period is higher in a random network than in a clustered network. This can be understood by considering a PS cluster where every agent is connected to every other agent. When the first PS user switches to OSS, all of its neighbors are influenced by the decision. Thus, no new node will be influenced later on. On the other hand, in a random network, new nodes are more likely to be affected over time as firms' second tier neighbors (i.e., neighbors' neighbor) are less likely to overlap with firms' immediate neighbors. This has the potential of triggering

OSS diffusion in different parts of a random network. Furthermore, this potential would be present regardless of the density of the network. However, whether OSS diffusion is actually triggered or the extent to which it is triggered is also dependent on the strength of interoperability costs (as evidenced by the first and the last four propositions). In an initially PS dominated network with low interoperability costs, firms evaluate OSS and PS on other cost factors, which are more favorable for OSS, with only limited additional pressure from their PS-neighbors. This coupled with the potential of random connections to trigger diffusion across the network, results in rapid diffusion of OSS (Figure 8).

Proposition 6: In the absence of or under low interoperability costs, random network has the fastest OSS diffusion regardless of the density of the network.

However, as interoperability costs increase, initially if there is any diffusion in a PS dominated network, it occurs within the cliques of the clustered networks. Despite their low overall numbers in the network, some of the OSS firms may be in majority in cliques across the network and they will drive the diffusion in the early part of the simulation. That cannot happen in a random network where the same number of OSS adopters are widely dispersed throughout the network and cannot enforce any ‘social pressures’ in the absence of cliques. Therefore, initially adoption is slowly driven within clusters in the presence of high interoperability costs. However, as the number of OSS adopters reaches a critical mass, the random networks use their ‘random’ connections to drive diffusion globally throughout the network.

Proposition 7: In the presence of relatively high interoperability costs, initial diffusion is driven by the local connections or cliques and later on the global or random connections drive the diffusion process.

The point where this changeover takes place is decided by the density of the network. Notice for instance that in the bottom three curves of Figure 6a, the rate of diffusion is the fastest in the clustered network throughout the course of the simulation. In this case OSS is never able to attain a critical mass of adopters from where diffusion can then really kick off. In a dense network with high interoperability costs, the OSS adopter may require a much longer period to attain that critical mass than in a less dense network (top three curves of Figure 7).

From a practical perspective this implies that under high interoperability costs a) OSS vendors need to focus more on group-based adoption of OSS to gain critical mass, b) it will take longer to achieve critical mass in denser networks, c) once critical mass is achieved, group-based adoption may no longer be as valuable as before and small world networks will result in the fastest diffusion of OSS (Figure 9) By nature small world networks have elements of the other two types of networks.

2.5 Discussion

The objectives of this research were to a) devise a framework for simultaneous investigation of social and economic factors affecting OSS diffusion, b) explore the effect of network topology, network density, uncertainty regarding OSS support costs, interoperability issues, the length of the PS upgrade cycle, and initial proportion of OSS adopters on the diffusion dynamics of OSS.

An agent-based computational economics approach was applied in pursuit of these objectives by modeling the market forces affecting software diffusion based on both academic and practitioner literature. To that extent, the actual numbers used in the experiments are not as important as the framework and resulting insights.

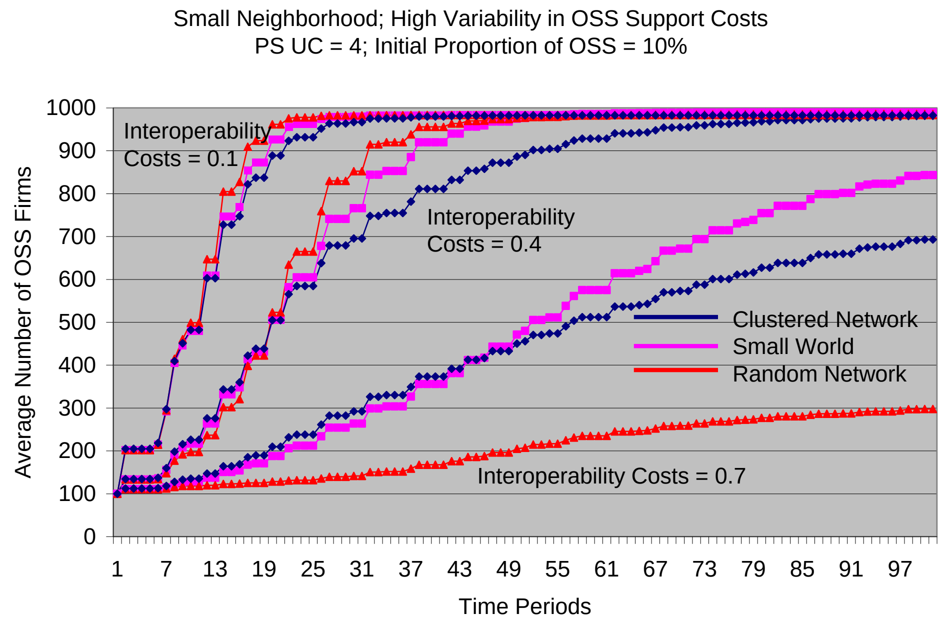


Figure 9. Rate of diffusion becomes fastest in small world networks as interoperability costs increase in less dense networks

This is a well established notion in simulation based studies especially when simulations are used to "stimulate discussion" on a particular topic (Dutta and Roy, 2005; Manzoni and Angehrn, 1997; Ahituv et al, 1998; Abdel-Hamid 1992; Jones et al, 2006; Kwon et al, 2007). However, it is worthwhile to revisit key assumptions made in the study and briefly address their effects on the propositions.

2.5.1 Assumptions and Propositions

First, we had assumed zero license/upgrade costs for OSS. In reality, there might be some upgrade costs for OSS but they would still be very low compared to PS upgrade costs (Wheeler 2005) and would not have any significant impact on our propositions. Second we had assumed that support costs for PS adopters would go up if their existing version is two or more versions older than the latest release from the PS vendor. Although it is known that PS users do not have guaranteed support once support is withdrawn for older version of the software by the PS vendor, we re-ran our experiments by removing this assumption and holding the support cost constant for PS adopters. We found that relaxing the assumption did not change our results. This result further establishes the robustness of our model and the resulting insights. Third, we assumed that OSS upgrades are more frequently available than PS upgrades and that upgrades mean more costs than benefits. Although, in our exemplar of Microsoft's Windows and Ubuntu's Linux releases it was evident that Linux releases are more frequent, we used two different PS upgrade cycles in our model – short and long. Although proposition 2 showed that speed of diffusion was faster when PS upgrade cycles were shorter, there was no qualitative difference in the rest of the propositions as a result of the different PS upgrade cycles. As for the costs associated with upgrades, we do take the qualitative benefits from an upgrade into account by assuming that in light of the potential benefits afforded by an upgrade, firms will either choose to switch or upgrade. Proposition 2 will no longer hold if upgrades indeed bring significantly more quantitative benefits than cost (i.e., positive cost saving from upgrade in the left-hand-side of the decision function). However, such a case is rare in reality. Fourth, we assumed that if firms do not switch to the alternative software they will upgrade their existing software. In reality, however, it is possible that firms may not decide to upgrade their software as well and continue to support the software on their own. However, other than slowing down the eventual diffusion of software, this assumption should have no qualitative impact on the insights generated through the propositions. Fifth, we assumed that when firms conduct transactions with each other, they incur interoperability costs per transaction. In reality, firms may choose to handle interoperability issues differently. We chose to use some numbers to model varying strengths of interoperability costs to accommodate those differences in reality. However, the point remains that some interoperability costs will be incurred. Again, changing the actual numbers of the interoperability costs will not affect the insights from our model. Sixth, we assumed that PS dominates the software market at the start of the simulation. This is a realistic assumption in the desktop operating system market. However, even if this assumption is relaxed, or the proportion of OSS versus PS is reversed at the start of the simulation, the propositions will not be affected.

In summation it can be seen that barring one assumption that would affect one proposition, all other assumptions made in the model can be relaxed or modified without affecting the results and insights generated from the model. As discussed earlier in Chapter 1, Lazer and Friedman (2007) proposed four key criteria for assessing simulation based studies: a) face validity i.e. the behavior of the model should closely follow reality, b) robustness, i.e. the results should hold in the face of trivial changes to the model c) replicable, i.e. other researchers should be able to completely replicate the results of this study and, d) “non-obvious non-trivial results” i.e. the model should allow the researchers to make new and insightful observations. This research satisfies all four criteria. The first two criteria are met by modeling the behavior of the individual agents (instead of aggregate behavior in traditional simulations) and choosing specific simulation parameters on academic research as well as practitioner reports. In addition, some of the results produced by the model are intuitive (Propositions 1 & 2). The paper provides sufficient detail about the simulation process and parameters for any other researcher to replicate the study (third criterion). Finally, the propositions stated earlier, particularly those regarding the interaction effects are indeed ‘non-trivial’ results that have not been highlighted in previous OSS research (fourth criterion).

2.6 Contribution to Research

This paper has made the following contributions to the growing body of research on OSS. First, it has provided a framework that can be used to study the diffusion processes of competing software. The framework concurrently includes multiple under-researched variables and could serve as a building block for diffusion dynamics under different pricing schemes. It was applied to specifically study the diffusion processes of OSS and PS where the former is characterized by low license costs, high variability in support costs, various OSS technical capabilities and no threat of withdrawal of support. Although the desktop operating system was used as an exemplar, we believe that our framework is general enough to be applied to the investigation of other software as well. For example, in the server operating system (OS) market, the initial number of OSS adopters, setup costs, support costs and training costs will be higher than those in the desktop OS market, but the model will still be applicable. Similarly, in the open source ERP market, the strength of interoperability issues may be higher than those described in our model in the context of the desktop OS, however the propositions should still hold. Second, it has demonstrated that the diffusion of software is dependent on strategic factors other than price, such as interoperability costs, network topology and density. This result has been demonstrated while incorporating significant heterogeneity among adopters and considering factors such as the threat of withdrawal of support by the PS vendor and the influence of centrality of neighbors on adoption decisions. Third, it has illustrated that variability in OSS support costs hastens the diffusion of OSS, given other factors. Fourth, it has incorporated the use of social networking concepts such as *degree* *centrality* to study diffusion of OSS and standards. Compared to earlier papers, our model provides a richer depiction of the critical variables and their interactions with each other.

2.7 Contribution to Practice

In order to illustrate the effects of key variables on the diffusion of OSS we deliberately chose a PS vendor that did not react to the changes in the market. Earlier models that investigated various pricing strategies for a PS vendor in competition with an OSS vendor did so without realizing the significance of other variables such as interoperability costs, duration of PS upgrade cycle, threat of withdrawal of support, network topology and network density on the diffusion of OSS. As a consequence we have offered a model that captures the interesting effects of these variables and can now be used as a building block to better investigate the impact of various pricing schemes on the PS and OSS software market. Furthermore, by modeling the effects of these key variables we have shown that PS and OSS vendors need to focus on strategic variables other than price to compete against each other. The simulation results revealed that the PS vendor is only threatened by OSS if interoperability costs are low and there is high variability in OSS support costs. With high variability in OSS support costs, the OSS vendor can offer very low support costs to some firms and build a critical mass that drives diffusion across the network. Our result also suggests that the applicability of the group-based adoption strategies depends on the existing interoperability cost and network density. From a PS vendor's perspective the upgrade policy could be revised by changing a) the frequency of upgrades, b) the additional cost of upgrades and c) the timing of withdrawal of support for earlier versions, all of which can potentially impact the diffusion of its software.

2.8 Conclusions, Limitations and Future Research

This essay proposes a model to better understand the impact of six key, yet under researched factors on the diffusion dynamics of OSS. We believe that our model builds on prior research and helps to create a richer theoretical foundation for studying OSS diffusion. Such modeling helps to structure the debate and open up the field for additional research (Liberatore et al 2000). The model presented in this paper could serve as a foundation on which different competitive actions by PS vendors can be superimposed. For example, different pricing policies and strategies for withdrawal of support can be examined. The results also indicated that apart from license costs, there are other key variables which the PS vendor can manipulate to prevent the diffusion of OSS. For example, the PS vendor could influence interoperability and upgrade costs, change the timing of withdrawal of support to influence the decision of existing and potential adopters. Hence, our model can be considered to be similar in purpose to the agent-based models that have been used in other domains such as supply chain management to create building blocks for studying dynamics (Swaminathan et al, 1998). Future research will focus on integrating competitive dynamics with the model presented in this paper to build a Complex Adaptive System (Miller and Page, 2007).

There are some aspects of this study that limit its scope and applicability. First, some of the probabilistic variables in the simulation were modeled using uniform and normal distributions. While these are reasonable choices in the absence of other information, one approach would be to use multiple distributions to condition the results. Second, the decision function of the individual firms is based on net cost savings. These cost savings are used as a proxy for measuring benefits. We believe this is a reasonable approach since benefits are typically difficult to quantify. However, some domain-specific benefit modeling may be possible. Again, we view this as an extension that can be superimposed on the basic model proposed in this paper. Third, diffusion dynamics highlighted by the propositions regarding interaction effects may not always be visible if one or more parameters dominate the diffusion process. For example, at one extreme, if interoperability costs are extremely high, diffusion will not occur and interaction effects will not be visible. Similarly, at the other extreme if OSS license costs are zero and interoperability costs are close to zero then diffusion dynamics can be driven by these two parameters and the effect of network topology or density is not significant. Fourth, the network structure remains static throughout the course of a simulation. However, from a practical perspective, the business relationships between firms evolve over time. A firm that is forced into adoption of an innovation by its neighbor may not necessarily continue its business ties with that neighbor. Finally, it is assumed that firms will adopt open source or proprietary software and not both. In reality however, we see that large organizations may actually choose to deploy both types of software at some point in time. Though this paper has focused on OSS diffusion, we believe that the model is fairly general and is applicable to studying the dynamics of software diffusion in a variety of contexts. Examining the generalizability of this model is another area of future research.

In this essay we highlighted key firm-level determinants of the diffusion of OSS. In Chapter 3, while drawing on the literature on social network analysis, we explore the importance of interrelationships between firms in explaining the diffusion of OSS.

CHAPTER 3: A SOCIAL NETWORK ANALYSIS OF THE DIFFUSION OF OPEN SOURCE SOFTWARE

3.1 Introduction

The acceptance of open source software (OSS) in private and government organizations has increased over the last few years (Blau 2006; Sayer, 2005; Shiels 2009). For example, in 2003 the city of Munich announced that it would move 14,000 PCs from Windows to Linux operating system (Shankland 2003). Around the same time, Paris (Sayer 2005) and Brazil (Kingstone 2005) moved towards OSS as well. Some practitioner reports attributed such large-scale OSS conversions to traditional one-vendor lock-in problems (Kingstone 2005; Moody 2008). Researchers offered explanations based on the literature on innovation diffusion (Bonacorsi et al 2006), software pricing and competition (Kim et al, 2006). However, these studies did not adequately model the effect of the organizational social networks within which a firm is embedded, in driving the adoption and diffusion of OSS. Growing emphasis on global collaboration and interdependence between organizations while making technology adoption decisions (Weitzel et al, 2006) warrant a better understanding of the effect of network structure on the social behavior of organizations and economic outcomes (Jackson, 2008). There are numerous studies that have already demonstrated the effect of organizational networks on individual firm performance, resource utilization, firm innovativeness etc. (Afuah, 2000; Kogut, 2000; Hite and Hesterly, 2001; Gulati et al, 2000; Capaldo, 2007; Bell, 2005; Zaheer and Bell, 2005).

Our objective is to investigate how a firm's inter-organizational relationships or social network can influence the adoption and diffusion of OSS. In the analysis of social networks, entities being studied are viewed as nodes in a network. Two nodes are connected via an edge if they have a relationship between them. Network-wide phenomena can then be investigated based on an analysis of the various structural characteristics. These structural characteristics can be measured at the level of relationships between individual nodes or groups of nodes. The emphasis shifts from the internal characteristics of individual nodes to the interrelationships between the nodes and their importance in explaining network-wide behavior. In the context of our research, firms using an open source (OSS) or proprietary alternative (PS) of a desktop operating system are viewed as nodes in a network. Links between firms indicate that business transactions are conducted between pairs of firms (or partners). Firms have to decide whether to upgrade their existing desktop operating system or switch to the alternative. The premise is that the choice of software used in the transactions with a firm's partners should in part be influenced by a firm's inter-organizational relationships. In other words, technology adoption decision, particularly platform decisions that involve interaction with partner firms, cannot be made without considering the decision of the partners. Therefore, important nodes in the network need to be identified based on various structural characteristics and their impact in driving diffusion of software needs to be investigated. This knowledge can then be exploited by vendors as they target influential nodes in the network to kick start or promote the diffusion of their software. We pose the following research questions:

1. *What is the relative importance of various individual-level structural measures in explaining the rate of diffusion of OSS?*

2. *What is the relative importance of group-level structural measures in explaining the rate of diffusion of OSS?*

3. *Which of the structural measures are most effective in explaining the rate of diffusion of OSS?*

While we focus on OSS, these questions and the approach presented in the chapter are applicable to diffusion of other types of software as well. We intend to use the agent-based model presented in Chapter 2 to simulate the diffusion of OSS. UCINET, a tool commonly used in social network analysis, will be used to measure the structural characteristics of the various networks. The rest of this chapter has been organized as follows. The next section provides a brief review of the relevant literature on social network analysis. This is followed by a description of the planned experiments, analyses and current results.

3.2 Literature Review

Please refer to the literature review section in Chapter 2 for a detailed review of the literature on OSS and diffusion of OSS. Suffice to state at this point that most of the studies discussed in the review suffered from two basic drawbacks: a) they modeled the determinants of diffusion of OSS in isolation i.e. there was no study that jointly investigated the impact of the critical determinants of OSS; b) they focused primarily on economic aspects of diffusion of OSS such as pricing, total cost of ownership, etc. In Chapter 2 a more comprehensive model was developed that attempted to fill these gaps in the literature by examining the diffusion of OSS based on both social and economic aspects of diffusion (Zaffar et al, 2008). Our investigation highlighted several important issues in the context of diffusion of OSS (please refer to Chapter 2 for more details). In light of the research questions stated above, the most important result from the study in Chapter 2 was that network topology, network density and interoperability costs both hampered and facilitated the process of diffusion of OSS at different points in time. Therefore, we decided to further investigate the precise nature of the network structure (network topology + network density) that, at any point in time, helps or hinders the diffusion of OSS. The concept of social network analysis is appropriate for this type of investigation as it evaluates the structure of a network in more detail. There are techniques in graph theory that help identify critical nodes or links in a network, which if removed, would disconnect the network. However, there are two issues with the application of those approaches in the context of present research: a) recent research suggests that it might be more meaningful to identify groups of nodes instead of individual important nodes in very large networks (Pandit et al 2008). If that is the case, then with the use of the traditional techniques our objective would be to find groups of nodes or links that might disrupt the network. This in turn might lead to an optimization problem (which is beyond the scope of what is under investigation) that what is the minimal set of nodes or links required to disrupt the network of a certain size, topology and density? b) the traditional graph theory techniques do not take into consideration the unique internal characteristics of the nodes in a network. We hope to incorporate both internal and external characteristics to determine structural importance specifically in the context of software diffusion.

In the following subsection, the literature on social network analysis is reviewed in the context of our research problem.

3.2.1 Social Network Analysis (SNA)

The analysis of social networks is concerned with the investigation of relationships between social entities (Wassermann and Fraust, 1994). In the context of inter-organizational relationships, these structural patterns are intertwined with the economic outcomes (Jackson, 2008). Under SNA, "the network structural environment [is viewed] as providing opportunities for or constraints on individual action" (Wassermann and Fraust, 1994, p 4). In other words, instead of their unique attributes, the behavioral or structural patterns between individuals become the focus of investigation. In the context of inter-organizational relationships, Hite and Hesterly (2000), Zaheer and Bell (2005) and Gulati et al (2000) argued that the structure of a network can influence a firm's actions, its innovative capabilities as well as performance. These structural patterns "can also be used to study the process of change within a group over time" (Wassermann and Fraust, 1994, p 10). Several metrics have been proposed in the previous literature to describe the structure of various types of social networks: *degree centrality, closeness centrality, betweenness centrality* etc. (Ahuja and Carley, 1998; Fernandez et al, 2006; Girvan and Newmann, 2002; Guo and Chang, 2007, Wassermann and Fraust, 1994). These structural measurements can be made at the level of individual nodes or groups of nodes. Recent research suggests that in case of very large networks it is more appropriate to study the structural measures at the level of groups of nodes instead of individual nodes (Pandit et al 2008). The rationale is that in very large networks the effect of important individual nodes is less significant to network-wide behavior than the effect of important groups of nodes. For example, in the context of software diffusion, where there is a large network of firms interacting with each other, it would be useful for software vendors and third-party software providers to identify influential groups that can quickly affect the diffusion dynamics of software. Since our research is aimed at determining the appropriate choice of structural measures for investigation of software diffusion, we will analyze the structural measures at both the individual as well as group level.

The choice of a suitable measure or measures is dependent upon the context of the research problem (Haythornthwaite, 1996). For example, Lazer and Friedman (2007) and Weitzel et al (2006) studied the effect of network density and network topology/path length on flow of information and standard diffusion. There are two important differences between their studies and our research: a) flow of information through a network is not comparable to the systematic evaluation, adoption and diffusion of software, b) network density and path length only provide one level of measurement of the network effect – in this study, we intend to investigate the network effect in more detail by studying the structural characteristics widely used in the literature on social network analysis. Guo and Cheng (2007) studied the impact of group level and individual level *degree* centrality, *closeness, betweenness* and *aggregate constraint* on the spread of a virus in a network. There is an important difference between their study and present research: in their network the virus spreads through contact whereas in our model, OSS spreads through the repeated evaluation of a decision function by heterogeneous firms in the network and this decision function takes both economic and social factors into account. In a recent study, Borgatti (2006) devised a new approach to identifying key players in a network that a) could facilitate diffusion of something in the network, and b) if removed, would severely disrupt the network. It is an interesting approach that goes beyond the use of traditional centrality measures or approaches in graph theory to identify crucial nodes in a network. However, that approach relies on structural information of the network alone to identify crucial nodes whereas in our study the network, the inter-relationships and the nodes are more complex. A node may or may not be influential by virtue of its connections alone. There is an interplay between social and economic factors which is fluidly captured in our agent-based model which is essential to identifying important nodes for the purposes of software diffusion.

Borgatti stated that centrality measures provide "expected values … of certain kinds of node participation in network flows. As such, they do not actually measure node participation at all but rather indicate the expected participation if things flow in the assumed way" (Borgatti, 2005: p. 70). Hence, if the network flow is not well understood or its workings are modeled on inaccurate assumptions, any centrality measure applied on such a network may lead to "incorrect" conclusions (Borgatti, 2005).

Therefore, in the following subsections, we first evaluate the network flow process of software diffusion and then identify suitable centrality measures for evaluating the impact of network structure on diffusion of OSS.

3.2.2 What Determines the Choice of Centrality Measures?

Borgatti (2005) defined four attributes for characterizing network flow processes: a) mechanism of network flow or diffusion: does diffusion occur "via replication (copy mechanism) or transfer (move mechanism)" (p. 58). Under replication, the information or packet flowing through the network gets copied from one node to the next. Whereas under transfer, the packet physically moves from one node to the next; b) serial or parallel network flow: is the network flow simultaneous like a broadcast or is it serial?

This attribute is meaningful only if diffusion occurs through replication; c) deterministic or un-deterministic flow of traffic: does network flow strategically follow a deterministic path such as shortest route, or does it flow "in a blind, undirected way" (p. 58); d) flow trajectory: does network flow follow paths, trails or walks? A trail is "a sequence of incident links in which no link is repeated … [paths are] sequences in which not only links but also nodes are not repeated … [walks are] unrestricted sequences. All paths are trails and all trails are walks, but not every walk is a trial and not every trail is a path" (p. 57). Borgotti recommended an appropriate choice of the commonly used centrality measures based on the characteristics of the network flow processes (Borgatti 2005).

Let us first evaluate the software diffusion process in light of Borgatti's four attributes. Diffusion of software occurs through parallel replication, is un-deterministic and follows walks. Table 5 provides a more detailed assessment of each attribute in the context of our research problem. Given these characteristics, Borgatti stated that only three of the commonly used centrality measures are appropriate in the context of our problem: Freeman's (1979) *closeness centrality*, Freeman's (1979) *betweenness centrality* and Bonacich's (1972) *eigenvector centrality* (Borgatti, 2005).

However, Borgatti's recommendations cannot be readily applied in the context of our problem for three reasons. First, software does not exactly replicate through the network – the deployment of the software be different from firm to firm. However, in our model we ignore the differences in deployment of the software and assume that two firms are OSS/PS adopters even if their versions are different. The fact that two firms in a dyad are using the same software eliminates the possibility of them incurring any interoperability costs in their transactions.

Table 5. Attributes for differentiating between network flow processes

Attribute	Explanation	Behavior in the context of our research problem
Mechanism of diffusion	Does diffusion occur "via replication (copy mechanism) or transfer (move mechanism)" (p. 58)	Software diffusion, it can be said, occurs through the copy mechanism instead of the transfer mechanism even though the same software may not be deployed by partner organizations in exactly the same way. As Borgatti (2005) states, money and used goods are transferred or move through a network (p. 57). However, an intangible good such as software can be at multiple places at the same time and need not actually move from node to node for the purposes of diffusion.
Serial or parallel network flow	This is applicable only if the mechanism of diffusion is *replication*: is the flow simultaneous like a broadcast or serial	Diffusion of software in the context of our problem is parallel. A firm using a particular software simultaneously affects all of its neighbors into adopting a particular software. This influence is not serial or does not occur in a specific order.
Deterministic or un-deterministic flow of traffic	Does the network flow follow a deterministic path such as the shortest route to a particular destination or does it flow "in a blind, undirected way" (p. 58)	Diffusion of software need not take the shortest path through a network. It is undirected in some ways because even though all neighbors of a firm are influenced by a firm's decision, they may not adopt this particular firm's software. The tendency for adoption is determined by several factors and none of them hinges on the concept of a shortest path in the network
Flow trajectories	Does network flow follow paths, trails or walks?	In the context of our research problem, diffusion of software occurs through walks i.e. the same node or link can be traversed again. Firm A using OSS can influence its neighbor, Firm B (through a link) to adopt OSS. If Firm B adopts, it could in some other time period be influenced to adopt OSS by another one of its neighbors Firm C (node repetition). And if at some point Firm B goes back to PS from OSS, Firm A could influence it again to adopt OSS (link repetition).

Furthermore, the actual versions used by the adopters are used to model the impact of threat of withdrawal of support by a vendor for older versions of its software (refer to the Simulation Model section in Chapter 2).

Second, in our model of diffusion of software, one firm directly influences the decision of its neighbors and is simultaneously influenced by the neighbors' decisions as well. This second characteristic is very important in software diffusion and is captured in the decision function of each agent in our model. However, Borgatti does not consider this dimension in differentiating between different types of network flow processes (Borgatti, 2005: p. 59). Third, as acknowledged by Borgatti, the recommendations made for using certain centrality measures for certain network flow processes were limited by the assumption that the flows "have a source and a target" (p. 70). Furthermore, he stated that "we should also examine the case where flows originate at each node systematically, but have no particular target. This will pose some challenges for walk-based processes but is an important line of future research." (p. 71). Since diffusion of software follows walks, it is unclear if Borgatti's recommended measures are most appropriate in the context of our research problem. Therefore, in the following subsections, we discuss our choice of centrality measures and analyze their appropriateness on theoretical grounds in the context of our research problem.

