Open Technological Standardization Processes Through Learning Networks

by

Christakis Mina

B.Sc. in Engineering, 1996
M.Sc. in Engineering, 2007

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Science

in

Engineering

in the

GRADUATE SCHOOL OF ENGINEERING
DEPARTMENT OF URBAN MANAGEMENT
of
KYOTO UNIVERSITY

Open Technological Standardization Processes Through Learning Networks

Abstract

Open Technological Standardization Processes Through Learning Networks

by

Christakis Mina
Doctor of Science in Engineering
Kyoto University

Open standards, open access and open source software are three phenomena that have been receiving increased scholarly attention lately. Recently, the number of research studies related to Open Standards and F/OSS saw an extraordinary growth. The scholarly attention given to the F/OSS phenomenon comes from a variety of disciplines including engineering, social sciences and economics. In addition to that, non-profit organizations as well as governments recognized the potential of the emerging F/OSS approach and dedicated resources in researching and supporting the phenomenon. Increasingly, governments around the globe are encompassing F/OSS and open standards in general with notable examples found in Japan, the United States, Germany, Brazil, Italy, Russia and Peru, among others. In addition to that, major for-profit organizations such as IBM, HP, Amazon.com and Google, among others, recognized the business opportunities F/OSS can generate and invested substantial resources to it which appears as a puzzle since F/OSS is regarded as a type of public good and public goods according to economic theory are not considered to be sustainable.

Even though a lot of research has been conducted in understanding the technicalities of the F/OSS phenomenon, surprisingly little research has been conducted in understanding the formation and evolution mechanisms behind F/OSS collaboration. This can be partially explained by the fact that F/OSS is a highly complex and heterogeneous phenomenon. Several researchers consider F/OSS to be a provision of a public good. According the the economic literature, public goods suffer from market failures such as free-riding. As a result, researchers expected that F/OSS not to be sustainable and to eventually vanish. Contrary to that, F/OSS is thriving and this fact presents a puzzle to researchers. In addition to that, other characteristics of F/OSS such as the fact that F/OSS is created mainly by volunteer contributions which in turn is made available to the public and its highly intricate institutional structure having the

ability to adapt over time made the phenomenon a focus of several research studies.

The main objective of the research is to investigate open standardization processes and learning networks. The pivotal phenomenon that is investigated is the collaboration network of F/OSS development. The main model is build within the complex social network analysis framework and it is used to analyze the formation and evolution of the F/OSS collaboration network. In addition to that, a formal model that investigates learning networks and their applications in engineering, specifically in risk analysis and management, via the Bayesian Neural Network (BNN) framework is also developed in this research. The BNN model is used to evaluate the cost risk of soil decontamination projects and illustrates how complex network analysis utilizing learning networks can be used in practical engineering applications.

The research presented in this dissertation spans several disciplines that include engineering, computer science, statistical physics, economics and law. The research attempts to bridge the gap in the literature of open standardization and scientific collaboration networks by investigating the formation and evolution of the F/OSS collaborators social network. The main research questions presented are: "why open standardization is emerging", "why open standardization phenomena such as F/OSS are currently thriving?" and "are those phenomena sustainable in the long run?". In the course of answering the research questions, the dissertation pivots on the F/OSS phenomenon presents its history and open standardization related literature review, analyzes the formation and evolution of the F/OSS collaboration network and investigates the sustainability of the phenomenon. Finally the dissertation presents two case studies based on open content collaborative projects that were developed during the research.

Dedicated to my parents Andreas and Maria, my wife Miho and my children Anthony, Marc and Julia.

Contents

List of Figures

List of Tables

Acknowledgments

It is a common misconception that doctoral research is a solitary activity. The image of a doctoral student confined in his lab, conducting his experiments and simulations alone and then documenting the results in his dissertation is what most people have in their mind. Those engaged in doctoral research know otherwise. They know that doctoral research is also a social activity. In fact, if the doctoral student fails to build and maintain a relevant social network he will face the risk of failure. Of course, hard work, dedication, persistence, perseverance and creativity are also very important factors of success, but no doctoral research can be successfully completed without the support and help of a group of dedicated people. My doctoral research is no exception to that. Here I would like to to acknowledge all of those that help make this research reach its goals.

I am deeply indebted to my academic adviser Professor Kiyoshi Kobayashi for his guidance, mentoring and support during this research. His encouragement, constructive criticism, sharp comments and feedback help tremendously in shaping this dissertation. Even though recently his workload became more than double, he always kept his door open to me, listened to my questions and concerns patiently and advised when necessary or provided new challenges to me in order to deepen my understanding on the research topic and expand my knowledge.

I would like to express my appreciation and gratitude to the members of my defense committee Professors Masashi Kawasaki and Satoshi Fujii for their constructive criticism and valuable comments.

I am grateful to Associate Professor Kakuya Matsushima for his guidance and advising during my doctoral studies. Our discussions and his feedback during my research seminars resulted in new ideas and research approaches that stimulated the progress of this research. I would like to also thank him for his help in dealing with the many difficulties I encountered during my stay in Kyoto and for being a good friend to me and my family.

My gratitude also goes to Professor Charles Scawthorn for his valuable advising, encouragement and support during this research. Professor Scawthorn has been an outstanding academic mentor to me. During this research I learned a lot from him, especially in the field of risk analysis and management.

I would like also to thank Assistant Professor Masamitsu Onishi for his help and friendship during this research. I'm also thankful to Assistant Professor Mamoru Yoshida for his support in this research, for helping me during my stay in the lab and for his friendship.

My appreciation also goes to Dr. Lei Shi and Dr. Hayeong Jeong for their support, feedback and friendship during my doctoral studies. My thanks also to the academic assistant of our lab, Ms. Aya Fujimoto for her kindness and support. I'm also thankful to my fellow student-colleagues for their friendship. I would like to especially thank (in alphabetical order) my friends Dr. Giancarlo Flores, Mr. Kensuke Hishida, Dr. Michiro Ishikawa, Dr. Le Thanh Nam, Dr. Puay How Tion, Mr. Fei Xu, and Dr. Emine Yetiskul for their friendship and support.

My appreciation and gratitude also goes to the members of the Alliance for Global Open Risk Analysis (AGORA) for their participation in the case study and for their support and encouragement.

I'm especially grateful to the Ministry of Education, Culture, Sports, Science and Technology of Japan. This dissertation would not have been possible without the scholarship funding I received and the kindness of the people of Japan.

Finally, my sincere gratitude goes to my parents and my wonderful family for their unconditional love and support.

Symbols

Abbreviations

ANN	Artificial Neural Network
BNN	Bayesian Neural Network
CGI	Common Gateway Interface
COTS	Commercial-Off-The-Shelf (Software)
CVS	Concurrent Versions System
FSF	Free Software Foundation
FTP	File Transfer Protocol
F/OSS	Free/Open Source Software
GNU	GNU's Not Unix
GPL	General Public License
HP	Hewlett-Packard
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines
IDE	Integrated Development Environment
IP	Intellectual Property
IRC	Internet Relay Chat
IT	Information Technology
ITS	Incompatible Time Sharing system
MH	Metropolis-Hastings Algorithm
MLP	Multi-Layer Perceptron
PC	Personal Computer
PDF	Portable Document Format
PHP	PHP Hypertext Preprocessor
SCCL	Soil Contamination Countermeasures Law of Japan
SCP	Secure Copy Protocol
SFTP	Secure File Transfer Protocol
SSH	Secure Shell
WWW	World Wide Web

Chapter 1

Introduction

Where the mind is without fear and the head is held high;
Where knowledge is free;

– Rabindranath Tagore, "Mind Without Fear"

1.1 Open Standards and the Emergence of the F/OSS Phenomenon

Open standards, open access and open source software are three phenomena that have been receiving increased scholarly attention lately. The importance of open standardization in engineering systems, specifically in infrastructure systems is actively debated by researchers in several disciplines including engineering, economics and the social sciences. For example, there exist various definitions of open standards in the literature and it seems that there is currently no consensus on an agreed definition. Pountain (2003) for example defines open standard as

> *a standard that is independent of any single institution or manufacturer, and to which users may propose amendments.*

The Free Software Foundation Europe (FSFE), a non-profit organization supporting open standards provides the following definition (FSFE, 2009):

> *An Open Standard refers to a format or protocol that is*
>
> 1. *subject to full public assessment and use without constraints in a manner equally available to all parties;*
> 2. *without any components or extensions that have dependencies on formats or protocols that do not meet the definition of an Open Standard themselves;*
> 3. *free from legal or technical clauses that limit its utilisation by any party or in any business model;*

4. *managed and further developed independently of any single vendor in a process open to the equal participation of competitors and third parties;*
5. *available in multiple complete implementations by competing vendors, or as a complete implementation equally available to all parties.*

Coyle (2002) suggests that open standards should allow anybody to use them, to acquire them for free or nominal low cost and should not impose any limitations in participation of their development by anyone. In addition to the above, Ghosh (2005) suggests that

> *open standards should be defined in terms of a desired economic effect: supporting full competition in the marketplace for suppliers of a technology and related products and services.*

Currently, Free/Open Source Software is a paradigm of open standardization. A layperson can not easily grasp the importance of software to our modern society. It is not obvious that software controls everyday use domestic appliances such as refrigerators, televisions, washing machines and micro-wave ovens, is used on communication devices such as mobile phones and fax machines and as well as on critical systems such as the electronic control systems of trains, cars, ships, airplanes power plants and space shuttles. Our daily lives depend on a variety of devices and systems, including critical infrastructure, which in-turn are run and controlled by software. The emergence of software standardization in sciences and academia goes back to the era of the first computers. Researchers, scientists and practitioners alike use a variety of software tools in their daily activities, either to control electronic and mechanical devices, to do experiments and simulations, and to record, analyze and process data. Software typically used by researchers is either commercial off-the shelf (COTS) or custom-made. In fact, custom-made software represents the lion's share of the software used in research. Custom-made software created in the academia, has been following the traditional rules of knowledge dissemination and peer review. That is, in order for the scientific community to validate the results of a specific research, its data, tools (e.g. software) and methodologies have to be transparent and accessible for review:

> *The scientific method rests on a process of discovery, and a process of justification. For scientific results to be justified, they must be replicable. Replication is not possible unless the source is shared: the hypothesis, the test conditions, and the results. The process of discovery can follow many paths, and at times scientific discoveries do occur in isolation. But ultimately the process of discovery must be served by sharing information: enabling other scientists to go forward where one cannot; pollinating the ideas of others so that something new may grow that otherwise would not have been born.* (DiBona et al., 1999)

This approach based on openness, transparency and replicability was the norm in the early times of computing and software development. Recently, with the emerge and proliferation of commercial proprietary software, the traditional norm has been pushed aside. This raised concerns among the research and software engineering communities due to the closed nature of COTS, eventually resulting in the emergence of a new philosophy and methodologies to develop software that preserves the traditional ways of knowledge dissemination and provides transparency and accessibility to all. This new software development philosophy approach came to be known as Free/Open Source Software[1] and its impact on the industry, academia, free markets and the society in general has been remarkable.

1.2 Rationale of the Research

Recently, the number of research studies related to Open Standards and F/OSS saw an extraordinary growth. The scholarly attention given to the F/OSS phenomenon comes from a variety of disciplines including engineering, social sciences and economics. In addition to that, non-profit organizations as well as governments recognized the potential of the emerging F/OSS approach and dedicated resources in researching and supporting the phenomenon. Increasingly, governments around the globe are encompassing F/OSS and open standards in general with notable examples found in Japan, the United States, Germany, Brazil, Italy, Russia and Peru, among others (Hahn, 2002). Recently major for-profit organizations such as IBM, HP, Amazon.com and Google, among others, recognized the business opportunities F/OSS can generate and invested substantial resources to it which appears as a puzzle since F/OSS is regarded as a type of public good and public goods according to economic theory are not considered to be sustainable (Hardin, 1968).

Even though a lot of research has been conducted in understanding the technicalities of the F/OSS phenomenon, surprisingly little research has been conducted in understanding the formation and evolution mechanisms behind F/OSS collaboration. This can be partially explained by the fact that F/OSS is a highly complex and heterogeneous phenomenon. Several researchers consider F/OSS to be a provision of a public good. According the the economic literature, public goods suffer from market failures such as free-riding. As a result, researchers expected that F/OSS not to be sustainable and to eventually vanish. Contrary to that, F/OSS is

[1]Free/Open Source Software (F/OSS) is a composite term merging "Free Software" and "Open Source Software" together. Both Free and Open Source basically describe similar software development models. The main differences in the two models are their comprising cultures and philosophies. Both models are described in more detail in Chapter 2.

thriving and this fact presents a puzzle to researchers. In addition to that, other characteristics of F/OSS such as the fact that F/OSS is created mainly by volunteer contributions which in turn is made available to the public and its highly intricate institutional structure having the ability to adapt over time made the phenomenon a focus of several research studies (Rossi, 2006).

1.3 Objectives of the Research

The main objective of the research is to investigate open standardization processes via complex social network analysis and learning networks. The pivotal phenomenon that is investigated is the collaboration network of F/OSS development. The main model is build within the complex social network analysis framework and it is used to analyze the formation and evolution of the F/OSS collaboration network. In addition to that, a formal model that investigates learning networks and their applications in engineering, specifically in risk analysis and management, via the Bayesian Neural Network (BNN) framework is also developed. The BNN model is used to evaluate the cost risk of soil decontamination projects and illustrates how complex network analysis utilizing learning networks can be used in practical engineering applications.

1.4 Research Methods

The models of this research are developed within the social network analysis framework utilizing methodologies from the analysis of complex social networks and through learning network methodologies such as the Bayesian Neural Network framework. The main objective is to investigate and model the formation and evolution of open standardization and collaboration. The F/OSS collaboration network is considered as the main paradigm of open standardization collaboration. In order to build the main model, data from the largest open source project hosting community, SourceForge.net, are utilized. Once the basic model is build its parameters are extracted following a methodology similar to the one proposed by Barabási et al. (2002). The research also includes two case studies of scientific collaboration and standardization processes. The research attempts to shed light on what causes the formation and evolution of open standardization processes and to investigate their sustainability.

1.5 Contribution of the Research

The research presented in this dissertation spans several disciplines that include engineering, computer science, statistical physics, economics and law. The research attempts to bridge the gap in the literature of open standardization and scientific collaboration networks by investigating the formation and evolution of the F/OSS collaborators social network. The main research questions presented are: "why open standardization is emerging", "why open standardization phenomena such as F/OSS are currently thriving?" and "are those phenomena sustainable in the long run?". In the course of answering the research questions, the dissertation pivots on the F/OSS phenomenon presents its history, a literature review of related research, analyzes the formation and evolution of the F/OSS collaboration network and investigates the sustainability of the phenomenon through simulations. Finally the dissertation presents two case studies based on open content collaborative projects that were developed during the research.

1.6 Structure of the Dissertation

The dissertation is structured as follows. Chapter 1 gives a brief introduction to Open Standardization, the rationale of the research, its objectives, methodologies and potential contributions to the body of knowledge. Chapter 2 provides an in-depth overview to the main paradigm of open standardization, the F/OSS phenomenon along with its history and reviews the related literature. Chapter 3 develops and presents a formal analytical model of the formation and evolution of F/OSS collaboration via complex social network analysis. Chapter 4 discusses learning and adaptive networks and presents a model of evaluating the cost risk of soil contamination projects via a Bayesian Neural Network framework. Chapter 5 and 6 discuss two case studies related to the research. The first case study discusses the formation and evolution of the Alliance of Global Open Risk Analysis (AGORA), a platform for supporting open standards in risk analysis. The case study uses data gathered from the AGORA website and through questionnaires to analyze how a specialized open collaborative platform such as AGORA evolves over time. The case study utilizes the framework developed in Chapter 3 for the analysis. The second case study discusses the Open Collaboration Book Project (OCBP), an effort to create and disseminate an open academic body of knowledge utilizing open standards. This case study focuses on the challenges encountered during the projects with a focus on institutional problems in adopting open standards and issues related to intellectual property rights. Finally, Chapter 7 presents the conclusions of the dissertation and suggests topics for further research on the topic.

Chapter 2

F/OSS: A Paradigm of Open Standardization

2.1 Introduction

This chapter presents an overview of the F/OSS phenomenon which is considered a paradigm of open standardization. In this chapter F/OSS is first defined, followed by a discussion of its history and a review of research literature related to the phenomenon.

2.2 The Importance of Source Code

In order to clearly understand what the F/OSS approach is trying to achieve it is necessary to first look at how software is developed and to grasp the importance of software's source code. Figure 2.1 shows a typical, simplified work-flow of software development using a high-level structured computer language such as C++ and Table 2.1 gives examples of the development processes (Miller et al., 1996-2007). The first step in creating a piece of software is

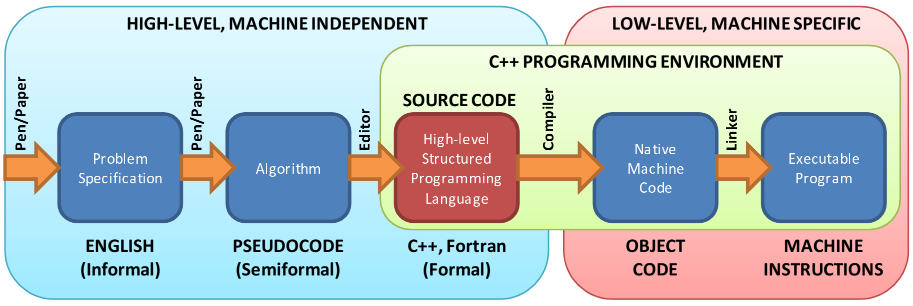


Figure 2.1: A typical simplified work-flow of software development

Table 2.1: Software development work-flow processes

Problem Specification	*Determine if a number is prime*

Algorithm	`input x`
	`for each number z that lies between 1 and x`
	`  if there is no remainder when z divides x`
	`   then output "not prime" and halt`
	`  if no such number can be found`
	`   then output "prime" and halt`
	`loop`

High-level Structured Programming Language	`#include <iostream>`
	`using namespace std;`
	`int main() {`
	`  int x;`
	`  cout << "enter number:\n";`
	`  cin >> x;`
	`  for (int z = 2; z<x; z++)`
	`    if (x % z == 0) {`
	`      cout << "not prime\n";`
	`      return 0;`
	`    }`
	`  cout << "prime\n";`
	`  return 0;`
	`}`

Native machine object code	Binary machine instructions (partial)
Executable	Binary machine instructions (full)

to determine the purpose of the software or the problem it will try to solve. Problem specification is created using natural language. The next step is to create the software's algorithm, again in natural language. The algorithm is then used to create the software's source code, typically written in a high-level structured computer programming language such as C++, Fortran or Java. At this point I would like to point out that the source code is very important as it carries all the necessary knowledge needed to create the final product, i.e. the executable program. This fact makes software a unique creation because a specific component of it, the source code is both the construct and the product at the same time (Bitzer and Schröder, 2006).

2.3 Definition of F/OSS

F/OSS is a composite term merging "Free Software" and "Open Source Software" together. Both Free and Open Source basically describe similar software development standards. The main differences in the two standards are their comprising cultures and philosophies.

2.3.1 Free Software Definition

According to the The Free Software Definition, as published by the Free Software Foundation[1], Free Software is:

> *"... a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech," not as in "free beer." Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:*
>
> - *The freedom to run the program, for any purpose (freedom 0).*
> - *The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.*
> - *The freedom to redistribute copies so you can help your neighbor (freedom 2).*
> - *The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.*
>
> *A program is free software if users have all of these freedoms. Thus, you should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere* (Stallman, 1984b).

2.3.2 Open Source Software Definition

The Open Source Initiative, the organization behind the Open Source methodology, defines Open Source[2]:

> *Open source is a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in* (Perens, 1998).

2.4 The History of F/OSS

In the early 1980's I was still in elementary school when I got my hands on my first computer. It was a used Apple II that my father bought for me. Needless to say that I was very fascinated with my new hi-tech "toy". Initially I started using the computer for fun. I was very fascinated by the games I could play on the computer. Soon my curiosity kicked-in and I wanted to know how the games were created on the computer. I wanted to learn how to create a game on my own. I found out that in order to do so I first had to learn programming. The programming learning resources available to me that time were very limited. I experimented

[1]Free Software Foundation (FSF) is the organization behind the Free Software development model
[2]The full annotated Open Source Definition is presented *ad verbatim* in the Appendices.

with programming on my own for a while but my progress was very slow. In the process of learning I had many questions and got very few answers. I started looking for the answers to my programming questions among my peers. That time, not many of my friends were interested with computers and programming. Those few friends I found to share the same computer interests with me were very excited to be discovered and contacted by me. Soon we created a small informal social network. We then started exchanging our experiences with programming. It was not of great help at the beginning but it was better than nothing. As a group eventually we started sharing knowledge, help each other, exchange information, programming tips and above all we shared our proud software creations. It was natural to us, along with the programs we made, to also share their source code and thus allow everybody to study them, find their bugs[3], fix, improve, remix and finally reuse the source code in new software projects. We had created a small social network and even though it lacked the efficiency and size of today's on-line virtual communities on the Internet, it was something that we were very proud of and above all it served its purpose surprisingly well. We were of course unaware that we had the basic characteristics of the F/OSS methodology in place. We had a network, a peer review process, sharing of knowledge, and we allowed others to examine and modify the source code of our software, find its shortcomings and to improve it. We lacked one very important component that could give our small network a great boost and unleash its true potential. We needed an inexpensive and efficient communication system that would have allowed us to reach more people with the same interests, enhance the peer-review process and to enable us to share our knowledge and creations with the world. For that we had to wait for the emergence and proliferation of the Internet and the World Wide Web.

The roots of F/OSS go back to the early era of electronic computers, in the mid 1940's. The beginning of the F/OSS phenomenon is believed to have started with the publishing of the MEMEX idea of Vannevar Bush, in 1945. According to Benussi (2005), the history of F/OSS can be broken down into 6 periods. Figure 2.2 presents the 6 periods of the F/OSS history along with the most important milestones. The first period, referred to as the era of "the new way of thinking" starts in 1945 with the publishing of the seminal paper of Vannevar Bush and ends in 1968 with the creation of the ARPANET[4]. The most important milestones of the first period

[3]A software bug (or glitch) is a shortcoming in a computer program (i.e an unexpected error, flaw, mistake, failure or fault) that results in erroneous behavior by the program, e.g. producing wrong outputs. Bugs generally are the result of coding mistakes made by the programmers or are caused by inappropriate program design (i.e. following wrong approaches and procedures in coding). The term "bug" goes back to the early times of computers in the late 1940's, when a Mark II computer was producing erroneous results and a moth found in its circuitry was believed to have caused the problem.

[4]The ARPANET (Advanced Research Projects Agency Network) developed by ARPA of the United States

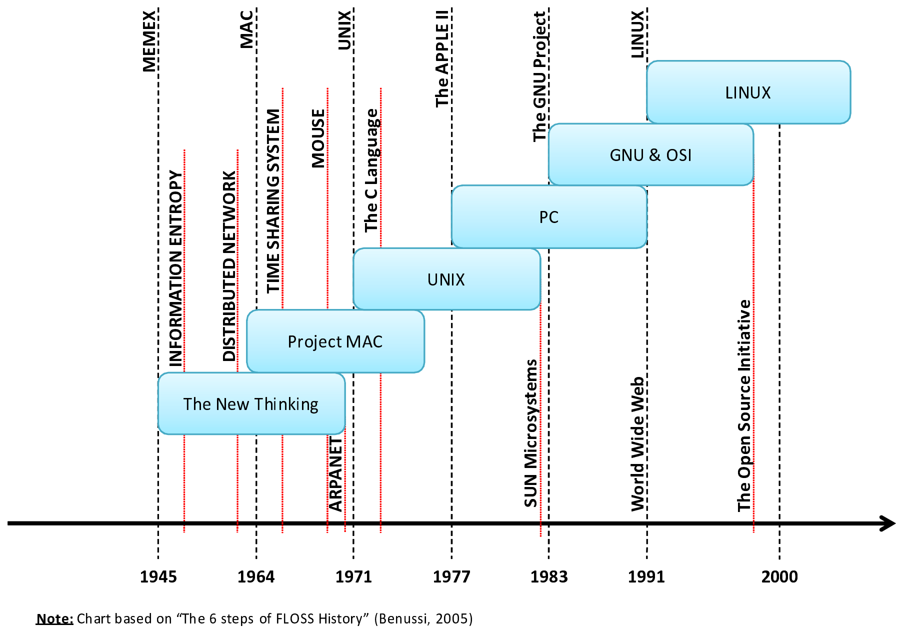


Figure 2.2: F/OSS History Time-line

are the Memex machine, the creations of the National Science Foundation[5] where many new ideas and technologies were created, the topology of a distributed telephone network and the ARPANET project. Several important ideas were generated during the first period including the "information entropy" (Shannon, 1948), "the distributed Network" (Baran, 1964), the "Galactic Network" concept and the computer as a communication device (Licklider, 1960). The main focus of the first period was on dealing with the communication and management of data in an efficient manner by improving and expanding communication technologies.

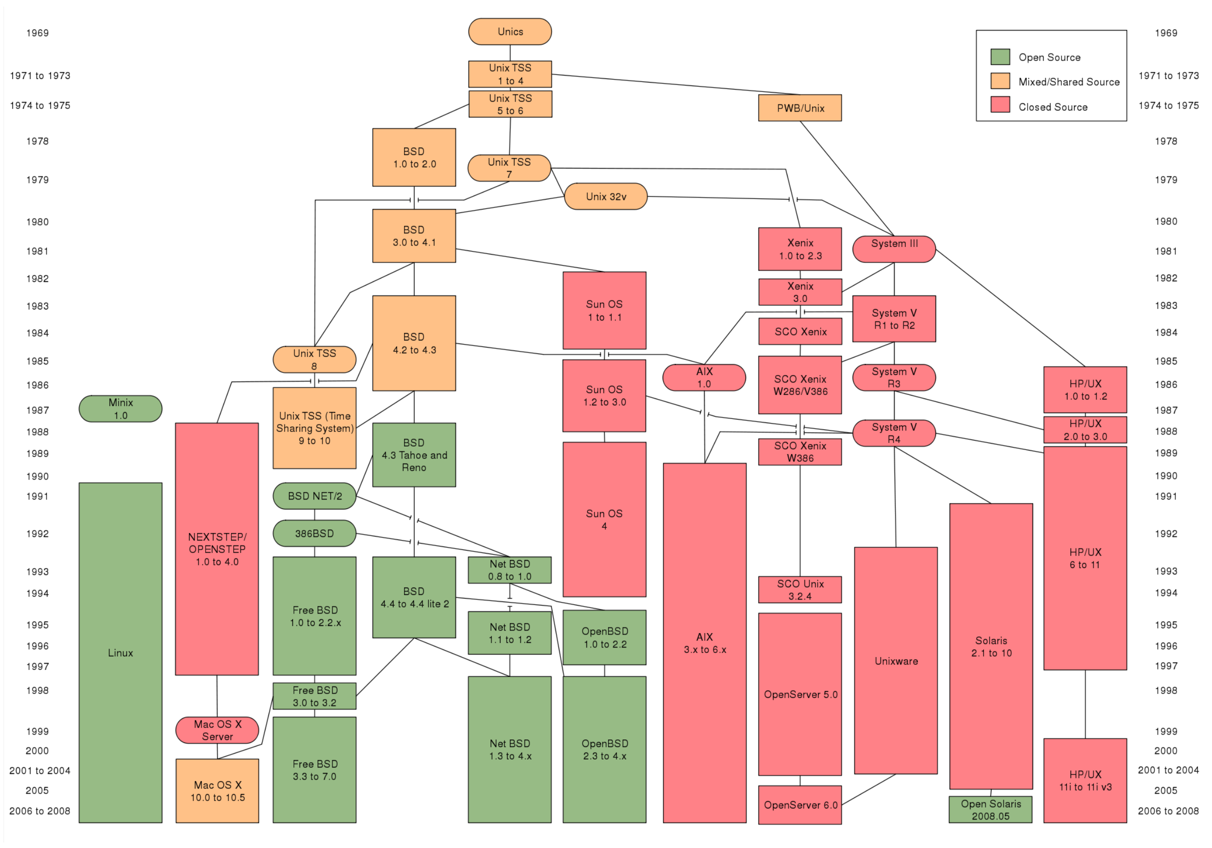
The second period, referred to as the "MAC era", starts in 1963 with the creation of the project MAC[6] and finishes in 1975 with the project's transformation to the Laboratory of Computer Science (LCS). Project MAC is regarded as one of the most innovative, productive and influential projects in the history of computing. During the project's lifespan groundbreaking research in operating systems, artificial intelligence, and the theory of computation was conducted. The main accomplishments of the second period were the creation of the Time-Sharing System (Fano and Corbato, 1966), the On-Line System, Hypertext (Nelson, 1995) and the mouse device (Engelbart and English, 1968). The main concerns of the second period were in creating a

Department of Defense, was the world's first operational packet switching network. It is considered to be the predecessor of the Internet.

[5] "Where Discoveries Begin" is shown proudly on the NSF's website!

[6] Project MAC (the MIT Project on Mathematics and Computation), was a research laboratory at MIT. The acronym "MAC" is derived from "Multiple Access Computer", "Machine Aided Cognition" and "Man And Computer".

modular system able to communicate within a computer network and also a functional, efficient computer to assist researchers in their work. The research work conducted in the second period evolved in the creation of a computer network that eventually was transformed to a public utility. The computer network was a very important infrastructure where the foundation of the F/OSS methodology was created. Also, during the second period the first experiments on the Graphical User Interface technology (GUI) and the conceptualization of hypertext were conducted.



Source: Wikipedia

Figure 2.3: The UNIX history

The third period, referred to as the "the Language era", starts in 1971 with the creation and funding of the UNIX[7] operating system and finishes in 1982 with the foundation of SUN Microsystems. Figure 2.3 shows the history and evolution of UNIX. The creation of UNIX also

[7]The origins of UNIX goes back to 1969. The first implementation was named UNICS, based on the the MULTICS (Multiplexed Information and Computing Service). Ken Thompson, the main developer behind UNIX, after MULTICS was retired by AT&T Bell, continued to develop it. He wrote a game for the new system which he called Space Travel. However, he found that the game was too slow on the system and in addition to that it was expensive to run it (computer time was very expensive that time). He decided to re-write the game in assembly with the help of Dennis Ritchie. His experiences in creating the game, coupled with his work with the MULTICS led in the creation of the UNIX system which was eventually funded by AT&T Bell to become one of the most radical operating systems in the computer history. It seems that I wasn't alone in being motivated by a computer game to learn programming!

coincided with the invention of the C programming language in 1972 by Dennis Ritchie. Both innovations were crucial to the emergence of F/OSS. GNU/Linux, the cornerstone operating system of the F/OSS movement, draws heavily from UNIX, whereas the C language (including its evolution, C++) is the programming language of choice of F/OSS developers. Some important concepts that emerged in the third period are the "pipe" concept, the "way of the hacker" in sharing technological information and the "art of hacking" found in tinkering with hardware and software. The main focus of the third period was in the development of a simple yet powerful and versatile operating system that could be used with different hardware platforms. Notable outcomes of the third period are the programing simplicity principle known as KISS[8], software modularity and portability and the creation of new computer network technologies and communication protocols such as the TCP/IP.

The fourth period dubbed as the "PC era" starts in 1977 with the release of Apple II personal computer and it finishes with the publication of the World Wide Web[9] project by Tim Berners-Lee in 1991. The fourth period is governed by the proliferation of the personal computer. The PC phenomenon created affordable and powerful computers for the masses. Along with the PC came the enhancement of the Graphical User Interface (GUI) technology and a great variety of software such as office applications (spreadsheets, word processors and database programs), and multimedia programs. The last and very important milestone of the fourth period was the Internet that played a pivoting role in the expansion of the F/OSS phenomenon. Some of the concerns of the fourth period that helped in the creation of many innovations include the "plug and play" personal computer that is ready to be used out of the box, the importance and proliferation of software applications and finally the further development and expansion of network computing.

The fifth period started with the creation of the GNU project[10] in 1983 and finished in 1998 with the foundation of the Open Source Initiative[11]. During the fifth period the Free Software Foundation was created and with it Richard Stallman and the proponents of his "free software" notion created a revolution in the software development industry that resulted in

[8]The KISS principle which stands for "Keep It Simple, Stupid" states that design simplicity should be a key goal and unnecessary complexity avoided. The principle closely resembles the Occam's Razor from which most probably was derived. The KISS principle is currently used in many disciplines including software development, engineering, architecture, and planning.

[9]Tim Berners-Lee published the details of the World Wide Web project on the alt.hypertext newsgroup on August 6, 1991, marking the creation of the WWW and the Internet.

[10]The GNU project was publicly announced on September 27, 1983, on the net.unix-wizards and net.usoft newsgroups by Richard Stallman.

[11]The Open Source Initiative was founded in February 1998 by Bruce Perens and Eric S. Raymond with the mission of promoting Open Source software.

innovations that still continue to influence the industry. The pinnacle of the Free Software revolution was the attempt to create a UNIX like operating system, called the GNU (a recursive acronym standing for "Gnu's Not Unix") that would be available along with its source code to everybody, thus providing the freedom to study, learn and modify the operating system to the needs of its users. Richard Stallman and his team started creating one by one the software components of the GNU operating system and by 1991 they had almost completed the operating system, with one major piece still under development. The major software piece needed by the GNU operating system in order to be completed was the kernel[12]. The development of the GNU's kernel, called the HURD, at the time faced major difficulties. On August of the same year, a young software engineer from Finland studying the development of operating systems, posted a message[13] on *comp.os.minix* newsgroup requesting the help of software engineers around the world in developing the kernel of the operating system he created. The name of the young software engineer from Finland was Linus Torvalds and the project he started became the Linux kernel that is used in the GNU/Linux operating system. The Linux kernel was quickly used by software developers around the world together with the GNU software tools in the creation of a full operating system. The creation of the GNU/Linux operation system was a very important milestone of the the fifth period. Bringing together the global community of software developers, researchers and practitioners to voluntarily collaborate in creating a major software innovation and then releasing the innovation back to the community along with the freedom of accessibility and transparency to its methods and its source code, I believe, was the most important contribution of the fifth period. During the fifth period several concerns emerged such as how to protect the freedoms of the F/OSS software and to how support and motivate the F/OSS development community. The former was address by the creation of specific licenses that guard the freedoms of F/OSS software. The General Public License (GPL), featuring the concept of "copyleft", is the most prominent example of such a license. The GPL is the most widely used license for F/OSS today, even though it is criticized as inflexible and "viral".[14] This fact led in the alienation of the efforts of the free software community from the business world. It was argued that "free software" does not present a good business image. As a result the Open Source Initiative was created in 1998 in order to soften the image of "free software" and make it more appealing to the business world (DiBona et al., 1999). With the creation of OSI, several

[12]The kernel is the central component of a computer operating systems. Basically, the kernel kernel connects application software residing on the computer's operating system with the hardware. The kernel is also responsible for the management of the computer's resources.

[13]The full message can be found *ad verbatim* in the appendices

[14]GPL requires the modifications of the original software to be released under the same license conditions.

business models based on the F/OSS methodology were created and proved very successful, such as the notion of "software as a service".

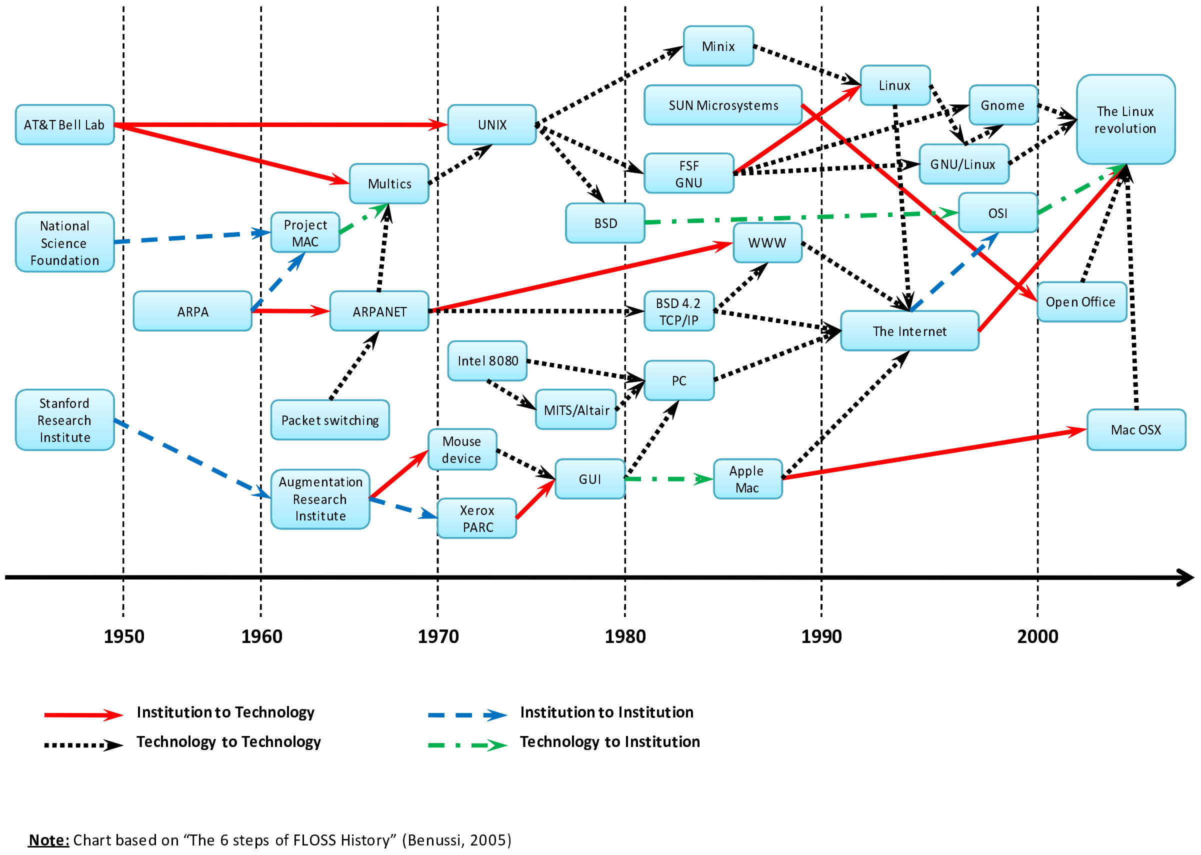


Figure 2.4: Historical Evolution of F/OSS

The sixth period is dubbed the "Linux Revolution". It starts with the announcement of the Linux kernel on the Internet in 1991 by Linus Torvalds. During the sixth period the GNU/Linux operating system saw a remarkable growth and gained market share against incumbent proprietary systems. For example, GNU/Linux is currently used on more that 85% of the supercomputers used in the world today[15]. The GNU/Linux is for sure not the only innovation initiated by the F/OSS phenomenon. Various F/OSS applications that compete directly in quality and features with commercial proprietary software have been created and developed during the sixth period[16]. Various methodologies and business models were also created and the F/OSS model is being accepted in the business world as a valid and sustainable model. Recently the F/OSS methodologies escaped the software industry and entered areas such as the

[15]Data from www.top500.org, accessed on July 2008.