3.2.3 Selected Individual and Group Centrality Measures

In the absence of recommended centrality measures for evaluating software diffusion type processes, we decided to use and evaluate four commonly used centrality measures in the literature on social networks: *degree centrality, betweenness centrality, closeness centrality and eigenvector centrality*. As will be discussed later in this section, these centrality measures are well defined for individual or node-level analysis. A typical approach to obtain group-level centrality measurements is to average the centrality measures of the individual nodes in a group (Guo and Chang, 2007). Although such aggregate measures might lose the nuances in some groups (Everett and Borgatti, 1999), practically more involved group centrality measures would be harder to compute for a large network of firms. Hence, this is a reasonable approximation for identifying and differentiating between groups of firms.

Therefore, in the context of our problem, group *degree centrality* will be measured as the average degree centrality of all firms in a group. Similarly, group *closeness centrality* will be computed as the average *closeness centrality* of all firms in a group, and so on. Before going into the details of the various centrality measures, it is important to define the concept of a group in the context of our research problem. A group is considered to be a simple supply chain network, which includes a focal firm and its immediate neighbors. This is a practical definition of a group in the context of software diffusion since a firm affects and is affected by the decision of its immediate neighbors (Weitzel et al, 2006). Therefore, by this understanding, Wal-Mart and its immediate set of suppliers and partners would be considered as one group in our analysis. Table 6 reviews the definitions of the selected centrality measures and their meaning in the context of software diffusion.

Degree centrality defines the importance of a node in terms of the number of relationships it is involved in (Wagstrom et al, 2005). Literature suggests that when firms make their own technology adoption decisions, a) they try to anticipate the decision of their neighbors (Weitzel et al, 2006), b) firms with high degree centrality (large number of neighbors) tend to be less concerned about the decisions of their neighbors (Ashton, 2004). Therefore, theoretically we would expect firms to contend with two kinds of effects: a) an individual decision made by a high degree-centrality firm will potentially influence the decision of a greater number of other firms (its immediate neighbors or direct partners) than it would in the case of a low-centrality firm, b) depending on the strength of the interoperability issues, there could potentially be greater pressure from the immediate neighbors on a high degree-centrality firm than there would be on a low-centrality firm, in adopting a technology different from that of its neighbors. Therefore, it is natural to further explore the effect of degree centrality on the diffusion dynamics of OSS. *Closeness centrality* defines the importance of a node in terms of how close it is, on average, to other nodes in the network. The distance between a pair of nodes is measured in terms of the number of links or connections between them. Borgatti (2005) stated that closeness is suitable for 'parallel duplication' processes. As stated earlier, in a parallel process the network flow can follow all possible paths. Under duplication, the network flow does not have to physically move or transfer through the network. Instead, it duplicates or copies itself from one node to the next.

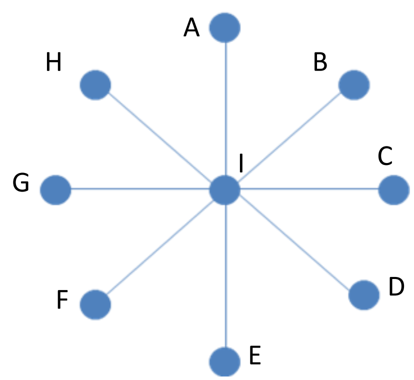
Table 6. Commonly used centrality measures and their application in the context of software diffusion

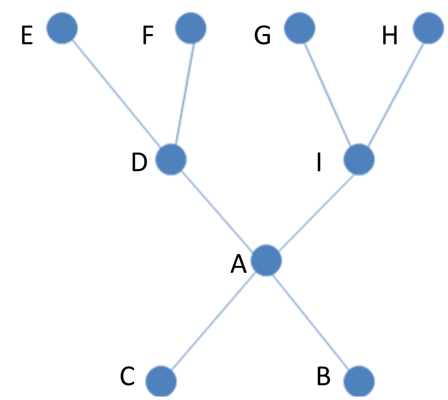
Centrality Measure	Definition	Meaning in the context of software diffusion
Degree centrality	$D_i = \text{Degree Centrality of firm } i = \frac{\sum_{j=1}^{m} X_{ij}}{N-1} \qquad 0 \le D_i \le 1$ $X_{ij} = \begin{cases} 1 & \text{if firm } j \text{ is a neighbor of firm } i \\ 0 & \text{otherwise} \end{cases}$ N is the total number of firms in the network	Importance of a firm is defined in terms of number of partners or neighbors. The more neighbors a firm has, the higher the number of firms it is likely to influence with its own software adoption/upgrade decision. Also, the more neighbors a firm has, the greater the pressure from neighboring firms to conform to the dominant software.
Closeness centrality	$C_i = \frac{N-1}{\left[\sum_{j=1}^{N} d(n_i, n_j)\right]} \qquad 0 \le C_i \le 1$ $d(n_i, n_j)$: shortest distance between nodes i and j N is the total number of firms in the network	Firms with low closeness centrality that are on average farthest from other firms in the network are less likely to influence the network wide diffusion process than firms with high closeness centrality. Firms with high closeness centrality are likely to indirectly influence the decision of other firms faster than firms with low closeness centrality.
Betweenness centrality	$B_i = \frac{2 \times \sum_{j<k} \frac{a^i_{jk}}{n_{jk}}}{N-1 \times N-2}$ n_{jk} : # of shortest paths between nodes j and k a^i_{jk} : # of shortest paths between j and k going through i N : total number of firms in the network	Firms that fall on the shortest path between pairs of other nodes have high betweenness centrality. These firms can be critical to the network-wide diffusion process by virtue of being part of multiple groups or supply chains in a network.
Eigenvector centrality	$Av = \lambda v$ A : adjacency matrix of the network λ : eigenvalue v : eigenvector Eigenvector centrality of a node 'i' = $E_i = \alpha \Sigma A_{ij} E_j$ α is the reciprocal of the eigenvalue	Eigenvector centrality measures importance in terms of both immediate number of neighbors and the next tiers of neighbors as well. Therefore, technology adoption decisions of high eigenvector centrality firms can not only have a profound immediate effect but a deeper and wider affect across the network as well.

In software diffusion the probabilities of spreading through all possible paths is dependent on a combination of factors wrapped up in the decision function of each firm and the software does not have to physically move or transfer through the network. Since it is not clear as to how closeness centrality might affect the software diffusion process, it is worthwhile to investigate its effect in our research. Betweenness centrality measures importance in terms of how often a node falls on the shortest path between pairs of other nodes (Guo and Chang, 2007).

Borgatti (2005) states that betweenness centrality is an appropriate measure of centrality provided that there is a 'target node' involved in the network flow process. However, in software diffusion there is no target node per se, yet conceptually, firms with high betweenness centrality (that may lie on critical paths and be part of multiple supply chains) would be expected to significantly influence the decisions of their neighbors and neighbors' neighbors. Therefore, it makes sense to quantitatively investigate this effect under various network conditions. Eigenvector centrality (Bonacich 1972) measures the importance of a node in terms of the overall structure of the network instead of just the local connections (Hanneman and Riddle, 2005). It is considered an ideal measure for an "influence type process" (Borgatti, 2005; p. 62) and our objective is to quantitatively investigate its effects in the context of software diffusion. Table 7 provides a numerical example of the centrality measures, further highlighting their different perspectives on the 'importance' of a node. The centrality values have been computed for two types of network topologies: sample network topology 1 (SNT 1) and sample network topology 2 (SNT 2).

Table 7. Numerical example of computing individual centrality measures under different network topologies





Sample Network Topology 1

	A	B	C	D	E	F	G	H	I
BC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**1.00**
CC	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53	**1.00**
DC	0.13	0.13	0.13	0.13	0.13	0.13	0.13	0.13	**1.00**
EC	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33

Sample Network Topology 2

	A	B	C	D	E	F	G	H	I
BC	**0.79**	0.00	0.00	0.43	0.00	0.00	0.00	0.00	0.43
CC	**0.67**	0.42	0.42	0.53	0.36	0.36	0.36	0.36	0.53
DC	**0.44**	0.11	0.11	0.33	0.11	0.11	0.11	0.11	0.33
EC	**0.60**	0.27	0.27	0.43	0.18	0.18	0.18	0.18	0.43

BC: Betweenness Centrality, CC: Closeness Centrality; DC: Degree Centrality; EC: Eigenvector Centrality

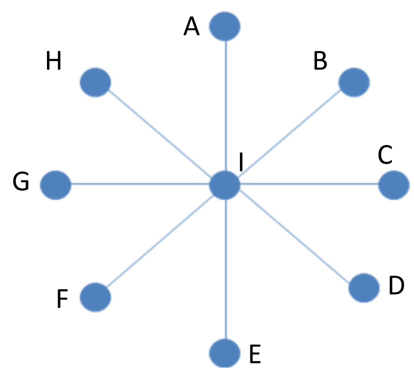
Note: The highlighted values in the table show the node with the highest centrality value given a particular measurement of centrality

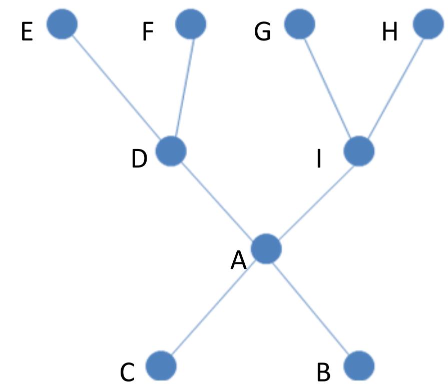
SNT 1 is a star topology network. In terms of betweenness, closeness and degree centralities, node ‘I’ in the center is the most important node. However, the relative importance of the nodes changes depending on which centrality measure is used. For example, under betweenness centrality node ‘I’ is at one extreme (with a centrality value of 1) whereas all other nodes are at the other extreme (with a centrality value of 0); under closeness centrality, even though node ‘I’ is still the most important node, other nodes are not completely unimportant (their centrality values are much higher than 0). Interestingly, eigenvector centrality does not help distinguish between the nodes. This is understandable because eigenvector centrality measures importance of a node on the basis of the depth and breadth of connections of a firm. In a star topology network, a) the center node has the whole network as its neighbors (and those neighbors have no other neighbors); b) the other nodes have one neighbor each and that neighbor is connected to the whole network.

In SNT 2, node A is the most important node by all measures of centrality. However, as in SNT 1, the relative importance of the nodes varies depending on the selected measure of centrality. Also, it is important to note that nodes B, C, E, F, G and H are peripheral or leaf nodes in the network with just one neighbor and are treated to be equally important in terms of betweenness and degree centrality. However, with eigenvector or closeness centrality, the relative importance of these nodes changes by virtue of their neighbors’ neighbors.

Table 8 looks at the examples from Table 7 and displays group-level centrality values. As explained earlier, a group was defined as a simple supply chain: a firm and its immediate set of neighbors.

Table 8 . Numerical example of computing group centrality measures under different network topologies





Sample Network Topology 1										Sample Network Topology 2									
	A	B	C	D	E	F	G	H	I		A	B	C	D	E	F	G	H	I
BC	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	**0.50**	0.11	BC	0.33	**0.40**	**0.40**	0.31	0.21	0.21	0.21	0.21	0.31
CC	**0.77**	**0.77**	**0.77**	**0.77**	**0.77**	**0.77**	**0.77**	**0.77**	0.58	CC	0.51	**0.55**	**0.55**	0.48	0.45	0.45	0.45	0.45	0.48
DC	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	**0.57**	0.23	DC	0.26	**0.28**	**0.28**	0.25	0.22	0.22	0.22	0.22	0.25
EC	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	0.33	EC	0.40	**0.44**	**0.44**	0.35	0.26	0.26	0.26	0.26	0.35

BC: Betweenness Centrality, CC: Closeness Centrality; DC: Degree Centrality; EC: Eigenvector Centrality

Note: The numeric value in each column shows the appropriate group centrality value for a group (with a focal firm whose name appears at the head of the column). For example, values under column A in SNT 1 mean average centrality values for a group focused on A (i.e. average of the individual centrality values for firms A and I (reported in the corresponding column in Table 7)).

By this definition, each firm will be the focal firm of its own group. Therefore, in a 1000-node network, there will be 1000 groups. In the examples in Table 8, there are 9-nodes in the two network topologies (SNT 1 and SNT 2), therefore, there are 9 groups in both these networks. The individual centrality values of the members of each group were averaged to compute the group centrality measures for each group. For example, for SNT 1, the average degree centrality for firm 'I' in the center is the average of the individual centrality values of all nodes in this network since the 'group' of node 'I' is node 'I' itself and its immediate neighbors. Notice that just like in Table 7, in Table 8, the relative importance of the groups of nodes changes depending on the chosen group centrality measure.

Furthermore, it appears that with our method of computing group centrality values, smaller-sized groups appear to be favored over larger sized groups. However, that is the *nature* of any 'average' and if large-sized groups have very important firms (or firms with high centrality values, regardless of which centrality measure is chosen), then their average will not be seriously attenuated. Although there are other ways of computing group centrality measures (for some types of centralities, see for example Everett and Borgatti 1999), it is not clear whether those measures are appropriate in the investigation of inter-organizational relationships (Everett and Borgatti, 1999). Therefore, given the exploratory nature of our research problem, it is appropriate to select an approximate measure of group centrality at this point.

Tables 7 and 8 demonstrate that the importance of nodes and groups of nodes changes on the basis of network topology as well as network density (average number of neighbors). However, there is no basis to argue whether one (or which one) of the centrality measure would be better than another in identifying structurally important firms or groups of firms in the context of software diffusion. To investigate this problem, the software diffusion model from chapter 2 was revised and a series of experiments were designed to study the impact of structural importance on software diffusion. The following section describes the revised model and experiments.

3.3 Revised Simulation Model

The software diffusion model proposed in chapter 2 assumed that OSS was randomly assigned to a small starting population of the network. The remaining firms were assigned PS. During the simulation, whenever a firm was at the start of its planning horizon, it had to decide whether to upgrade its existing software or switch to the alternative software. The decision was made on the basis of a decision function that took a combination of social and economic factors into account. Please refer to the section on the simulation model in chapter 2 for details.

In the present study, the objective was to identify and investigate the effect of structurally important firms and groups of firms in the context of software diffusion. Therefore, instead of random selection, the initial population of OSS adopters was selected on the basis of structural importance (using the available centrality measures) and differences across the ensuing patterns of diffusion were investigated. Since the importance of nodes can be measured at the level of individual as well as groups of nodes, two separate variants of the model and experiments were designed: a) in one instance the starting population of OSS adopters was chosen using individual or node-level centrality values, and b) in the other case the initial population of OSS adopters was chosen using group-level centrality values.

Given the exploratory nature of this investigation, it was decided to first investigate the importance of individual-centrality measures on the diffusion of OSS. The results of those experiments were used to determine what experiments to run for group-centrality measures. Although some recent research suggests that important groups instead of important individuals should be of interest in very large networks (Pandit et al, 2008), it was decided that in the absence of guidelines for a) appropriate selection of centrality measures in the context of software diffusion, and b) determining whether a network is 'very large' for a particular research context, that both individual and group centrality measures should be investigated in this order.

3.3.1 Individual Centrality-Based Experiments

At the start of the simulation, OSS was systematically assigned to a small proportion of structurally important firms in the network. Structural importance was measured by computing individual centrality values for the four different centrality measures described earlier, using UCINET (Borgatti et al, 2002): a tool commonly used in social network analysis (see Appendix E for details). The rest of the model, including agent behavior and interaction between the agents, remained unchanged from Chapter 2. Figure 10 provides an overview of the revised model. In Chapter 2, different levels of six selected critical variables were analyzed over a series of experiments. The experiments had demonstrated that under some conditions diffusion did not occur. However, in that model, the starting population of OSS was randomly selected. In this study it was decided to retain most of the parameter values from those experiments to determine if strategic selection of the initial population of OSS adopters caused diffusion to occur in many cases where it did not occur with random selection of the initial population of OSS adopters.

- A network (with pre-specified topology and density) of 1000 firms is generated based on Watts and Strogatz algorithm (1998)
- Network-level attributes such as frequency of proprietary and open source software upgrades, strength of interoperability costs etc. are assigned
- Firm-level attributes such as size of the firm, current choice of software, level of OSS technical expertise etc. are assigned
- For each firm the following steps are repeated for 100 simulated time periods

  Step 1: If the firm is at the beginning of its planning horizon, proceed to step 2, otherwise proceed to step 4

  Step 2: The firm evaluates its decision function to decide whether to upgrade existing software or switch to the alternative software. The decision function takes into account economic factors (such as costs in case of upgrading/switching the software) and social factors (such as the decision of the firm's neighbors)

  Step 3: If the firm switches to the alternative software, it obtains new costs (setup, training, license, support costs etc.); otherwise, it upgrades existing software

  Step 4: Proceed to next simulated time period

Figure 10. Revised Simulation Process

The original values included, a) three network topologies (random, small world, clustered), three network densities (low, medium, high), three OSS support cost distributions (low mean-low variability, low mean-high variability, high mean-low variability), three levels of interoperability costs (low, medium, high), two frequencies of PS upgrades (low and high) and two proportions of initial OSS adopters (low and high). Please refer to Chapter 2 for the exact parameter values. Apart from the following two parameter values, all other parameter values were retained for the analysis of individual centrality-based assignment of OSS to the initial population: a) clustered network was dropped because in a clustered network all nodes are structurally equivalent and cannot be differentiated based on the selected centrality measures, b) very high mean-low variability OSS support cost distribution was dropped because simulations in Chapter 2 revealed that OSS diffusion did not occur with very high OSS support cost. As a result, the following experiments were run to investigate the effect of social structural measures on the diffusion of OSS: 2x3x2x3x2x2x5=720: – network topology (random and small-world), network density (low, medium and high), OSS support costs (low, high), interoperability costs (low, medium and high), duration of PS upgrade cycle (short, long), initial proportion of OSS adopters (low, high) and OSS assignment criterion at the start of the simulation (random, degree centrality-based, betweenness centrality-based, closeness centrality-based and eigenvector centrality-based assignment). Please refer to Chapter 2 for the actual parameter values used for these variables.

Network structure (topology + density) was randomly generated using the Watts and Strogatz algorithm (Watts and Strogatz 1998) and 50 samples were drawn for each network structure.[1] During the simulation 50 samples were drawn for each one of the additional random variables as well, such as level of technical capability of the firms, OSS support costs, size of the firm etc. Consequently, 2500 sample paths (i.e. n=50x50=2500) were generated for each experimental condition[2]. These sample paths were generated five times for every experimental condition. In the first set of sample paths, OSS was *randomly* assigned to a predefined proportion of the network population and the diffusion of OSS was monitored over the course of the simulation. The simulation ran for 100 time periods and diffusion was said to occur when the number of OSS adopters doubled over the course of the simulation (Dalle and Jullien, 2001). In each of the next four runs, OSS was selectively assigned to the same proportion of the network population based on individual centrality measures: degree, closeness, betweenness and

---

[1] There were 6 unique network structures: 2 topologies x 3 densities and 50 samples for each resulted in 300 different network structures.

[2] "A sample path is a collection of time-ordered data describing what happened to a dynamic process in one instance." (Hyksova 2003)

eigenvector centrality – each time the simulation was run for 100 time periods. For example, if the initial proportion of OSS adopters was defined to be 30%, then in case of degree centrality, the firms were sorted in descending order of degree centrality and 30% of firms with the highest degree centrality were designated as initial OSS adopters. The rate of diffusion of OSS, defined as the time it takes for diffusion to occur, was measured at the end of each run and compared across the different runs using repeated measures ANOVA. All reported results were significant at $p<0.05$.

Given their computationally expensive nature, the simulations were run on a Linux-based cluster which had over 200 CPU cluster blade servers, Intel Xeon CPUs and gigabit Ethernet interconnections with 2TBs of dedicated network attached storage.

3.3.1.1 Individual Centrality-Based Assignment Results

Diffusion occurred in 234 or about 33% of the 720 experiments. *Diffusion did not occur in most cases when the interoperability costs or OSS support costs or both were too high or when the initial proportion of OSS adopters at the start of the simulation was low*. Overall the results demonstrated that *when diffusion of OSS did occur, it was always faster with strategic selection of the starting population of OSS than random selection of the starting population*. This is an important, albeit seemingly trivial, result as it validates the findings from the literature on economics of social networks in the context of software diffusion i.e. strategic location of firms in a network can significantly influence the process of software diffusion. The more interesting aspects of the results, however, were in the relative importance of the various centrality measures under different simulation conditions.

Eigenvector centrality based assignment of OSS at the start of the simulation resulted in the most number of diffusion cases (64 or 27%), followed by closeness (20%), degree (20%), betweenness (18%) and random assignment (15%). Diffusion occurred more in small-world networks (150 cases) than in random topology networks (84 cases). Within the random topology network, diffusion never occurred when the initial proportion of OSS adopters was low. Furthermore, the frequency of PS upgrades did not change the impact of the strategic assignment criteria on OSS diffusion.

It was found that in random topology networks, regardless of network density, selection of the initial population of OSS on the basis of betweenness centrality resulted in the fastest rate of diffusion compared to any other OSS assignment criteria. This result suggests that in a random network, where firms do not form a very cohesive group, structural importance may not be determined by sheer number of neighbors (degree centrality), or connections to other significant firms (eigenvector centrality), or even average distance from other firms (closeness centrality). Rather the importance is determined by how central a firm is to the network-wide communication between various other pairs of firms. The explanation lies in the fact that no other centrality measure is able to exploit the characteristics of a random network to help differentiate between nodes as effectively as betweenness centrality: a firm with random connections is as close to other firms in the network as any other firm (so closeness centrality can be expected to be similar for the firms); random connections imply that there is no 'order' in the connections which are spread all over the network and therefore firms cannot as easily distinguish themselves on the basis of their neighbors and neighbors' neighbors (eigenvector centrality); similarly, having more connections (high degree centrality) can help some firms but by virtue of the fact that the connections are random, such firms are unable to drive diffusion as rapidly as betweenness centrality. With random connections, some firms are likely to fall on the shortest paths between other pairs of nodes. Hence importance on the basis of betweenness centrality will automatically become a better differentiating factor for firms than any of the other centrality measures.

Interestingly, however, in small-world networks no single OSS assignment criterion was found to be superior under all simulation conditions. Overall, eigenvector centrality-based assignment of OSS resulted in the highest number of diffusion cases (30%), followed by degree centrality (19%), closeness centrality (19%), betweenness centrality (16%) and random assignment (16%). Unlike in random topology networks, betweenness centrality based assignment of OSS was almost as ineffective as random assignment of OSS. The explanation lies in the structure of small world networks. Typically in small worlds, most firms are tightly clustered and some have global connections or less clustered connections (Watts and Strogatz, 1998). Therefore, betweenness centrality of most firms will be somewhat the same. On the other hand, given the presence of global connections, some firms will be more influential than others i.e. the eigenvector centrality of some of the firms will be substantially different from those of the other firms in the network. In the context of software diffusion, this means that in small worlds, firms with both local (clustered) as well as global (random) connections are more important as they will drive the diffusion of OSS both locally and globally throughout the network. Notice also that having more connections (high degree centrality) in general is more valuable in a small world than in a random topology network. This is understandable because large cohesive set of firms of OSS adopters are more likely to drive diffusion of OSS than large number of firms that are not well integrated.

Tables 9 and 10 provide a more detailed look at OSS diffusion with different OSS assignment criteria under all possible experimental conditions. The actual numbers have been reported in Appendix A. Table 9 provides the details under small world networks whereas Table 10 provides the details for random topology networks. The columns in each table show different network densities, interoperability cost levels, OSS support costs and frequency of PS upgrades. The top half of each table shows results with low starting population of OSS adopters. The bottom half shows results with high starting population of OSS adopters. “No diffusion” means that no diffusion of OSS occurred regardless of how the initial population of OSS adopters was selected. In all other cells, if diffusion occurred with one or more assignment criterion, the criteria were listed in ascending order of rate of diffusion. To facilitate the interpretation of the tables, let us look at the highlighted cell in Table 9. The position of the cell signifies the following experimental setup: low density small world network, with low interoperability costs, low variability in OSS support costs, high frequency of PS upgrades (2 years) and a low starting population of OSS adopters. The cell contains the abbreviations of all the different assignment criteria: BC (betweenness centrality), CC (closeness centrality), DC (degree centrality), EC (eigenvector centrality) and RD (random assignment). This means that regardless of how OSS was assigned to the starting population in these network conditions, diffusion of OSS always occurred.

Table 9. Diffusion in small world networks with different OSS assignment criteria

	Low Initial Proportion of OSS											
Network Density	Frequency of Proprietary Software Upgrades = 2 years						Frequency of Proprietary Software Upgrades = 4 years					
	Low Variability in OSS Support Costs			High Variability in OSS Support Costs			Low Variability in OSS Support Costs			High Variability in OSS Support Costs		
	Low Interop	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**
Low	**EC*** CC DC RD/BC	EC	EC	**EC** CC DC RD/BC	EC	EC	**EC** CC DC RD/BC	EC	EC	**EC** CC DC RD/BC	EC	EC
Medium	EC	No diffusion		EC	No diffusion		EC	No diffusion		EC	No diffusion	
High	No diffusion			No diffusion			No diffusion			No diffusion		
	High Initial Proportion of OSS											
Low	**DC** BC CC RD EC	**CC/DC** EC RD BC	**CC** EC DC RD BC	**DC** BC CC RD EC	**DC** CC BC RD EC	**CC/DC** EC RD BC	**DC** BC CC RD EC	**CC/DC** EC RD BC	**CC/EC** DC RD BC	**DC** BC CC RD EC	**DC** CC BC RD EC	**CC** DC EC RD BC
Medium	**EC** DC CC BC RD	**EC** CC/DC	EC	**DC** BC CC RD EC	**EC** CC/DC	EC	**EC** DC CC BC RD	**EC** DC CC	EC	**DC** BC CC RD EC	**EC** DC CC	EC
High	**EC** CC/DC BC RD	EC	No diffusion	**BC** EC CC/DC RD	EC	No diffusion	**EC** CC/DC BC RD	No diffusion	No diffusion	**BC/EC** CC/DC RD	No diffusion	No diffusion

* Note the sample size in each cell was 2500

Table 10. Diffusion in random topology networks with different OSS assignment criteria

Network Density	Low Initial Proportion of OSS											
	Frequency of Proprietary Software Upgrades = 2 years						**Frequency of Proprietary Software Upgrades = 4 years**					
	Low Variability in OSS Support Costs			**High Variability in OSS Support Costs**			**Low Variability in OSS Support Costs**			**High Variability in OSS Support Costs**		
	Low Interop	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**
Low	No diffusion*			No diffusion			No diffusion			No diffusion		
Medium												
High												
	High Initial Proportion of OSS											
Low	**BC** DC CC EC RD	**BC** DC CC/EC	No diffusion	**BC** DC CC EC RD	**BC** DC CC EC RD	**BC** DC CC/EC	**BC** DC CC EC RD	**BC** DC CC/EC	No diffusion	**BC** DC CC EC RD	**BC** DC CC/EC RD	**BC** DC CC/EC
Medium	**BC** CC/DC EC RD	No diffusion		**BC** CC/DC EC RD	No diffusion		**BC** CC/DC EC RD	No diffusion		**BC** CC/DC EC RD	No diffusion	
High	**BC** CC/DC EC			**BC** CC/DC EC RD			**BC** CC/DC EC			**BC** CC/DC EC RD		

* Note the sample size in each cell was 2500

However, the relative order of these assignment criteria also tells us that diffusion was fastest when the initial population was chosen based on eigenvector centrality (EC), followed by CC and DC. RD and BC, unlike other assignment criteria, are on the same line and separated by a forward slash. This means that diffusion did occur with these assignment criteria, however, the rate of diffusion in these cases was statistically the same (i.e. $p >= 0.05$). Similarly, it can be seen that the last cell in this row in Table 9 only contains 'EC' which means that in low density small worlds, with high interoperability costs, high variability in OSS support costs, low frequency of PS upgrades and low initial proportion of OSS, diffusion only occurred when the initial population was chosen based on eigenvector centrality.