[16]A visit at the sourceforge.net, the largest repository of F/OSS on the Internet will provide an insight of the phenomenon. The sourceforge.net repository currently hosts more than 230,000 F/OSS projects. (Site accessed on July 2009).

automobile, movie, music and publication industries[17]. Figure 2.4 presents the overall historical evolution of F/OSS through the six periods and shows the interconnectivity of emerging technologies.

2.5 Review of related literature

Recently a plethora of papers investigating the F/OSS phenomenon appeared in the scientific literature. The literature related to F/OSS can be split into several categories that include software engineering, economics, business model development, scientific history and social network analysis among others. An excellent review of the history and ideology of the F/OSS phenomenon and the evolution of the Open Source movement is offered by Raymond (2001). Several papers analyze the F/OSS phenomenon in view of market forces and the sustainability of its business models. Bitzer and Schröder (2006) provide an excellent literature review of the F/OSS phenomenon from an economic perspective. Papers in the economic literature focus in explaining the behavior of the programmers/developers in creating and developing F/OSS, analyzing platform wars between proprietary and F/OSS and their effects on free markets and to investigate the collaborating behaviors of F/OSS stakeholders. Lerner and Tirole (2001) identified some major research questions regarding the F/OSS phenomenon by analyzing the incentives of individual programmers and commercial firms to participate in F/OSS projects. In another paper, Lerner and Tirole (2002) investigate the economics of Open Source viewed through the lenses of labor economics and industrial organization theory. The main finding of the paper was that programmers/developers participate in F/OSS projects for various reasons such as to signal their abilities to potential employers and for ego gratification (i.e. seek peer recognition). On the same subject, Raymond (2001) suggests that F/OSS programmers participate in F/OSS projects also for altruistic reasons such as to give back something to their community. He suggests that the F/OSS community resembles a "gift culture" where the members of the community gain recognition based on the quality of the "gifts" they offer back to the community. Another cluster of papers in the economic literature attempt to explain the F/OSS phenomenon as a provision of a public good. Johnson (2002) analyzes the enhancement of a pre-existing F/OSS by presenting a game theoretic model where several software developers decide independently whether or not to develop the software. The paper concludes that the F/OSS development methodology is best suited for software that is already under development. Bitzer and Schröder (2005) also present a

[17]Examples include the Open Source Car (OSCAR) found at www.theoscarproject.org, the "Elephants Dream" which is an Open Source movie found at orange.blender.org and Open Music found at openmusic.linuxtag.org.

game theoretic model of F/OSS development as a provision of a public good. Their paper adds to the knowledge by examining some intrinsic motivations of F/OSS programmers. The paper suggest that F/OSS programmers are motivated by their intrinsic urge to "play" initiated by their urge to solve challenging programming problems and for the fun of programming which is referred to as *homo ludens* behavior in the paper. In another paper, Bramoullé and Kranton (2007) proposed a model used to study the provision of a public good via the network analysis framework. The paper arrives at some very interesting conclusions one being that networks can lead to specialization in public good provision. The paper also concludes that equilibriums exist in every social network where some agents contribute and others free ride and depending on the type of network this extreme is the only outcome. Another important conclusion was that specialization of agents (i.e. agents exhibiting full effort) that are connected to many agents can benefit society as a whole. The paper also concludes that when a new link is added to a not-fully connected (incomplete) network it results in increasing the access to the public good, but at the same time reduces individual incentives to contribute. For that reason the paper also concludes that the overall welfare can be higher when the network is incomplete (i.e. there are structural holes in a network).

Analysis of the F/OSS phenomenon via the social network analysis framework is rather new with a limited but growing body of literature. A reason for the relatively small number of publications in that area is the computational difficulties involved in analyzing large complex social networks such the F/OSS collaboration network. For example, the social network of developers and F/OSS projects of the SourceForge Community has more than 2,000,000 nodes and more than 230,000 clusters (projects)[18]. Analysis of such large scale networks is very computationally expensive. Recently, there is strong interest in analyzing such large complex social networks. The fact that many phenomena in engineering, biology, physics, medicine and social science can explained and analyzed as network based phenomena enabled researchers from many disciplines to get involved and bring fresh ideas, methodologies and findings in the field. One of the most powerful contributions in the study of complex social networks came from the statistical physics discipline. Seminal works by Newman (2001), Barabási and Albert (1999), Albert et al. (1999) and Faloutsos et al. (1999) that explored massive complex networks such as the Internet, the World Wide Web and social networks of scientific collaborations presented new methodologies that allow complex social network analysis that was not feasible earlier to be conducted. This research utilizes such complex social network analysis methodologies.

[18]Data from SourceForge; accessed on July 2009.

Chapter 3

Open Standardization Formation and Evolution Model

3.1 Research Fundamentals

3.1.1 Social Network Analysis

Social network analysis (SNA) is defined as the study of social relationships between agents in a social construct. The main focus is on analyzing the relationships among agents, and the patterns and implications of these relationships. In the words of Wasserman and Faust (1994):

> *The social network perspective encompasses theories, models, and applications that are expressed in terms of relational concepts or processes. That is, relations defined by linkages among units are a fundamental component of network theories.*

Traditional social science methods differ from those utilized in social network analysis. One of the main differences is the focus of SNA on the network as the unit of analysis instead on the agent (Scott, 2000). Quantitative social science studies typically collect and analyze attributes and opinions of agents. On the other hand, social network analysis focuses on the examination of relationships of the agents; SNA investigates and identifies the roles, positions, and groups in a network (Haythornthwaite, 1996). Recently, the SNA field has grown rapidly and is being used in various disciplines to analyze social and behavioral problems and to quantify political, economic, or social structural environments. Wasserman and Faust (1994) define four main tenets of social network analysis:

- *Actors and their actions are viewed as interdependent rather than independent,* *autonomous units. Human behavior is embedded in networks of interpersonal relationships.*

- *Relational ties (linkages) between actors are channels for transfer or "flow" of resources (either material or non material). Network connections constitute social capital, and rich and well-structured networks can provide high levels of social capital to actors within them.*
- *Network models focusing on individuals view the network structural environment as providing opportunities for, or constraints on, individual action*
- *Network models conceptualize structure (social, economic, political, and so forth) as lasting patterns of relations among actors*

Classical applications of SNA can be found in the investigation of kinship patterns, community structures and interlocking directorships (Scott, 2000). Recently, social network analysis techniques are increasingly being applied to new and diverse applications in fields such as engineering and computer science (Carrington et al., 2005). They cover topics ranging from health and the spread of infectious diseases to international economic growth and the topology of the Internet (Faloutsos et al., 1999). For a comprehensive list of applications of SNA see Wasserman and Faust (1994).

The study of social networks emerged at the end of the 18th century with the work of Durkheim and Tönnies who argued that social structures (i.e. cliques and groups) exist as in the form of personal social ties that either link people who share similar values and beliefs or impersonal, formal and instrumental social links. In the beginning of the 20th century, Simmel is considered the first scholar to publish work directly utilizing social network terminology. Simmel's research focused mainly on the nature of networks, their sizes and on the likelihood of interaction in ramified, loosely-knit networks rather than groups. SNA as we know it today is a merge of three research efforts that occurred during the 20th century. The first effort started in the early 1930's in Europe with a group of German researchers influenced by the "gestalt" theory of Wolfgang Köhler. In the 1930's, Moreno's systematic recording and analysis of social interaction in small groups (referred to as "sociometry"), and Warner's and Mayo's groups at Harvard that explored interpersonal relations at work environments set the first foundations for a systematic study of social networks. During the same period, a group of researchers at Harvard developed the ideas of British anthropologist Radcliffe-Brown to study the flow of information within groups. Their studies focused on the investigation of patterns of interpersonal relations and the formation of informal sub-groups, i.e. cliques, within large social constructs. The third group consisted of researchers at the department of anthropology of Manchester University. The Manchester researchers also developed the ideas of Radcliffe-Brown to analyze conflict and contradiction within groups. The work of the three groups was finally merged in the 1960's at Harvard by Harrison White and his group of researchers that included Ivan Chase, Bonnie Erickson, Harriet Friedmann, Mark Granovetter, Nancy Howell, Joel Levine, Nicholas Mullins, John Padgett, Michael Schwartz and Barry Wellman. Other important researchers in the group were Charles Tilly, who focused on networks in political sociology and social movements and Stanley Milgram widely known for his "six degrees of separation" thesis. White's researchers extended the previous work building on mathematical frameworks such as graph theory. The result was the new formal analytical framework of social network analysis. Ever since the field saw a remarkable growth, especially in the last 15 years with the emerge of powerful computing methods that enable analysis of large complex networks that was previously infeasible. This new field of social complex network analysis is discussed in the following section. For a detailed presentation of the history of social networks and SNA the reader is directed to Freeman (2004).

Social Network Analysis Methodologies and Data

The methodologies used in SNA can be categorized into two major groups, ego network analysis and complete network analysis. The two groups are distinguished from the type of data they utilize. Ego network analysis utilizes data from traditional surveys and questionnaires. For example, in a survey participants are asked about the people they personally know and the relationships with them. In ego network analysis such data are typically sampled randomly from large populations. Typically, the analysis of ego networks involves assessing the networks qualitatively. Assessments includes the calculation of the network's size and its node diversity. Ego network analysis is less demanding and less complicated as compared with complete network analysis. On the other hand, complete network analysis attempts to capture all the relationships among a set of agents, such as friendships or club memberships of the agents. The majority of studies in social network analysis utilize complete network analysis methodologies.

A characteristic of SNA that distinguishes it from other sciences is how data are constituted. Social science data are linked to cultural values and symbols. In contrast to physical data used in natural sciences, the data used in SNA stem from meanings, motives, definitions and typifications. Scott (2000) defines three types of data, attribute data, relational data and ideational data:

- *Attribute data relate to the attitudes, opinions, and behaviors of agents, in so far as these are regarded as the properties, qualities or characteristics that belong to them as individuals or groups*

- *Relational data are the contacts, ties and connections, the group attachments and meetings, which relate one agent to another and so cannot be reduced to the properties of the individual agents themselves. Relations are not the properties of the agents, but of systems of agents; these relations connect pairs of agents into larger relational systems*
- *Ideational data describe the meanings, motives, definitions and typifications themselves*

In general, SNA is best suited for relational data. Variable analysis is used for attribute data and typological analysis is used for ideational data. Figure 3.1, adopted from Scott (2000), summarizes the methods and their respective data.

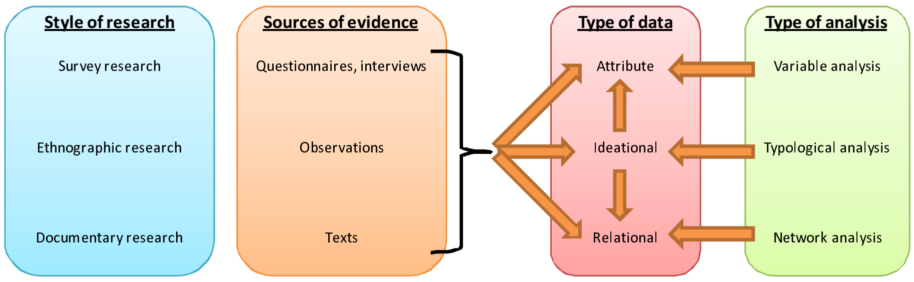


Figure 3.1: Types of social science data and analyses

Social Network Analysis Terminology

Data used in SNA are typically organized in matrix form stored in databases. Fundamental features of social networks are typically analyzed through the direct manipulation of those matrices. As a result, SNA adopts methodologies from matrix algebra. In addition to that methodologies from graph theory are also adopted. Graph theory provides the formal framework for describing and analyzing the networks and their features. In addition to that, SNA also adopts methodologies from probability theory. Following are some of the most common terms used in SNA along with a brief descriptions of their meanings and usage.

Incidence Matrix A representation of the affiliations of agents in a network. Assume we have relational data about software projects and their programmer contributors. An incidence matrix could show for example which programmer belongs to what software project by listing the programmers as the column entries, the software projects as the row entries and contains ones where affiliations exist or zeros where they don't. Incidence matrices are typically rectangular. An example of an incidence matrix can be seen at Figure 3.2.

	Programmer 1	Programmer 2	Programmer 3	Programmer 4	Programmer 5
Project 1	0	1	1	0	0
Project 2	1	1	0	1	0
Project 3	0	0	1	1	1
Project 4	1	0	1	1	1

Figure 3.2: Example of an incidence matrix

Adjacency matrix A representation of the ties among agents in a network. An adjacency matrix could show for example which programmers are linked together via participation in the same project, i.e. two programmers are linked if they belong to the same software project. Adjacency matrices are always square.

Sociograms Graphical representations of a social network. A sociogram could show the agents as nodes (vertices) and the ties among them as links (edges). Links can be unidirectional or bi-directional. In addition to that, links can be valued, i.e. have weights. Graphs having unidirectional links are also referred to as directed graphs whereas those having bi-directional links are referred to as undirected graphs. Adjacency matrices of undirected networks are always symmetric.

Adjacent Nodes Any two nodes that are connected.

Neighborhood For a given node, the set of its adjacent nodes is referred to as its neighborhood.

Degree (of Connection) For a specific node, the count of the number of nodes of its neighborhood is referred to as the degree of the node. This is also known as the "geodesic distance".

Degree Distribution Given a network, its degree distribution shows the fraction of nodes having degree κ where $\kappa = 0, 1, \ldots, n-1$; $n =$ total number of nodes. Alternatively, the degree distribution can be understood as the probability that a randomly selected node has degree κ.

Component Typically within a social network sub-groupings exist. SNA attempts to discover and analyze any sub-groups (sub-graphs) that might exist in the network. In this context, a component is defined as a maximal connected sub-graph. A sub-graph is an isolated graph within the main network where all of its points are linked to one another through paths. In a maximal connected graph, it is impossible to add any new members without destroying the quality of connectedness.

Diameter The diameter of a graph is defined as the largest (geodesic) distance between any pairs of its points.

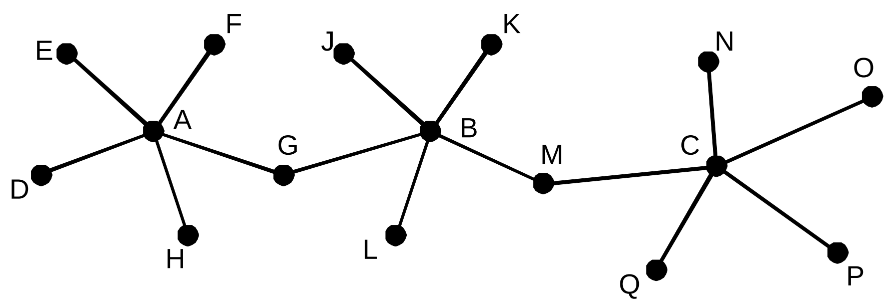
Clustering For a given node that has at least 2 neighbors, its clustering is defined as the fraction of pairs of neighbors of that node that are themselves also neighbors. In other words, clustering measures the likelihood that two neighbors of a node are neighbors themselves.

Cohesion A measure applicable to arbitrary subsets of nodes. A subset is defined to have high cohesiveness if its nodes have low proportions of connections with the nodes outside of the subset. Cohesion measures in a way how a subset is “shielded” from external influence.

Centrality A measure of the importance of a node in providing connectivity (bridging) to other nodes in the network. Typical measures of centrality are the betweenness, closeness and degree of the network. Scott (2000), defines two types of centrality, local and global. Local centrality is simply the degree of a node. Nodes with high degrees have high local centrality. On the other hand, global centrality is defined in terms of distances. Distance is defined as the length of the shortest path between two points. Global centrality of a node is defined as the sum of all distances of that node to the rest of the nodes of the network. A node having the smallest sum of distances is considered to be globally central. In other words, a node is globally central if it lies at short distances from the other nodes of the network. Figure 3.3, adopted from Scott (2000), shows an example of local and global centrality for a small social network.

Betweenness A measure of the extend to which a particular node lies between other nodes. A node with high betweenness acts as an “intermediary” that facilitates connection to other nodes in the network.

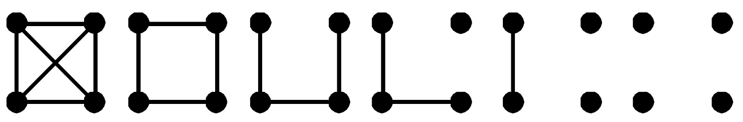
Closeness A measure of how close a node is to the rest of the nodes in the network. Closeness is defined as the inverse of the sum of the shortest distances between each node and every other node in the network.



		A,C	B	G,M	J,K,L	All other nodes
Local Centrality	Absolute	5	5	2	1	1
	Relative	0.33	0.33	0.13	0.07	0.07
Global Centrality		43	33	37	48	57

Figure 3.3: Local and global centrality example

Density A measure of the completeness of a network, i.e. the extend to which all possible relations are actually present. Various densities of a simple network can be seen in Figure 3.4, adopted from Scott (2000). The density of an undirected graph is given by $\frac{2l}{n(n-1)}$, the ratio of



No. of connected nodes	4	4	4	3	2	0
Inclusiveness	1.0	1.0	1.0	0.7	0.5	0
Sum of degrees	12	8	6	4	2	0
No. of lines	6	4	3	2	1	0
Density	1.0	0.7	0.5	0.3	0.1	0

Figure 3.4: Densities for a simple network

the number of links a graph has l to the maximum possible links that graph can have $\frac{n(n-1)}{2}$.

Structural Hole A structural hole exists in a network when two nodes are connected at distance 2 but not at distance 1. The existence of structural holes in a network allows opportunities for nodes to become intermediaries (bridges). Bramoullé and Kranton (2007) show that social networks with structural holes have higher social welfare than complete networks.

3.1.2 Complex Social Network Analysis

The field of complex social network analysis came into existence during the last decade due to the great interest that was generated in analyzing phenomena that can be understood as large complex networks and the availability of large open data-sets that are easily accessible by researchers, via the Internet for example. Analyzing complex social networks in the past was difficult due to the high computational costs involved in the analysis as well as the lack of analytical frameworks and methodologies suitable for large complex network analysis and the lack of easily accessible data-sets. Recently, with the advert of cheap powerful computer resources, the emergence of social network applications and scientific collaboration projects on the Internet that store social network data in open databases and the development of suitable analytical frameworks and methodologies, the field saw a remarkable growth. In addition to social phenomena, many physical and biological phenomena can also be analyzed via complex social network analysis. The gamut of phenomena that can analyzed via complex social network analysis include transportation networks, the physical Internet network, the virtual WWW network, information flow networks such as scientific publication citation and patent networks, molecular process such as metabolic reactions (Jeong et al., 2000), gene regulation (Kauffman, 1993), the folding of proteins (Scala et al., 2001) and collaboration networks such as the F/OSS developers community. A very interesting application utilizing the framework of complex social network analysis is found in the field of biology. Research conducted by White et al. (1986) successfully mapped completely the neural network of a living organism for the first time. The realization that a multitude of phenomena can be analyzed via the complex social network analysis framework attracted researchers from various disciplines to join the growing field of complex social network analysis. Their contributions were invaluable in causing the field to rapidly expand. The contributions from the field of statistical physics are undoubtedly among the most important in helping to establish a formal analytical framework for the study of complex social network analysis.

Complex Social Network Analysis Fundamentals

This section provides an overview of the basic theory of complex social networks. In this research we adopt the terminology and notation as suggested by Vega-Redondo (2007).