A detailed look at Table 10 did not reveal any additional interesting results other than the ones that have already been described earlier i.e. in random topology networks, diffusion did not occur with low initial proportion of OSS and with high initial proportion BC-based assignment of OSS always resulted in the fastest diffusion of OSS. On the other hand, a detail look at table 9 revealed additional interesting results:

1. If the network environment is otherwise not conducive for OSS diffusion, eigenvector centrality is the best criterion for determining strategic importance of firms in a small world network. By stating that the environment is not conducive to diffusion of OSS the implication is that (based on our findings from Chapter 2), the interoperability costs, or frequency of PS upgrades or OSS support costs are not, in any case, favorable for OSS diffusion. Under such adverse conditions, who a firm is connected to (eigenvector centrality) is crucial to driving diffusion of OSS. For example, notice that with low initial population (top half of Table 7) it

would be difficult for OSS to diffuse throughout the network (look at the number of empty cells), and its best bet of spreading throughout the network is to start with a strategically selected population based on eigenvector centrality.

2. Conversely, *if the network conditions are otherwise favorable to diffusion of OSS, then eigenvector centrality is not as effective a measure for determining structural importance as some of the other measures in small world networks*. Under favorable conditions for OSS diffusion such as low interoperability costs, low OSS support costs, high frequency of PS upgrades, firms in any case will be leaning towards adoption of OSS. These conditions will favor OSS diffusion and their effect will not be dampened by the way the firms are connected to each other. In fact, the local as well as global connections inherent in small world networks will quickly drive diffusion in clusters (locally) as well as globally across the network if there are low interoperability costs. Hence, it will be difficult to distinguish between firms in terms of their influence on the process of software diffusion just by observing their connections or neighbors' connections. Under such circumstances, targeting firms on strategic location alone may not be the best strategy.
3. With a high starting population of OSS adopters the measure of structural importance changes with the level of interoperability costs and density of the network
    a. In low density small worlds OSS diffusion is fastest with DC-based assignment under low interoperability costs but as interoperability costs are increased, CC and EC-based assignments become more effective. DC-

based assignment is effective under low interoperability costs because the more neighbors a firm has (high degree centrality), the higher its potential to quickly influence other firms throughout the network. In random topology networks high degree centrality firms are not as effective because of the random nature of the connections. In other words, a high degree centrality OSS adopter with neighbors randomly spread all over the network cannot exercise much influence on its neighbors or through its neighbors to drive OSS diffusion in a PS-dominated network because most of its neighbors (and/or neighbors' neighbors) will be PS adopters. On the other hand, in small worlds, despite overall PS dominance, the same high degree centrality firm's neighbors are in a position to spark diffusion locally (by virtue of the 'clustered' nature of some of the connections) and drive it globally (by virtue of some 'random' connections) as well. In other words, in small worlds, high degree centrality firms are more likely to be a part of a cluster of firms that may be OSS dominated which could become the driving force for network-wide diffusion of OSS.

However, what is more interesting to note is that when interoperability costs are increased, then in a PS-dominated network, the 'strength of the neighbors' used to drive OSS diffusion earlier (with low interoperability costs) now becomes more of a 'weakness'. An OSS firm with large number of neighbors (and high interoperability costs) is likely going to have many PS neighbors as well (since most of the network has PS adopters). Therefore, high interoperability costs will make it difficult for the OSS firms from influencing its neighbors as effectively as it did earlier in driving diffusion. In fact, the large number of neighbors and high interoperability costs with (potentially most of) the neighbors might even force the OSS adopter to switch to PS. Under such circumstances, an influential firm will be the one that is less encumbered by its large number of immediate neighbors. Theoretically such influence could be defined by either betweenness, closeness or eigenvector centrality because neither one determines ‘influence’ strictly on the basis of immediate set of neighbors. A closer inspection helps understand as to why closeness and eigenvector turn out to be better than betweenness as interoperability costs are increased.

Betweenness centrality measures importance on the basis of how often a firm falls on the shortest path between pairs of other firms. In small worlds, most of the firms (by virtue of the mostly lattice-like structure of the network) will have similar betweenness centrality values. Therefore, betweenness centrality alone will not be as good a differentiating factor as closeness centrality. Closeness centrality measures importance on the basis of how close a firm is to all other nodes in the network. Despite the generally lattice-like structure of small worlds, even one ‘random’ or ‘global’ connection of a firm (connecting two clusters across the network) can dramatically change its closeness value (because suddenly that firm will be close to many other firms across the network). Similarly, eigenvector centrality measures importance by considering a firm’s neighbors and neighbors' neighbors. Again, the 'random' or 'global' connections will have the potential of quickly changing the eigenvector centrality value of a firm. It is worth reiterating at this point that it is precisely because of these reasons that betweenness centrality is able to exert greater influence than other centrality measures in random topology networks – when the connections to other firms are random, a firm is as close to the other firms in the network as its neighbor (closeness centrality); similarly, a firm's neighbors' neighbors will be equally well connected (or equally less well connected) as the next firm's neighbors and neighbors' neighbors.

b. In high density small worlds, the effects described in 3a are in play but in a different manner. High density networks indicate that on average each firm has more neighbors. Therefore, in a PS dominated network, the importance of interoperability costs with respect to OSS diffusion will be magnified (regardless of whether interoperability costs are low or high). Theoretically, degree centrality-based assignment of OSS should still play the same role on the diffusion dynamics of OSS. However, with high density the total 'weight' of interoperability costs on the decision function goes up. As a result, other cost factors interact. For example, in high density networks with high variability in OSS support costs, the increased effect of interoperability costs on the decision function is not discernable and degree-centrality based assignment of OSS still turns out to be the best. However, observe that in high density networks with low variability in OSS support costs, degree centrality based assignment no longer appears to be the fastest. This is not because degree centrality is no longer affecting software diffusion in small worlds but because the low variability in OSS support costs means that firms are generally facing high OSS support costs. In other words, in dense networks, low interoperability costs are not low enough to drive OSS diffusion through high degree centrality firms. This explanation is lent further support by the fact that additional experiments were run for these network conditions with very low interoperability costs (0.1, 0.4 and 0.7). It was observed that for those levels of interoperability costs, even in higher density small worlds, degree centrality based assignment of OSS was generally the best in driving OSS diffusion.

To come to the essence of this point, recall from bullet 1 that under adverse network conditions for OSS diffusion, eigenvector centrality is the best measure for identifying structurally important firms. Therefore, in the presence of high interoperability costs in dense small worlds, eigenvector centrality based assignment outperforms all other criteria. In fact, it is interesting to note that at times when diffusion does not occur at all with any other criteria, there are a few instances where it occurs with eigenvector centrality based assignment. Why is it that firms selected on the basis of eigenvector centrality are able to overcome the increased effect of density and interoperability costs? Simply because even if on average the number of connections of such firms (with high eigenvector centrality) is high, they start the simulation as OSS adopters not because they have the most number of connections but because of their neighbors and neighbors' neighbors. Therefore, their decision functions incorporate a comparatively lesser influence from the density of connections and higher interoperability costs and they are able to drive diffusion better than any other centrality measure.

Furthermore, it was found that the relative importance of the various centrality measures was robust even when a) diffusion was said to occur with a 50% increase in the number of OSS adopters, b) limited additional levels of interoperability costs were tried (as were tested in Chapter 2).

3.3.1.2 Discussion of Individual Centrality-Based Assignment Results

The results reinforced the findings in the literature of economics of social networks by demonstrating that the strategic location of a firm in a network can significantly influence the diffusion of OSS. This is a non-trivial outcome: reinforcement of known or existing concepts or replication of reality is a well-established practice in simulation modeling and builds confidence in the validity of the model (Kwon et al, 2007; Manzoni and Angehrn, 1997). More interestingly, our results demonstrated that the criterion for determining strategic location changes depending on the network conditions and external environment within which diffusion takes place. This is an important finding from a research perspective as it discourages the use of just about any centrality measure for strategically targeting nodes in a network without understanding the prevalent network conditions and their relationship with the chosen centrality measure. Although the model was parameterized based on unique aspects of OSS, the framework itself can be extended to the investigation of other types of software as well. Furthermore, the research has contributed to the literature on economics of social networks by developing a better understanding of the software diffusion-type processes in this context. This is a valuable addition to software diffusion theory for two reasons. First, globalization and technology trends are driving a greater degree of interconnectivity among organizations and software diffusion needs to be studied in this context. Second, increasing delivery of software as a service over network facilitates a high degree of data mining by vendors, potentially leading to more sophisticated marketing. We hope our research contributes to theory building in this context. From a practical perspective, this result suggests that information regarding structural importance of firms can be exploited by vendors and third-party software providers to facilitate or inhibit diffusion of software. For example, strategically located firms could be offered a better price structure to retain them as customers or induce them to switch from their existing software to another software. This idea will be explored in detail in Chapter 4. The following subsection discusses the practical implications of the results in more detail.

3.3.1.3 Practical Implications of Individual Centrality-Based Assignment Results

It is important to better understand the practical implications of these results. Let us first examine the results in the context in which the model was parameterized: desktop operating system software market. Software vendors are not in a position to change the inter-organizational relationships between the firms i.e. the network structure is an uncontrollable factor for the vendor. Other factors such as license costs, interoperability costs, setup costs, support costs etc. are controllable factors. Our research basically suggests that vendors should, a) invest in finding out the structure of their network; b) measure the structural characteristics of the network; c) examine current cost/price structure of their software with respect to the network structure and use our results to determine what centrality criterion to use to target influential firms in the network. For example, if the vendor's analysis reveals that the target network has the characteristics of a low density small world network, then i) if there are no significant interoperability issues with the competitor's software, firms with the most number of connections to other firms should be targeted, ii) interoperability issues can be strategically manipulated to change the level of influence of some firms over other firms – the results had demonstrated that structural importance of firms would change with higher interoperability costs. What is more interesting is that with the changing market share, the vendor needs to adapt and target a different set of firms. For example, early on when the vendor does not have a large installed base of adopters then important firms are ones that have highest eigenvector centrality (in small world networks). On the other hand, as the market share of the vendor improves, depending on the density of connections and strength of interoperability costs, eigenvector centrality should not be used to identify important firms. These strategies would be applicable in case of other types of software markets as well particularly when proprietary and open source software vendors are competing for the same network of clients. Figure 11 provides a quick overview of the results and highlights the conditions under which different centrality measures would be appropriate for identifying strategically located firms in a network. Notice that all paths leading to eigenvector centrality being the appropriate measure of structural importance reflect difficult or adverse conditions for diffusion of OSS, whereas all other paths reflect favorable conditions for diffusion of OSS.

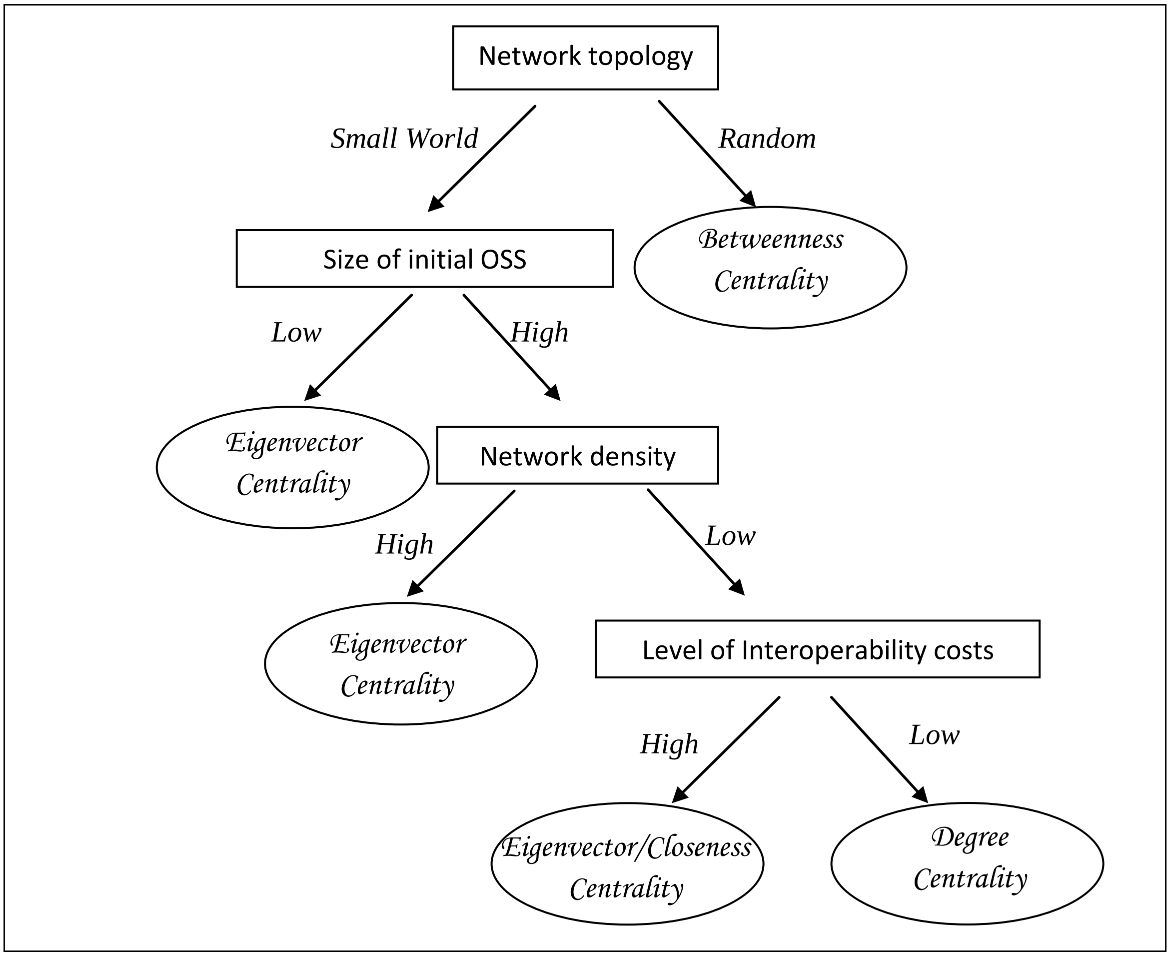


Figure 11. Overview of individual centrality-based assignment results

Next, the effect of selecting structurally important groups of nodes on OSS diffusion was analyzed. The following sections describe the experiment design, analysis and results for group centrality-based assignment of OSS to the starting population.

3.3.2 Group Centrality-Based Experiments

As mentioned earlier, research suggests that in very large networks, important groups need to be identified instead of important individuals to study any network-wide behaviors (Pandit et al 2008). However, there is no precedent in the academic or practitioner literature than can quantify the size of the network for which studying individuals may or may not be better. Therefore, in this exploratory study on the effect of structural characteristics of diffusion of OSS, it was decided to investigate both individual centrality and group centrality measures. The results from experiments using individual centralities had demonstrated that a) strategic location of individual firms is important to the diffusion of software; b) the frequency of PS upgrades does not affect the relative importance of the various centrality measures; c) random assignment of OSS to the starting population was never better than strategic centrality-based assignment. These results informed the design of experiments for group centrality-based experiments.

In this research, a group was defined to be a simple supply chain: one focal firm with its immediate partners. Therefore, in a 1000-node network, there were 1000 groups in which each firm was considered to be the focal firm in its own supply chain (or group). The group centrality measures were calculated by averaging the individual centrality measures of each member of a group (please refer to section 3.2.3 for details). The model used in Chapter 2 was used and a similar setup was employed as was used in the investigation of individual centrality-based experiments. The entire simulation process described in the previous section and depicted in Figure 1 remained the same with two major differences: a) instead of selecting the initial population of OSS adopters using individual centrality based assignment, group centrality measures were used, b) since individual centrality based experiments revealed that the frequency of PS upgrades did not significantly change the relative impact of the centrality measures, only one PS upgrade frequency was used in the experiment designs. Higher frequency of PS upgrades was chosen because that was the more realistic scenario for a PS vendor (Keizer 2007).

First, all group centralities were computed and the firms were sorted in the order of highest group centrality (where in each case the firm was the focal firm of its own group). Second, group members with the highest centrality value were assigned OSS. If the size of any group was bigger than the required initial proportion of OSS to be maintained for the experiments, then only the required number of individual nodes (with highest centrality within the group) were assigned OSS. It is important to understand this difference in assignment of OSS to the starting population in comparison with the individual centrality-based experiments. Under individual centrality-based assignment of OSS, individual firms could have been picked from anywhere within the network and their neighbors may or may not have been assigned OSS. On the other hand, with group centrality-based assignment, OSS always got assigned to a group of firms. This fact alone would impact the diffusion dynamics very differently because in one case strategically located OSS adopters were spread across the network (individual centrality assignment) and may or may not have been connected to each other, whereas in the other case they were directly connected to each other (group centrality assignment).

Then the simulation (Steps 1 – 4 described in Figure 14) was run as usual. The rate of diffusion was measured across the different experimental conditions: 2x3x2x3x1x2x5=360 – network topology (random and small-world), network density (low, medium and high), OSS support costs (low, high), interoperability costs (low, medium and high), duration of PS upgrade cycle (long), initial proportion of OSS adopters (low, high) and OSS assignment criterion at the start of the simulation (individual random assignment, group degree centrality based assignment, group betweenness centrality based assignment, group closeness centrality based assignment and group eigenvector centrality based assignment). The simulations were run on a cluster and the results were analyzed using repeated measures ANOVA. All reported results were significant at $p<0.05$.

Notice that as a comparison case, individual random assignment was used instead of group random assignment. There were two rationales for making that decision. First, individual centrality based assignment of OSS always produced superior results compared to random assignment of OSS. Therefore, group-random assignment-based experiments were not expected to produce any surprising results. Second, the objective with group-centrality based experiments fundamentally was to compare their performance with individual-centrality based experiments. Therefore, in some ways, the comparison cases for these experiments were the results from individual-centrality based experiments.

3.3.2.1 Group Centrality-Based Assignment Results

Diffusion occurred in 100 or about 28% of the 360 experiments. *Diffusion mostly occurred in small world networks* (66 of the 100 cases). Tables 11 and 12 show the detailed results for small world and random topology networks respectively. In small world networks *individual random assignment was able to outperform group centrality based assignment in low density small worlds with high initial proportion of OSS adopters*. Recall from our earlier discussion that by nature, in small worlds, there are groups of firms which are well connected with each other and have overlapping nodes and connections with some groups across the network. The overlapping groups point towards redundant connections in terms of software diffusion. Recall also that in low density small worlds with a high starting population of OSS adopters, the conditions are somewhat favorable for OSS diffusion. Therefore, targeting strategically located groups that might be overlapping and interconnected will not increase the chances of spreading OSS throughout the network as much as targeting randomly selected firms across the network. The latter will increase the chances of promoting diffusion locally in groups across the network and then globally throughout the network as well.

Table 11. Diffusion in small world networks with different group OSS assignment criteria

<table>
<tr><th></th><th colspan="6">Low Initial Proportion of OSS</th></tr>
<tr><th rowspan="3">Network Density</th><th colspan="6">Frequency of Proprietary Software Upgrades = 4 years</th></tr>
<tr><th colspan="3">Low Variability in OSS Support Costs</th><th colspan="3">High Variability in OSS Support Costs</th></tr>
<tr><th>Low Interop</th><th>Medium Interop</th><th>High Interop</th><th>Low Interop</th><th>Medium Interop</th><th>High Interop</th></tr>
<tr><td>Low</td><td>BC
DC
CC
EC
RD</td><td colspan="2" rowspan="3">No diffusion</td><td>BC/DC
CC
EC
RD</td><td colspan="2" rowspan="3">No diffusion</td></tr>
<tr><td>Medium</td><td>BC
DC
EC
CC</td><td>BC
DC
EC
CC</td></tr>
<tr><td>High</td><td>No diffusion</td><td>EC
DC/BC/CC</td></tr>
<tr><th></th><th colspan="6">High Initial Proportion of OSS</th></tr>
<tr><td>Low</td><td>RD
BC
DC
CC
EC</td><td>RD
CC
DC/BC/EC</td><td>RD
CC</td><td>RD
EC
CC
BC
DC</td><td>RD
EC
CC
DC/BC</td><td>RD
CC</td></tr>
<tr><td>Medium</td><td>BC/CC/DC
EC
RD</td><td colspan="2" rowspan="2">No diffusion</td><td>RD
CC
DC/BC
EC</td><td colspan="2" rowspan="2">No diffusion</td></tr>
<tr><td>High</td><td>DC
BC
CC
EC
RD</td><td>RD
DC/BC
CC/EC</td></tr>
</table>

On the other hand, *with low initial proportion of OSS, group betweenness centrality-based assignment resulted in the fastest diffusion of OSS.* The explanation lies in the fact that if the installed base is going to be small, targeting groups whose members are strategically located on the shortest paths between other pairs of nodes in the network makes more sense than randomly selecting nodes across the network. Random selection is able to take advantage of the large initial proportion. However, in the absence of that large installed base, it is more appropriate to target groups whose members are strategically located throughout the network.

Table 12 shows that *in random topology networks if the initial proportion of adopters is low then diffusion does not occur regardless of which OSS assignment criterion is used*. However, *if the installed base is large, then it should be targeted based on group degree centrality*. This makes sense because in the absence of the inherent clustered and overlapping connections found in small worlds, diffusion can be driven rapidly in random topology networks by influencing as many nodes as possible. From that perspective, firms whose members are connected to most other firms in the network should be targeted.

3.3.2.2 Discussion of Group Centrality-Based Assignment Results

Our analysis revealed several interesting results for understanding structural importance of groups in the context of software diffusion:

First, *in random topology networks, strategic assignment of OSS based on group centrality measures is better than random assignment*. Therefore, in random topology networks, vendors should invest in identifying and targeting influential groups to encourage diffusion of their software. Second, *influential groups in random networks are those that have the most number of connections with other firms in the network*. The underlying concept is that when the connections are not cohesive then the objective should be to target high degree centrality nodes to increase the chances of diffusion. Third, *in small-worlds, where groups of firms are fairly cohesive but have some random* *ties with other firms in the network, influential groups are those that are central to the network-wide communication between other groups in the network* (high group betweenness centrality).

Table 12. Diffusion in random topology networks with different group OSS assignment criteria

<table>
<tr><th></th><th colspan="6">Low Initial Proportion of OSS</th></tr>
<tr><th rowspan="3">Network Density</th><th colspan="6">Frequency of Proprietary Software Upgrades = 4 years</th></tr>
<tr><th colspan="3">Low Variability in OSS Support Costs</th><th colspan="3">High Variability in OSS Support Costs</th></tr>
<tr><th>Low Interop</th><th>Medium Interop</th><th>High Interop</th><th>Low Interop</th><th>Medium Interop</th><th>High Interop</th></tr>
<tr><td>Low</td><td colspan="3" rowspan="3">No diffusion</td><td colspan="3" rowspan="3">No diffusion</td></tr>
<tr><td>Medium</td></tr>
<tr><td>High</td></tr>
<tr><th></th><th colspan="6">High Initial Proportion of OSS</th></tr>
<tr><td>Low</td><td>DC
BC/CC/EC
RD</td><td>BC/CC/DC/EC</td><td>No diffusion</td><td>BC/CC/DC/EC</td><td>DC
BC
EC
CC
RD</td><td>No diffusion</td></tr>
<tr><td>Medium</td><td>DC
BC
CC
EC
RD</td><td colspan="2" rowspan="2">No diffusion</td><td>DC
EC
BC
CC
RD</td><td colspan="2" rowspan="2">No diffusion</td></tr>
<tr><td>High</td><td>No diffusion</td><td>DC
EC
BC
CC
RD</td></tr>
</table>

Fourth, *in low density small worlds randomly selected firms have a better chance of driving network-wide diffusion than strategically selected groups*. This is somewhat counterintuitive. However, the explanation lies in the fact that in low density small worlds, diffusion of OSS might get ‘stuck’ or slow down as it spreads within cohesive groups whereas randomly selected firms from all over the network are able to more quickly start and drive diffusion across different parts of the network. This offers some very interesting implications, a) from a software vendor's perspective, targeting groups to encourage diffusion might not always be the best strategy, b) if the target network resembles a low density small world, lack of information regarding the details of interconnections between consumers may not be a disadvantage.

3.3.3 Comparison between Individual and Group Centrality-Based Assignment Results

An overview of the overarching results with individual centrality and group centrality based assignment is shown in Table 13.

Table 13. Comparison between individual and group centrality based assignment results

Topology	Low Starting Population		High Starting Population	
	Low Density	High Density	Low Density	High Density
Random	No diffusion		Ind. BC Grp. DC	
Small World	Ind. EC Grp. BC	Ind. EC	Ind. DC RD	Ind. EC

It is interesting to note that in random topology networks, if individuals have to be targeted, they should be targeted based on betweenness centrality. However, if groups have to be targeted then groups with most connections with other firms in the network should be targeted. It makes sense that given the nature of the 'random' connections in random topology networks, targeting individuals with large number of connections (high degree centrality) may not be as effective as targeting individuals who are on the shortest paths between pairs of other nodes in the network. In other words, having more connections with other firms might not guarantee faster software diffusion by virtue of the lack of cohesion between those firms. Recall that in individual centrality based assignment firms getting OSS at the start of the simulation may not be connected to each other. This means that they must be strategically placed to significantly affect the process of diffusion. On the other hand, targeting groups whose members are connected to many other nodes (high group degree centrality) increases the chances of reaching out to more firms in the network. In such a case, even if the connections in the network are random, just by targeting firms that can target many other firms, the chances of diffusion are rapidly improved. Again it is important to take note that with group centrality-based assignment of OSS at the start of the simulation, the OSS firms are connected to each other in which case each one of those firms need not be located on some strategic path in the network.

In low density small worlds, if the initial proportion is low, groups whose members are strategically placed on the shortest paths between other pairs of nodes (high group betweenness centrality), or individuals who have high eigenvector centrality, should be targeted. Recall that low initial proportion of OSS is not favorable for diffusion of OSS. Therefore, if individual firms have to be targeted, then the ones with well connected neighbors and neighbors' neighbors (highest eigenvector centrality) should be targeted. Individual betweenness centrality is not effective because given the nature of the low density small worlds most individual firms tend to have similar betweenness centrality values. However, what is more interesting is the fact that group betweenness centrality is very effective in driving diffusion instead of group eigenvector centrality. It is easy to understand that with a small installed base of adopters, targeting groups whose members are strategically placed on the shortest paths between pairs of other nodes should help. However, it was observed that individual betweenness centrality of firms in small worlds were not significantly different from each other and that is why in case of individual centrality-based assignment BC was not as effective as EC. The explanation lies in the fact that group centralities are computed by averaging individual centrality values of the group's member firms. With eigenvector and closeness centrality, when averages are computed for a group, those averages reduce the differences between individuals (hence groups) in terms of group eigenvector and group closeness centralities. The group selected now may not have the most strategically located firms. On the other hand, since the individual betweenness centralities are somewhat similar, those values are not attenuated as much.