Definition of Network A network $\Gamma = (N, L)$ is defined by a set of nodes $N = \{1, 2, \ldots, n\}$ and a set of links $L \subseteq N \times N$. In this research it is assumed that the links are unweighted and that two nodes can be connected by only one link, i.e. no multiple links between two nodes exist. The network is represented by its adjacency matrix (see previous section), denoted by A. The adjacency matrix A has elements

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in L \\ 0 & \text{otherwise} \end{cases}$$

It is assumed that the network is undirected, i.e. the adjacency matrix is symmetric about its diagonal. This means that if $(i,j) \in L$ then $(j,i) \in L$. Another assumption is that the network has no reflexive links, i.e. $(i,i) \notin L$ for each $i \in N$. The neighborhood of a node $i \in N$ is given by $\mathcal{N}^i \equiv \{j \in N : ij \in L\}$. Hence a network $\Gamma = (N, L)$ can be described also by the set of the neighborhoods of all of its nodes, $\{\mathcal{N}^i\}_{i \in N}$.

Random Networks Random networks, originally proposed by Erdös and Rényi (1959), refer to probabilistic constructs rather than representations of actual networks. A random network consists of a family $\mathcal{G}$ of possible networks and a pdf P specifying an *ex-ante* probability $P(\Gamma)$ with which each network $\Gamma \in \mathcal{G}$ is selected. An example of such a random network, as suggested by Erdös and Rényi (1959) is as follows. Assume a fixed set of nodes $N = \{1, 2, \ldots, n\}$ and a fixed number of links ℓ, that can be used to connect the nodes. The set $\mathcal{G}$ is given by the collection of all possible undirected networks that can be constructed by connecting the nodes in N with ℓ links. Every network in the set $\mathcal{G}$ can be chosen with uniform probability. Hence, there are $\binom{n(n-1)/2}{\ell}$ networks that can be chosen with probability $P(\Gamma) = \binom{n(n-1)/2}{\ell}^{-1}$ for every $\Gamma \in \mathcal{G}$.

Connectivity For a given node i, its degree (defined in the previous section) is given by

$$z^i \equiv \left|\mathcal{N}^i\right| = |\{j \in N : ij \in L\}| \tag{3.1}$$

In this research were are not concerned with higher order neighborhoods and degrees, so when we refer to the neighborhood of a node we specifically mean the first order neighborhood, i.e. the set of nodes that are at a geodesic distance of 1. For a given network Γ we define its degree distribution $p(\kappa)$ as the fraction of nodes that have degree κ, where $\kappa = 0, 1, \ldots, n-1$. The degree distribution is given by

$$p(\kappa) = \frac{1}{n}\left|\{i \in N : z^i = \kappa\}\right| \tag{3.2}$$

As it was mentioned in the previous section, the degree distribution of a network also expresses the probability that a randomly selected node has degree κ. In that way, networks can be expressed as statistical constructs defined by the set of possible networks $\mathcal{G}$ and a degree distribution. There are several types of degree distributions used to describe random and real networks. Those include Binomial, Poisson, Geometric, and Power-law distributions. Of interest to this research is the Power-law distribution, having density

$$p(\kappa) = A\kappa^{-\gamma} \qquad (\kappa = 1, 2, \ldots) \tag{3.3}$$

where $\gamma > 1$ governs the rate at which the probability decays with degree and $A = 1/\mathcal{R}(\gamma)$ where

$$\mathcal{R}(\gamma) \equiv \sum_{\kappa=1}^{\infty} \kappa^{-\gamma} \tag{3.4}$$

defines the Riemann Zeta function that normalizes the distribution. Power-law degree distributions are considered scale-free, having $p(\alpha\kappa) = \alpha^{-\gamma}p(\kappa)$ for any κ and α. In this study we will show that F/OSS collaboration networks are scale-free social networks governed by preferential attachment, similar to scientific collaboration social networks studied by (Barabási et al., 2002).

Components Components are maximal subsets of nodes $\chi \subset N$ such that for every pair of nodes $i, j \in \chi$ there exists a path that connects them. Scale-free networks contain what are termed "giant components" and sometimes the whole network has just one component. A giant component χ^o with relative size $|\chi^o|/n$, i.e. the number of the nodes it includes divided by the total number of nodes of the network, remains bounded above zero as the number of nodes $n \to \infty$ (Vega-Redondo, 2007).

Geodesic Distances For a given network $\Gamma = (N, L)$ the average geodesic distance $d(i, j)$ between a pair of nodes i, j is defined as the minimum number of links that need to pass through to reach node i from node j. If no such path exists then $d(i, j) = +\infty$. The average network distance $\bar{d}$ is computed as follows. Assuming there are paths connecting any pair of nodes of the network, we define the distribution of node pairs at distance r by

$$\varpi(r) = \frac{|\{(i, j) \in N \times N : d(i, j) = r\}|}{n(n-1)}$$

and $\sum_{r>0} \varpi(r) = 1$. The average network distance is then given by

$$\bar{d} = \sum_{0<r<\infty} r\varpi(r) \tag{3.5}$$

and the diameter of the network, i.e. the maximum geodesic distance among any node pair of the network, is given by

$$\hat{d} = \max\{r : \varpi(r) > 0\} \tag{3.6}$$

Clustering Given a network $\Gamma = (N, L)$, for each node i that has at least two neighbors, i.e. $z^i \geq 2$, the clustering of the node $\mathcal{C}^i$ is defined as the fraction of pairs of neighbors of i that are themselves neighbors. Clustering can be calculated by

$$\mathcal{C}^i \equiv \frac{|\{jk \in L : ij \in L \wedge ik \in L\}|}{\frac{z^i(z^i-1)}{2}} \tag{3.7}$$

or in terms of the adjacency matrix $A = (a_{ij})_{i,j=1}^n$ is given by

$$\mathcal{C}^i = \frac{\sum_{j<k} a_{ij} a_{ik} a_{jk}}{\sum_{j<k} a_{ij} a_{ik}} \tag{3.8}$$

For a node j with $z^j < 2$ its clustering is $\mathcal{C}^j = 0$. The clustering of the whole network is given by

$$\mathcal{C} = \frac{1}{n} \sum_{i=1}^{n} \mathcal{C}^i \tag{3.9}$$

Alternatively the clustering can be calculated by

$$\tilde{\mathcal{C}} = \frac{\sum_{i<j<k} a_{ij} a_{ik} a_{jk}}{\sum_{i<j<k} a_{ij} a_{ik}} \tag{3.10}$$

Cohesiveness Given a network $\Gamma = (N, L)$, let $M \subset N$ be a subset of the network's nodes. For each node $i \in M$ the fraction of its connections that are within M is given by

$$\mathcal{H}^i(M) = \frac{|\{ij \in L : j \in M\}|}{z^i} \tag{3.11}$$

The overall cohesiveness of M, defined as the minimum fraction across all nodes, is given by

$$\mathcal{H}(M) = \min_{i \in M} \mathcal{H}^i(M) \tag{3.12}$$

Centrality and Betweenness Assuming the network is connected, we consider all the shortest paths joining any two nodes j and k where $j \neq k$ and denote $\nu(j,k)$ their total number. Let $\nu^i(j,k)$ denote the number of those paths that also connect node i where $(i \neq j \neq k \neq i)$. The betweenness of node i is then given by

$$b^i \equiv \sum_{j \neq k} \frac{\nu^i(j,k)}{\nu(j,k)} \tag{3.13}$$

where $\nu^i(i,k) = \nu^i(j,i) = 0$. Aggregating the betweenness over all nodes we get

$$\sum_{i=1}^{n} b^i = n(n-1)(\bar{d} - 1) \tag{3.14}$$

3.2 Model Formulation

3.2.1 The SourceForge.net Dataset

SourceForge.net is currently the world's largest on-line repository of F/OSS projects with more than 230,000 projects and 2,000,000 registered collaborators. SourceForge.net, established in 1999, provides free hosting and other services such as version control and forums to F/OSS developers and users (Maguire, 2007). All the data regarding the registered F/OSS projects and their contributors on SourceForge.net are stored in a large relational database residing at the servers of SourceForge.net. True to the spirit of F/OSS, SourceForge.net makes the data available to researchers. Currently, there are two major research projects that disseminate the SourceForge.net data to researchers, FLOSSMole (Howison et al., 2006) and the SourceForge Research Data Archive (Van Antwerp and Madey, 2008).

The FLOSSMole project was created to freely provide data about open source projects in multiple formats for anyone to download, to integrate F/OSS related donated data from other research teams, to provide tools for researchers to gather their own data and, finally, to provide a community for researchers to discuss public data about open source software development. FLOSSMole utilizes special data mining programs (web spiders) that gather data by accessing the websites of SourceForge.net and then store them in databases at the project's servers. Currently, in the FLOSSMole databases there are about 300 GB of data covering the period 2004 to the present. For this research we initially considered using the data from FLOSSMole but we abandon the effort for the following reasons. First, because the FLOSSMole data were extracted using web spidering operations are not consistent over time and often data important for the analysis are missing from the data-set. Second, the FLOSSMole data-set does not contain data about the dates projects were registered on SourceForge.net which are necessary to study the evolution of the complex social network of the SourceForge.net collaboration. In this research, among others, we aim to study the formation and evolutions of F/OSS collaboration and in order to do so the full data-set is necessary. For those reasons the data from the FLOSSMole project could not be utilized.

The SourceForge Research Data Archive (SRDA) project is an NSF funded project that was initiated at the University of Notre Dame in order to support academic and scholarly research on the F/OSS phenomenon. The SRDA project acquired permission from SourceForge.net to process and disseminate its data to academic researchers studying the F/OSS phenomenon. Unlike the data from the FLOSSMole project, the SRDA data are the actual data from Source-

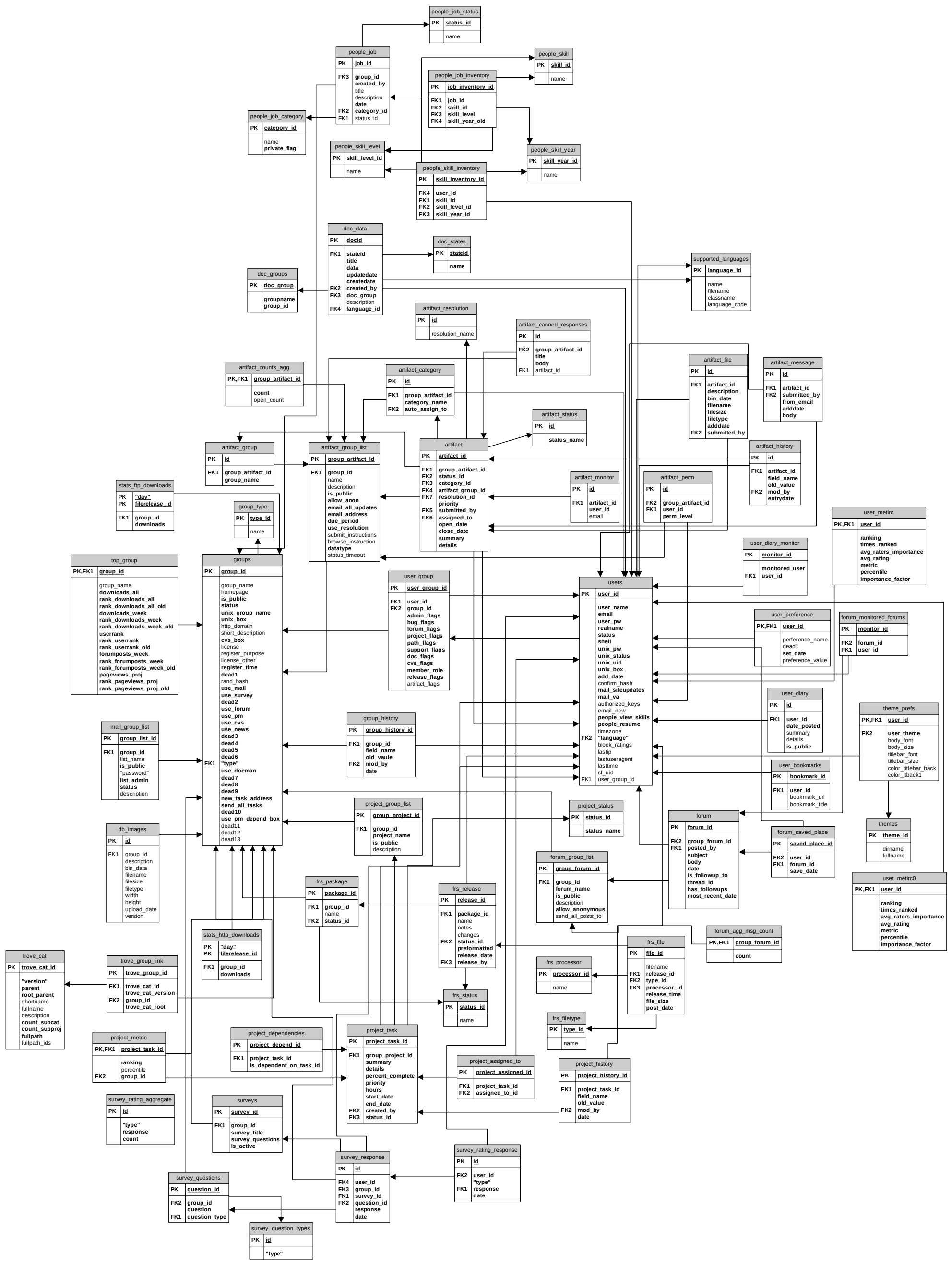
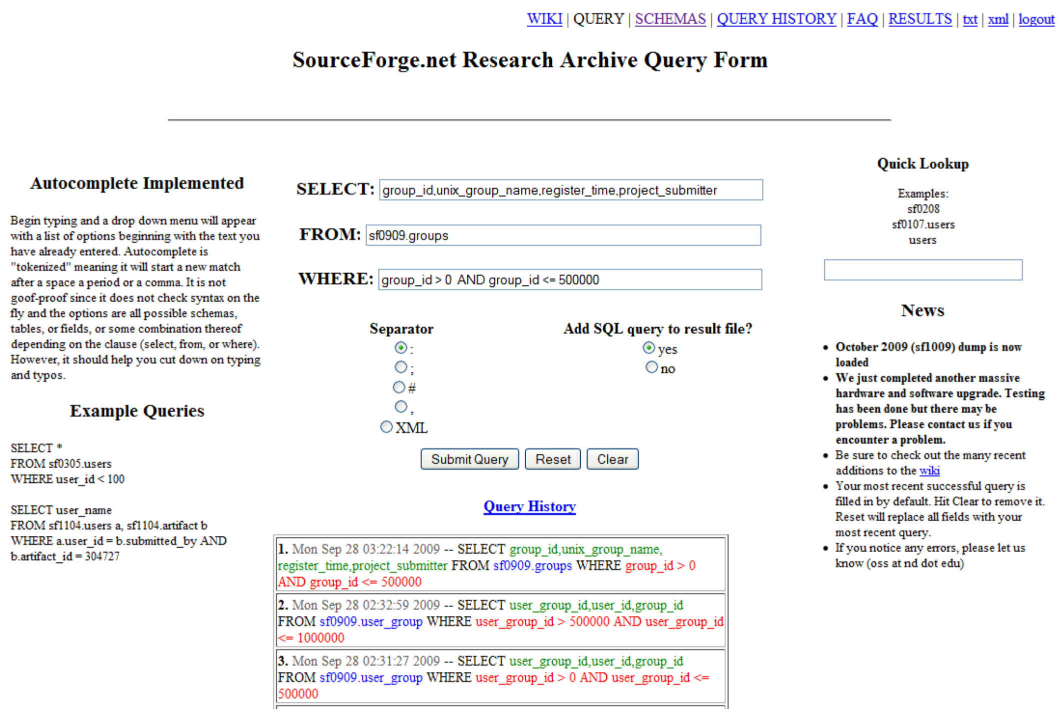


Figure 3.5: The structure of the SRDA Database

Forge.net databases. For privacy reasons some data such as the email addresses of the developers are removed from the SRDA data-set. In order to get access to the SRDA data-set, researchers have to request access contacting the administrators of the SRDA project, prove that they are eligible for access (only academic and scholarly researchers are eligible to receive the data) and sign an agreement. The SRDA data are stored in a relational database (PostgreSQL) which can be accessed via a web interface using a web browser. The database has over 100 relational tables as shown in Figure 3.5. Every month, a complete dump of the SourceForge.net databases


WIKI | QUERY | SCHEMAS | QUERY HISTORY | FAQ | RESULTS | txt | xml | logout

SourceForge.net Research Archive Query Form

Autocomplete Implemented

Begin typing and a drop down menu will appear with a list of options beginning with the text you have already entered. Autocomplete is "tokenized" meaning it will start a new match after a space a period or a comma. It is not goof-proof since it does not check syntax on the fly and the options are all possible schemas, tables, or fields, or some combination thereof depending on the clause (select, from, or where). However, it should help you cut down on typing and typos.

Example Queries

```
SELECT *
FROM sf0305.users
WHERE user_id < 100

SELECT user_name
FROM sf1104.users a, sf1104.artifact b
WHERE a.user_id = b.submitted_by AND
b.artifact_id = 304727
```

SELECT:	group_id,unix_group_name,register_time,project_submitter
FROM:	sf0909.groups
WHERE:	group_id > 0 AND group_id <= 500000

Separator: [✓ :] ; # , XML

Add SQL query to result file? [✓ yes] no

Query History

1. Mon Sep 28 03:22:14 2009 -- SELECT group_id,unix_group_name, register_time,project_submitter FROM sf0909.groups WHERE group_id > 0 AND group_id <= 500000
2. Mon Sep 28 02:32:59 2009 -- SELECT user_group_id,user_id,group_id FROM sf0909.user_group WHERE user_group_id > 500000 AND user_group_id <= 1000000
3. Mon Sep 28 02:31:27 2009 -- SELECT user_group_id,user_id,group_id FROM sf0909.user_group WHERE user_group_id > 0 AND user_group_id <= 500000

Quick Lookup

Examples:
sf0208
sf0107.users
users

News

- **October 2009 (sf1009) dump is now loaded**
- **We just completed another massive hardware and software upgrade. Testing has been done but there may be problems. Please contact us if you encounter a problem.**
- Be sure to check out the many recent additions to the wiki
- Your most recent successful query is filled in by default. Hit Clear to remove it. Reset will replace all fields with your most recent query.
- If you notice any errors, please let us know (oss at nd dot edu)

Figure 3.6: The SQL query web interface of the SRDA Project

(minus the data dropped for privacy and security reasons) is delivered to the SRDA project. The SRDA project created a data warehouse comprised of these monthly dumps, with each stored in a separate database schema. Thus, each monthly dump is a snapshot of the status of all the SourceForge.net projects at that point in time. The data warehouse is currently estimated to be 1.3 TBytes[1] in size, and is growing at about 25 GBytes per month. Queries across the monthly schema may be used to discover when changes took place, to estimate trends in project activity and participation, or even if no activity, events or changes have taken place. The SRDA database contains the following types of data:

- Project sizes over time (number of developers as a function of time presented as a frequency distribution)
- Development participation on projects (number of projects individual developers participate on presented as a frequency distribution)
  - The above two items can be used to create a "collaboration social-network"

[1]The estimation is based on data about the size of the data warehouse in 2007, provided at the website of the project.

  – The above two items can be used to discover scale-free distributions among developer activity and project activity

- The extended-community size around each project including project developers plus registered members who participated in any way on a project (discussion forum posting, bug report, patch submission, etc.)
- Date of project creation (at SourceForge.net)
- Date of first software release for a project
- SourceForge.net ranking of projects at various times
- Activity statistics on projects at various times
- Number of projects in various software categories, e.g., games, communications, database, security, etc.