Similarly, if there is high initial proportion of OSS in low density small worlds, then individual firms with most neighbors (high degree centrality) or randomly selected firms should be targeted (instead of any group centrality-based targets). The rationale for both choices is to take advantage of the high initial proportion and reach out to as many new firms across the network as possible. Individual degree centrality based assignment will allow that – by targeting firms first locally within the small worlds and then globally. Random assignment will achieve that by first reaching out to firms spread globally across the network and then initiating diffusion locally in different areas within the network.

Surprisingly, it was found that the fastest rate of diffusion using group centrality-based assignment was always slower compared to the fastest rate of diffusion using individual centrality-based assignment for the same experimental conditions. This is a somewhat counterintuitive result as one might expect that a group of firms targeted simultaneously might speed up diffusion more than strategically located individuals spread throughout the network. Prior research points in some interesting directions in this regard. Some research suggests that in large networks groups might be more influential than individuals in affecting network-wide behavior (Pandit et al, 2008). Other findings, though in a different context of diffusion, also suggest that "subgroup structure of social network affects the scale of computer epidemics indirectly through interaction with individual-level centrality measures" (Cheng and Guo, 2008). Without further experiments and detailed analysis it would be difficult to accurately explain this result. However, based on the model and experiment design and known results, two explanations can be offered:

First, it is possible that the size of our network (1000 firms) is too small and comparatively its density is very high. High density in small network would mean a number of redundant connections (for the purposes of diffusion) within a 'group'. Therefore, when we target groups, the redundant connections "lock down" diffusion within that dense group and if at all the members of the group start affecting other groups, the process is slow by virtue of the dense connections. On the other hand, for the same dense network, if individuals across the network are targeted, they are likely to speed up diffusion more than when groups are targeted because they may not be encumbered by each other's dense connections i.e. they may not end up being part of the same group. If this explanation is correct, then one could expect that networks whose size (number of nodes) to density (average number of links per node) ratio is high might allow group centrality based assignment to be superior compared to individual centrality-based assignment in the context of software diffusion.

Second, the method of assigning OSS to the initial population using group centrality values might be slowing diffusion down in general as well. This is related to the first point mentioned above. If a group of firms are assigned OSS, as against individuals spread across the network, it can be expected that diffusion will be "locked in" (due to early PS domination) and will be slower to spread throughout the network. In a low density network that may not happen as much compared to a higher density network. So for example, it was observed that the differences between the fastest rates of diffusion using individual and group centrality assignments increased as the network density increased (regardless of network topology). In fact, in many cases diffusion failed to occur with group centrality based assignment as the density of the network was increased.

3.4 Contributions to Research and Practice

The overall results of our experiments demonstrated that a) strategically located firms or groups of firms within a network can significantly influence the diffusion of OSS. b) there is value in knowing the underlying structure of the network of consumers. Although the concept of targeting groups to drive network-wide adoption/diffusion is not new – AT&T offers myFavs which allows subscribers to identify their clique or group – these results enrich our understanding of network effects in the context of software adoption and diffusion.

From a research perspective we have aimed at developing a better understanding of the commonly used centrality measures in the context of software diffusion at both individual as well as group levels: a) commonly used centrality measures can be effectively applied in the investigation of software diffusion type network flow processes, b) despite the same network conditions, individual and group centrality measures may behave differently in affecting network-wide phenomena, c) having deeper knowledge regarding interconnections in a social network may not always be a source of advantage in an attempt to influence network-wide phenomena (such as diffusion). This in particular is an important insight that should provide some guidance for future research on social networks. Also, from a software diffusion perspective we have established the importance of understanding the social network of the target population in explaining network wide adoption and diffusion of OSS. Although the model was parameterized based on unique aspects of OSS, the framework itself can be extended to the investigation of diffusion of other types of software as well. It is possible that the degree of influence of some of the centrality measures, depending on the actual numbers being used in the simulation, might change. However, the direction of their effects should not vary significantly from what we have described.

From a practitioner's perspective, we have demonstrated that if the strategic importance of firms can be established based on some criteria (betweenness, closeness etc.) in the context of software diffusion, vendors can take advantage of such insights and improve their targeted marketing and sales practices. This possibility is later explored in more detail in Chapter 4.

3.5 Assumptions, Limitations and Future Research

There are a few limitations of this research. First, only two types of network topologies were used to investigate the effect of the various centrality measures. Although those two topologies are two of the most commonly used topologies in the literature on analysis of social networks, it would be worthwhile to run these experiments under other simulated (such as scale-free) or real world network topologies.

Second, for measuring group centrality values, a group was defined to be a simple supply chain. In social network analysis and graph theory, there are various definitions of groups of nodes in a network. Many of those were explored in the context of our research, such as: cliques, N-cliques, clans, N-clans etc. However, it was concluded that given the focus of our research (investigation of centrality measures) and the nature of our network flow process (software diffusion), a simple supply chain would be a simple yet powerful approximation of a ‘group’ in a network of interconnected firms. Future research could further explore the robustness of our findings based on different definitions of a group.

Third, we approximated group centrality measures based on individual centrality measures of members of the group. This method of computing group centrality does not pose a serious problem for the purpose of this investigation for the following four reasons: a) if large-sized groups have very important firms (or firms with high centrality values, regardless of which centrality measure is chosen), then their average will not be seriously attenuated; b) in our network, the average number of neighbors (density of the network) is the same and multiple replications of networks (with similar topology and density) are made to ensure that the effect of the various centrality measures can be robustly measured; c) in the absence of an agreed-upon definition of a group in the context of inter-organizational relationships, it is appropriate to assume the simple supply chain to be one representation of a group. This definition of a group allows firms that are part of multiple supply chains to contribute to the group centrality values different groups at the same time. This is important because there is no accepted norm for capturing overlapping group memberships in the computation of group centrality values in the literature on social networks; d) the average measure should theoretically bring the computed centrality values of the groups closer to each other and make it harder to differentiate between their effects in the statistical analysis. However, as was demonstrated earlier, the results for various group centrality measures did not always come out to be statistically the same. Again, although using average as a surrogate for an actual group centrality measurement is an accepted approximation in the analysis of social networks, some nuances of the groups might be lost with such aggregate measures of group centrality. Future research could explore alternative measurements of group centrality and investigate their impact on diffusion of OSS.

Fourth, it is important to note that the unit of analysis in our study was a firm. Although firm-level decisions may not always be made regarding various software solutions, the model and its findings can be extended to other levels of analyses. For example, the unit of analysis could be a department within a network of departments or a project in a portfolio of projects within an organization. At such levels of analysis the actual numbers adopted in the simulation model may vary, however, the behavior of the agents, the components of their decision functions and the nature of social and economic interdependence between agents (whether they be departments or projects) would still permit the application of the model.

CHAPTER 4: THE IMPACT OF NETWORK-AWARE PRICING VERSUS TRADITIONAL SOFTWARE PRICING SCHEMES ON THE DIFFUSION OF OPEN SOURCE SOFTWARE

4.1 Introduction

In the previous chapter it was shown that a starting population strategically located within the network can significantly influence the process of software diffusion. As a consequence it is natural to ask that if a software vendor were to have knowledge regarding the structural characteristics of the network, can the vendor target strategically located firms within the network to trigger or further drive the diffusion of its software? More specifically this chapter explores the concept of "network-aware" pricing i.e. price discrimination on the basis of location information of the firms. Since the objective is not to propose a new pricing scheme or to discuss the optimal conditions under which location-based pricing may or may not work, findings from Chapter 3 are used as a basis to simply demonstrate the concept of network-aware pricing.

There are different ways in which vendors can target important firms. For example, vendors often resort to price discounting to attract larger customers (Geisman and Maruskin, 2006; Holden 2008). In this chapter, we explore the concept of price discounting on the basis of location information of the firms. The idea that a vendor has information regarding the network structure of the clients is not new. There are ways of estimating data regarding the network of customers through empirical and computational techniques (Westarp 2003).

Also, there is a possibility that customers may share some information regarding their network with the service provider in order to secure a better deal. For example in the cell phone industry AT&T's myFavs offers customers low calling charges to a select group of 'favorite' numbers identified by the customers. Given the rising trend toward usage-based licensing or on demand software delivery, customers might be willing to share network or network-based usage information to secure a more economical deal with the service provider. Therefore, we evaluate the effectiveness of this pricing scheme against traditional size-based price discounting scheme in which discounts are offered based on the size of the deal (Geisman and Maruskin 2006). More formally, our research question is: *From a vendor's perspective, is network-aware pricing more effective than traditional pricing schemes?*

The model developed in the previous chapters is used in this chapter. In Chapter 2 it was demonstrated that under some conditions the proprietary software (PS) vendor can eventually lose market share to OSS despite being vastly dominant in the market at the start of the simulation. In that model, the PS vendor was passive and did not react to the changing market conditions. However, in this study it is assumed that the PS vendor is aware of the strategic location of the firms and acts proactively by changing its pricing scheme to retain its dominance in the market.

The rest of this chapter has been organized as follows: the next section provides a brief review of the traditional software pricing schemes. This is followed by a description of the model and experiments. Then the results are presented along with a discussion on the implication of these results. The chapter is concluded with a brief description of the limitations and ideas for future research.

4.2 Literature Review

This section provides a brief review of the literature on traditional software pricing schemes. Since the objective of present research is not to evaluate or recommend optimal software pricing strategies in general, only relevant literature will be reviewed. For a detailed review of various software pricing strategies please refer to an exploratory study recently conducted by Lehmann and Buxmann (2009).

4.2.1 Traditional Software Pricing Schemes

There have been several studies in the past that have investigated the issue of pricing software or information goods (Bontis and Chung, 2000; Brynjolfsson and Kemerer, 1996; Chakravarty et al, 2006; Foley, 2004; Gallaugher and Wang, 2002; Sundararajan, 2004; Gandal, 2004). Determining the right price for software remains a “complex and subjective process” (Bontis and Chung, 2000: p. 247) primarily because it is hard to a) put a price on the development of such an “intangible asset” (Bontis and Chung, 2000), and b) determine what the customer will be willing to pay for it (Gallaugher and Wang, 2002; Foley, 2004) as every customer may not derive the same level of value from the same software (Bontis and Chung, 2000). Other factors that might determine or affect the price of software include “network externalities, cross-market complementarities, standards, mindshare, trialability” (Gallaugher and Wang, 2002).

Researchers do agree that any pricing scheme must take into consideration the value associated by the buyer with the software. In an attempt to more accurately capture this value proposition, vendors resort to price discrimination. Lehmann and Buxman (2009) discuss three types of price discrimination strategies in the context of pricing software: 1) in first-degree price discrimination consumers are offered a price based on their willingness to pay or value proposition. As mentioned above, this is hard to accurately achieve in the case of software; 2) in second-degree price discrimination the consumer chooses between different price-product offerings that may vary on the basis of performance, timing of availability and quantity; 3) in third-degree price discrimination, as against second-degree, the vendor discriminates by segmenting the market into nature of use (e.g. business use, student use etc.) or region of use (e.g. based on country). Typically, vendors offer a combination of second- and third-degree price discrimination strategies (Lehmann and Buxman 2009). Functionality and version based pricing (Simonetto and Davidson 2005; Sundararajan 2004) can be considered to be examples of third-degree price discrimination whereas named user, volume-based, concurrent, site and upgrade pricing (Simonetto and Davidson, 2005; Sundararajan, 2004) can be considered examples of second-degree price discrimination. Some of these pricing schemes (such as volume-based and named user licensing) are usage dependent whereas others (such as functionality and version based licensing) are usage independent. Research suggests that vendors expected usage-dependent pricing schemes to gain prominence in the future, however, users preferred usage-independent schemes as these did not involve "problematic cost calculation, and the selection of a concrete, meaningful assessment base" (Lehmann and Buxmann 2009: p. 460).

In addition to these pricing schemes that target customers on the basis of the value they associate with the software, vendors often resort to offering high discounts to further penetrate the market or to meet revenue targets (Holden 2008; Lehmann and Buxmann 2009). This is similar to quantity-based second-degree price discrimination where key customers are large customers and are offered a reduced per-unit price. However, recent empirical investigations have revealed that in the software industry vendors do not strategically plan the discount offerings and have limited understanding of their long-term implications (Geisman and Maruskin, 2006). The evidence suggests that despite oversight on individual cases, the absence of an overarching strategy or budget results in arbitrary spending on discounts. Although conventional wisdom dictates that higher discounts should be offered to secure bigger deals, the empirical findings do not validate this idea (Geisman and Maruskin 2006). Therefore, there is a need for software vendors to be able to a) identify which potential or existing clients should be offered discounts, and b) better manage the discount dollars that are spent in the process of securing new customers and/or meeting revenue targets.

In this chapter, we propose that information regarding a firms' strategic location in a network should be taken into consideration in order to determine which customers should be targeted for a discount. This notion stems from the understanding that strategically located firms can significantly influence software diffusion or market penetration (Chapter 3). Furthermore, if the use of software entails network externalities and interoperability issues are involved, a firm's value proposition of the software will likely be affected by its neighbors (or neighbors' neighbors). Therefore, from the perspective of targeting customers on the basis of their value proposition as well as their ability to penetrate the market, it is essential to take information regarding their location into account to inform any pricing strategy. The notion of understanding the network structure of existing or target clients is neither new nor extremely difficult. There are ways of estimating data regarding the network of customers through empirical and computational techniques (Westarp 2003). Also, there is a possibility that customers may share some information regarding their network with the service provider in order to secure a better deal. For example in the cell phone industry AT&T's myFavs offers customers low calling charges to a select group of 'favorite' numbers identified by the customers. Given the rising trend toward usage-based licensing or on demand software delivery, customers might be willing to share network or network-based usage information to secure a more economical deal with the service provider.

4.3 Simulation Model

The simulation model developed in Chapter 2 was used as the underlying framework in this study. The model simulated a network of 1000 firms using a desktop operating system. The firms chose between proprietary and open-source alternatives of a desktop OS while trying to maximize their cost savings. At the start of the simulation OSS was assigned to a small, randomly selected, population of the network. The model assumed that the PS vendor was passive to the zero license/upgrade price offered by the OSS vendor. In other words, if the PS vendor gained or lost market share or its profitability was affected, the price offered to the clients was never changed. The results demonstrated that under many conditions, the PS vendor did not lose market share and under other conditions, network topology, network density and interoperability costs were some of the more critical variables that affected the PS vendor's market share.

In Chapter 3, the effects of network topology and network density were explored in more detail on the basis of the literature on economics of social networks. Instead of randomly assigning OSS to a small starting population in the simulation, OSS was assigned to selected firms based on structural importance (using various centrality measures). This investigation revealed that a) structurally important firms significantly affected the course of diffusion of OSS, b) under different conditions (including network topology, network density and interoperability costs) the criterion for identifying structurally important firms was different. These results suggested that from a marketing perspective, if a PS vendor were to offer discounts to structurally important firms, the diffusion of competing software might be prevented or slowed down. Hence, the present study was designed to demonstrate the concept of "network-aware pricing".

There are three important points to be made at this point before describing the model for the present study in detail. First, it is essential to reiterate that the importance of structural information of firms was demonstrated in Chapter 3 when the starting population of OSS adopters was chosen on the basis of that information. In this study, however, as the rest of the section will demonstrate in detail, the structural information of the firms was not used in a similar manner. That defines a crucial difference between the two studies: *just because structural information was found to significantly affect software diffusion in Chapter 3, it cannot be assumed that the use of that information in the present study will also significantly influence the outcomes*. Second, in Chapter 3, it was found that firms selected on the basis of either individual or group centralities significantly affected the rate of diffusion of OSS. However, rate of diffusion of OSS with individual centrality-based assignment of OSS was always better than the rate of diffusion of OSS with group centrality-based assignment of OSS. Therefore, in this study it was decided to test the concept of network-aware pricing using only individual centrality-based assignment. Third, it was found in Chapter 3 that different centrality measures helped identify structurally important firms under different environmental conditions. These environmental conditions were defined by the six critical variables investigated in Chapter 2: network topology, network density, interoperability costs, OSS support costs, frequency of PS upgrades and size of initial population of OSS adopters. Therefore, it was decided to use all four centrality measures to demonstrate the effectiveness of “network-aware” pricing. In order to do that, different environemtnal conditions, appropriate to the selected centraltiy measure, had to be setup. For example, to demonstrate the effectiveness of network-aware pricing using information on the betweenness centrality of firms, a random network had to be setup with a high initial population of OSS adopters. This is because it was found in Chapter 3 that a) diffusion in random networks only occurred with high initial population of OSS adopters, b) when diffusion did occur with high initial population of OSS adopters, it was fastest with betweenness centrality based assignment of the initial population of OSS. On the other hand, if the effectiveness of network-aware pricing had to be demonstrated with information on the eigenvector centralities of firms, a random topology network with high initial population of OSS adopters could not be used because results from Chapter 3 had demonstrated that under those conditions structural importance of firms was more effectively captured using betweenness centrality instead of eigenvector centrality. With these important ideas in mind, let us now consider the design of the model for the present study in more detail (see Figure 12).

Instead of strategically selecting a starting population of OSS adopters, OSS was assigned to a small randomly selected set of firms. At the start of the simulation, a competitive PS vendor offered per-license discounts to some firms in the network (regardless of whether they were PS or OSS adopters). In light of the research question, three criteria were used for selecting the firms for discounts: a) size-based discounts –

higher discounts to be offered to firms with larger number of licenses, b) location-based discounts – higher discounts to be offered to more structurally important firms, c) combined discounts – higher discounts to be offered to firms that were both larger and more structurally important. It is important to note that despite the different selection criteria, the discounts were always offered per license and the decision to accept or reject the offer was made at an aggregate/organizational level.

- A network (with pre-specified topology and density) of 1000 firms is generated based on Watts and Strogatz algorithm (1998)
- Network-level attributes such as frequency of proprietary and open source software upgrades, strength of interoperability costs etc. are assigned
- Firm-level attributes such as size of the firm, current choice of software, level of OSS technical expertise etc. are assigned
    - Firms are sorted based on size and the level of per-license discounts for which the firms might be eligible are determined
    - Firms are sorted based on location (given a chosen centrality measure) and the level of per-license discounts for which the firms might be eligible are determined
    - Firms are sorted based on an average rank (of size and location) and the level of per-license discounts for which the firms might be eligible are determined
- For each firm the following steps are repeated for 50 simulated time periods

  Step 1: If the firm is at the beginning of its planning horizon, proceed to step 2, otherwise proceed to step 4

  Step 2: The firm evaluates its decision function to decide whether to upgrade existing software or switch to the alternative software. The decision function takes into account economic factors (such as costs in case of upgrading/switching the software) and social factors (such as the decision of the firm's neighbors). If discounts are being offered by the vendor then those discounts are taken into account when the upgrade/switching costs are computed

  Step 3: If the firm switches to the alternative software, it obtains new costs (setup, training, license, support costs etc.); otherwise, it upgrades existing software

  Step 4: The vendor monitors total market share (# of adopters) and revenue and compares the revenue at time 't' with the maximum revenue achieved with the previous version of the software. If the revenue falls by more than a certain level, the vendor decides to offer discounts to all eligible firms (i.e. firms who will be at the beginning of their planning horizon) in the next time period (time t+1)

  Step 5: Proceed to next simulated time period

Figure 12. Simulation Process with a Reactive PS Vendor

The interesting implication of such discounts from a vendor's perspective is that i) with size based discounts there is a potential to offer deeper discounts to fewer firms with a certain amount of money, whereas, ii) with location-based discounts, for the same amount of money, smaller discounts can be offered to a much bigger number of firms. Practically, for instance, a vendor can decide what type of clients to target in the short term or the long term.

Therefore, to evaluate the effectiveness of discounts offered under different criteria during the simulation, the PS vendor monitored the market share (# of adopters) and profit (NPV based on revenue from licenses sold every year and support). If the profit in a given time period fell by more than a certain level compared to the maximum profit earned with the release of the previous version of the software, the PS vendor offered per-license discounts to all eligible firms in the next time period. The maximum profit with the release of the previous software was used as a benchmark. There is no empirical data which suggests what benchmark for profit is actually used by software vendors. It was assumed that with the release of a newer version the vendor would be looking to improve its position in the market and expect to earn more. Therefore, best performance with the previous version was deemed to be an appropriate comparison case. If the profit did not fall significantly, then the PS vendor did nothing to change its original pricing structure. Firms were only eligible for a discount if they were at the start of their planning horizon and not all firms were offered the same discount.

4.3.1 Experiments and Simulation Parameters

A 3x3x5 study was designed to address the research question by modifying three different variables: *revenue threshold*, *type of discount* and *size of discount*. Table 14 provides an overview of the variables and chosen parameter values for the experiments. Three levels of *revenue threshold* (low, medium, high) were used to represent the drop in

Table 14. Key variables and parameter values for investigating the effectiveness of network-aware pricing

Variable	**Parameter Values**
Type of discount	Per-license discounts offered were based on a) size: measured in terms of number of machines, b) location: determined by the centrality value, c) combined: determined by averaging the size and location-based rank. For each type of discount, the firms were sorted in order of highest to lowest value. Top 20% received the highest level of discount, next 60% received the low level of discount and the lowest 20% did not receive any discount. For example, 20% of the largest firms received the highest size-based discount.
Level of discount	Five levels of per-license discounts were offered: no discount, very low (5%, 10%), low (15%, 30%), high (25%, 50%) and very high (30%, 70%). Apart from the no discount scenario which was used as a comparison case, in all other levels of discounts, two kinds of discounts were offered so that not all firms received the same discount. For example if size-based discounts were being given, then in case of very low discount, 20% of the largest firms were offered 10% per-license discount, 60% of the largest firms after that were offered 5% per-license discount and the smallest 20% of the firms were offered zero discount.
Threshold for fall in revenue that will induce a reaction from the PS vendor	Three levels of threshold were used: low (3%), medium (5%) and high (10%). Low threshold meant that if the revenue of the PS vendor at time t fell by more than 3% compared to the benchmark, then at time t+1 the PS vendor would start offering discounts. The benchmark was chosen to be the highest revenue earned with the release of the last version of the software. Therefore, low threshold was meant to model a more revenue-sensitive vendor whereas high threshold was meant to model a less revenue-sensitive vendor.
Chosen centrality measures and other network parameters	In Chapter 3 it was determined that different centrality measures (betweenness, closeness, degree and eigenvector centrality) were effective in driving diffusion of OSS under different network conditions (initial proportion of OSS adopters, level of interoperability costs, density and topology of network etc.). Furthermore, in that model these centrality measures were used to assign OSS to the starting population. However, since in this study the objective was to determine whether pricing on the basis of strategic location is more effective than pricing on the basis of traditional demographic information, it did not make sense to re-run experiments with all the different network conditions that were explored and investigated in Chapters 2 & 3. Therefore, each one of the centrality measures was used to test whether network-aware pricing was better than a traditional pricing scheme under different levels of discount and revenue thresholds. Since each centrality measure was most effective under different network conditions, the network conditions were appropriately changed each time and have been reported along with the results in the subsequent sections (see section 4.3.1 for details).

revenue which would trigger a reaction (in the shape of offering discounts) from the PS vendor. Low threshold was indicative of a more revenue-sensitive vendor followed by medium and high threshold. Three *types of discounts* (size-based, location-based, combined) were used to represent the different criteria adopted by the PS vendor for offering discounts. Size-based discounts were determined on the basis of the size of the firm – larger firms (with high number of machines) received higher per-license discounts whereas smaller firms received smaller discounts.

The simulations were run for 50 time periods and 400 sample paths were generated for each parameter combination described in the previous paragraph. Number of OSS adopters and revenue were recorded throughout the course of the simulation.

These dependent variables were chosen because one (# of adopters) measures market penetration whereas the other (revenue) measures profitability and both represent practical but different objectives of a typical software vendor when deciding pricing schemes (Lehmann and Buxmann 2009). NPV of the cash flows over the course of the simulation was computed using an interest rate of 4%. Estimating interest rate for evaluating investments is a complex problem which requires an understanding of the industry, past trends, type of investment etc. In the absence of empirical data on the actual interest rate used by vendors to evaluate revenue streams in the software market, the risk-free interest rate typically associated with government treasury bills was used as a conservative estimate.

The computation of NPV facilitated a more accurate comparison of the profitability of the different price discounting schemes. Similarly, location-based discounts were determined by offering high discounts to the more strategically located firms in the network. Strategic location was determined by choosing an appropriate centrality measure (either betweenness, closeness, degree or eigenvector centrality). Results from Chapter 3 were used to determine the appropriate centrality measure for each experiment. Combined discounts were determined by offering high discounts to strategically located, large firms. The combined ranking of the firms was determined by computing the average rank based on size as well as location. Five *sizes of discounts* were used to represent varying levels of per license discounts offered by the PS vendor to retain or improve market share and increase net revenue. The five levels were: no discount (as a comparison or base case), very low discount, low discount, high discount and very high discount. Within each level of *size of discount*, regardless of the criteria for selecting the firms for discounts, a "step-wise" function was used to determine actual per-license discount for each firm. For example, if *very low* size-based discounts were being offered then, 20% of the largest firms were offered 10% per-license discount; the next 60% of smaller firms were offered a 5% per-license discount; and the smallest 20% of the firms were offered no discount. Similarly, if *very low* location-based discounts were being offered then, 20% of the most strategically located firms (based on the selected centrality measure) received a 10% per-license discount; 60% of the next most strategically located firms received a 5% per-license discount and 20% of the least strategically located firms received no discount.

As mentioned earlier, in addition to these three key variables, 6 other parameter values had to be chosen that described the market conditions: network topology, network density, interoperability costs, OSS support costs, frequency of PS upgrades and size of initial population of OSS adopters. One route would have been to setup all possible network conditions (324 that were used in Chapter 2) and use each one of the centrality measures in those conditions to offer location-based discounts and compare them with size-based discounts. However, this was an infeasible route for two reasons: First, results from Chapter 3 had shown that the most appropriate criteria for defining the structural importance of a firm (degree, closeness, betweenness or eigenvector centrality) changed with the network conditions. For example, betweenness centrality was effective in spreading diffusion only in random topology networks with high initial proportion of OSS adopters; closeness centrality was effective in small world networks with high interoperability costs and high initial proportion of OSS adopters, and so on. Therefore it did not make sense to ignore those findings. Second, the objective in this chapter was to simply demonstrate whether location-based discounts can be more effective than traditional size-based discounts or not. Since a formal analytical proof-of-concept cannot be developed, a demonstration was required to convey the point. Therefore, it was decided to choose experiments with the following criteria in mind

1. Experiments should be conducted that demonstrate the concept of location-based discounts for all four centrality measures that were studied and found to be important in chapter 3. This was important because a) it had been found in Chapter 3 that each one of the four measures was important under some conditions, b) it would allow us to show that the concept of location-based discounts is not limited to one centrality measure – so long as an appropriate criteria for measuring structural importance can be identified, location-based discounts can be applied.

2. Experiments should span the different network conditions that were depicted in Chapter 3. In other words, location-based discounts should not just be tested for random topology or small world networks, or low density or high density networks etc. Again, the motivation was to separate the concept of network-aware pricing from actual numbers used in the simulation model by demonstrating the concept under different conditions.
3. In the selected experiments, diffusion of OSS should not occur very fast i.e. the time for diffusion of OSS to occur should not be a few years. This was important because both Chapters 2 and 3 had shown that if the diffusion of OSS occurred very rapidly or if the network environment was very conducive to the diffusion of OSS, then location was less important to the process of diffusion. Therefore, in those conditions location-based discounts cannot be expected to be significantly better than size-based discounts.
4. In the selected experiments, diffusion of OSS based on random selection of the initial OSS population should be clearly slower than the rate of diffusion with centrality-based selection of the initial OSS population (from Part 2). Simple statistical difference between the rates of diffusion with random and centrality-based assignment could not have been used as a suitable criterion. This is because in many conditions, statistical difference did not reflect 'practical difference'. For instance, consider the following network condition: low density random topology network, low interoperability costs, high starting population of OSS and low variability in OSS support costs, diffusion of OSS: in this case diffusion of OSS occurred in 4 years with betweenness centrality-based selection of the starting

OSS population and 4.168 years with random selection of the starting OSS population. These two numbers were found to be statistically different. However, it can be argued that they may not be too different, practically, from a software vendor's perspective.

Consequently, four experimental conditions were setup, one for each one of the centrality measures that matched the above mentioned criteria:

For betweenness centrality-based discounts: high density random topology network, high variability in OSS support costs, low interoperability costs, low PS upgrade frequency and high starting OSS population. For closeness centrality:-based discounts low density small world network, high variability in OSS support costs, high interoperability costs, low PS upgrade frequency and high starting OSS population. For degree centrality-based discounts: low density small world network, low variability in OSS support costs, medium interoperability costs, low PS upgrade frequency and high starting OSS population. For eigenvector centrality-based discounts: low density small world network, high variability in OSS support costs, low interoperability costs, high frequency of PS upgrades, low starting OSS population

Table 15 provides a quick overview of the selected market conditions for each one of the centrality measures. Repeated measures ANOVA was used to determine the effectiveness of the three types of price discounting schemes. All reported results were significant at $p < 0.05$. The results being reported and discussed in the following section are those in which the dependent variables (# of OSS adopters and profit of the vendor) were measured at the end of the simulation i.e. time t=50.

Table 15. Centrality measures and corresponding network conditions to effectively investigate network-aware pricing

Centrality Measure	Selected network conditions to investigate the effect of location-based discounts
Betweenness Centrality	High density, random topology network; High variability in OSS support costs; Low interoperability costs per transaction (1); Low frequency of PS upgrades (an upgrade every 4 years); High initial proportion of OSS adopters at the start of the simulation (30%). Under these conditions, betweenness centrality based assignment of OSS to the starting population resulted in the fastest diffusion of OSS. The rate of diffusion (10 years) was clearly better than the rate of diffusion under random assignment (57 years).
Closeness Centrality	Low density, small world network; High variability in OSS support costs; High interoperability costs per transaction (5); Low frequency of PS upgrades (an upgrade every 4 years); High initial proportion of OSS adopters at the start of the simulation (30%). Under these conditions, closeness centrality-based assignment of OSS to the starting population resulted in the fastest diffusion of OSS. The rate of diffusion (60 years) was clearly better than the rate of diffusion under random assignment (87 years).
Degree Centrality	Low density, small world network; Low variability in OSS support costs; Medium interoperability costs per transaction (3); Low frequency of PS upgrades (an upgrade every 4 years); High initial proportion of OSS adopters at the start of the simulation (30%). Under these conditions, degree centrality-based assignment of OSS to the starting population resulted in the fastest diffusion of OSS. The rate of diffusion (32 years) was clearly better than the rate of diffusion under random assignment (56 years).
Eigenvector Centrality	Low density, small world network; High variability in OSS support costs; Low interoperability costs per transaction (1); High frequency of PS upgrades (an upgrade every 2 years); Low initial proportion of OSS adopters at the start of the simulation (10%). Under these conditions, eigenvector centrality-based assignment of OSS to the starting population resulted in the fastest diffusion of OSS. The rate of diffusion (11 years) was clearly better than the rate of diffusion under random assignment (74 years)

However, another set of analyses were also performed in which the dependent variables were measured at T=25 to understand the robustness of the effect of location-based discounts. That analysis did not reveal any new or surprising results compared to the ones that were found when the dependent variables were measured at T=50. Therefore, the following section does not discuss those results. Details of those results can be found in Appendix B.

4.3.2 Results and Analysis

In general the results demonstrated that all three types of discounts offered by the PS vendor (size-based, location-based and combined) were effective in reducing the rate of diffusion of OSS. In most cases, increasing the size of the per-license discount helped the vendor and a delayed response from the vendor to the falling profits (i.e. when revenue threshold was high), adversely affected the vendor's market share and profitability. These are trivial results but they are important because they lend some level of face validity to the model. The results also revealed that with higher discounts, location-based discounts can be better than size-based discounts in terms of both profitability as well as market share. On the other hand, low discounts do not provide sufficient incentive to enough number of strategically located firms to out-perform size-based or combined discounts. Detailed results based on all the different centrality measures used to offer location-based and combined discounts have been reported in Tables 16-21.

High level of location-based discounts offered on the basis of *closeness centrality*, outperformed size and combined discounts both in terms of number of adopters (Table 16) and NPV (Table 17). When lower level discounts were offered then location-based discounts were less or equally effective compared to the other two types of discounts. Offering higher discounts reduced the diffusion of OSS (lower number of adopters at the end of the simulation) and increased the NPV of the PS vendor (higher NPV at the end of the simulation).

Table 16. Number of adopters when closeness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	416.725	≈404 (S/L/C)	≈389 (S/L/C)	377.67(L) 380.25(C) 381.40(S)	359.76(L) 370.41(C) 373.79(S)
Medium (5%)	416.725	≈408 (S/L/C)	≈398 (S/L/C)	388.83(L) 391.90(C) 393.25(S)	371.10(L) 382.73(C) 386.67(S)
High (10%)	416.725	≈412 (S/L/C)	≈406 (S/L/C)	396.77(L) 401.19(C) 402.86(S)	378.92(L) 392.60(C) 396.92(S)

Table 17. NPV in $ when closeness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	1.92E+10	1.929E+10 (S/C) 1.9285E+10 (L)	1.93E+10 (S/L/C)	1.931E+10 (L) 1.927E+10 (C) 1.925E+10 (S)	1.940E+10 (L) 1.929E+10 (C) 1.924E+10 (S)
Medium (5%)	1.92E+10	1.928E+10 (S/C) 1.927E+10 (L)	1.928E+10 (S/L/C)	1.930E+10 (L) 1.926E+10 (C) 1.924E+10 (S)	1.939E+10 (L) 1.928E+10 (C) 1.923E+10 (S)
High (10%)	1.92E+10	1.927E+10 (S/C) 1.926E+10 (L)	1.927E+10 (S/L/C)	1.929E+10 (L) 1.925E+10 (C) 1.923E+10 (S)	1.939E+10 (L) 1.927E+10 (C) 1.922E+10 (S)

* Sample size is 400 in each cell

Also, a delayed response to changing market conditions from the PS vendor (i.e. higher revenue thresholds) resulted in lower revenue for the PS vendor. Very similar results were observed when *betweenness centrality* was used to offer size-based discounts (Tables 18 and 19). These results validate the concept that was proposed in Chapter 3, i.e. PS vendors can benefit from offering better pricing scheme to strategically located firms in a network. The results provide strong support also because compared to the model in Chapter 3, the structural information was used differently in this study and still found to be significant.

In Chapter 3 structural information contributed to the selection of the starting population of OSS adopters whereas in this study it was used to offer discounts to the firms. Furthermore, despite the fact that for both betweenness and closeness centrality, the other market conditions (network topology, network density, level of interoperability costs etc.) were different (refer to Table 15) and that both reflect differently on the concept of importance of a firm in a network, location based discounts still proved to be quite effective.

When location-based discounts were offered using eigenvector centrality, it was observed that size-based discounts were statistically more effective or equally effective in slowing down the diffusion of OSS (Table 20). However, in terms of NPV, location-based discounts appeared to be better (Table 21). Furthermore, higher discounts resulted in a loss of revenue. These results require some explanation. In Chapter 3 it was found that under adverse conditions for OSS, if at all diffusion occurred, it occurred with eigenvector centrality. Therefore, when discounts were offered under those conditions to prevent diffusion of OSS, the conditions were made more difficult for OSS diffusion to occur.

Table 18. Number of adopters when betweenness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	709.452	601.28 (S) 614.14 (C) 638.67 (L)	415.58 (S) 421.93 (C) 460.77 (L)	374.47 (L) 380/380.25 (S/C)	315.77 (L) 350.53 (C) 363.91 (S)
Medium (5%)	709.452	≈690 (S/L/C)	579.17 (S) 599.22 (C) 616.33 (L)	≈490 (S/L/C)	331.42 (L) 401.59 (C) 433.10 (S)
High (10%)	709.452	≈695 (S/L/C)	664.63 (L) 677.93 (C) 683.71 (S)	572.75 (L) 641.02 (C) 670.12 (S)	359.71 (L) 544.05 (C) 617.12 (S)

Table 19. NPV in $ when betweenness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	1.72E+10	18702266981 (S) 18620699013 (C) 18397616190 (L)	18990691230 (S) 18958447719 (C) 18876345263 (L)	19008931070 (L) 18939157447 (C) 18887554565 (S)	**19406481658** (L) 19067142478 (C) 18864968947 (S)
Medium (5%)	1.72E+10	18237309572 (S) 18150600109 (C) 17981150641 (L)	18775248835 (S) 18697020184 (C) 18571145501 (L)	18839008503 (L) 18786418754 (C) 18733396726 (S)	**19380739534** (L) 18997522093 (C) 18764853111 (S)
High (10%)	1.72E+10	17715984227 (S) 17656682257 (C) 17554185678 (L)	18125298874 (S) 18092358680 (C) 18041260182 (L)	18548816889 (L) 18309350543 (C) 18189822375 (S)	**19352939185** (L) 18759264730 (C) 18398309728 (S)

* Sample size is 400 in each cell

As a result, firms that were in any case not considering adoption of OSS, received discounts from the PS vendor which unnecessarily lowered the revenue of the PS vendor. Since size-based discounts given on number of machines meant higher dollar amount spent on discounts, those discounts were most effective in preventing OSS diffusion (Table 20) but least effective in improving the revenue of the PS vendor (Table 21).

When location-based discounts were offered using degree centrality, it was observed that size-based discounts were generally better in terms of slowing diffusion of OSS (Table 22) and improving the revenue of the PS vendor (Table 23). It was also observed that when very high discounts were offered, PS vendors revenue fell compared to when high discounts were offered. These are interesting results and warrant some explanation. In Chapter 3 it was observed that degree centrality based assignment of OSS to the starting population was only effective in low density small worlds with low/medium interoperability costs and high initial proportion of OSS.

Table 20. Number of adopters when eigenvector centrality was used for location-based discounts*

Revenue Threshold	**Size of Discount**				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	172.83	169 (S)* ≈170 (L/C)	≈165 (S/C) 167 (L)	162.66 (S) 163.25 (C) 163.91 (L)	≈160 (S/L/C)
Medium (5%)	172.83	170 (S) ≈171 (L/C)	167.59 (S) 168.15 (C) 169 (L)	≈166 (S/C) 167 (L)	≈165 (S/L/C)
High (10%)	172.83	170 (S) ≈171 (L/C)	168.49 (S) 169.05 (C) 169.98 (L)	167.66 (S) ≈168 (L/C)	≈166 (S/L/C)

* Sample size is 400 in each cell

Table 21. NPV in $ when eigenvector centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	**2.54E+10**	2.5354E+10 (S/L/C)	2.527E+10 (L) 2.526E+10 (S/C)	2.518E+10 (L) 2.515E+10 (C) 2.514E+10 (S)	2.508E+10 (L) 2.504E+10 (C) 2.502E+10 (S)
Medium (5%)	**2.54E+10**	2.5353E+10 (S/L/C)	2.5268E+10 (L) 2.5262E+10 (C) 2.5259E+10 (S)	2.517E+10 (L) 2.515E+10 (C) 2.514E+10 (S)	2.508E+10 (L) 2.504E+10 (C) 2.502E+10 (S)
High (10%)	**2.54E+10**	2.5353E+10 (S/L/C)	2.57E+10 (L) 2.526E+10 (S/C)	2.517E+10 (L) 2.515E+10 (C) 2.514E+10 (S)	2.508E+10 (L) 2.504E+10 (C) 2.502E+10 (S)

Table 22. Number of adopters when degree centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	598.695	544.08(S) 546.13(C) 553.20(L)	489.52(S) 494.15(C) 502.33(L)	464.07(S) 466.09(C) 473.43(L)	449.73(S)/450.32(C) 454.66(L)
Medium (5%)	598.695	560.93(S) 562.89(C) 568.61(L)	520.68(S) 522.73(C) 530.07(L)	498.43(S) 501.29(C) 507.67(L)	486.38(S) 488.50(C) 491.16(L)
High (10%)	598.695	580.28(S) 582.01(C) 585.12(L)	557.02(S) 559.96(C) 565.79(L)	544.27(S) 546.90(C) 552.39(L)	536.46(S) 538.04(C) 540.12(L)

* Sample size is 400 in each cell

Table 23. NPV in $ when degree centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	1.82E+10	1.8454E+10 (S) 1.8431E+10 (C) 1.8377E+10 (L)	1.8669E+10 (S) 1.8641E+10 (C) 1.8571E+10 (L)	**1.8676E+10** (S) 1.8656E+10 (C) 1.8601E+10 (L)	1.86E+10 (S/L/C)
Medium (5%)	1.82E+10	1.8376E+10 (S) 1.8354E+10 (C) 1.8306E+10 (L)	1.8552E+10 (S) 1.8529E+10 (C) 1.8456E+10 (L)	**1.8565E+10** (S) 1.8541E+10 (C) 1.8482E+10 (L)	1.85E+10 (S/C) 1.847E+10 (L)
High (10%)	1.82E+10	1.8301E+10 (S) 1.8279E+10 (C) 1.8239E+10 (L)	1.8438E+10 (S) 1.8407E+10 (C) 1.8341E+10 (L)	**1.8441E+10** (S) 1.8412E+10 (C) 1.8350E+10 (L)	1.8375E+10 (S) 1.8364E+10 (C) 1.8340E+10 (L)

* Sample size is 400 in each cell

All of these conditions strongly favored diffusion of OSS. Under such conditions, it made sense that a starting OSS population of firms with highest number of neighbors on average (high degree centrality) would further speed up diffusion of OSS. However, in the present study, the favorable conditions are not significantly offset by simply offering discounts to high degree centrality firms. Size-based discounts translate into greater discounts which are able to slow diffusion more than location-based discounts. This suggests that perhaps degree-centrality based discounts might not be effective at all because the market conditions under which degree centrality was found to be effective (in Chapter 3) dramatically favor diffusion of OSS. Hence, location-based (degree centrality-based) discounts do not provide sufficient incentive to firms to switch compared to size-based discounts.

4.4 Discussion

Overall the results have demonstrated that location-based discounts offered under appropriate market conditions can be effective both in terms of penetrating the market and achieving revenue targets. This concept was suggested on the basis of experiments run in Chapter 3. In that study, the starting population of OSS adopters was chosen on the basis of the structural information. In the present study, however, the same structural information was simply used to offer discounts to strategically located firms in the network. A comparison between those types of discounts and traditional size-based discounts revealed that if strategically located firms are offered a big enough discount, that can be more favorable for a vendor in terms of both profitability as well as market share.

The peculiar results in the cases of eigenvector and degree centrality-based discounts offer two additional insights: first, if other factors (such as interoperability costs, support costs etc.) favor the diffusion of the vendor's software then of course the vendor need not offer any additional discounts because that results in falling profits; second, if other factors are highly conducive to the diffusion of the competitor's software then targeting discounts (low or high) alone to strategically located firms may not be enough. Under such circumstances, the vendor would have to react with more than just a price change to compete. At this point present research merely suggests that under those circumstances offering per-license discounts to strategically located firms would not be sufficient. These results also suggest that having location information need not always mean that the vendor should incorporate it in its pricing strategy. Interestingly, 'combined' discounts strategy for selecting firms never outperformed purely location or size-based selection strategies. In the combined strategy, firms were given an average rank on the basis of their structural as well as size-based importance. On the basis of the average rank, the best firms were offered the largest discounts. The fact that combined discount strategy never outperformed the other strategies in our case could have been because of the way the combined rank was computed using simple averages. Different 'combined' strategies could be explored at a later stage which compute a weighted average by placing more weight on location information than size information (or vice versa) depending on other network conditions. However, such an investigation is beyond the scope of the present research.

Recall the discussion on the practical implications of the results in Chapter 3 (section 3.3.1.3). The results in the present study provide credence to some of the ideas discussed in that section. Different network conditions dictate which firms have the potential of dramatically changing the diffusion dynamics in a software market. Attempts by vendors to capture large firms may make business sense from an isolated case-by-case perspective. However, our research strongly suggests that sufficient inducements offered to strategically located (and not necessarily the largest) firms can significantly trigger the diffusion of the vendor's software both locally as well as globally across the network. Furthermore, since strategically located individual firms may not end up receiving deep discounts (as against large firms), targeting such firms results in better revenue figures as well.

4.5 Limitations

There are a few areas that may limit the generalizability of the findings of this research. First, it is assumed that the OSS vendor does not react to the changing market conditions. Given the research question, this is not a limiting assumption. The model was designed to study if location-based information can effectively inform pricing decisions of a vendor. To that end, the assumption does not limit the findings of the study. As an extension, the OSS vendor's reaction can be modeled along similar lines as well i.e. OSS vendor starts offering discounts as well on the basis of size and strategic location of the firms. However, there are two issues that would have to be considered: a) that would be an economic game which would have to be modeled differently with entirely different objectives for observing the various strategies and counter strategies of the vendors; b) typically OSS vendors do not offer discounts on license costs so a price-discounting scheme in such a case might have to focus on discounting the overall package of the OSS clients.

Second, it is assumed that different types of per-license discounts are offered by the PS vendor. Typically, discounts are offered on the deal and the volume of the discounts matters instead of the actual component on which the discounts get applied. Again, given the context of the study, the assumption that discounts are applied on license costs is not a limiting assumption. They could have been applied to a different cost component. The objective fundamentally was to evaluate the effectiveness of location-based discounts and compare their effectiveness against traditional types of discounts. Furthermore, literature suggests that discounting licenses is not an uncommon practice.

Third, it was assumed that the PS vendor does not modify its pricing structure if the market share or profit does not fall significantly enough to warrant a reaction. Although practically a vendor can proactively modify the price to chase out the competition, the modeling of such a strategy was not crucial to the investigation of our research question. Whether a vendor chooses to act proactively or reactively is inconsequential to the point that was being made in this study: location-based discounts under some conditions can be more effective than traditional size-based discounts.

CHAPTER 5: CONCLUDING REMARKS

5.1 Research Overview

The thesis discussed the issue of adoption and diffusion of open source software (OSS). A review of the literature had revealed that there was some understanding of the factors that affected the decision of firms to adopt open source software. Two additional findings emerged from the review of the literature: a) there was no comprehensive model that addressed the inter-relationships between different factors and their effect on adoption and diffusion of OSS. Although some factors had been identified in various studies but their collective effect on diffusion of OSS had not been studied; b) there was a specific call in prior research that asked for more research to investigate “strategic variables other than price” to “better understand the drivers of adoption” of OSS particularly in the context of Windows and Linux (Masanell and Ghemawat, 2006: p. 1083). As a result, this thesis was designed to systematically address the gaps in the literature through a series of three inter-related studies.

In the first study (Chapter 2), a comprehensive model was developed to identify critical factors other than price that could significantly affect the adoption and diffusion of OSS. Some of the factors modeled in the study were taken from prior research whereas a few new factors were introduced as well whose affect on the diffusion of OSS had not been investigated in prior research (Table 2 in Chapter 1).

An agent-based simulation model was designed that modeled a network of 1000 interconnected firms interacting with each other. The firms were assigned a desktop operating system (proprietary or open source) at the start of the simulation. During the course of the simulation each firm had to choose between upgrading its existing software and switching to the alternative software. The decision was made on the basis of a decision function that considered a series of economic and social factors. The economic factors included license, support, training, interoperability costs etc. and the social factors included the network of neighbors, software being used by neighbors etc. (see Section 2.3 for details). The results revealed that the starting population of the OSS adopters, followed by interoperability costs, network density, high variability in OSS support costs and network topology were the most critical variables (other than price) that affected the diffusion dynamics of OSS. Seven propositions were presented that highlighted the main and interaction effects of these critical variables.

The second study (Chapter 3) was designed to better understand the interaction effects between network topology, network density and interoperability costs. The objective was that if network structure is so critical to the diffusion of OSS, a more systematic and detailed analysis must be conducted to explain its effect on the diffusion of OSS. Therefore, an economics of social networks approach was adopted that emphasized the network structure and its effect on the economic decisions of the firms. Prior research on the analysis of social networks looked at a set of centrality measures that identified the importance of nodes in a network from different perspectives. In the absence of guidelines for using particular set of centrality measures for investigating diffusion type processes, four commonly used centrality measures were employed in the study. A starting population of OSS adopters was chosen on the basis of these measures to see which one of them would be more suited in identifying strategically located firms that could significantly affect the process of OSS diffusion. These measurements were taken at the level of individual as well as groups of firms to identify important individuals and groups of firms that could shape the process of diffusion. The results revealed that strategically located individual firms and groups of firms can significantly impact the process of diffusion. Furthermore, it was found that the criteria for identifying structural importance varied under different network topologies, network densities, size of the starting population of OSS adopters and interoperability costs. The results contributed to the literature on the analysis of social networks by identifying suitable centrality measures that could be used in the investigation of software diffusion type process. From a practical perspective the results also suggested that if strategically located firms in a network can significantly affect the process of diffusion, software vendors can exploit that information by offering a different pricing structure to the strategically located firms. The third study was designed to further explore and validate this point.

In the third study (Chapter 4), the concept of ‘network-aware’ pricing was introduced that simply stated that a better package should be offered by software vendors to the strategically located firms in a network. In this case, strategic location was contextualized on the basis of the centrality measures whose importance in the context of software diffusion had already been investigated in Chapter 3. The objective was to measure the effectiveness of such a pricing scheme against a traditional software pricing scheme that generally offer better packages to larger firms. Experiments were setup that allowed the PS vendor to offer per-license discounts to selected firms. The firms were selected on the basis of their size, or location, or both criteria (size and location). The effectiveness of these various approaches was measured by monitoring the market share and profit of the PS vendor. The experiments revealed that in several cases location-based discounts outperformed traditional size-based discounts in terms of both market share and profit. Table 24 provides an overview of the three studies.

5.2 Contributions

The following subsections review the contributions made through the thesis to both research and practice.

5.2.1 Research Contributions

There are several ways in which the thesis has contributed to the literature on OSS diffusion, software diffusion, analysis of social networks and software pricing. First, a framework has been developed that can be used to study the diffusion process of competing software. Although the model was contextualized for studying desktop operating system market only, there is nothing in the characteristics of the agents, their behavior or other simulation conditions that could prevent it from being used to investigate diffusion of other types of software. For example, in the server operating system (OS) market, the initial number of OSS adopters, setup costs, support costs and training costs will be higher than those in the desktop OS market, but the model will still be applicable. Similarly, in the open source ERP market, the strength of interoperability issues may be higher than those described in our model in the context of the desktop OS, however the propositions should still hold. The framework presented in this thesis models the simple behavior of the individual agents while capturing the inherent heterogeneity between them and within their interactions.

Table 24: Overview of the dissertation

	Essay One: Determinants of Diffusion Dynamics of Open Source Software	Essay Two: A Social Network Analysis of Diffusion of Open Source Software	Essay Three: The Impact of Network-Aware Pricing versus Traditional Software Pricing Schemes on Diffusion of Open Source Software
Research Questions	How do key variables other than price individually and collectively affect the diffusion dynamics of OSS?	What is the relative importance of various individual-level structural measures in explaining the rate of diffusion of OSS? What is the relative importance of group-level structural measures in explaining the rate of diffusion of OSS? Which of the structural measures are most effective in explaining the rate of diffusion of OSS?	Is network-aware software pricing more effective than a traditional software pricing scheme?
Theoretical Foundation	Diffusion/diffusion of Innovations/ Standards, Open Source Software, Agent-based Computational Economics	Social Network Analysis	Software Pricing
Motivation	Call in previous research to explore "strategic variables other than price" to "better understand the drivers of adoption" (Masanell and Ghemawat, 2006: p. 1083). Lack of a comprehensive model/framework that collectively investigated the effect of critical variables on diffusion of OSS	The results from the first essay showed that network structure is an important variable. We wanted to investigate it in more detail using literature on social networks. Given the growing emphasis on global collaboration in the business world and the interdependence between organizations while making technology decisions, in this paper, we propose an investigation of diffusion of OSS through social network analysis (SNA)	The results from the second essay showed that strategically located firms in a network can affect software diffusion. This information can be used in different ways by a software vendor. We decided to demonstrate its use by incorporating it in a simple price-discounting strategy to highlight the importance of location information.
Results Highlights	Variability in OSS support costs, network topology, network density and interoperability costs are some of the critical factors other than price that can significantly affect OSS diffusion	Strategic location of individual firms and groups of firms can significantly impact OSS diffusion. The criteria for identifying structural importance vary on the basis of the network conditions	Network-aware (or location-based) pricing can outperforms traditional size-based pricing in terms of market share and profit of the PS vendor in a software market
Practical Implications	Software adoption and diffusion are affected by a series of interrelated economic and social factors. These are critical factors other than the price of the software that can be strategically manipulated by the vendor to improve both market penetration and meet revenue targets. When pricing decisions have to be made, information regarding strategically located firms can inform those decisions. Furthermore, the agent-based model and simulations presented in the thesis can serve as the basis for vendors' market simulations to better understand their own markets.		

The use of the agent-based computational economics approach results in a simple yet powerful model that facilitates the investigation of macro-level behavior (diffusion dynamics) by accurately modeling the micro-level (firm-level) characteristics and behaviors.

Second, the thesis has demonstrated that the diffusion of software is dependent on strategic factors other than price, such as interoperability costs, variability in support costs, network topology and network density. What makes these findings more powerful and robust is the fact that they have been established while incorporating significant heterogeneity among firms and considering factors such as the threat of withdrawal of support by the PS vendor and the influence of centrality of neighbors on adoption decisions – interrelated factors that were never considered in prior research on software diffusion. The research has also added to the software diffusion literature by providing a framework that can model a heterogeneous set of economic and social factors to study the process of diffusion.

Third, the research has demonstrated that strategic location of individual or groups of firms can significantly impact the process of software diffusion. Prior research did not have any study that a) formally demonstrated the effect of location on the process of software diffusion, b) provide guidelines for appropriate use of structural (centrality) measures in the context of software diffusion type processes. Borgatti (2005) did propose a set of centrality measures to investigate certain types of diffusion processes. However, that taxonomy did not cover software diffusion type processes. Therefore, not only did this thesis formally demonstrate the effectiveness of individual as well as group centrality measures, it also highlighted the conditions under which four of the commonly used centrality measures can be applied in the analysis of software diffusion.

Fourth, as a proof-of-concept, the thesis also demonstrated that strategic location of firms can be used effectively by software vendors to influence the process of software diffusion. Such use of location information to inform pricing decisions had not been investigated in prior research.

5.2.2 Practical Contributions

There are several important practical implications of this research. Overall, the thesis has demonstrated that the process of software adoption and diffusion is significantly affected by a series of interrelated economic and social factors. Although the proposed framework was contextualized in the context of a software market involving open source and proprietary software, certain overarching arguments that can be applied in practice, need to be highlighted.

First, the agent-based simulation model presented in the thesis can be applied by a vendor, with adjustments to the specific parameters in the model, to run market simulations. The model is useful as it a) tries to capture the heterogeneity encountered in reality, and b) exhibits behavior which closely resembles reality (i.e. has face validity). The model can be used by the vendor as an effective tool to study the effect of different critical variables in the context of his/her own specific market. Given the behavior of the agents and the modeling of the interaction between the agents, the model can also be applied to the investigation of other types of software markets as well (and not just desktop OS software market).