In order to access the SRDA data, the project provides web access via a wiki where SQL queries can be executed, as shown in Figure 3.6. The user, when logged in, can select the type of the query output among plain text or XML formatted text. Figure 3.7 shows an example of the XML output of a query.

```
<root query="SELECT * FROM sf0909.users WHERE user_id=1457550">
  <row>
    <user_id>1457550</user_id>
    <user_name>"chrmina"</user_name>
    <realname>"Christakis Mina"</realname>
    <status>"A"</status>
    <unix_uid>198694</unix_uid>
    <add_date>1140515760</add_date>
    <people_resume>"My Resume Goes HERE"</people_resume>
    <timezone>"Asia/Tokyo"</timezone>
    <language>275</language>
    <stay_anon>0</stay_anon>
    <donation_request />
    <donate_optin>0</donate_optin>
    <last_sitestatus_view>0</last_sitestatus_view>
    <row_modtime>1245225250</row_modtime>
  </row>
</root>
```

Figure 3.7: Example of SQL query output from the SRDA Project

3.2.2 Data Analysis

The first step in constructing the complex social network of the SourceForge.net collaboration was to setup a database on a local server to store the data necessary for the analysis. For that purpose we setup a GNU/Linux server and installed the MySQL relational database system on it. We then created a database schema containing three tables, "users", "groups" and "users_groups". The "users" table was designed to hold the data about the registered users of SourceForge.net. For our analysis we needed only 3 types of data, the user id, the user name and the date when the user was registered at SourceForge.net. The "groups" table was designed to hold the data about the software projects hosted at SourceForge.net. Similar to the "users" table, for the analysis we needed only 3 types of data, the group id, the group name and the date when the project was registered at SourceForge.net. Finally, the "users_groups" table is the join table for the two tables that holds the affiliation information, i.e. which user belongs to what project. In relational database jargon, the join table is used to create a "many-to-many" relation among the "users" and "groups" tables. A "many-to-many" relationship in this case means that a user can belong to many projects and a project can have many users. Figure 3.8, shows the tables of SRDA that were used to create the local database schema.

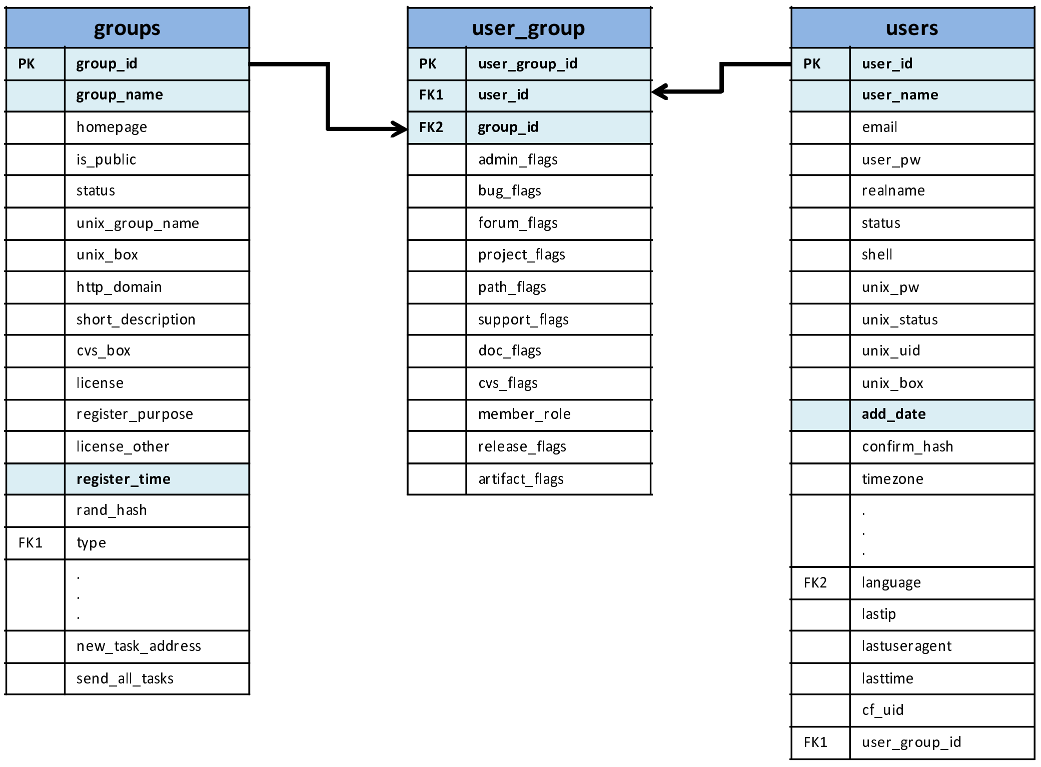
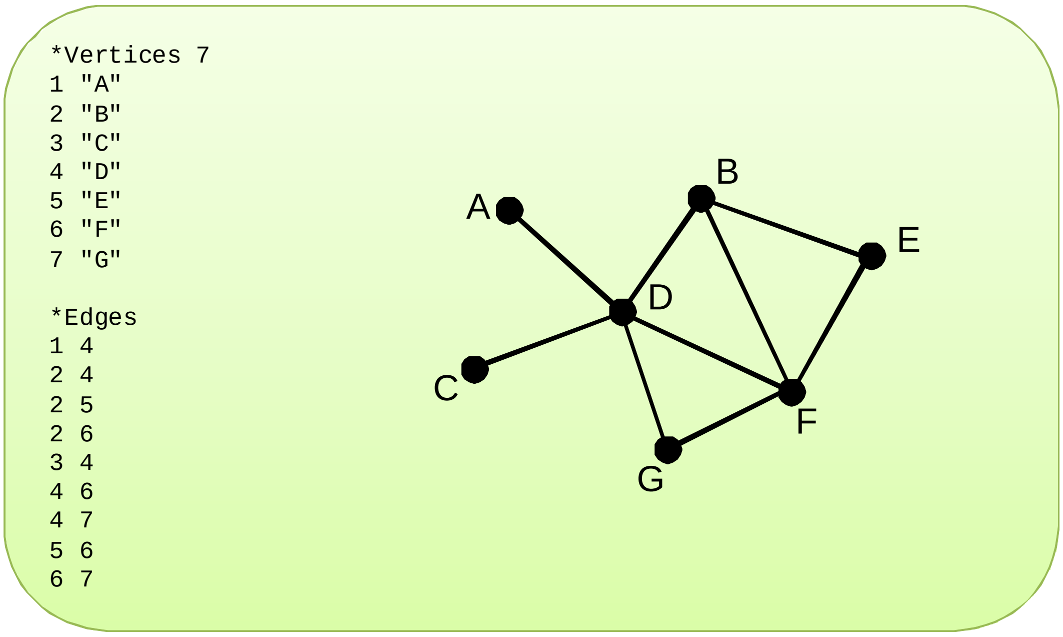


Figure 3.8: The tables used for constructing the local database

Once the local database schema was created the next step was to acquire the necessary data from SRDA and import them to the local database. In Figure 3.8, the fields from where actual data were extracted from SRDA to be used for the analysis are shown highlighted. Due to the vast amount of data, we executed several queries on the tables of SRDA to reduce the workload on the SRDA server and expedite the data gathering process. Once all the necessary data were gathered from the query we then proceeded to import them in the local database. By doing so we removed the dependency we had on the SRDA servers in getting and processing the data.

For the social network analysis of the SourceForge.net collaboration we utilized Pajek, a freely available SNA application specifically design to aid researchers in the analysis of large complex social networks (Batagelj and Mrvar, 2003). Pajek is a Windows program, implemented in Delphi, for analysis and visualization of large networks. The first release of Pajek was in November 1996 and currently is at version 1.25, the version we used for the analysis in this research. The main motivation for development of Pajek was to provide to SNA researchers freely-available tools for the analysis and visualization of large complex social networks such as collaboration networks, organic molecule networks in chemistry, protein-receptor interaction networks, genealogies, Internet networks, citation networks, diffusion (infectious diseases, news, innovations) networks, and data-mining (2-mode networks). Pajek has specific requirements about its input files. A typical Pajek network input file is shown in Figure 3.9. The Pajek net-



```
*Vertices 7
1 "A"
2 "B"
3 "C"
4 "D"
5 "E"
6 "F"
7 "G"

*Edges
1 4
2 4
2 5
2 6
3 4
4 6
4 7
5 6
6 7
```

Figure 3.9: Example of Pajek input file

work input file requires minimum two type of information, node labeling and node connectivity. Information about the nodes such as their total number and labeling is given in the "*Vertices" section of the input file. The number next to "*Vertices" denoted the total number of nodes in the network followed by the numbering and labeling (optional) for each node. The next section of the input file, "*Edges", describes the connectivity of the network. Each row in the "*Edges" section describes a link between two nodes. Pajek is very sensitive about the format of the input file. Connectivity between two nodes must be separated with a single space character or else the program fails to run properly. In the case of undirected networks, the link between two nodes should be input only once, i.e. if node 3 is connected to node 4 only the link "3 4" is needed; inputting link "4 3" will result in an error. In order to meet the input requirements for Pajek it became necessary to create custom data mining programs that process the raw data from the local database to usable Pajek input files. The first step was to create a dump of the three tables in comma separated value (csv) text. Next, user id values were re-labeled in consecutive order. This was done for two reasons. Firstly, Pajek input file specifications require that node numbering be consecutive and continuous. Second, by re-labeling the user id we anonymized the SourceForge.net data. The next challenge was to create the "*Edges" section of the input files. In our database we had only affiliation data, i.e. which user belongs to what project. For the analysis we assumed that two developers (nodes) are connected when they belong to the same project. In order to go from the affiliation data to the connectivity input file we first created a csv file from the affiliation table (users_groups) and then sorted it by project number in ascending order. We then created Algorithm 1 and we used it to write a program that generated the input file for Pajek. The program was written in C++ and is available in the appendices.

3.2.3 Complex Social Network Analysis of SourceForge.net

Recently, Barabási et al. (2002) applied complex social network methodologies to establish a new approach in studying collaboration networks. Their study considered the scientific co-authorship collaboration networks of authors of scientific papers in mathematics and neuro science journals, as paradigms of scientific collaboration and prototypes of evolving networks. Their analysis data came from two large databases containing co-authorship data for all published papers of mathematics and neuro science journals for the period of 1991 to 1998. Even though the data enabled them to study the evolution of the co-authorship collaboration network within the period 1991-98 they had to resort to simulations in order to study the the network evolution from its inception. Our work takes a slightly different, yet complimentary approach

Algorithm 1 Create the Pajek input file from the affiliation table

```
while Not EOF do
    Read Line of Data
    if First line then
        Split string $\rightarrow a, b$
        Store $a$ in vector
        Copy $a, b \rightarrow c, d$
    else
        Split string $\rightarrow a, b$
        if $b = d$ then
            Store $a$ in vector
            Copy $a, b \rightarrow c, d$
        else
            Sort vector
            for $j = 0$ to vector.size-1 do
                for $i = j + 1$ to vector.size do
                    Output vector(j), vector(i)
                end for
            end for
            Clear vector contents
            Store $a$ in vector
            Copy $a, b \rightarrow c, d$
        end if
    end if
end while
```

to the study of Barabási et al. (2002). In contrast to Barabási et al. (2002), our data from SourceForge.net enables us to study the formation and evolution of the F/OSS collaboration network from its inception to the present. We believe that our study of F/OSS collaboration at SourceForge.net presents a true paradigm of a complex evolving network. A visualization of the scale of data we used for the analysis as well as an overall view of the evolution of the F/OSS collaboration network at SourceForge.net is shown at Figure 3.10 and 3.11. Figure 3.10 shows the cumulative number of developers at SourceForge.net based on the SRDA data. The inset of the graph shows the number of developers registered every year from the establishment of

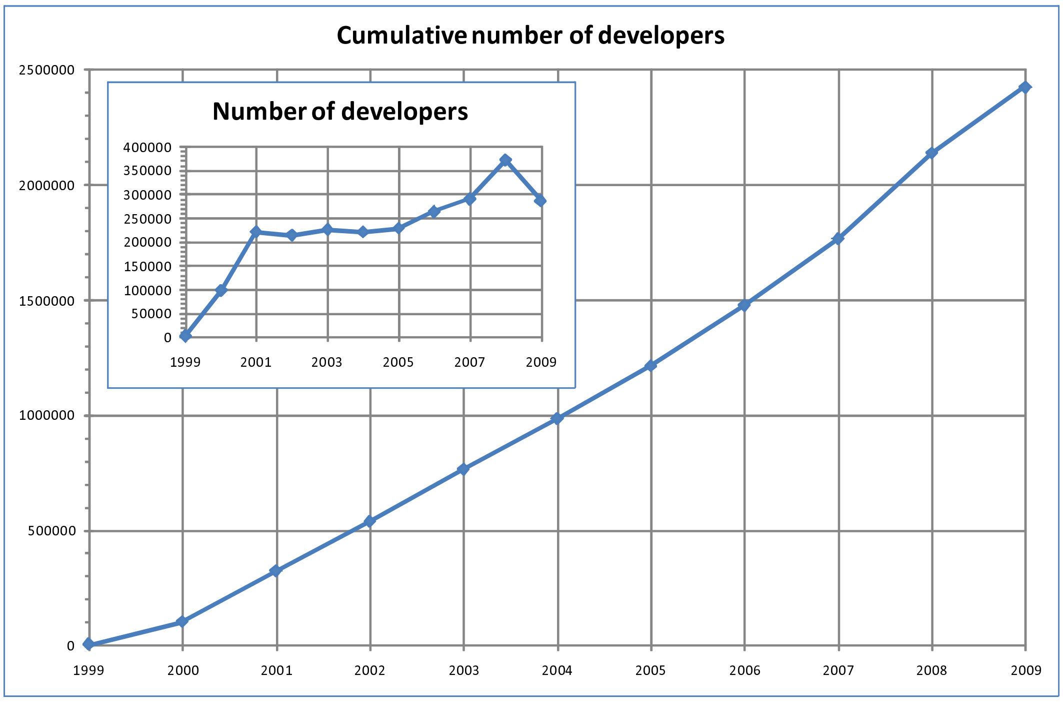


Figure 3.10: Cumulative number of developers on SourceForge.net

SourceForge.net in 1999 until the present. Similarly, Figure 3.11 shows the cumulative number of software projects at SourceForge.net based on the SRDA data. The inset of the graph shows the number of software projects registered every year from the establishment of SourceForge.net in 1999 until the present.

Degree Distribution

The degree distribution of a complex social network is one of the important statistics calculated in SNA. Figure 3.12(a) shows a plot of the degree distribution of the SRDA cumulative data up to year 2009. From the plot, it can be seen that the degree distribution of SourceForge.net collaboration network follows a power-law distribution $P(\kappa) = \alpha\kappa^{-\gamma}$, where $\gamma = 1.488$. In Figure 3.12(b), we used logarithmic binning to plot the degree distribution in order to show more clearly the scaling regime of the power-low distribution. Complex social networks that have power-law degree distributions are classified as scale-free networks (Barabási et al., 2002). A close examination of Figure 3.12 shows that the data from the SourceForge.net collaboration network follow a power-law distribution, thus we can safely assume that the SourceForge.net collaboration network is a scale-free network.

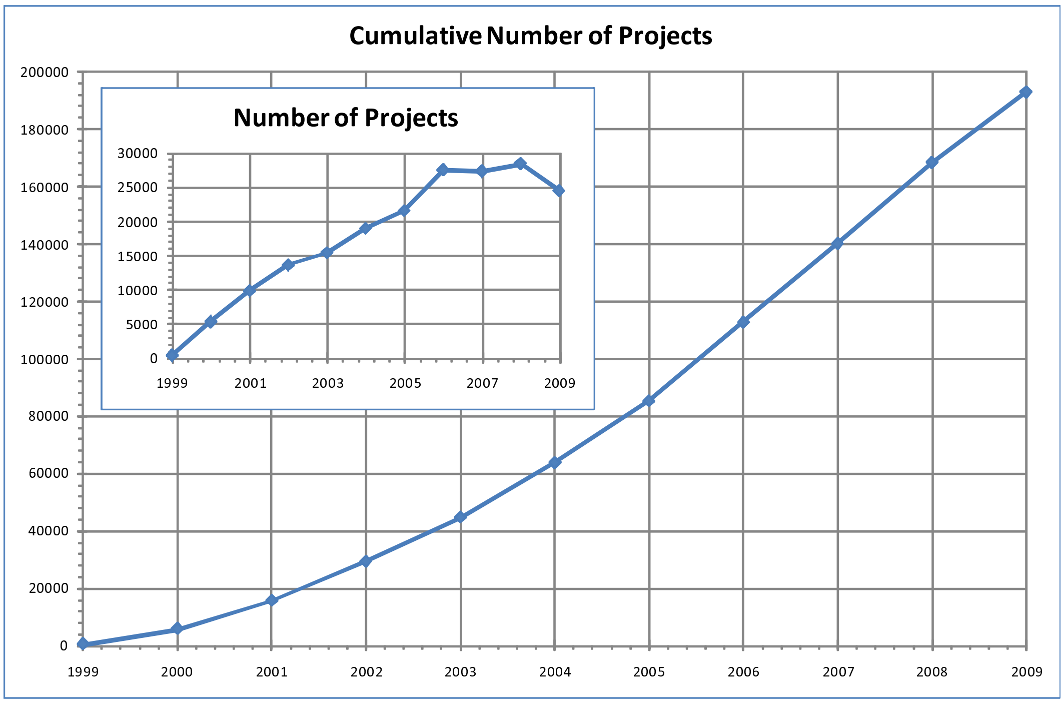
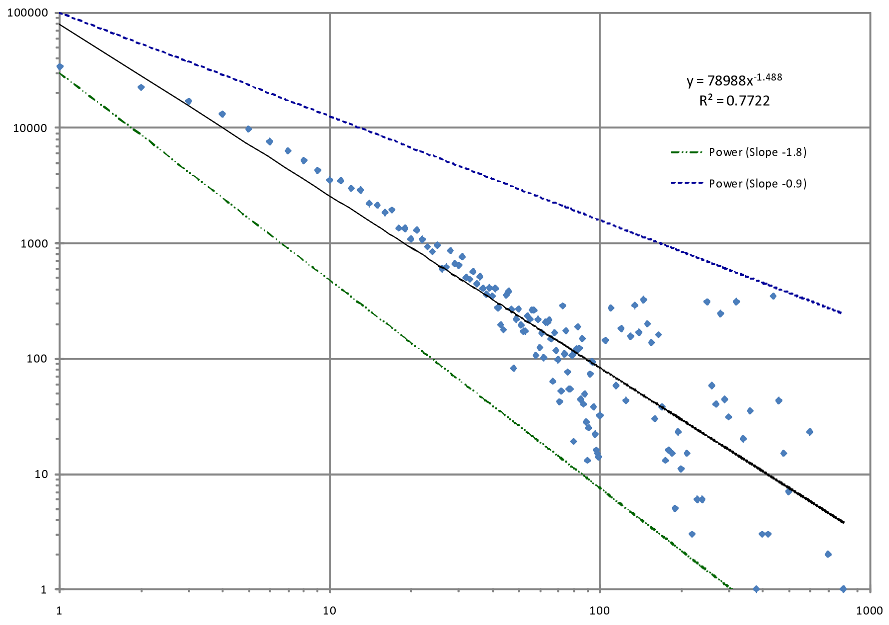