Second, the research has demonstrated that there are several critical variables other than price that a software vendor can focus on to compete in a software market. For example, interoperability and support costs can significantly impact the process of diffusion. Notwithstanding the specific numbers used the model, which might be different for other types of software markets or might have been parameterized differently had another research investigated the same problem, the elaborate set of experiments in Chapter 2 provide sufficient basis for a software vendor to at least more carefully evaluate some of these cost components when deciding any market strategy.

Third, the research showed that the timing of software upgrades can significantly affect the process of software diffusion. The implications are that a PS vendor should consider evaluating its upgrade policy on the basis of the frequency and additional cost of upgrades – factors that can potentially impact the diffusion of its software.

Fourth the analysis of the social network of firms revealed that strategically located firms within a network can significantly impact the process of software diffusion. The implication is that vendors should be aware of the basic structural characteristics of the network of their clients. To that extent the research also revealed the criteria for identifying strategically located individual firms and groups of firms under different network conditions. For example, if the vendor's analysis reveals that the target network has the characteristics of a low density small world network, then i) if there are no significant interoperability issues with the competitor's software, firms with the most number of connections to other firms should be targeted, ii) interoperability issues can be strategically manipulated to change the level of influence of some firms over other firms – the results had demonstrated that structural importance of firms would change with higher interoperability costs. What is more interesting is that with the changing market share, the vendor needs to adapt and target a different set of firms. For example, early on when the vendor does not have a large installed base of adopters then important firms are ones that have highest eigenvector centrality (in small world networks). On the other hand, as the market share of the vendor improves, depending on the density of connections and strength of interoperability costs, eigenvector centrality should not be used to identify important firms. These strategies would be applicable in case of other types of software markets as well particularly when proprietary and open source software vendors are competing for the same network of clients.

Fifth, the research demonstrated the potential of using location information of firms to effectively compete in a software market. Although, a limited set of experiments were run in Chapter 4, the results did highlight the potential use of location information in any pricing strategy adopted by a software vendor. Given the rising trend toward usage-based licensing or on demand software delivery, vendors can further explore the possibility of becoming 'network-aware' while coming up with better, newer pricing schemes.

Sixth, the findings from Chapter 2 highlighted the fact that under some market conditions even a passive PS vendor – one who does not react to fluctuations in the market conditions by changing its pricing structure – might not be severely threatened by OSS. In fact, the simulation results revealed that the PS vendor is only threatened by OSS if interoperability costs are low and there is high variability in OSS support costs.

5.3 Future Research

There are some limitations of the thesis which have been discussed separately at the end of each study (in chapters 2, 3 and 4). This section presents some ideas for future research.

First, the model was contextualized for studying the desktop operating system market. Future research can investigate the diffusion dynamics in other markets as well such as application software, enterprise software etc. The actual numbers used in the model will change, however, the behavior of the model and findings are not expected to change significantly. Second, in highlighting strategically located groups, one definition of a group was used: a simple supply chain. In other words, each firm was the focal firm of its own group (or immediate set of neighbors). Future research could explore other definitions of group definitions as well. This may require the use of additional network topologies (like scale-free networks) and densities as well. Third, in testing the concept of network-aware pricing a simple price discounting scheme was used. Future research could explore the use of location information in coming up with newer ways of pricing software. For example, with the growing trend toward on demand or cloud computing that will look to track the actual use of the software, software providers could come up with newer pricing schemes that offer favorable prices to strategically located users of the software. Furthermore, additional investigation is required into the factors that can affect the performance of various network-aware pricing schemes.

REFERENCES

Abdel-Hamid T. K. “Investigating the Impacts of Managerial Turnover/Succession on Software Project Performance”, Journal of Management Information Systems, 9, 2 (Fall 1992), pp. 127-144

Afuah A. “How much do your “co-opetitors” capabilities matter in the face of technological change?”, Strategic Management Journal, 21 (3), Special Issue: Strategic Networks, (March 2000), pp. 387-404

Ahituv N., Igbaria M. and Sella A. “The Effects of Time Pressure and Completeness of Information on Decision Making”, Journal of Management Information Systems, 15, 2 (Fall 1998), pp. 153-172

Ahuja, G. “Collaboration networks, structural holes, and innovation: a longitudinal study”, Administrative Science Quarterly, 45, 3 (2000), pp. 425–457

Ahuja M. K. and Carley F. M. “Network Structure in Virtual Organizations”, Journal of Computer-Mediated Communication, 3, 4 (June 1998), pp. 741-757

Ashton, K. “Why Wal-Mart is Right”, The RFID Journal, (July 2004)

Barahona J. C. and Pentland A. “Advice Networks and Local Diffusion of Technological Innovations”, published in, Communities and Technologies 2007: Proceedings of the Third Communities and Technologies Conference, Pentland, Ackerman, and Contractor (Eds.), Michigan State University, (2007), London: Springer, pp. 509-529

Bell G. G. “Clusters ,Networks and Firm Innovativeness”, Strategic Management Journal, 26,(2005), pp. 287-295

Blau, J. “Munich begins to switch Windows out for Linux”, CIO Magazine, (September 21, 2006).

Bonaccorsi, A., S. Giannangeli, and Rossi C. “Entry Strategies Under Competing Standards: Hybrid Business Models in the Open Source Software Industry.” Management Science, 52, 7 (July 2006), pp. 1085-1098

Bonaccorsi, A. and C. Rossi "Why Open Source Software Can Succeed." Research Policy, 32, 7 (July 2003), pp. 1243-1258.

Bonacich P. “Factoring and weighting approaches to status scores and clique identification”, Journal of Mathematical Sociology, 2, (1972), pp. 113-120

Bontis N. and Chung H. "The evolution of software pricing: from box licenses to application service provider models", Internet Research: Electronic Networking Applications and Policy, 10, 3, (2000), pp. 246-255

Borgatti, S. P. "Centrality and network flow", Social Networks, 27, 1 (2005), pp. 55-71

Borgatti, S.P., Everett, M.G. and Freeman, L.C. 2002. Ucinet for Windows: Software for Social Network Analysis. Harvard, MA: Analytic Technologies.

Bowman, J. "The end of Windows 98", In Depth Technology: CBS News, (July 2006), http://www.cbc.ca/news/background/tech/windows98.html

Brynjolfsson E. and Kemerer C. F. "Network Externalities in Microcomputer Software: An Econometric Analysis of the Spreadsheet Market", Management Science, 42, 12 (December 1996), pp. 1627-1647

Capaldo A. "Network Structure and Innovation: The Leveraging of a Dual Network as a Distinctive Relational Capability", Strategic Management Journal, 28, (2007) pp.585-608

Chakravarty S., Dogan K. and Tomlinson N. "A hedonic study of network effects in the market for word processing software", Decision Support Systems, 41, (2006), pp.747-763

Chakravarty S., Haruvy E. and Wu F. "The link between incentives and product performance in open source software development: An empirical investigation", Global Business and Economics Review, 9, 203 (July 2007), pp. 151-169

Chatterjee R. and Eliashberg J. "The Innovation Diffusion Process in a Heterogeneous Population: A Micromodeling Approach", Management Science, 36, 9 (September 1990), pp. 1057-1079

Chen P. Y. and Forman C. "Can Vendors Influence Switching Costs and Compatibility in an Environment with Open Standards?", MIS Quarterly, 30, Special Issue (August 2006), pp. 541-562

Choudhary V., Ghose A., Mukhopadhyay T. and Rajan U. "Personalized Pricing and Quality Differentiation", Management Science, 51, 7 (July 2005), pp. 1120-1130

Dalle JM, Jullien N "Open-Source vs. Proprietary Software" http://opensource.mit.edu/papers/dalle2.pdf, 2001

Delre S. A., Jager W. and Janssen M. A. "Diffusion dynamics in small-world networks with heterogeneous consumers", Computational Math and Organization Theory, 13, 2 (June 2007), pp. 85-202

Deprez J. C., Monfils F. F., Ciolkowski M. and Soto M. “Defining Software Evolvability from a Free/Open Source Software Perspective”, Third IEEE Workshop on Software Evolvability, 1, 1 (October 2007), pp. 29-35

Dick S. J. “Reconsidering Adoption Behavior: Models Beyond Diffusion”, Atlantic Journal of Communication, 12, 4, (2004), pp. 233-248

Doz, Y. L. “The evolution of cooperation in strategic alliances: initial conditions or learning processes? Strategic”, Management Journal, 17, Special Issue, (1996) pp. 55–83

Dutta A. and Roy R. “Offshore Outsourcing: A Dynamic Causal Model of Counteracting Forces”, Journal of Management Information Systems, 22, 2 (Fall 2005), pp. 15-35

Economides N. and Katsamakas E. “Two-sided competition of proprietary vs. open source technology platforms and implications for the software industry”, Management Science, 52, 7 (July 2006), pp. 1057-1071

Everett M.G. and Borgatti S.P. (1999), “The Centrality of Groups and Classes”, Journal of Mathematical Sociology, 23 (3), pp. 181-201

Fichman R. G. “The Diffusion and Assimilation of Information Technology Innovations”, In Zmud R. W. (ed), Framing the Domains of IT Management, Projecting the Future … Through the Past, Pinnaflex Education Resources, Inc., Cincinnati, Ohio, 2000

Foley J. "Power Shift: Users gain flexibility and influence with new software pricing models", Information Week, April 19, 2004, http://www.informationweek.com/story/showArticle.jhtml?articleID=18901868

Freeman L. C. “Centrality in networks: A conceptual clarification”, Social Networks, 1, 3, (1979), pp. 215-239

Gallaugher J. M. and Wang Y. M. "Understanding Network Effects in Software Markets: Evidence from Web Server Pricing", MIS Quarterly, 26, 4 (December 2002), pp. 303-327

Gandal N. “Hedonic price indexes for spreadsheets and an empirical test for network externalities”, The RAND Journal of Economics, 25, 1 (Spring 1994), pp. 160-170.

Girvan M. and Newman M. E. J. “Community structure in social and biological networks”, PNAS, 99, 12 (June 2002), pp. 7821-7826

Gray J. R. An Executive Decision: Windows OS vs. Linux OS vs. Hybrid, Graduate Research Report, Bowie State University, Maryland in Europe, April 2005

Gulati R., Nohria N. and Zaheer A “Strategic Networks”, Strategic Management Journal, 21, (2000), pp. 203-215

Guth R. “Linux starts to find home on Desktops”, The Wall Street Journal Online, (March 13, 2007), B3

Guo H. and Chang H. K. “Computer Virus Propagation in Social Networks”, Twenty Eighth International Conference on Information Systems, Montreal, Canada, 2007

Harkola J. and Greve A. “Diffusion of Technology: Cohesion or Structural Equivalence?”, The Academy of Management Best Paper Proceedings, Academy of Management Meeting, Vancouver, (August 1995), pp. 422-426

Haythornthwaite, C. “Social Network Analysis: An Approach and Technique for the Study of Information Exchange”, Library and Information Science Research (LISR), 18, 4 (Autumn 1996), pp. 323-342

Hissam S. A., Plakosh D. and Weinstock C. “Trust and Vulnerability in Open Source Software”, IEE Proceedings, Software, 149, 1 (February 2002), pp. 47-51

Hite J. M. and Hesterley W. S. “The Evolution of Firm Networks: From Emergence to Early Growth of the Firm”, Strategic Management Journal, 22, (2000), pp. 275-286

Huang J. S. and Kapur M. “Diffusion of Pedagogical Innovations as a Complex Adaptive Process – Agent-based Modeling as Research Method”, In, Hirashima T., Ulrich H. and Young S. S. C. (eds.), Supporting Learning Flow through Integrative Technologies, IOS Press, 2007, pp. 67-74

Hykosova M. “Lecture notes for the course Stochastic Models at FD ÈVUT”, Prague, 2003.

Jiang B., Chen P. and Mukhopadhyay T. "Software pricing: Pay-per-use versus perpetual licensing", Tepper School of Business, Working Paper, July 2007, Carnegie Mellon University

Jones J. L., Easley R. F. and Koehler G. J. “Market Segmentation Within Consolidated E-Markets: A Generalized Combinatorial Auction Approach”, Journal of Management Information Systems, 23, 1 (Summer 2006), pp. 161-182

Kamhorst E. M. “Open vs. closed. Some consequences of the open source movement for software markets”, CPB Discussion Paper No. 13, CPB Netherlands Bureau for Economic Policy Analysis, Netherlands, September 2002

Katsamakas E. and Xin M. “An Economic Analysis of Enterprise Adoption of Open Source Software” (October 2005). NET Institute Working Paper No. 05-29

Keizer G. “Microsoft COO: Big client OS not dead yet”, Computerworld UK, (July 14, 2007)

Kim B. C., Chen P. and Mukhopadhyay T. “Pricing Open Source Software”, in the Twenty-Seventh International Conference on Information Systems, Milwaukee, Canada, (2006), pp. 341-366

Kogut B. “The Network as Knowledge: Generative Rules and the Emergence of Structure”, Strategic Management Journal, 21, 2000, 405-425

Kwon D., Oh W. and Jeon S. “Broken Ties: The Impact of Organizational Restructuring on the Stability of Information-Processing Networks”, Journal of Management Information Systems, 24, 1 (Summer 2007), pp. 201-231

Lazer D. and Friedman A. “The Network Structure of Exploration and Exploitation”, MIS Quarterly, 52, (2007), pp. 667-694

Leading Edge Forum: CSC “Open Source: Open for Business”, Leading Edge Forum Report, (September 2004), http://www.csc.com/features/2004/247.shtml

Lerner, J. and Tirole, J. “The open source movement: Key research questions”, European Economic Review, 45, 4 (May 2001), pp. 819-826

Levinthal, D. A. and Fichman, M. “Dynamics of inter-organizational attachments: auditor–client relationships”, Administrative Science Quarterly, 33, 3,(1988),pp.345–369.

Liberatore, M., Hatchuel, A., Weil, B., and Stylianou, A. “An organizational change perspective on modeling”. European Journal of Operational Research, 125, 1 (August 2000), pp. 184-194.

Lin L. “Impact of User Skills and Network Effects on the Competition between Open Source and Proprietary Software”, Electronic Commerce Research and Applications, 7, 1 (April 2008), pp. 68-81

Madey G., Feeh V. and Tynan R. “The Open Source Software Development Phenomenon: An Analysis Based on Social Network Theory”, Eighth Americas Conference on Information Systems, (2002), pp. 1806-1813

Manzoni J. F. and Angehrn A. A. “Understanding Organizational Dynamics of IT-Enabled Change: A Multimedia Simulation Approach”, Journal of Management Information Systems, 14 (3), Winter 1997-98, pp. 109-140

Markus, M. L. (1990). Toward a “critical mass” theory of interactive media. In J. Fulk & C. Steinfield (Eds.), Organizations and communication technology, Newbury Park, CA: Sage, pp. 194–218

Masanell R. C. and Ghemawat P. “Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows”, Management Science, 52, 7 (July 2006), pp. 1072-1084

McAllister, N. “Open Enterprise: Desktop Linux Remains Hard Sell”, Computer World: The Voice of Management (March 21, 2006), http://www.computerworld.com.au/index.php/id;608238263

McMillan R. “The business case for Linux”, InfoWorld.com, (July 30, 2004)

Michlmayr M. “Quality improvement in volunteer free software projects: Exploring the impact of release management”, In Scotto M. and Succi G. (eds.), Proceedings of the First International Conference on Open Source Systems, Genova, (July 2005), pp. 309-310

Miller J. H. and Page S. E. Complex Adaptive Systems: An Introduction to Computational Models of Social Life, Princeton University Press, 2007

Mockus A., Fielding R. T. and Herbsleb J. “A Case Study of Open Source Software Development: The Apache Server”, Proceedings of the 22nd International Conference on Software Engineering, 2000, pp. 263-272

Moody G. “Munich Makes Good”, ComputerWorldUK, June 03, 2008.

Mukherji N., Rajagopalan B. and Tanniru M. “A decision support model for optimal timing of investments in information technology upgrades”, Decision Support Systems, 42,3 (December 2006), pp. 1684-1696.

Mustonen M. "Copyleft - the economics of Linux and other open source software", Information Economics and Policy, 15, 1 (March 2003), pp. 99-121

Ngwenyama O., Guergachi A. and McLaren T. “Using the learning curve to maximize IT productivity: A decision analysis model for timing software upgrades”, International Journal of Production Economics, 105, 2 (February 2007), pp. 524-535.

Norris J. S. “Mission-Critical Development with Open Source Software: Lessons Learned”, IEEE Software, 21, 1 (January/February 2004), pp. 42-49

Pandit V., Modani N., Mukherjee S., Nanavati A. A., Roy S. and Agarwal A. “Extracting dense communities from telecom call graphs”, 3rd International Conference on Communication Systems Software and Middleware Workshops, Bangalore, India, January 2008, pp. 82-89

Perens B. “The Open Source Definition”, In, DiBona C., Ockman S. and Mark S. (eds.), Open Source Voices from the Open Source Revolution, O’Reilly & Associates Inc., Sebastopol, CA , 1999, pp. 171-188.

Postmus D., Wijngaard J. and Wortmann H. "An economic model to compare the profitability of pay-per-use and fixed-fee licensing", Information and Software Technology, 51, 3 (March 2009), pp. 581-588

Raghunathan S., Prasad A., Mishra B. K. and Chang H. "Open Source Versus Closed Source: Software Quality in Monopoly and Competitive Markets", IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, 35, 6 (November 2005), pp. 903-918.

Raymond, E. S. "The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary", O'Reilly & Associates Inc., Sebastopol, CA, 1999.

Rogers E. M. Diffusion of Innovations, Fourth Edition, The Free Press, New York, 1995.

Ruffin M. and Ebert C. "Using Open Source Software in Product Development: A Primer", IEEE Software, 21, 1 (January/February 2004), pp. 82-86.

Scacchi W. "Understanding the requirements for developing open source software", IEE Proceedings, Software, 149, 1 (February 2002), pp. 24-39

Sayer P. "Paris accelerates move to open source", ComputerWorld, 21/11/2005

Shankland S. "Munich deal boosts desktop Linux", ZDNet UK, CNET News.com, (May 29, 2003)

Shiels, M. "Calls for open source government", BBC News, Silicon Valley, January 21, 2009, accessed on January 21, 2009, http://news.bbc.co.uk/2/hi/technology/7841486.stm

Simonetto M. and Davidson A. "Pricing strategy and execution: an overlooked way to increase revenues and profits", Strategy and Leadership, 33, 6, 2005, pp. 25-33

Singh P. V. "Open Source Software Development and the Small World Phenomenon: An Empirical Investigation of Macro Level Collaboration Network Properties on Project Success", in the Twenty Eighth International Conference on Information Systems, Montreal, Canada, 2007, pp. 1-17.

Singh, K. and Mitchell, W. "Precarious collaboration: business survival after partners shut down or form new partnerships", Strategic Management Journal, 17, 2 (1996), pp. 99–116.

Sowe S. K., Stamelos I. and Angelis L. "Understanding knowledge sharing activities in free/open source software projects: An empirical study", The Journal of Systems and Software, 81, 3 (March 2008), pp. 431-446

Srbljinović, A., & Skunca, O. “An introduction to agent based modeling and simulation of social processes” Interdisciplinary Description of Complex Systems, 1,1-2 (2003), pp. 1-8.

Sundararajan A. "Nonlinear Pricing of Information Goods", Management Science, 50, 12 (December 2004), pp. 1660-1673

Sundararajan A. “Local Network Effects and Network Structure”, Working Paper CeDER-05-02, Center for Digital Economy Research, Stern School of Business, New York University, May 2005.

Swaminathan J. M., Smith S. F. and Sadeh N. M. “Modeling Supply Chain Dynamics: A Multi-agent Approach”, Decision Sciences, 29,3 (Summer 1998), pp. 607-632

Tawileh A. and Rana O. “Free and Open Source Quality Assurance”, Information and Communication Technologies, 2, (April 2006), pp. 2866-2871

Tesfatsion L. and Judd K. L. Handbook of Computational Economics: Agent-based Computational Economics Volume 2, North-Holland, Oxford, UK, 2006

Tirole J. and Lerner J. “The Scope of Open Source Licensing”, Journal of Law, Economics and Organization, 21, 1 (April 2005), pp. 20-56.

Tsai, W. P. “Social capital, strategic relatedness and the formation of intraorganizational linkages”, Strategic Management Journal, 21, 9 (2000) pp. 925–939.

Van de Ven, A. H. and Walker, G. “The dynamics of inter-organizational coordination”, Administrative Science Quarterly 29 (4) (1984) pp. 598–622.

Valente T. W. “Network Models and Methods for Studying the Diffusion of Innovations”, pp. 98-116, in Models and Methods in Social Network Analysis, Carrington P. J., Scott J. and Wassermann S. (Eds.), Cambridge University Press, New York, 2005

Vaughan-Nicholas, S. J. “Upgrade price wars: Vista vs. Linux”, DesktopLinux.com, (September 06, 2006).

Wagstrom P., Herbsleb J. and Carley K. “A Social Network Approach to Free/Open Source Software Simulation”, Proceedings of the First International Conference on Open Source Systems, 11-15th July 2005, Genova, Marco Scotto and Giancarlo Succi (Eds.), pp. 16-23

Waldrop, M. M. Complexity: The Emerging Science at the Edge of Order and Chaos, Touchstone, New York, NY, 1992

Wassermann S., and Faust, K. Social Network Analysis, Cambridge University Press, Cambridge, UK, 1994.

Watts DJ and Strogatz SH "Collective dynamics of "small-world" networks", Nature, 393, (1998), pp. 440–442

Weitzel, T., Beimborn, D., and Konig, W., "A Unified Economic Model of Standard Diffusion: The Impact of Standardization, Cost, Network Effects, and Network Topology", MIS Quarterly, 30, Special Issue (August 2006), pp. 489-514.

Westarp F. V. "Modeling Software Markets: Empirical Analysis, Network Simulations, and Marketing Implications", Physica-Verlag, Heidelberg New York, 2003.

Westarp F. V. and Wendt O. "Diffusion Follows Structure - A Network Model of the Software Market", in the Proceedings of the 33rd Hawaii International Conference on System Sciences, (January 2000), 10 pp.

Wheeler, D. A. "Why Open Source Software/ Free Software (OSS/FS, FLOSS or FOSS)? Look at the Numbers!", 2005, http://www.dwheeler.com/oss_fs_why.html

Wilensky, U. "NetLogo Small Worlds model", Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, http://ccl.northwestern.edu/netlogo/models/SmallWorlds

Wilkins, D., Cole, D. and Nelson, M. "Windows Interoperability with Linux in the Enterprise (WINWILE): A Solution to the High Cost of Licensing, Downtime, and Security Problems", Consortium for Computing Sciences in Colleges: Southeastern Conference, 2004, pp. 260-266

Williams, D. H. "The Strategic Implications of Wal-Mart's RFID Mandate", Directions Mag, July 29, 2004

Xu B. "Volunteers' participative behaviors in open source software development: The role of Extrinsic incentive, intrinsic motivation and relational social capital", PhD Dissertation, Texas Tech University, May 2006.

Yu L. "Self-organization process in open-source software: An empirical study", Information and Software Technology, 50, 5 (April 2008), pp. 361-374

Zaffar M. A., Kumar R. L. and Zhao, K. "Diffusion Dynamics of Open Source Software: An Agent-Based Computational Economics Approach", International Conference on Information Systems, Paris, France, December 20008.

Zaheer A. and Bell G. G. “Benefiting from Network Position: Firm Capabilities, Structural Holes, and Performance”, Strategic Management Journal, 26, 2005, pp. 809-825

Zhao L. and Elbaum S. “Quality assurance under the open source development model”, The Journal of Systems and Software, 66, 1 (April 2003), pp. 65-75

Zhu, K. Kraemer, K. L. Gurbaxani, V. and Xu S. X. “Migration to Open-Standard Interorganizational Systems: Network effects, Switching costs and Path dependency”, MIS Quarterly, 30, Special Issue (2006), pp. 515-539

APPENDIX A: DETAILED RESULTS FOR CHAPTER 3

Table 25. Time for diffusion of OSS to occur in small world networks with different OSS assignment criteria

	Low Initial Proportion of OSS											
Network Density	**Frequency of Proprietary Software Upgrades = 2 years**						**Frequency of Proprietary Software Upgrades = 4 years**					
	Low Variability in OSS Support Costs			**High Variability in OSS Support Costs**			**Low Variability in OSS Support Costs**			**High Variability in OSS Support Costs**		
	Low Interop	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**
Low	18.2564* 97.6131 100.3948 ≈100	90.0056	100.8688	11.1816 42.6528 52.2348 ≈74	85.6544	100.4244	26.6212 98.7124 100.6504 ≈100	93.8936	100.9264	16.4052 54.0776 63.4288 ≈82	89.998	100.6496
Medium	95.5824	No diffusion		87.598	No diffusion		96.593	No diffusion		90.076	No diffusion	
High	No diffusion			No diffusion			No diffusion			No diffusion		
	High Initial Proportion of OSS											
Low	5.0004 5.04 5.19480 5.8396 9.90159	≈21 33.1524 39.76039 45.778	64.404 72.26159 77.7136 94.2288 98.35719	4.0036 4.0108 4.2988 4.4204 7.5	9.9996 11.7356 12.5456 15.16 31.6572	≈41 70.3052 72.9108 81.3996	7.0076 7.1768 7.4548 8.5592 14.7196	≈32 44.822 55.9692 63.7012	≈82 91.9976 99.189 100.7972	6.0052 6.0152 6.34 6.45439 11.0812	15.0556 17.6096 18.7384 22.58 41.8668	59.9128 60.9884 79.25239 87.2724 94.2648
Medium	11.5568 13.6908 14.3824 15.9732 38.4192	72.56559 97.2704/9 7.27519	100.3628	7.0756 7.1072 7.2564 8.4244 8.5688	70.3664 90.5752/9 1.2356	100.512	17.1928 20.3140 21.3704 23.7112 48.289	90.24 99.8096/9 9.83839	100.9748	10.858 10.9508 11.1528 12.704 12.8872	89.3036 97.5868 98.08039	100.7604
High	13.6764 ≈15 16.2528 97.3188	100.8304	No diffusion	10.852 10.954 ≈11 29.992	100.97	No diffusion	20.298 ≈22 23.812 98.50519	No diffusion	No diffusion	≈16 ≈17 39.7519	No diffusion	No diffusion

*This table should be interpreted in conjunction with Table 9

Table 26. Time for diffusion of OSS to occur in random topology networks with different OSS assignment criteria*

	Low Initial Proportion of OSS											
Network Density	**Frequency of Proprietary Software Upgrades = 2 years**						**Frequency of Proprietary Software Upgrades = 4 years**					
	Low Variability in OSS Support Costs			**High Variability in OSS Support Costs**			**Low Variability in OSS Support Costs**			**High Variability in OSS Support Costs**		
	Low Interop	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**	**Low Interop**	**Medium Interop**	**High Interop**
Low	No diffusion			No diffusion			No diffusion			No diffusion		
Medium												
High												
	High Initial Proportion of OSS											
Low	4 4.0080 4.14119 4.1684 5.7588	11.734 13.7944 ≈21	No diffusion	3.7844 3.9284 3.9868 3.9924 4.32159	6.7596 7.0568 7.8188 7.86079 27.9772	36.8144 55.402 ≈89	6.00039 6.0128 6.2128 6.2492 8.3459	17.7719 20.6596 ≈31	No diffusion	5.8248 5.9460 5.9904 5.9952 6.348	10.0632 10.7964 ≈12 38.0412	52.8584 71.2788 ≈95
Medium	9.0256 ≈9.54 10.2152 87.7328	No diffusion		5.0144 ≈5.05 5.1172 8.7144	No diffusion		13.4896 ≈14.51 15.7124 94.45479	No diffusion		7.059 ≈7.16 7.30239 13.12	No diffusion	
High	38.5716 ≈51.45 68.9248			7.074 ≈7.25 7.5379 46.12			57.27279 ≈70 84.44279			10.8176 ≈11.12 11.5024 57.0672		

* This table should be interpreted in conjunction with Table 10

Table 27. Time for diffusion of OSS to occur in small world networks with different group OSS assignment criteria*

<table>
<tr><td></td><td colspan="6">Low Initial Proportion of OSS</td></tr>
<tr><td rowspan="3">Network Density</td><td colspan="6">Frequency of Proprietary Software Upgrades = 4 years</td></tr>
<tr><td colspan="3">Low Variability in OSS Support Costs</td><td colspan="3">High Variability in OSS Support Costs</td></tr>
<tr><td>Low Interop</td><td>Medium Interop</td><td>High Interop</td><td>Low Interop</td><td>Medium Interop</td><td>High Interop</td></tr>
<tr><td>Low</td><td>38.0108
39.3520
55.9228
66.3484
100.862</td><td colspan="2" rowspan="3">No diffusion</td><td>46.2/46.4
60.1132
75.5856
81.5419</td><td colspan="2" rowspan="3">No diffusion</td></tr>
<tr><td>Medium</td><td>53.4200
55.3116
67.0960
81.3572</td><td>52.3924
54.4776
62.8508
76.4516</td></tr>
<tr><td>High</td><td>No diffusion</td><td>100.64
≈100.9</td></tr>
<tr><td></td><td colspan="6">High Initial Proportion of OSS</td></tr>
<tr><td>Low</td><td>8.5592
25.0004
30.0980
31.6824
33.1472</td><td>55.9692
94.1688
≈97</td><td>99.189
100.8576</td><td>6.45439
15.1976
20.8060
22.7312
24.5244</td><td>22.58
94.6240
95.4052
≈98</td><td>87.2724
100.9724</td></tr>
<tr><td>Medium</td><td>≈33.9
35.4616
48.289</td><td colspan="2" rowspan="2">No diffusion</td><td>12.704
33.8312
≈34
36.1708</td><td colspan="2" rowspan="2">No diffusion</td></tr>
<tr><td>High</td><td>49.706
50.38
50.962
51.510
98.50519</td><td>39.7519
≈56
≈58</td></tr>
</table>

* This table should be interpreted in conjunction with Table 11

Table 28. Time for diffusion of OSS to occur in small world networks with different group OSS assignment criteria*

<table>
<tr><td></td><td colspan="6">Low Initial Proportion of OSS</td></tr>
<tr><td rowspan="3">Network Density</td><td colspan="6">Frequency of Proprietary Software Upgrades = 4 years</td></tr>
<tr><td colspan="3">Low Variability in OSS Support Costs</td><td colspan="3">High Variability in OSS Support Costs</td></tr>
<tr><td>Low Interop</td><td>Medium Interop</td><td>High Interop</td><td>Low Interop</td><td>Medium Interop</td><td>High Interop</td></tr>
<tr><td>Low</td><td colspan="3" rowspan="3">No diffusion</td><td colspan="3" rowspan="3">No diffusion</td></tr>
<tr><td>Medium</td></tr>
<tr><td>High</td></tr>
<tr><td></td><td colspan="6">High Initial Proportion of OSS</td></tr>
<tr><td>Low</td><td>7
≈7.02
8.3459</td><td>100.3472
≈100.7</td><td>No diffusion</td><td>≈6
6.348</td><td>17.9296
18.4028
18.8652
19.0548
38.0412</td><td>No diffusion</td></tr>
<tr><td>Medium</td><td>45.474
46.796
47.98
75.512
94.45479</td><td colspan="2" rowspan="2">No diffusion</td><td>10.0176
10.1220
10.1776
10.9372
13.12</td><td colspan="2" rowspan="2">No diffusion</td></tr>
<tr><td>High</td><td>No diffusion</td><td>17.094
17.286
17.575
30.027
57.0672</td></tr>
</table>

* This table should be interpreted in conjunction with Table 12

APPENDIX B: RESULTS FOR CHAPTER 4 WITH DEPENDENT VARIIABLES MEASURED AT T=25

Table 29. Number of adopters when closeness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	349.31	≈342 (S/C) 343 (L)	≈335 (S/L/C)	328.04(L) 329.81(C) 330.49(S)	317.31(L) 323.56(C) 325.99(S)
Medium (5%)	349.31	≈344 (S/C) 345 (L)	≈338(S/C) 340 (L)	332.72(L) 334.70(C) 335.55(S)	322.39(L) 328.94(C) 331.13(S)
High (10%)	349.31	≈345 (S/C) 346 (L)	≈341 (S/C) 342 (L)	335.42(L) 337.61(C) 338.63(S)	325.11(L) 332.27(C) 334.61(S)

Table 30. NPV in $ when closeness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	8.83E+09	8.82094E+09 (S/C) 8.818E+09 (L)	8.8788E+09 (L) 8.78+09(S/C)	8.7619E+09 (L) 8.7352E+09 (C) 8.7238E+09 (S)	8.7600E+09 (L) 8.7062E+09 (C) 8.6823E+09 (S)
Medium (5%)	8.83E+09	8.82099E+09 (S/C) 8.8176E+09 (L)	8.879E+09 (L) 8.786+09(S/C)	8.7661E+09 (L) 8.7410E+09 (C) 8.7295E+09 (S)	8.7663E+09 (L) 8.7136E+09 (C) 8.6906E+09 (S)
High (10%)	8.83E+09	8.82088E+09 (S/C) 8.81786E+09 (L)	8.79E+09 (L) 8.787+09(S/C)	8.7672E+09 (L) 8.7420E+09 (C) 8.7312E+09 (S)	8.7681E+09 (L) 8.7158E+09 (C) 8.6933E+09 (S)

* The sample size is 400 in each cell

Table 31. Number of adopters when degree centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	423.937	398.66 (S) 399.85 (C) 403.44 (L)	372.98 (S) 374.88 (C) 379.55 (L)	361.38 (S) 362.71 (C) 367.34 (L)	355.69 (L) 357.08 (C) 359.21 (S)
Medium (5%)	423.937	405.59 (S) 406.76 (C) 409.56 (L)	386.66 (S) 387.96 (C) 392.27 (L)	377.22 (S) 378.78 (C) 382.89 (L)	≈373 (S/C) 376.04 (L)
High (10%)	423.937	412.79 (S) 413.98 (C) 416.32 (L)	399.85 (S) 402.03 (C) 405.88 (L)	393.57 (S) 395.70 (C) 400.08 (L)	390.96 (L) 392.34 (C) 394.83 (S)

Table 32. NPV in $ when degree centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	**8.6E+09**	8.648E+09 (S) 8.641E+09 (C) 8.629E+09 (L)	8.650E+09 (S) 8.644E+09 (C) 8.626E+09 (L)	8.600E+09 (S) 8.596E+09 (C) 8.586E+09 (L)	8.537E+09 (L) 8.525E+09 (C/S)
Medium (5%)	**8.6E+09**	8.639E+09 (S) 8.632E+09 (C) 8.620E+09 (L)	8.643E+09 (S) 8.637E+09 (C) 8.618E+09 (L)	8.599E+09 (S) 8.594E+09 (C) 8.583E+09 (L)	8.540E+09 (L) 8.530E+09 (C/S)
High (10%)	**8.6E+09**	8.633E+09 (S) 8.627E+09 (C) 8.615E+09 (L)	8.638E+09 (S) 8.631E+09 (C) 8.613E+09 (L)	8.596E+09 (S) 8.591E+09 (C) 8.580E+09 (L)	8.540E+09 (L) 8.53E+09 (C/S)

* The sample size is 400 in each cell

Table 33. Number of adopters when eigenvector centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	128.12	126.41 (S) 126.66 (C) 126.94 (L)	≈125 (S/C) 126 (L)	123.48 (S) 123.96 (C) 124.57 (L)	≈123 (S/L/C)
Medium (5%)	128.12	126.48 (S) 126.75 (C) 127.02 (L)	≈125 (S/C) 126 (L)	123.77 (S) 124.14 (C) 124.75 (L)	≈123 (S/L/C)
High (10%)	128.12	126.48 (S) 126.75 (C) 127.02 (L)	≈125 (S/C) 126 (L)	123.77 (S) 124.14 (C) 124.75 (L)	≈123 (S/L/C)

Table 34. NPV in $ when eigenvector centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	**5%,10%**	**15%,30%**	**25%,50%**	**35%,70%**
Low (3%)	**1.15E+10**	1.1468E+10 (L) 1.1466E+10 (S/C)	1.1399E+10 (L) 1.1388E+10 (C) 1.1384E+10 (S)	1.132E+10 (L) 1.130E+10 (C) 1.129E+10 (S)	1.125E+10 (L) 1.121E+10 (C) 1.120E+10 (S)
Medium (5%)	**1.15E+10**	1.1469E+10 (L) 1.1466E+10 (S/C)	1.1399E+10 (L) 1.1388E+10 (C) 1.1384E+10 (S)	1.132E+10 (L) 1.130E+10 (C) 1.129E+10 (S)	1.125E+10 (L) 1.121E+10 (C) 1.120E+10 (S)
High (10%)	**1.15E+10**	1.1469E+10 (L) 1.1466E+10 (S/C)	1.1399E+10 (L) 1.1388E+10 (C) 1.1384E+10 (S)	1.132E+10 (L) 1.130E+10 (C) 1.129E+10 (S)	1.125E+10 (L) 1.121E+10 (C) 1.120E+10 (S)

* The sample size is 400 in each cell

Table 35. Number of adopters when betweenness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	471.045	341.71(S) 346.73(C) 356.46(L)	331.71(S) 332.34(C) 333.90(L)	323.75(L) 327.81(C) 329.69(S)	302.79(L) 318.01(C) 324.98(S)
Medium (5%)	471.045	364.20(S) 368.25(C) 378.08(L)	342.10(S) 344.38(C) 347.31(L)	333.51(L) 337.66(C) 339.83(S)	305.38(L) 325.64(C) 334.68(S)
High (10%)	471.045	413.57(S) 419.03(C) 427.85(L)	373.33(L) 375.07(C) 378.31(L)	347.13(L) 360.99(C) 366.48(S)	306.58(L) 337.89(C) 353.84(S)

Table 36. NPV in $ when betweenness centrality was used for location-based discounts*

Revenue Threshold	Size of Discount				
	No	5%,10%	15%,30%	25%,50%	35%,70%
Low (3%)	8.43E+09	8.653E+09 (S) 8.631E+09 (C) 8.589E+09 (L)	8.660E+09 (S) 8.646E+09 (C) 8.519E+09 (L)	8.614E+09 (L) 8.597E+09 (C) 8.578E+09 (S)	8.693E+09 (L) 8.594E+09 (C) 8.524E+09 (S)
Medium (5%)	8.43E+09	8.604E+09 (S) 8.582E+09 (C) 8.548E+09 (L)	8.641E+09 (S) 8.620E+09 (C) 8.586E+09 (L)	8.601E+09 (L) 8.591E+09 (C) 8.573E+09 (S)	8.696E+09 (L) 8.607E+09 (C) 8.535E+09 (S)
High (10%)	8.43E+09	8.558E+09 (S) 8.536E+09 (C) 8.505E+09 (L)	8.602E+09 (S) 8.583E+09 (C) 8.552E+09 (L)	8.589E+09 (L) 8.571E+09 (C) 8.548E+09 (S)	8.696E+09 (L) 8.603E+09 (C) 8.528E+09 (S)

* The sample size is 400 in each cell

APPENDIX C: NETLOGO CODE FOR THE SIMULATION MODEL

```
;******************************
;******DEFINING BREEDS*********
;******************************
;NOTE: any line starting with a semicolon contains comments (or code that will not be executed)
breed [ firms firm ]

firms-own
[
  standard  ;each firm's standard OSS or PS
  currentversion  ;version of the standard adopted by this firm
  PH  ;planning horizon given current standard
  sPH  ;planning horizon if it were to adopt the other standard
  numberofmachines  ;size of the firm represented by number of machines
  osstechnicalcapability  ;representing technical capability of the firm
  cliquishness  ;cliquishness of the neighborhood of this firm
  centrality  ;level of social influence of this firm in its neighborhood
          ;based on number of neighbors with respect to network size
          ;this is basically degree centrality
          ;it is used in the decision function
          ;and it is used to determine the order in which the firms
          ;evaluate their decision function
  betweennesscentrality
  closenesscentrality
  eigenvectorcentrality

  ;defining current costs
  clicensecosts
  csetupcosts
  ctrainingcosts
  csupportcosts
  cinteropcosts

  ;defining costs if the firm were to upgrade
  ulicensecosts
  ;defining costs if the firm were to switch
  slicensecosts
  ssetupcosts
  strainingcosts
  ssupportcosts
  sinteropcosts

  ;aggregate costs at time t+1 if the firm were to upgrade
  ucostsattplusone

  ;aggregate costs at time t+1 if the firm were to switch
  scostsattplusone
```

```
  ;number of neighbors using same or different standard
  similarneighbors
  dissimilarneighbors

  ;* SIZE OF DISCOUNT - WILL DEPEND ON THE CRITERION
  ; THAT IS USED TO COMPUTE IT
  sizebaseddiscount
  locationbaseddiscount
  combineddiscount
  ;RANKS WILL BE USED TO DETERMINE THE SIZE OF THE
  ;sizeandlocationdiscount or combined discount
  centralityrank
  sizerank
  centralitysizeaverage
  combinedrank

  numberofswitches
]

links-own
[
  rewired?  ;;to check whether this link has been rewired or not
  vot  ;;for storing volume of transactiosn associated with this link
]

;*****************************
;**DEFINING GLOBAL VARIABLES***
;*****************************

globals
[
  averagecliquishness
  versionOSS
  versionPS

  citeration  ;;current iteration number
  attributesassigned?  ;;boolean variable to see whether attributes have been assigned
                       ;;to each firm at the start of the simulation or not

  osssc_mean  ;;for storing actual values of OSS support cost distribution mean
  osssc_sd  ;;for storing actual values of OSS support cost distribution sd

  lastPSupgrade  ;for storing the iteration number when PS was last upgraded
  lastOSSupgrade  ;for storing the iteration number when OSS was last upgraded

  ;********** PART 3 VARIABLES FOR THE PS VENDOR ***************
  prev_licenses  ;licenses sold in the previous time period
  prev_revenue  ;revenue from the previous time period

  curr_licenses  ;licenses sold in the current time period
  curr_revenue  ;reveune from the current time period
```

```
  curr_license_revenue  ;revenue from licenses
  curr_support_revenue  ;revenue from support costs

  max_licenses   ;maximum licenses sold with the current upgrade
  max_revenue  ;maximum revenue from maximum licenses sold with the current upgrade
  ;max_license_revenue  ;maximum revenue from licenses
  ;max_support_revenue  ;maximum revenue from support costs

  prev_max_licenses ; maximum licenses sold with the previous upgrade
  prev_max_revenue  ;maximum reven from maximum licenses sold with the previous upgrade

  total_revenue  ;adds up prev_max_revenue
]

;******************************
;**SETTING UP THE SIMULATION***
;******************************

to setup
  ca  ;clear screen
  ;random-seed 1

  set attributesassigned? false

  ;set the OSS support cost distribution parameters
  if osssc = 0
  [
    set osssc_mean 200
    set osssc_sd 50
  ]

  if osssc = 1
  [
    set osssc_mean 200
    set osssc_sd 200
  ]

  if osssc = 2
  [
    set osssc_mean 800
    set osssc_sd 200
  ]

  setup-firms  ;initialize firms
  read-network  ;read in the network structure for these firms
  read-centrality  ;read centrality values depending on what is the 'typeofcentrality'
  compute-centrality  ;compute DEGREE centrality
  determine-centrality-rank  ;determine ranking of each firm based on centrality value

  ;****** INITIALIZE GLOBALS ********
  set curr_licenses 0
```

```
  set curr_revenue 0
  set curr_license_revenue 0
  set curr_support_revenue 0
  set max_licenses 0
  set max_revenue 0
  set prev_licenses 0
  set prev_revenue 0
  set prev_max_licenses 0
  set prev_max_revenue 0
  set total_revenue 0
  set offerdiscounts? true

  ;set the version numbers for the PS and OSS vendors
  set versionOSS 1
  set versionPS 1
  set lastOSSupgrade 0
  set lastPSupgrade 0

  setup-plot
  ;when the firms start, they start with version numbers 0
  ;we assume at the start of the simulation that they have been using some version
  ;and that at the first time period or step, the vendors offer a new version
end

to setup-firms
  ;This function basically creates 'numberoffirms' firms
  ;gives them white color and spreads them around in a circle
  set-default-shape firms "dot"  ;create turtles or firms with default shape 'dot'
                                 ;use ';show shapes' to see other possibilities
  create-firms numberoffirms  ;create firms
  layout-circle (sort firms) max-pxcor - 8
                                 ;layout the turtles in sorted order by 'who' number
                                 ;over a cirlce of radious 'max-pxcor' where
                                 ;max-pxcor is the maximum_width/2 of the screen
  ask firms
  [
    set color white
    set numberofswitches 0
  ]  ;;change the color of all firms to white
end

to read-network
  ;This will read the network from one of 9 files
  ;the procedure assumes that 1000 firms/agents/turtles
  ;have already been created
  let firm_i 0
  let firm_j 0

  ifelse (count links != 0)
  [
    clear-links
```

```
  ]
  [
    ;;assuming firms have been created
    file-open (word "nw" (word rewiringcode (word neighborhoodcode (word networknumber".txt"))))
    while [file-at-end? = false]
    [
      set firm_i file-read
      set firm_j file-read
      ask firm firm_i [ create-link-with (one-of firms with [who = firm_j])]
    ]
    file-close
  ]
end

to read-centrality
  ;this funciton will only be used when centralities
  ;have already been computed and need to be read from the files
  ;the centralities in the file are not standardized
  ;but it doesn't matter because the size of the network is 1000
  ;all the time so there is no need to come up with a standardized
  ;measure of centrality

  if typeofcentrality != "dc"
  [
    file-open (word typeofcentrality (word rewiringcode (word neighborhoodcode (word networknumber ".txt"))))
    let firmid 0
    let temporarycentrality 0
    repeat numberoffirms
    [
      set temporarycentrality file-read
      if typeofcentrality = "bc" [ask firm firmid [set betweennesscentrality temporarycentrality]]
      if typeofcentrality = "cc" [ask firm firmid [set closenesscentrality temporarycentrality]]
      if typeofcentrality = "ec" [ask firm firmid [set eigenvectorcentrality temporarycentrality]]
      set firmid (firmid + 1)
    ]
    file-close
  ]
end

to determine-centrality-rank
  if typeofcentrality = "bc" [compute-betweenness-rank]
  if typeofcentrality = "cc" [compute-closeness-rank]
  if typeofcentrality = "dc" [compute-degree-rank]
  if typeofcentrality = "ec" [compute-eigenvector-rank]
end

to compute-betweenness-rank
  let rankcounter 1000
  foreach sort-by [([betweennesscentrality] of ?1) > ([betweennesscentrality] of ?2)] firms
```

```
  [
    ask ?
    [
      set centralityrank rankcounter
      ifelse centralityrank >= 800
      [
        set locationbaseddiscount high_discount
      ]
      [
        ifelse centralityrank >= 200
        [
          set locationbaseddiscount low_discount
        ]
        [
          set locationbaseddiscount 0
        ]
      ]
    ]
    set rankcounter (rankcounter - 1)
  ]
end

to compute-closeness-rank
  let rankcounter 1000
  foreach sort-by [([closenesscentrality] of ?1) < ([closenesscentrality] of ?2)] firms
  [
    ask ?
    [
      set centralityrank rankcounter
      ifelse centralityrank >= 800
      [
        set locationbaseddiscount high_discount
      ]
      [
        ifelse centralityrank >= 200
        [
          set locationbaseddiscount low_discount
        ]
        [
          set locationbaseddiscount 0
        ]
      ]
    ]
    set rankcounter (rankcounter - 1)
  ]
end

to compute-degree-rank
  let rankcounter 1000
  foreach sort-by [([centrality] of ?1) > ([centrality] of ?2)] firms
  [
```

```
    ask ?
    [
      set centralityrank rankcounter
      ifelse centralityrank >= 800
      [
        set locationbaseddiscount high_discount
      ]
      [
        ifelse centralityrank >= 200
        [
          set locationbaseddiscount low_discount
        ]
        [
          set locationbaseddiscount 0
        ]
      ]
    ]
    set rankcounter (rankcounter - 1)
  ]
end

to compute-eigenvector-rank
  let rankcounter 1000
  foreach sort-by [([eigenvectorcentrality] of ?1) > ([eigenvectorcentrality] of ?2)] firms
  [
    ask ?
    [
      set centralityrank rankcounter
      ifelse centralityrank >= 800
      [
        set locationbaseddiscount high_discount
      ]
      [
        ifelse centralityrank >= 200
        [
          set locationbaseddiscount low_discount
        ]
        [
          set locationbaseddiscount 0
        ]
      ]
    ]
    set rankcounter (rankcounter - 1)
  ]
end



to compute-centrality
  ;This procedure computes centrality of each firm
  ;and stores it in the 'centrality' variable of each firm
```

```
  ask firms [ set centrality ((count link-neighbors) / (numberoffirms - 1)) ]
end

to compute-cliquishness
  ;This procedure computes cliquishness
  ;for the neighborhood of each firm
  ifelse all? firms [count link-neighbors <= 1]
  [
    ;; it is undefined
    ;; what should this be?
    set averagecliquishness 0
  ]
  [
    let total 0
    ask firms with [ count link-neighbors <= 1]
    [ set cliquishness "undefined" ]
    ask firms with [ count link-neighbors > 1]
    [
      ;let 'n' be the firms neighbors (gathered in localneighborhood variable)
      ;cliquishness wants to see how well connected your neighbors are with each other
      ;hence --> count links with [in-neighborhood? localneighborhood]
      ;tries to find exactly that
      ;(n * (n-1))/2 gives total number of possible links in your neighborhood
      ;imagine that to be a fully connected neighborhood
      ;so cliquishness then is the number of connections your neighbors have
      ;with each other, divided by the number of possible links in your neighborhood
      let localneighborhood link-neighbors
      set cliquishness (2 * count links with [ in-neighborhood? localneighborhood ] /
                                  ((count localneighborhood) * (count localneighborhood - 1)) )
      ;; find the sum for the value at turtles
      set total (total + cliquishness)
    ]
    ;; take the average
    set averagecliquishness (total / (count firms with [count link-neighbors > 1]))
    ;;;show averagecliquishness
  ]
end
; Part of the cliquishness and network generation code taken from Uri Wilenski's Net Logo distribution.
;Cliquishness is not actually used anywhere in the simulation.
to-report in-neighborhood? [ lnhood ]
  report ( member? end1 lnhood and member? end2 lnhood )
end
;****DRIVING THE SIMULATION****
;******************************
to go
    ; so that it gets reported at each run and then initialized at the start of each run
  assign-attributes  ;this will basically assign cost and other values to the firms
                     ;it must be called once during one run i.e. in the first step
                     ;of the run - hence the use of the 'attributesassigned?' variable
```

```
  ;show curr_license_revenue
  ;show curr_support_revenue
  ;show max_revenue
  set curr_license_revenue 0
  set curr_support_revenue 0

  compute-interoperabilitycosts

  take-decision citeration

  update-vendor-variables

  ;the following to be used only if experiments are not being run on a cluster
  ;if numberOSS >= (2 * proportionofOSS * numberoffirms)
  ;[
  ;  update-plot
  ;  stop
  ;]
  ;update-plot
end

to display-stats
  file-open "stats.txt"
    foreach sort firms
    [
      ask ?
      [
        file-type sizerank
        file-type " "
        file-type centralityrank
        file-type " "
        file-print combinedrank
      ]
    ]
  file-close
end

to setup-plot
  set-current-plot "Revenue"
  set-plot-y-range max_revenue (500 * 1000 * 199)
  set-current-plot "Licenses"
  set-plot-y-range max_licenses (500 * 1000)
  set-current-plot "PS Adopters"
  set-plot-y-range 0 1000
end

to update-plot
  set-current-plot "Revenue"
  set-current-plot-pen "max_revenue"
  plot prev_max_revenue
  set-current-plot-pen "curr_revenue"
```

```
  plot prev_revenue
  set-current-plot-pen "license_revenue"
  plot curr_license_revenue
  set-current-plot-pen "support_revenue"
  plot curr_support_revenue

  set-current-plot "Licenses"
  set-current-plot-pen "max_licenses"
  plot prev_max_licenses
  set-current-plot-pen "curr_licenses"
  plot prev_licenses

  set-current-plot "PS Adopters"
  plot numberPS
end

to update-vendor-variables
  ;update prev and curr licenses/revenue variables
  ;also update the maximum licenses/revenue variables if necessary
  if offerdiscounts? = true [set offerdiscounts? false]
  ;;show offerdiscounts?
  ;show prev_max_revenue
  ;show max_revenue
  ;show curr_revenue
  set prev_licenses curr_licenses
  set prev_revenue curr_revenue
  ;show curr_revenue
  set curr_licenses 0
  set curr_revenue 0
  set total_revenue (total_revenue + prev_revenue)

  ;first update the current (or running maximum)
  ;if (prev_licenses >= max_licenses) or (prev_revenue >= max_revenue)
  if (prev_revenue >= max_revenue)
  [
    ;if the revenue from the previous time period
    ;was better than the last recorded maximum
    ;then update maximum
    set max_revenue prev_revenue
    set max_licenses prev_licenses
  ]

  if prev_max_licenses != 0
  [
    ifelse (((prev_max_revenue - prev_revenue) / prev_max_revenue) >= revenue_threshold)
    [
        ;then offer discounts to some selected firms in the next time period
        set offerdiscounts? true
        ;show "offer discounts in next time period"
    ]
    [
```

```
    ;do nothing because even though the revenue has fallen
    ;it hasn't fallen enough to warrant any action from the vendor
    set offerdiscounts? false
  ]
 ]

 set citeration (citeration + 1)

 if (remainder citeration 2) = 0
 [
  set versionOSS (versionOSS + 1)
  set lastOSSupgrade citeration
 ]

 if (remainder citeration lengthofPSUC) = 0
 [
  set versionPS (versionPS + 1)
  set lastPSupgrade citeration

  set prev_max_licenses max_licenses
  set prev_max_revenue max_revenue
  set max_licenses 0
  set max_revenue 0
 ]

end

;*****************************
;*******ASSIGN ATTRIBUTES******
;*****************************
to assign-attributes
 let cuc 0  ;temporary variable used in assigning planning horizon PH to each firm
 let suc 0  ;temporary variable used in assign the planning horizon sPH to each firm
 let randomvalue 0  ;randomvalue generated to assign PH to each firm
 let temporarycounterforOSS 0  ;this will keep track of how many firms have been assigned the OSS standard

 if attributesassigned? = false ;i.e. if this is the first time this procedure is being
                                ;called in this run
 [
  set attributesassigned? true  ;so that this procedure is not called again

  foreach (sort firms)
  [
   ask ?
   [
    ;assign number of machines per firm
    set numberofmachines ((random (ubmachinesperfirm - lbmachinesperfirm)) + lbmachinesperfirm)

    ;;assign OSS technical capability
```

```
    set osstechnicalcapability (random-normal TCOSSmean TCOSSsd)
    ;if OSS technical capability is less then 0, truncate it to 0
    if osstechnicalcapability < 0 [set osstechnicalcapability 0]

    ;;assign standard
    ifelse ((random-float 1.0 < proportionofOSS) and (temporarycounterforOSS < (proportionofOSS * numberoffirms)))
    [
      ;if enough OSS standards have not been assigned
      ;then keep assigning them :)
      set temporarycounterforOSS (temporarycounterforOSS + 1)
      ;;OSS
      set standard "OSS"
      set color blue
      set currentversion 0
      ;;assign planning horizon
      set cuc 2
      set suc lengthofPSUC

      ;assign current and upgrade costs based on "OSS"
      ;multiply the costs by number of machines in the firm
      set clicensecosts (cLcOSS * numberofmachines)
      set csetupcosts (cSTcOSS * numberofmachines)
      set ctrainingcosts (cTRcOSS * numberofmachines)
      set csupportcosts (random-normal osssc_mean osssc_sd)
      ;if support costs are less than 0, truncate them to 0
      if csupportcosts < 0 [ set csupportcosts 0]
      set csupportcosts (csupportcosts * numberofmachines)

      set ulicensecosts (uLcOSS * numberofmachines)

      ;since this is an OSS firm
      ;multiply its support costs by its OSS technical capability
      set csupportcosts (csupportcosts * osstechnicalcapability)

      ;assign costs in case this firm switches to "PS"
      set slicensecosts (cLcPS * numberofmachines)
      set ssetupcosts (cSTcPS * numberofmachines)
      set strainingcosts (cTRcPS * numberofmachines)
      set ssupportcosts (50 * numberofmachines)
    ]
    [
      ;;PS
      set standard "PS"
      set currentversion 0
      set color red

      ;;assign planning horizon
      set cuc lengthofPSUC
      set suc 2
```

```
  ;;assign current and upgrade costs based on "PS"
  set clicensecosts (cLcPS * numberofmachines)
  set csetupcosts (cSTcPS * numberofmachines)
  set ctrainingcosts (cTRcPS * numberofmachines)
  set csupportcosts (50 * numberofmachines)

  set ulicensecosts (uLcPS * numberofmachines)

  ;assign costs in case this firm switches to "OSS"
  set slicensecosts (cLcOSS * numberofmachines)
  set ssetupcosts (cSTcOSS * numberofmachines)
  set strainingcosts (cTRcOSS * numberofmachines)
  set ssupportcosts (random-normal osssc_mean osssc_sd)
  if ssupportcosts < 0 [set ssupportcosts 0]
  ;if support costs are less than 0, truncate them to 0
  set ssupportcosts (ssupportcosts * numberofmachines)

  ;if this PS firm were to switch to OSS, its support costs
  ;for OSS should be multiplied by its OSS technical capability
  set ssupportcosts (ssupportcosts * osstechnicalcapability)

  ;******** UPDATE THE MAXIMUM LICENSES AND REVENUE VARAIBLES
**********
  ;THIS WILL BE DONE ONLY ONCE AT THE START OF THE SIMULATION
  set max_licenses (max_licenses + numberofmachines)
  ;set max_revenue (max_revenue + (numberofmachines * uLcPS)) ;cLcPS = 199
  set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS))
  set curr_support_revenue (curr_support_revenue + csupportcosts)
  set max_revenue (max_revenue + curr_license_revenue + curr_support_revenue)
  set prev_max_licenses 0
  set prev_max_revenue 0
 ]

 ;;ASSIGN THE PLANNING HORIZON
 set randomvalue (random-float 1.0)
 ifelse (randomvalue < 0.10)
 [
  ;;assign uc as ph
  set PH cuc
  set sPH suc
 ]
 [
  ifelse (randomvalue < 0.50)
  [
   ;;assign uc+1 as ph
   set PH (cuc + 1)
   set sPH (suc + 1)
  ]
  [
   ifelse (randomvalue < 0.90)
   [
```

```
          ;;assign uc+2 as ph
          set PH (cuc + 2)
          set sPH (suc + 2)
        ]
        [
          ;;assign uc+3 as ph
          set PH (cuc + 3)
          set sPH (suc + 3)
        ]
      ]
    ]
  ]
]
set max_revenue (curr_license_revenue + curr_support_revenue)

;;sort 'links' by their 'who' number and assign each one a volume of transactions
foreach (sort links)
[
  ask ?
  [
    ;go to the link and read in its volume of transactions
    set vot (random (ubvot - lbvot) + lbvot)
  ]
]

;****** UPDATE THE PREVIOUS LICENSES AND REVENUE VARIABLES
;BASED ON THE VALUES ASSIGNED TO THE MAXIMUM LICENSES AND
REVENUE VARIABLES
;THIS WILL BE DONE ONLY ONCE AT THE START OF THE SIMULATION
set prev_licenses max_licenses
set prev_revenue max_revenue
set prev_max_licenses max_licenses
set prev_max_revenue max_revenue
;the idea is that the benchmark or prev_max_revenue at the start
;should be total number of licenses * 199 (assuming that at t = -1)
;the firms who are using PS have upgraded

let rankcounter 1000
foreach sort-by [([numberofmachines] of ?1) > ([numberofmachines] of ?2)] firms
[
  ask ?
  [
    set sizerank rankcounter
    ifelse sizerank >= 800 [ set sizebaseddiscount high_discount]
    [ ifelse sizerank >= 200 [ set sizebaseddiscount low_discount] [ set sizebaseddiscount 0]]
    set centralitysizeaverage int (((sizerank + centralityrank) / 2))
  ]
  set rankcounter (rankcounter - 1)
]

set rankcounter 1000
```

```
    foreach sort-by [([centralitysizeaverage] of ?1) > ([centralitysizeaverage] of ?2)] firms
    [
      ask ?
      [
        set combinedrank rankcounter
        ifelse combinedrank >= 800 [ set combineddiscount high_discount]
        [ ifelse combinedrank >= 200 [ set combineddiscount low_discount] [ set combineddiscount
0]]
      ]
      set rankcounter (rankcounter - 1)
    ]
    ;update-plot
  ]
end

;******************************
;COMPUTE INTEROPERABILITY COSTS
;******************************
to compute-interoperabilitycosts
  ;This procedure computes interoperability costs for each firm
  ;based on volume of transactions on its links

  let volumeoftransactions 0

  ask links
  [
    ;reset the interoperability cost variables
    ;of the respective firms on this link to 0
    ask both-ends
    [
      set similarneighbors 0
      set dissimilarneighbors 0
      set cinteropcosts 0
      set sinteropcosts 0
    ]
  ]

  foreach (sort links)  ;sorts links by who number
  [
    ask ?
    [
      set volumeoftransactions vot

      ifelse ([standard] of end1 = [standard] of end2)
      [
        ;if the existing standards of both firms are the same
        ;use volume of transactions to adjust the sinteropcosts
        ;for both these firms in case they were to switch their standard
        ;sinteropcosts are interoperbaility costs that would be incurred
        ;if a firm were to switch its standard
```

```
        ask both-ends
        [
          set similarneighbors (similarneighbors + 1)
          set sinteropcosts (sinteropcosts + (interoperabilitycosts * volumeoftransactions))
        ]
      ]
      [
        ;if the existing standards of both firms are different
        ;then compute cinteropcosts or current interoperability costs
        ;and increment the counter for dissimilar neighbors
        ask both-ends
        [
          set dissimilarneighbors (dissimilarneighbors + 1)
          set cinteropcosts (cinteropcosts + (interoperabilitycosts * volumeoftransactions))
        ]
      ]
    ]
  ]
end

to take-decision [iterationnumber]
  ;This is the procedure where the firms will make a decision
  ;regarding upgrades or a switch

  let LHS 0  ;for storing the aggregated LHS of the decision function
  let RHS 0  ;for storing the aggregated RHS of the decision function
  let tempSC 0  ;for temporarily storing support costs
  let lastupgrade 0 ;for storing the iteration number for last upgrade depending on the standard of
this firm

  foreach (sort-by [([centrality] of ?1) > ([centrality] of ?2)] firms)
  [
    ask ?
    [
      ;A firm will only consider such a decision if its planning horizon PH has expired
      ifelse (iterationnumber = 0 or ((remainder iterationnumber PH) = 0))
      [
        ;FIRST, adjust the support costs
        ;if it is a PS firm adjust its current support costs
        ;the firm should look ahead to see if it will be more than X versions
        ;behind the vendor's version at the end of this planning horizon
        ;it must take the increased support costs into consideration
        ;the X number of versions can be decided by 'withdrawsupportafter'
        let safetytime 0  ;the time period after which support costs should be bumped up
        let safetyperiod 0  ;the duration of time for which support costs should not be bumped up
        let dangerousperiod 0  ;the duration of time for which support costs should be bumped up

        set safetytime (lastPSupgrade + (withdrawsupportafter * lengthofPSUC))
        set safetyperiod (safetytime - iterationnumber)

        ;Adjust support costs ONLY IF
```

```
;the safetytime comes before the planning horizon expires

if standard = "PS"
[
 ifelse (safetytime < iterationnumber + PH)
 [
  set dangerousperiod (PH - safetyperiod)
  set csupportcosts ( (safetyperiod * 50) + (dangerousperiod * 100) ) / PH
  set csupportcosts (csupportcosts * numberofmachines)
 ]
 [
  ;this is to ensure that if costs were adjsuted in one cycle
  ;next time around, if the firm has upgraded, it should face support costs = 50
  ;and not the adjusted one it had estimated over its previous PH
  set csupportcosts (50 * numberofmachines)
 ]
]

if standard = "OSS"
[
 ifelse (safetytime < iterationnumber + sPH) and standard = "OSS"
 [
  set dangerousperiod (sPH - safetyperiod)
  set ssupportcosts ( (safetyperiod * 50) + (dangerousperiod * 100) ) / sPH
  set ssupportcosts (ssupportcosts * numberofmachines)
 ]
 [
  set ssupportcosts (50 * numberofmachines)
 ]
]

;SECOND, THIRD assuming that the support costs
;have been adjusted, compute the costs if the firm were to upgrade or switch

;THIS FIRST SET OF NESTED IF-CONDITIONS WILL CHECK IF DISCOUNTS ARE
BEING OFFERED
; AND THAT IF THIS IS A PS FIRM THEN THE DISCOUNTS SHOULD BE APPLIED
TO THE UPGRADE COSTS
ifelse standard = "PS" and offerdiscounts? = true
[
 ifelse typeofdiscount = "sizebased" [set ucostsattplusone ( (ulicensecosts * (1 -
sizebaseddiscount)) + csupportcosts + cinteropcosts)]
 [ifelse typeofdiscount = "locationbased" [set ucostsattplusone ( (ulicensecosts * (1 -
locationbaseddiscount)) + csupportcosts + cinteropcosts)]
 [set ucostsattplusone ( (ulicensecosts * (1 - combineddiscount)) + csupportcosts +
cinteropcosts)]]
]
[
 ; NO DISCOUNTS WILL BE APPLIED TO THE UPGRADE COSTS
 ; IF THIS IS NOT A PS FIRM OR IF THE DISCOUNTS ARE NOT BEING OFFERED
 set ucostsattplusone (ulicensecosts + csupportcosts + cinteropcosts)
```

```
]

;THIS SET OF NESTED IF-CONDITIONS WILL CHECK IF THE DISCOUNT SHOULD BE APPLIED TO THE SWITCHING COSTS
;THEY WILL BE APPLIED ONLY IF THIS IS AN OSS FIRM
ifelse standard = "OSS" and offerdiscounts? = true
[
 ifelse typeofdiscount = "sizebased" [set scostsattplusone ( (slicensecosts * (1 - sizebaseddiscount)) + ssupportcosts + sinteropcosts + ((ssetupcosts + strainingcosts) / sPH))]
 [ifelse typeofdiscount = "locationbased" [set scostsattplusone ( (slicensecosts * (1 - locationbaseddiscount)) + ssupportcosts + sinteropcosts + ((ssetupcosts + strainingcosts) / sPH))]
 [set scostsattplusone ( (slicensecosts * (1 - combineddiscount)) + ssupportcosts + sinteropcosts + ((ssetupcosts + strainingcosts) / sPH))]]
]
[
 set scostsattplusone (slicensecosts + ssupportcosts + sinteropcosts + ((ssetupcosts + strainingcosts) / sPH))
]

;FOURTH, compute LHS of the decision function if upgrade costs are not 0
;i.e. avoid division by 0
ifelse ucostsattplusone != 0
[
 set LHS ((ucostsattplusone - scostsattplusone) / ucostsattplusone)
]
[
 ;if upgrade costs are 0, the firm will upgrade
 ;since RHS can never be <= 0
 set LHS 0
]

;FIFTH, compute RHS of the decision function
set RHS ( (1 - centrality) * (1 - (dissimilarneighbors / (similarneighbors + dissimilarneighbors))))

;SIXTH, if LHS >= RHS
;change standards, costs, version numbers and planning horizons

let tempPH 0

ifelse LHS >= RHS
[
 ;;then switch the standard of the firm
 ;show "switched"
 ;show standard
 set numberofswitches (numberofswitches + 1)
 ifelse (standard = "OSS")
 [
  ;;make new standard "PS"
  set standard "PS"
  set color red
```