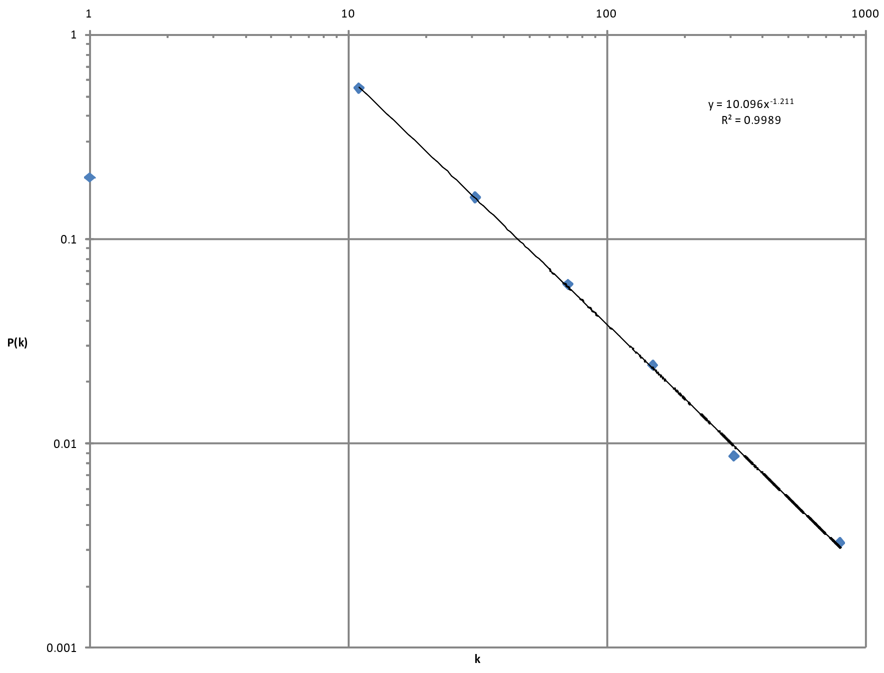
Figure 3.11: Cumulative number of projects on SourceForge.net

Evolution of the F/OSS Collaboration Network

One of the main objectives of this research is to examine the evolution of the F/OSS collaboration social network. In order to do so, we conducted SNA of the SourceForge.net collaboration network for each year from the formation of the network in 1999 until the present time, 2009. The analysis for each year is based on the cumulative data collected until that specific year. The first important finding of the evolution analysis is that the degree distribution of the network over time follows the power-law shape. Figure 3.13 shows the evolution of the degree distribution of the SourceForge.net collaboration network over time. It can be seen that the degree distribution follows a power-law shape at all times. Table 3.1 summarizes the results of the evolution analysis. The evolution analysis of the SourceForge.net collaboration network revealed some interesting characteristics. Figure 3.14 shows the evolution of the number of nodes and links of the network together. It can be seen that the network expands over time but the rate of addition on new nodes is higher than the rate of creating new links. Figure 3.15 shows the evolution of the network's density. The density of the network declines rapidly over time. Figure 3.16 shows the average degree of the network over time. The averages degree of the network increases initially and finally stabilizes around 1. Figure 3.17 shows the largest degree of the network over time. The value of the largest degree increases over time. A very



(a) Plain Binning



(b) Logarithmic Binning

Figure 3.12: Degree distribution (cumulative data up to 2009)

important characteristic of a network is its diameter. The diameter of a network is defined as the longest distance among all the shortest paths connecting any two nodes of the network. The diameter basically describes how many "jumps" are needed in average to reach any node in the

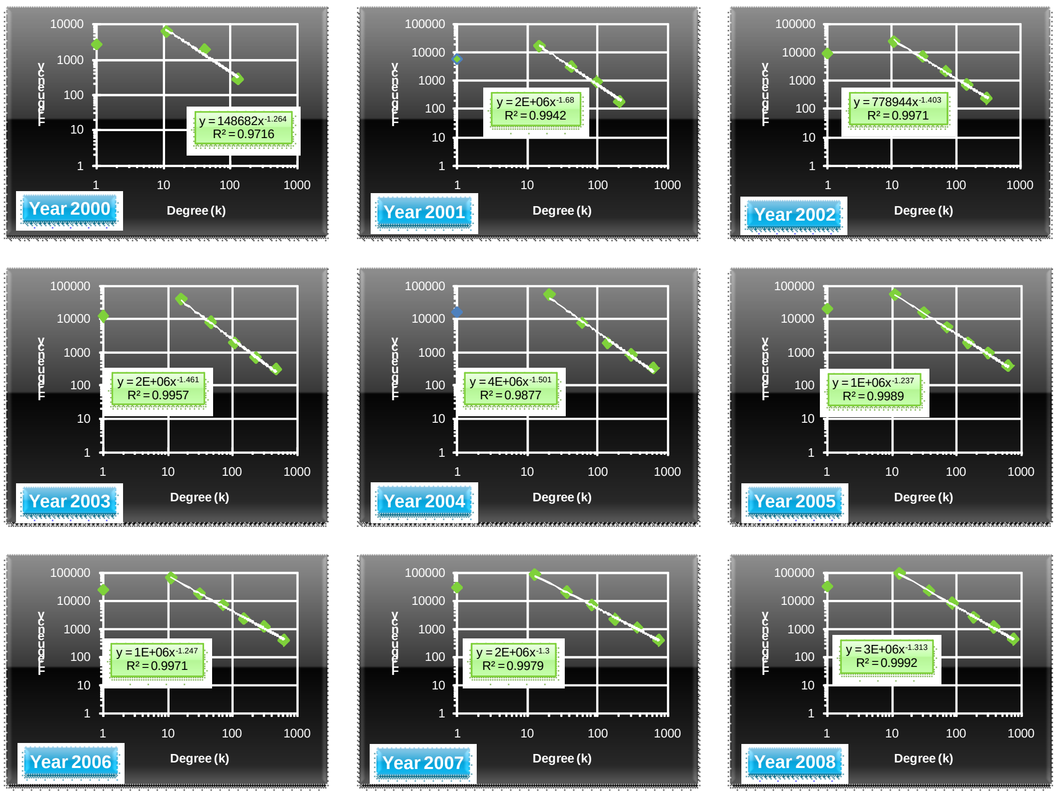


Figure 3.13: Degree Distribution Evolution of SourceForge.net

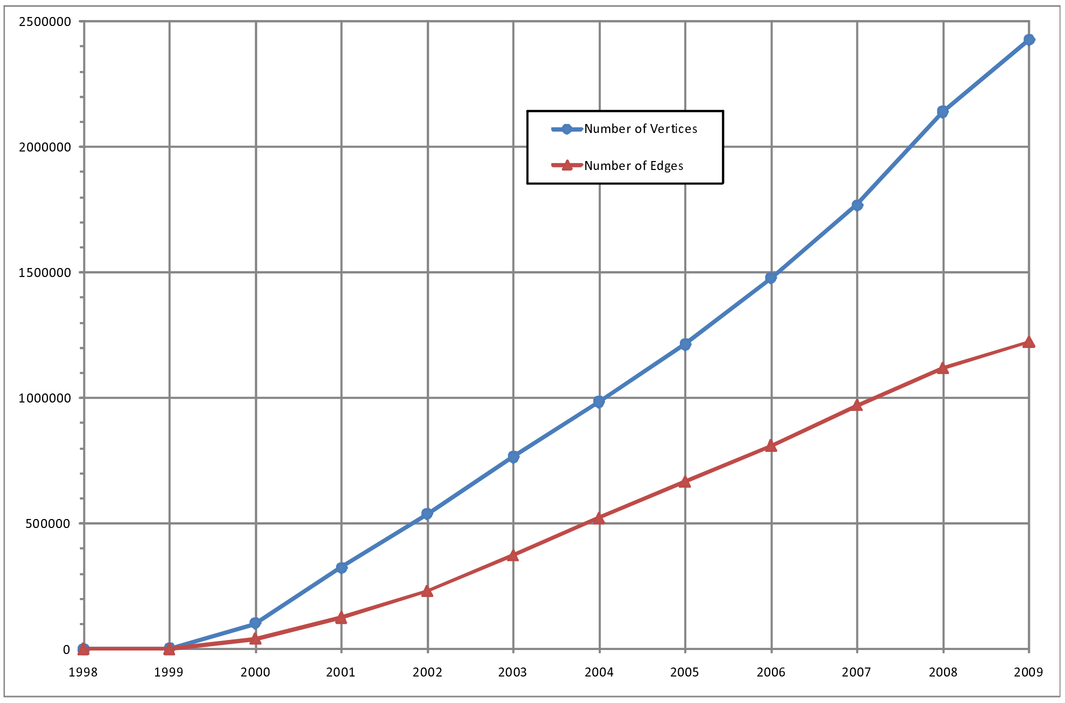


Figure 3.14: The Evolution of Vertices and Edges of SourceForge.net

Table 3.1: SourceForge.net SNA Evolution Summary

	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Number of Vertices	2877	101423	322876	537486	763628	984507	1212904	1476574	1766736	2138829	2425081
Number of Edges	208	41637	124817	230618	371361	520385	665331	809193	968533	1117382	1220522
Density	0.0000491	0.0000081	0.0000024	0.0000016	0.0000013	0.0000011	0.0000009	0.0000007	0.0000006	0.0000005	0.0000004
Average Degree	0.1411192	0.8210564	0.7731575	0.8581358	0.9726228	1.0571484	1.0970877	1.0960412	1.0964094	1.0448540	1.0065825
Diameter	5	20	20	21	22	23	23	23	22	22	22
Largest Degree	12	128	199	245	372	474	553	619	668	713	740
Mean Cluster Coefficient	0.0236376	0.0685985	0.0546701	0.0556311	0.0563757	0.0573561	0.0578662	0.0572359	0.0563000	0.0532898	0.0511894

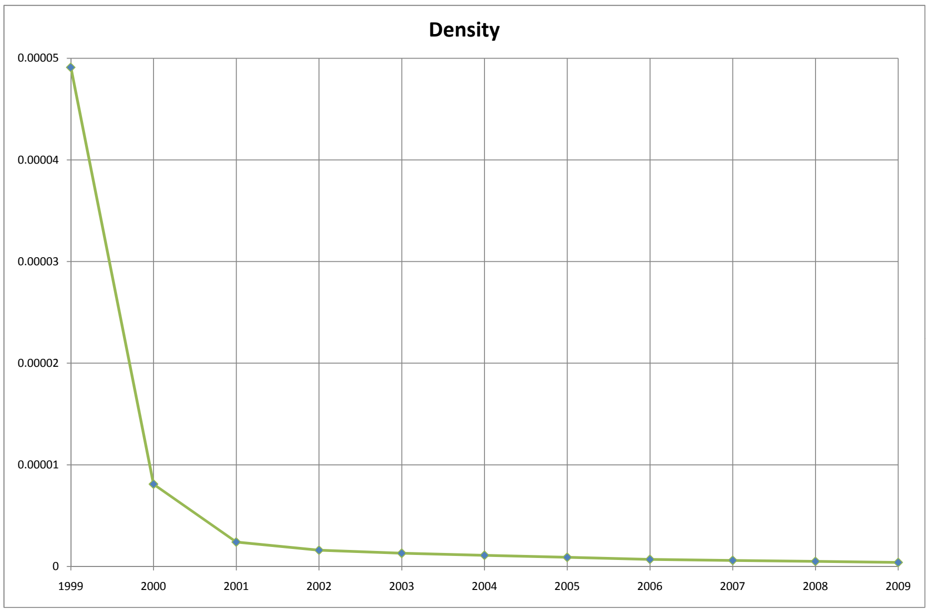


Figure 3.15: The Evolution of the Density of SourceForge.net

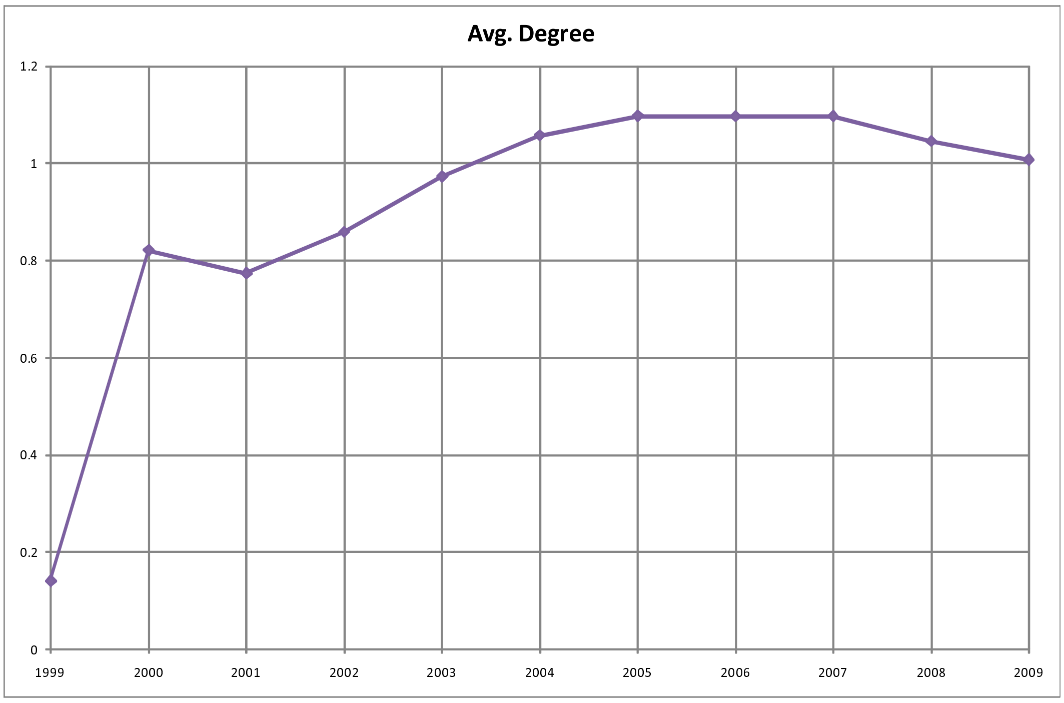


Figure 3.16: The Average Degree of SourceForge.net Network Over Time

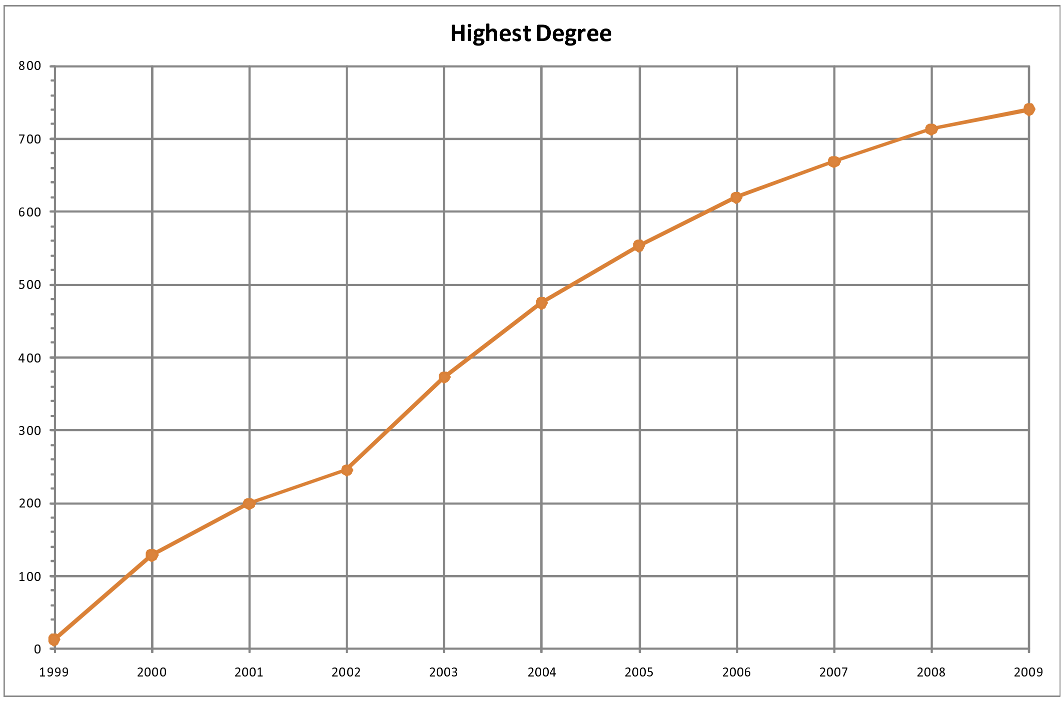


Figure 3.17: The Largest Degree of SourceForge.net Network Over Time

network. Collaboration networks with power-law degree distributions exhibit the "small world" characteristic. Such networks typically have small diameters, ranging from 5 to 30. Figure 3.18 shows the diameter of the network over time. It can be seen that the diameter is initially very

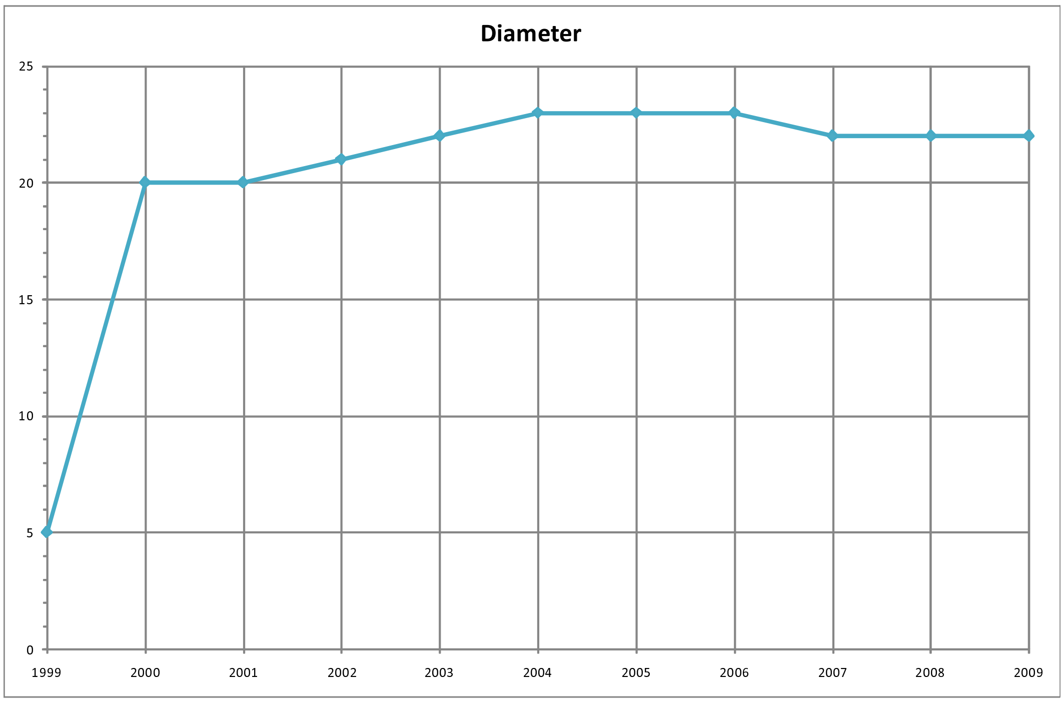


Figure 3.18: The Diameter of SourceForge.net Network Over Time

small and then stabilizes at 22, a typical value expected of a collaboration network having a power-law degree distribution. Finally, Figure 3.19 shows the mean clustering coefficient of the network over time. The clustering coefficient starts at a low value then increases and finally

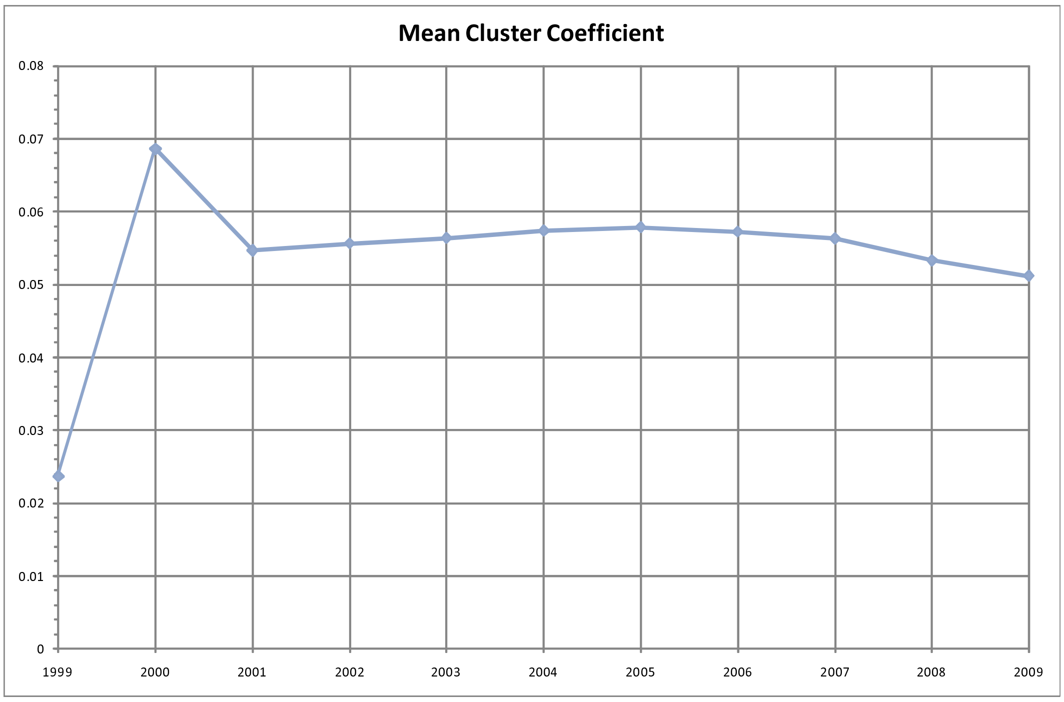


Figure 3.19: The Mean Clustering Coefficient of SourceForge.net Network Over Time

stabilizes around 0.05.

Discussion of Analysis Results

The complex social network analysis as well as the evolution analysis of the collaboration network of SourceForge.net reveals some very interesting results. The first one relates to the physical evolution of the network, i.e. the change in the number of nodes and links over time. Referring to the graph shown in Figure 3.14, it can be seen that both the number of nodes and links grow over time but the rate of growth of links is lower than the rate of growth of nodes. This can mean that collaboration among the members of SourceForge.net declines over time. Even though more and more new members join SourceForge.net over time they choose to either not to participate in F/OSS projects or start a project alone. This is also evident from Figure 3.16 that shows the average degree of the network over time to stabilize around 1, i.e. in average the members of SourceForge.net are collaborating in pairs. Figure 3.17, showing the largest degree of the network over time, exposes another interesting characteristic of the SourceForge.net collaboration. The increasing largest degree of the network means that few projects attract a lot of software developers creating huge components in the network.

One of the biggest concerns of the F/OSS community, which is also a hot topic for research in the field, is the future shortage of F/OSS software developers and its impact on the sustainability of the F/OSS phenomenon (Lattemann and Stieglitz, 2006). It is posited that as the demand in F/OSS projects increase in the future the number of available software developers will not be sufficient to meet that demand. As a result, it is believed that F/OSS will not be sustainable. The results from the evolution analysis of the SourceForge.net collaboration network can be used to hypothesize about the future sustainability of F/OSS. By observing Figures 3.10 and 3.11, it can be seen that the growth of the number of F/OSS software developers and of F/OSS projects follow similar growth rates. Additionally, Figure 3.15 shows that the density of the SourceForge.net collaboration network diminishes with time, meaning that the network is far from been completed. It is argued that social networks that are incomplete and have structural holes posses higher social welfares that complete social networks (Bramoullé and Kranton, 2007). In our opinion, based on the findings of our analysis, we believe that F/OSS is sustainable in the future. Our findings show a healthy, thriving collaboration community that expands over time with no signs of decline.

3.2.4 Conclusions and Topics for Future Research

This chapter investigated the F/OSS phenomenon as a paradigm of open standardization process formation and evolution via the complex social network analysis framework, utilizing data from SourceForge.net. The main findings of the investigation are:

- F/OSS collaboration has the main characteristic of other complex social networks such as the scientific co-authorship collaboration network, the patent collaboration network and the Internet. Those characteristics include power-law degree distributions, small network diameters and "small-world" properties.
- The SourceForge.net collaboration network is expanding over time and the rates of growth of the number of developers and projects follow similar trends.
- The SourceForge.net collaboration network's level of completion is low and the network has structural holes.
- If the SourceForge.net collaboration network's characteristics remain unchanged it will be sustainable over time.

In this research we analyzed the SourceForge.net collaboration network using data collected from its inception to the present. The network evolution parameters we calculated in this study can be used to run simulations to investigate the future growth and sustainability of the F/OSS phenomenon. In addition to that, our analysis can proceed further by gathering new data over time and verify our sustainability hypothesis. Those remain topics of future research. Another important topic for future research which was pointed out by the members of the defense committee was the investigation of the evolution of the network's density. From the analysis conducted in this research it was found that network's density is decreasing over time. In the future the density evolution could be studied in order to find which density is optimal for this kind of collaboration networks.

Chapter 4

Learning Networks

4.1 Introduction

The importance of mitigating the risks associated with contaminated sites is evident in the increasing trend in the number of soil decontamination projects. The decontamination and remediation of contaminated land sites, in Japan for example, follows a rapid upward trend. According to data from a recent APEC-VC study[1], an estimated 320,000 contaminated site locations exist in Japan. A more recent study on the trend of soil contamination investigation cases cites 211 contaminated site cases in 1999, 204 in 2000 and 273 in 2001 (Muraoka, 2005). Soil contamination investigations are considered essential before any land transaction because the costs incurred from soil decontamination can be very high. Another factor that makes soil contamination investigations essential before any land transaction is the preference of buyers to avoid purchasing potentially contaminated land that will eventually incur extra costs and result in a lower land market resale value.

Many studies related to the evaluation of soil decontamination cost can be found in the literature. Past research on the evaluation of soil decontamination costs can be divided roughly into two categories. The first category includes those studies that utilize statistical models that evaluate the soil decontamination cost risk based on the correlation of pertinent data. The second category includes the studies that utilize dynamic models using data on soil pollutants and their associated decontamination costs. What both categories have in common is that they depend on soil contamination data produced by costly investigations such as subsoil exploration via drilling. However, a practical methodology that attempts to estimate the soil decontamination cost during the conceptual stage of land development, using a dynamic model

[1]Data as of year 2005.

and data obtained from low cost investigation methods such as the past land usage and top soil contamination investigations, cannot be found in the literature. This is partially due to the fact that topsoil contamination investigations give limited information on the depth and extend of the soil pollution. Moreover, when the ground conditions are not uniform, which is typically the case with most sub-soil conditions, a lot of uncertainty exists when the soil decontamination cost is estimated. Therefore relying on the results from simple investigations to calculate the soil decontamination cost is not dependable and the usage of an advanced statistical model to assist in the estimation of the cost risk is necessary.

Research conducted using artificial neural networks (ANN) has a long history with many applications in the field of civil engineering. Using ANN models, non-linear phenomena with complex structures can be easily modeled. It can be proved that a multi-layer ANN with at least one hidden layer is able to approximate quite reliably any arbitrary function (Hornik et al., 1989). As a result, ANN models are accepted as appropriate flexible non-linear forecasting models. Bayesian statistics has also a long history of research and many applications based on Bayesian statistics also exist in the field of civil engineering. However, research using ANN models based on Bayesian statistics is quite new and very few case studies can be found in the literature. Reasons for the lack of such studies is the high complexity of BNN, problems associated in calculating the posterior probabilities of the parameters and difficulties involved in making predictions by integrating over the posterior distribution. Analytical methods are nearly infeasible and methods that utilize conjugate distributions of the parameters of the model also are infeasible when the Bayesian estimation method is applied on an ANN (Neal, 1996). This became a practical obstacle in using the Bayesian estimation method with ANN. However, it is possible to calculate the posterior probability distribution more easily using, for example, Markov Chain Monte Carlo methods (MCMC) (Neal, 1992; MacKay, 1992b).

4.1.1 Soil contamination investigations

Typical soil decontamination techniques include chemical processing of the soil (e.g. wash and reuse), pollutant vapor extraction and disposal and biological remediation. What all of those techniques have in common is their high complexities and costs. As a result, soil contamination investigations are necessary before any decisions related to the site redevelopment are considered.

In order to get a realistic indication of the soil contamination extends, topsoil contamination investigations, drilling investigations (soil boring exploration), and other soil contami-

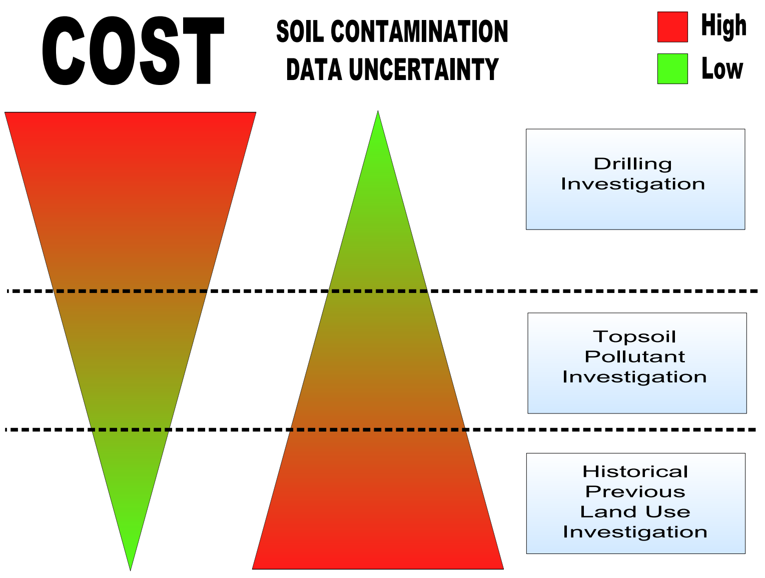

COST
SOIL CONTAMINATION
DATA UNCERTAINTY
🟥 High
🟩 Low
Drilling Investigation
Topsoil Pollutant Investigation
Historical Previous Land Use Investigation

Figure 4.1: Soil Contamination Investigation Techniques.

nation assessments are necessary. Soil contamination investigations can be classified by their complexities, costs and quality of the resulting data, as seen in Figure 4.1. There exist basically three soil contamination investigation methods:

Past (historical) land usage investigations which include reviewing of old maps and aerial photos, examination of the land register, interviews and investigation of the peripheral area and geographical investigations

Topsoil investigations which include investigation of soil gases, heavy metals and agricultural pollutants using topsoil samples taken from the contaminated site

Drilling investigations which involve full subsoil explorations using boring and sampling, underground water investigations and various advance simulations and data analyses.

Inherently, the three types of investigations have different associated costs and produce data with different levels of accuracy and significance. The past land usage investigation is the simplest of the three. This investigation utilizes data such as the land usage history, current land usage, soil pollution accidents, terrain type and data on the surrounding land. When a qualitative judgment concerning soil contamination is requested at the early stages of the land development project, the past land usage investigation is appropriate. When investment for a possibly contaminated land development project is strongly considered and managerial risk decision making is necessary, a more advanced investigation such as a topsoil contamination investigation should be considered. The topsoil contamination investigation directly examines the presence, range, kind and density of the pollutants at the surface soil of the land. However the total extends of the soil contamination cannot be determined by the topsoil contamination investigation alone and it is common to have a lot of uncertainty in the estimation of the soil decontamination cost using data from this kind of investigation. When the land redevelopment decision solidifies, a more precise soil decontamination cost evaluation method is needed. In this case it is necessary to execute drilling investigations to evaluate the extent of the land contamination more accurately. When drilling investigations are executed, data about the precise geological features of the subsoil, presence of pollutants including their kind, range, depth and densities, the ground water table, ground water contamination are collected. Drilling investigations provide the best and most accurate information for the assessment of the soil contamination but they are also the most costly and time consuming of the three investigation methods. During the conceptual stage of a land development investment, when the profitability of the investment is assessed, it is not reasonable to spend large amounts of money and time for a detailed soil contamination investigation. In that case, it becomes necessary to evaluate the soil decontamination cost using simple inexpensive investigations such as topsoil contamination and historical land usage investigations and to examine the sustainability of the investment in view of market risks and other risk factors. However, it is extremely difficult to calculate the extent of soil contamination deterministically from such simple investigations. Therefore, the soil decontamination cost probability distribution, hereafter referred to as the soil decontamination cost risk, becomes more relevant. The research presented in this paper proposes a Bayesian neural network (BNN) model to evaluate the soil decontamination cost risk based on information gathered from simple topsoil contamination and historical land usage investigations. The model described in this paper was developed to be used as a tool in aiding investment decision making by calculating the soil decontamination cost risk. The model merges two powerful statistical methodologies, artificial neural networks (ANN) and the Bayesian statistical framework, by forming a BNN model capable of producing the full distribution of soil decontamination cost risk based on data from simple soil contamination investigations.

The chapter is organized as follows. The next section discusses soil decontamination cost risk. The fundamentals of the research and the model are discussed in the following sections. The next section presents a case study using data from real soil decontamination projects around Japan followed by the discussion of the results. Summary and conclusions are provided in the final section.

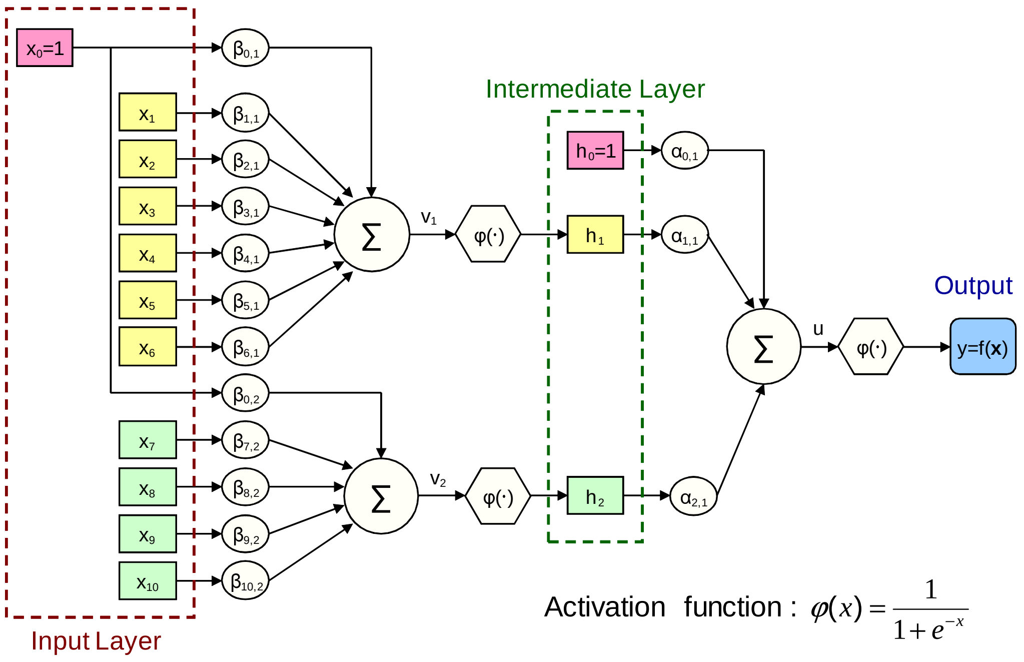


Figure 4.2: A multi-layer feed-forward network.

4.2 Research Fundamentals

4.2.1 Artificial Neural Networks

In an ANN, nodes called “neurons” are connected together to form a network, as shown in Figure 4.2. The similarity of ANN with biological neural networks is in the way the neurons function, i.e. in a parallel and collective fashion (Haykin, 1999). Each connection of the network is associated with a weight which expresses the contribution of the connection to the network. The effectiveness of an ANN lies in its ability, through an algorithm, to alter the weights (strength) of its connections by minimizing some sort of a cost function (Principe et al., 2000). A typical cost function is the network’s output error which is basically measured as the difference between the network output and the desired response. This ability renders ANN able to classify complicated data, extract patterns and detect trends that are too complex or very computationally demanding to be evaluated by other known techniques.

4.2.2 Bayesian Neural Networks

Bayesian models are very computationally costly and this fact posed a strong resource limitation to researchers. As computers became faster and more readily available and as machine computation costs became lower, numerical applications and sampling techniques used in

Bayesian models that required huge computation costs became increasingly available (MacKay, 1992a). Sampling techniques such as Markov Chain Monte Carlo (MCMC) and sampling algorithms such as the Metropolis-Hastings and the Gibbs sampler were put to work in Bayesian analysis where the estimation of complicated posterior distributions was previously infeasible. The Bayesian approach was introduced in ANN in the early 1990's by the work of Buntine and Weigend (1991), MacKay (1992b) and Neal (1992), and reviewed in Bishop (1995), MacKay (1995) and Neal (1996). In the Bayesian framework all the parameters of a model have their probability distributions and inference is conducted from the posterior conditional probabilities of the unobserved variables of interest, given the collected (observed) data and the prior model assumptions. Uncertainty about the relationship between the inputs and outputs of the network is initially taken care of by an assumed prior distribution of the network's parameters (i.e. its weights and biases). The Bayes' theorem is used to update the prior to the posterior distribution while new data are observed and incorporated in the model. The posterior distribution is then maximized with a suitable optimization technique. Once the network is trained, the predictive distribution of the network outputs is obtained by integrating over the posterior distribution of the model's parameters.

4.3 Model Formulation

The term "risk" has several definitions in the literature. In this research risk is used in the context of uncertainty, i.e. the indefiniteness about the outcome of a situation, which is described by a probability distribution. The uncertainty in the forecasted value of the soil decontamination cost is referred to as the soil decontamination cost risk. As more advanced soil contamination investigations are employed it is possible to acquire detailed information on the soil contamination i.e. the level of uncertainty decreases. For example, if the drilling survey is executed, the extents of the soil contamination can be realized in more detail utilizing the data from the investigation. Those data can be used to forecast the soil decontamination cost more accurately. However, in this study it is assumed that the land development is at the early stage of the investment and it is unreasonable to spend a large amount of money on advanced investigations. The information conveyed by the soil decontamination cost risk can be used to extract a better overview of the decontamination cost. The cost risk data can be also used in conjunction with other risk information, e.g. the land market price volatility, in order to aid the decision making process concerning the future investment.

4.3.1 Model Structure

SD-CORE stands for **S**oil **D**econtamination **CO**st **R**isk **E**valuation and is the name given to the model developed in this research. The structure of the model is shown in Figure 4.3. The model consists of six processes. The first process utilizes an ANN feed-forward MLP model.