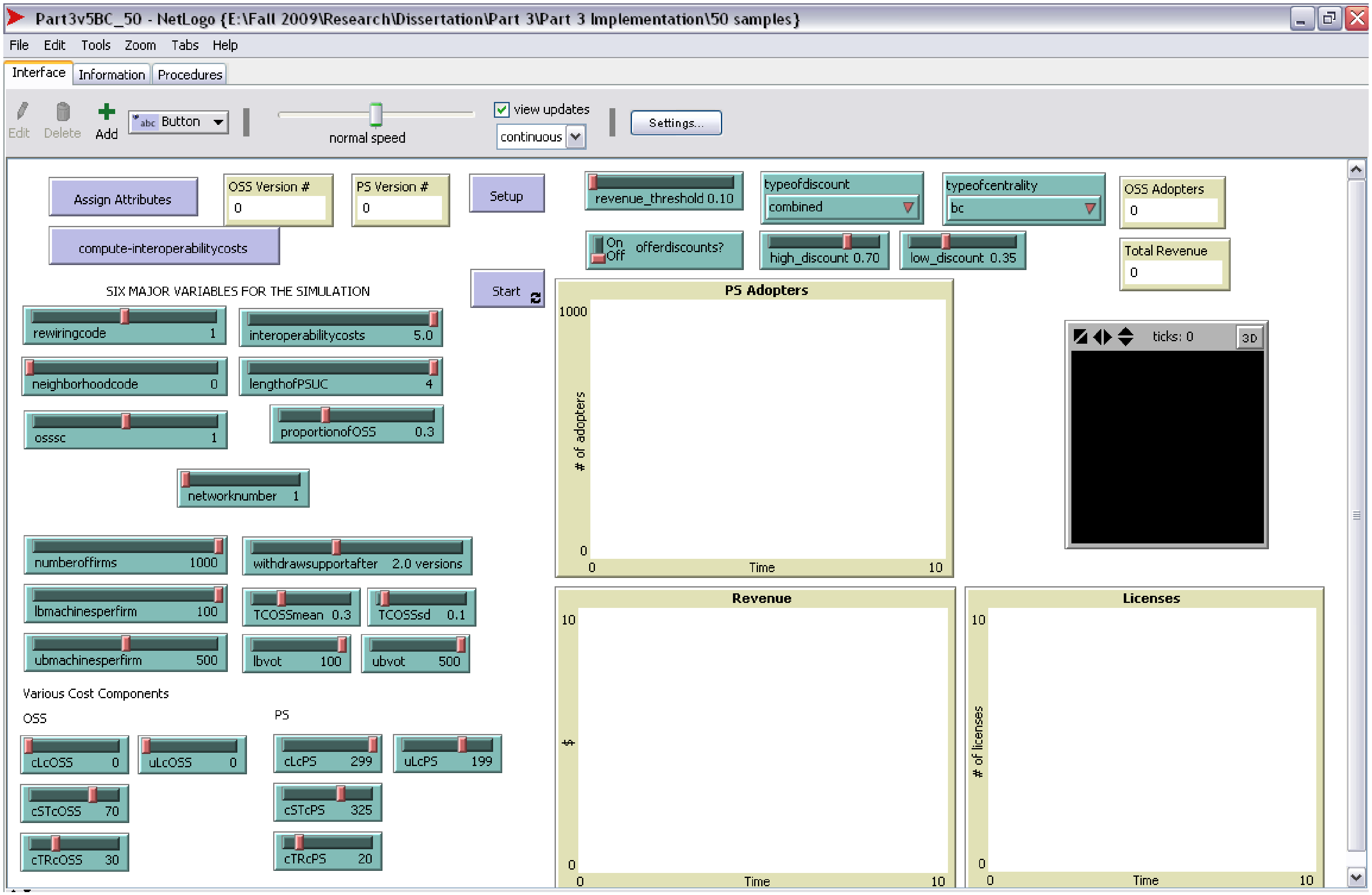
```
set currentversion versionPS  ;;assign latest version of PS
;;swap current and switching planning horizons
set tempPH PH
set PH sPH
set sPH tempPH
;;re-assign the other costs
;;assign current and upgrade costs based on "PS"
set clicensecosts (cLcPS * numberofmachines)
set csetupcosts (cSTcPS * numberofmachines)
set ctrainingcosts (cTRcPS * numberofmachines)

set ulicensecosts (uLcPS * numberofmachines)

;;assign costs in case this firm switches to "OSS"
set slicensecosts (cLcOSS * numberofmachines)
set ssetupcosts (cSTcOSS * numberofmachines)
set strainingcosts (cTRcOSS * numberofmachines)

;;swap the support costs
;;this was an OSS firm with csupportcosts based on a distribution
;;before you set its csupportcosts to 50 * numberofmachines
;;save them in a temporary location and make them the ssupportcosts
set tempSC csupportcosts
set csupportcosts (50 * numberofmachines)
set ssupportcosts tempSC  ;;no need to multiply with osstechnical capbaility
                          ;;since that has been done already

;***IF AN OSS FIRM SWITCHED TO PS
;THAT MEANS AN INCREASE IN LICENSES/REVENUE FOR PS
;SO ADJUST THE CURRENT LICENSES VARIABLE
set curr_licenses (curr_licenses + numberofmachines)
ifelse offerdiscounts? = true
[
  ;show "switched with discounts"
  ifelse typeofdiscount = "sizebased" [set curr_license_revenue (curr_license_revenue +
(numberofmachines * cLcPS * (1 - sizebaseddiscount)))] ;cLcPS = 299 because this is a new
adopter
  [ifelse typeofdiscount = "locationbased"[set curr_license_revenue (curr_license_revenue
+ (numberofmachines * cLcPS * (1 - locationbaseddiscount)))] ;cLcPS = 299 because this is a
new adopter
  [set curr_license_revenue (curr_license_revenue + (numberofmachines * cLcPS * (1 -
combineddiscount)))]] ;cLcPS = 299 because this is a new adopter
]
[
  ;show "switched without discounts"
  set curr_license_revenue (curr_license_revenue + (numberofmachines * cLcPS))
]
set curr_support_revenue (curr_support_revenue + csupportcosts)
;set curr_revenue (curr_revenue + curr_license_revenue + curr_support_revenue)
]
[
```

```
        ;;make new standard "OSS"
        set standard "OSS"
        set color blue
        set currentversion versionOSS ;;assign latest version of OSS
        set cinteropcosts 0  ;;since these will be re-computed
        set sinteropcosts 0  ;;since these will be re-computed
        set similarneighbors 0  ;;these will have to be recomputed when interoperability costs are
recomputed
        set dissimilarneighbors 0  ;;these will have to be recomputed when interoperability costs
are recomputed
        ;;swap planning horizons
        set tempPH PH
        set PH sPH
        set sPH tempPH
        ;;re-assign the other costs
        ;;assign current and upgrade costs based on "OSS"
        set clicensecosts (cLcOSS * numberofmachines)
        set csetupcosts (cSTcOSS * numberofmachines)
        set ctrainingcosts (cTRcOSS * numberofmachines)
        set ulicensecosts (uLcOSS * numberofmachines)

        ;;assign costs in case this firm switches to "PS"
        set slicensecosts (cLcPS * numberofmachines)
        set ssetupcosts (cSTcPS * numberofmachines)
        set strainingcosts (cTRcPS * numberofmachines)

        ;;swap the support costs
        ;;this was a PS firm with csupportcosts based on $50/year
        ;;OR those costs were spread over additional years depending on
        ;;the planning horizon
        set tempSC ssupportcosts
        set csupportcosts tempSC
        set ssupportcosts (50 * numberofmachines)

        ;show "switched to OSS"
        ;**** IF A FIRM MOVED AWAY FROM PS, NO NEED TO MAKE ADJUSTMENTS
        ;TO THE CURRENT LICENSES/REVENUE VARIABLES FOR THE PS VENDOR
        ;SINCE THOSE VALUES ARE BEING CALCULATED FROM SCRATCH AND THIS
        ;DECISION FROM PREVIOUS PS ADOPTERS WILL AUTOMATICALLY FIGURE
INTO
        ;THE COMPUTATION OF THE CURRENT LICENSES AND REVENUE
VARIABLES
       ]
      ]
      [
       ;set the upgrade license costs to current license costs
       set clicensecosts ulicensecosts

       ;update the version number of this firm since it has upgraded
       ifelse standard = "OSS"
       [
```

```
          set currentversion versionOSS
          ;show "upgraded OSS"
        ]
        [
          set currentversion versionPS
          ;**** THIS MEANS THAT A PS ADOPTER UPGRADED
          ;SO THIS WILL BE FACTORED INTO THE CALCULATION OF
          ;CURRENT LICENSES AND REVENUE
          set curr_licenses (curr_licenses + numberofmachines)
          ifelse offerdiscounts? = true
          [
            ifelse typeofdiscount = "sizebased" [set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS * (1 - sizebaseddiscount)))] ;uLcPS = 199
            [ifelse typeofdiscount = "locationbased" [set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS * (1 - locationbaseddiscount)))]
            [set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS * (1 - combineddiscount)))]]
            ;show "upgraded PS with discounts"
          ]
          [
            set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS))
            ;show "upgraded PS without discounts"
          ]
          set curr_support_revenue (curr_support_revenue + csupportcosts)
          ;set curr_revenue (curr_revenue + curr_license_revenue + curr_support_revenue)
        ]
      ]
    ]
    [
      ;IF THE PLANNING HORIZON IS NOT BEGINNING
      ;REVENUE STILL NEEDS TO BE COMPUTED IF THIS IS A PS ADOPTER
      if standard = "PS"
      [
        set curr_licenses (curr_licenses + numberofmachines)
        set curr_license_revenue (curr_license_revenue + (numberofmachines * uLcPS))
        set curr_support_revenue (curr_support_revenue + csupportcosts)
      ]
    ]
  ]
 ]
 set curr_revenue (curr_license_revenue + curr_support_revenue)
end

to-report numberOSS
 report (count firms with [standard = "OSS"])
end

to-report numberPS
 report (count firms with [standard = "PS"])
end
```

```
to-report report-versionOSS
  report versionOSS
end

to-report report-versionPS
  report versionPS
end

to-report total-revenue
  report total_revenue
end

to-report license-revenue
  report curr_license_revenue
end

to-report support-revenue
  report curr_support_revenue
end
```

APPENDIX D: SNAPSHOT OF THE NETLOGO MODEL



APPENDIX E: SAMPLE FILES FOR UCINET

With 2 network topologies, 3 network densities and 50 replications each, 300 networks (2x3x50) were generated using the Watts and Strogatz (1998) algorithm. UCINET was used to compute the different individual centrality values. To compute these values, the 300 networks had to be represented in a certain format to allow UCINET to read the network. DL files read by UCINET allow different formats for storing a network. Following is just a snapshot of a DL file that was created for use with UCINET.

```
dl n=1000 format = nodelist1
data:
1   4 5 8 13 14 18 19 23 28 29 31 36 41 47 50 52 55 57 62 65 69 70 71 74 82 83 84 85 88
2   3 6 7 8 11 12 15 21 23 35 37 40 45 46 47 49 50 56 57 63 69 71 73 84 87 91 94 98 102
3   2 9 14 21 22 23 29 30 32 35 38 41 42 44 45 50 52 56 57 63 64 66 73 76 77 78 83 85 8
4   1 6 15 16 17 25 28 29 31 32 34 36 41 48 49 56 57 61 63 65 66 67 69 71 74 80 83 87 9
5   1 6 7 9 11 12 13 19 22 35 37 40 48 49 50 52 53 55 63 64 65 69 71 73 74 76 77 80 86
6   2 4 5 8 13 14 15 18 21 24 28 29 30 36 39 40 41 42 43 45 46 47 50 52 57 58 65 66 67
7   2 5 8 9 13 14 15 16 18 19 20 22 23 24 29 31 37 39 43 49 50 57 61 63 64 65 70 72 77
8   1 2 6 7 12 14 23 26 27 30 33 35 36 39 42 44 53 54 55 58 62 66 69 73 80 86 88 95 96
9   3 5 7 10 19 25 26 36 38 39 43 45 48 51 52 59 64 68 74 75 79 82 84 86 89 90 93 97 10
10   9 12 14 19 23 24 25 27 28 30 31 32 35 37 39 41 43 44 51 55 58 60 62 67 68 70 71 74
11   2 5 14 15 18 25 33 36 42 46 53 55 56 57 58 60 63 68 69 71 75 77 79 85 89 93 97 101
12   2 5 8 10 18 22 24 35 41 45 46 48 49 55 56 59 61 73 75 76 78 86 87 88 92 94 95 98 1
13   1 5 6 7 14 17 19 28 31 32 37 40 41 44 48 49 51 54 61 66 67 68 72 73 74 77 78 81 84
14   1 3 6 7 8 10 11 13 19 20 26 27 29 36 38 39 40 43 44 45 53 55 56 58 59 64 68 71 73
15   2 4 6 7 11 20 21 26 28 30 32 34 36 38 39 40 41 44 55 57 61 65 67 74 78 86 89 91 92
16   4 7 19 22 26 32 33 40 41 42 47 48 52 54 57 58 63 67 68 70 74 76 77 79 80 87 89 94
17   4 13 23 30 31 33 35 37 41 44 47 48 49 50 53 54 68 74 75 77 79 86 92 102 105 106 10
18   1 6 7 11 12 19 25 31 32 38 40 44 45 50 58 61 64 65 72 73 80 83 84 90 94 95 96 98 1
19   1 5 7 9 10 13 14 16 18 22 25 26 29 36 48 49 51 52 59 62 66 71 72 73 74 76 79 81 83
20   7 14 15 38 48 53 54 60 64 65 67 68 81 84 85 92 93 94 102 103 105 107 108 111 113 1
```

The first line indicates that UCINET should expect to read a network of a 1000 nodes in the ‘nodelist1’ format. The ‘nodelist1’ format indicates that each node in the network will be listed in one line and next to it will be a list of all other nodes that this node is connected to. For example, in the illustration above, node 1 is connected to nodes 4, 5, 8, 13 etc.; node 20 is connected to nodes 7, 14, 15 etc. Once 300 files were created for the various networks, UCINET was used to compute the different centrality measures. These values were stored in text files which were then read, as and when required, at the start of each experiment. Group centrality values were computed separately by writing another short program and stored in separate text files.