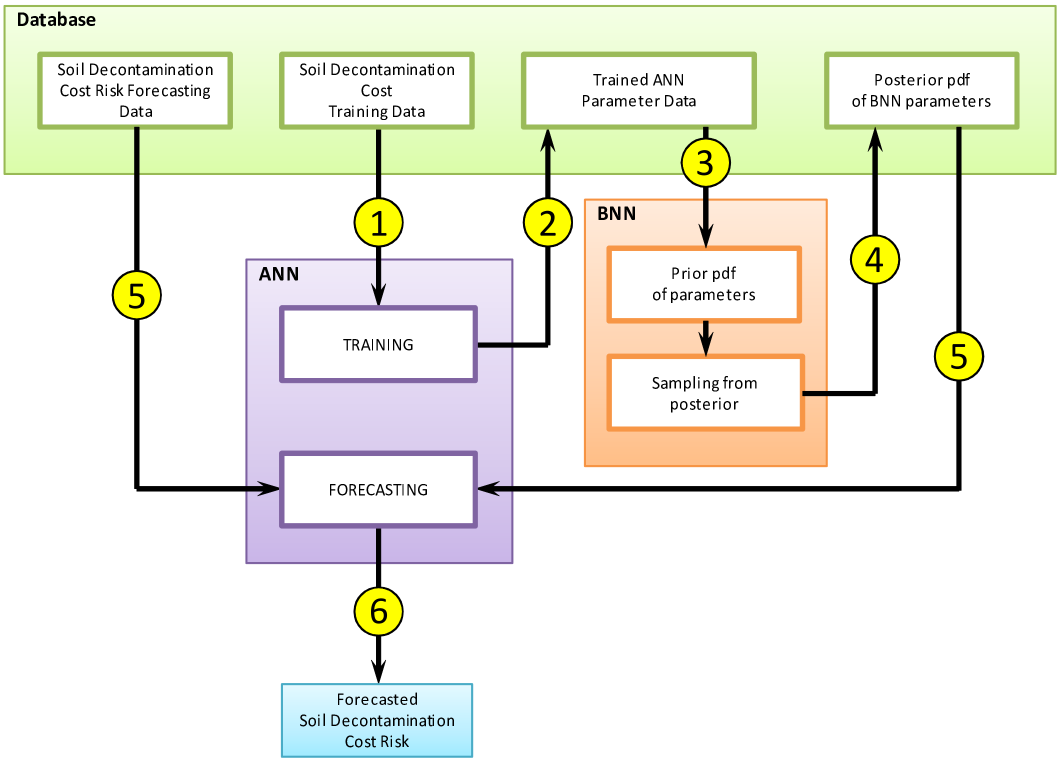

Database
Soil Decontamination Cost Risk Forecasting Data
Soil Decontamination Cost Training Data
Trained ANN Parameter Data
Posterior pdf of BNN parameters
1
2
3
4
5
5
6
ANN
TRAINING
FORECASTING
BNN
Prior pdf of parameters
Sampling from posterior
Forecasted Soil Decontamination Cost Risk

Figure 4.3: The SD-CORE model structure

The ANN is trained using soil decontamination cost data (referred to as the training data) from the main database. Once the ANN is fully trained the values of the ANN's parameters are saved in the database (process 2). In the third process, the values of the parameters of the ANN that were saved after the training are used to form the prior distribution of the parameters of the BNN. It is noted here that the BNN model uses the same network structure and configuration as the ANN. Using the likelihood function of the parameters of the BNN, the posterior distribution of the parameters given the training data is calculated using Bayes' rule and sampling from all the parameters is conducted. The posteriors of all the BNN's parameters are then saved to the database (process 4) to be used for forecasting. For the forecasting (process 5) the ANN model, soil decontamination forecasting data and the posteriors of the BNN parameters are used to calculate the predictive distribution for the new input data. The result (process 6) is the full distribution of the forecasted soil decontamination cost. Finally, when new soil decontamination cost data (e.g. from completed soil decontamination projects) are gathered they can be imported in the database, the system is retrained and new forecasts can be generated.

4.3.2 ANN formulation

The model uses an ANN feed-forward multi-layer perceptron (MLP) model with a single hidden layer as shown in Figure 4.4. For the training process the back-propagation algorithm is used (Rumelhart and McClelland, 1986). The back-propagation algorithm is applicable to

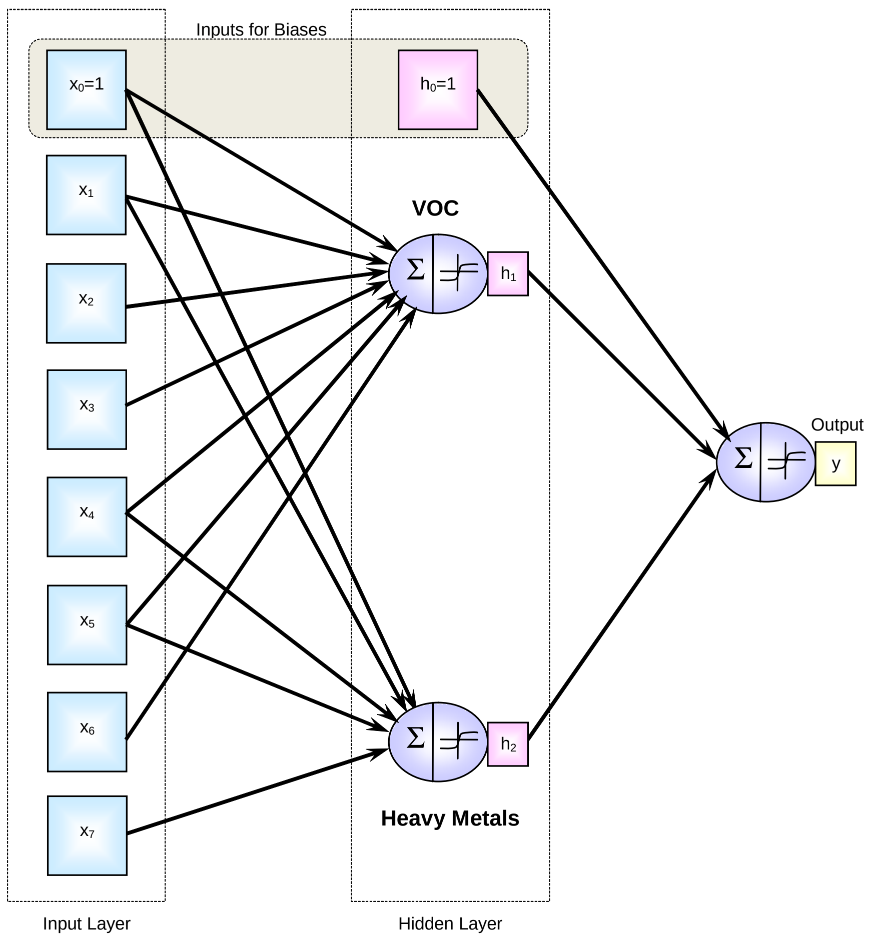


Figure 4.4: The ANN model structure

ANN that utilize the supervised learning paradigm. It is based on the error-correction learning rule and can be considered as a generalization of the LMS algorithm described in Table 4.1. In ANN, learning is basically achieved through an iterative process based on a set of rules that aim to adjust the parameters (i.e. synaptic weights and biases) of the network thus enabling the network to learn about its environment. Central to the learning process is the cost (objective

Table 4.1: The LMS algorithm

Training Data	Input vector $\mathbf{x}(n)$
	Desired output $d(n)$
Learning Rate	User selected η
Initialize weights	$\mathbf{w}(0) = \mathbf{0}$
Calculations	For $n = 1, 2, \ldots$
	$e(n) = d(n) - \mathbf{w}^T(n)\,\mathbf{x}(n)$
	$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\mathbf{x}(n)\,e(n)$
	Loop

or error) function which typically measures how far away the network output is from from the optimal solution. The objective of learning is to find the solution the minimizes the cost function. The learning approach shares a lot of similarities with maximum likelihood estimation method where the model with the parameters that fit best the given data is chosen. The most typical cost functions used in ANN are the sum of squared errors (SSE) and the mean squared error (MSE). The sum of squared errors is defined by

$$E_{SSE} = \sum_k \sum_i (d_{ki} - o_{ki})^2 \tag{4.1}$$

where k is the data pattern index, i is the output node index, d_{ki} is the desired (target) response and o_{ki} is the network's actual output. The mean squared error is defined by

$$E_{MSE} = \frac{1}{KN} \sum_k \sum_i (d_{ki} - o_{ki})^2 = \frac{1}{KN} E_{SSE} \tag{4.2}$$

where K is the number of the training data patterns and N is the number of network outputs. In this study the supervised learning method is utilized. The standard algorithm used in supervised learning is the back-propagation (BP) also called generalized delta rule (Rumelhart and McClelland, 1986). A representation of ANN supervised learning can be seen in Figure 4.5. The algorithm is an "off-line" algorithm, that is the two main operations of the ANN, the training and the predictive operations, take place at different times. In fact the predictive operation of

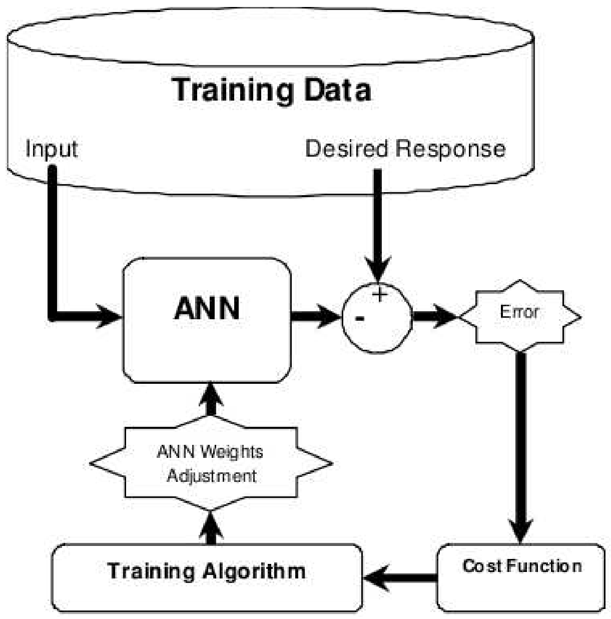


Figure 4.5: Supervised learning

the ANN requires that the training process be completed first. The basic steps of the algorithm are as follows (Reed and Marks, 1999):

- Present a training pattern to the ANN and propagate it through it to get the output.
- Compare the output with the desired response and calculate the error from the error function.
- Calculate the derivatives $\partial E/\partial w_{ij}$ of the error with respect to the network's parameters.
- Adjust the network's parameters to minimize the error.
- Repeat until the error is acceptably low or another convergence criterion is reached.

The algorithm involves two stages, the forward pass and the backward pass. During the forward pass the input data are passed through the network's layers and the output is calculated at the exit of the network. It should be noted that during the forward pass the network's parameters (weights and biases) are held fixed. Once the output is generated it is compared with the desired response and through the error function an error signal is generated. The error signal is then utilized during the backward pass where it is propagated backward in the network, from the output to the input layer. During the backward pass the parameters of the network are adjusted according to an error-correction rule in order for the output of the network to get closer to the desired response. Thus, choosing an appropriate error function for the algorithm is very important. The two most typical error functions used are the sum-of-squares function shown in (4.1) and the mean-square-error function shown in (4.2). The algorithm also requires that the activation function used in the neurons of the ANN be differentiable. The two most commonly used activation functions, the logistic and the hyperbolic tangent functions are both continuous and differentiable. In this research the logistic function given by

$$\phi(u) = \frac{1}{1+e^{-u}} \tag{4.3}$$

with first order given by:

$$\phi'(u) = \phi(u)\left(1-\phi(u)\right) \tag{4.4}$$

is utilized. The algorithm loops through all the data. The presentation of all the input data to the network is called an epoch. At the end of an epoch the average of the error is calculated and the training process is repeated until the average error reaches an acceptable low value or another convergence criterion is met.

4.3.3 Bayesian Estimation

Assume a training data sample $(\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M)$ consisting of M points. Let $k \in (1, \ldots, M)$ and consider a training sample point $\boldsymbol{\xi}^k = (y^k, \mathbf{x}^k)$. Here y^k, the desired response, is an actual soil decontamination cost value and the vector $\mathbf{x}^k = (x_1^k, \ldots, x_n^k)$ is the vector of the inputs consisting of n input variables. The likelihood of the data given the parameters is $\mathcal{L}(\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M | \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the unknown parameters of the BNN model. The prior distribution of the parameters is $\pi(\boldsymbol{\theta})$. The posterior distribution of the parameters given the data, using the Bayes' rule, is

$$\pi\left(\boldsymbol{\theta}|\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M\right) = \frac{\mathcal{L}\left(\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M|\boldsymbol{\theta}\right)\pi\left(\boldsymbol{\theta}\right)}{\int_\Theta \mathcal{L}\left(\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M|\boldsymbol{\theta}\right)\pi\left(\boldsymbol{\theta}\right)d\boldsymbol{\theta}} \tag{4.5}$$

Because the denominator of (4.5), i.e. the evidence of the model, can not be evaluated the following proportionality is used

$$\pi\left(\boldsymbol{\theta}|\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M\right) \propto \mathcal{L}\left(\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M|\boldsymbol{\theta}\right)\pi\left(\boldsymbol{\theta}\right) \tag{4.6}$$

The posterior distribution of the parameters is then used to make predictions for previously unseen data. Let the new data point we want to get the predictive distribution, i.e. the soil decontamination cost risk, be $\boldsymbol{\xi}^{M+1} = \left(y^{M+1}, \mathbf{x}^{M+1}\right)$ where $\mathbf{x}^{M+1}$ is the vector of the new inputs. The predictive distribution is then given by

$$\rho\left(y^{M+1}|\mathbf{x}^{M+1}, \boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M\right) = \int p\left(y^{M+1}|\mathbf{x}^{M+1}, \boldsymbol{\theta}\right)\pi\left(\boldsymbol{\theta}|\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M\right)d\boldsymbol{\theta} \tag{4.7}$$

4.3.4 BNN formulation

One of the main assumptions of the BNN model is that the probability density of the soil decontamination cost given the input data and the model's parameters, $p\left(y|\mathbf{x},\boldsymbol{\theta}\right)$, is Log-Normally distributed given by

$$p\left(y|\mathbf{x},\boldsymbol{\theta}\right)=\left(2\pi\sigma^2y^2\right)^{-\frac{1}{2}}\exp\left\{-\frac{\left(\ln y-\ln f\left(\mathbf{x},\boldsymbol{\beta}\right)+\frac{\sigma^2}{2}\right)^2}{2\sigma^2}\right\} \tag{4.8}$$

where the BNN parameters are $\boldsymbol{\theta}=\left(\boldsymbol{\beta},\sigma^2\right)$ and the vector $\boldsymbol{\beta}$ is the vector of the ANN parameters. The reasoning behind this assumption is that the soil decontamination cost can get values only in the range $[0,\infty)$. The expectation of the cost is given by

$$E\left[y\right]=\exp\left\{\left(\ln f\left(\mathbf{x},\boldsymbol{\beta}\right)-\frac{\sigma^2}{2}\right)+\frac{\sigma^2}{2}\right\}=f\left(\mathbf{x},\boldsymbol{\beta}\right) \tag{4.9}$$

which equals to the output of the ANN. Assuming a training data set with M samples, $\hat{\boldsymbol{\xi}}^k=\left(\hat{y}^k,\hat{\mathbf{x}}^k\right)$ and $k=1,\ldots,M$ (the hat symbol indicates collected data), the likelihood function of the training data set $\left(\hat{\boldsymbol{\xi}}^1,\ldots,\hat{\boldsymbol{\xi}}^M\right)$ is given by

$$\mathcal{L}\left(\hat{\boldsymbol{\xi}}^1,\ldots,\hat{\boldsymbol{\xi}}^M|\boldsymbol{\theta}\right)=\prod_{k=1}^{M}p(\hat{y}^k|\hat{\mathbf{x}}^k,\boldsymbol{\theta}) \tag{4.10}$$

where the RHS of (4.10) equals to

$$\left(2\pi\sigma^2\right)^{-\frac{M}{2}}\prod_{k=1}^{M}\frac{1}{\hat{y}^k}\exp\left\{\sum_{k=1}^{M}-\frac{\left(\ln\hat{y}^k-\ln f\left(\hat{\mathbf{x}}^k,\boldsymbol{\beta}\right)+\frac{\sigma^2}{2}\right)^2}{2\sigma^2}\right\} \tag{4.11}$$

Next, the posterior distribution of parameters $\boldsymbol{\theta}$ is calculated. In order to do that, first of all the prior distribution of the parameters $\boldsymbol{\beta}$ and the data dispersion σ^2 has to be defined. In the BNN model developed in this research it is assumed that each parameter β_p, where $p=1,\ldots,s$ has a Gaussian distribution $\mathscr{N}\left(\tilde{\beta}_p,\tilde{\sigma}_p^2\right)$ with expected value $\tilde{\beta}_p$ and variance $\tilde{\sigma}_p^2$. The values for the $\tilde{\beta}_p$ are taken from the parameters of the ANN, after training. Because correlation exists among the parameters β_p, it is desirable to formulate the distribution of the parameter vector $\boldsymbol{\beta}$ with a proper multi-dimensional density function. However, because information on the covariance between the parameters β_p is not available, it is assumed that the parameters are *iid*. It should be noted that the effects of this assumption vanish as the number of samples used in the model increases. Assuming that the prior distributions of the parameters are *iid* does not prove to be a problem in practical applications as the number of sample data used in the model increases.

In addition to that, in Bayesian statistics, many models in order to ease the Bayesian estimation calculations follow similar assumptions. This way the results from the ANN calculated in Process 1 together with assuming simple prior probability density functions makes the calculation of the soil decontamination cost risk possible. Hence the prior probability densities of the BNN model's parameters β_p are given by

$$\pi\left(\beta_p\right) = \left(2\pi\tilde{\sigma}_p^2\right)^{-\frac{1}{2}} \exp\left\{-\frac{\left(\beta_p - \tilde{\beta}_p\right)^2}{2\tilde{\sigma}_p^2}\right\} \tag{4.12}$$

The BNN model developed in this research uses the MCMC method for sampling from the posterior distribution of the parameters. When the MCMC method is used and the dispersion of the data is small the sampling efficiency decreases. In order to prevent the standard deviation $\tilde{\sigma}_p$ from becoming very small it is assumed that the data dispersions are directly proportional to the parameter expected values

$$\tilde{\sigma}_p = \alpha\tilde{\beta}_p \tag{4.13}$$

where α is a constant. Naturally, the influence that the variance of the prior distribution exerts on the soil decontamination cost risk decreases gradually as the number of samples increases. Thus, taking all of the above into consideration, the prior distribution of the BNN model's parameters becomes

$$\pi\left(\beta_p\right) = \left(2\pi\alpha^2\tilde{\beta}_p^2\right)^{-\frac{1}{2}} \exp\left\{-\frac{\left(\beta_p - \tilde{\beta}_p\right)^2}{2\alpha^2\tilde{\beta}_p^2}\right\} \tag{4.14}$$

The data dispersion σ^2 is an important parameter in order to calculate the soil decontamination cost risk. Since there is no prior information on the type of the distribution of the data dispersion a non-informative, Jeffreys distribution, is assumed for the prior distribution of σ^2 (Jeffreys, 1961)

$$\pi\left(\sigma^2\right) \propto \sigma^{-2} \tag{4.15}$$

Using the likelihood function of the training data given the model's parameters and the prior distributions of the parameters $\boldsymbol{\beta}$ and σ^2 defined above, the posterior distribution of $\boldsymbol{\beta}$ is

$$\pi\left(\boldsymbol{\beta}|\sigma^2, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right) \propto \mathcal{L}\left(\hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M|\boldsymbol{\beta}, \sigma^2\right) \prod_{p=1}^{s} \pi\left(\boldsymbol{\beta}_p\right) \tag{4.16}$$

that is proportional to

$$\left(\sigma^2\right)^{-\left(\frac{M}{2}\right)} \exp\left\{\sum_{p=1}^{s} -\frac{\left(\beta_p - \tilde{\beta}_p\right)^2}{2\alpha^2\tilde{\beta}_p^2} - \sum_{k=1}^{M} \frac{\left(\ln\hat{y}^k - \ln f\left(\hat{\mathbf{x}}^k, \boldsymbol{\beta}\right) + \frac{\sigma^2}{2}\right)^2}{2\sigma^2}\right\} \tag{4.17}$$

and the posterior distribution of σ^2 is

$$\pi\left(\sigma^2|\boldsymbol{\beta},\hat{\boldsymbol{\xi}}^1,\ldots,\hat{\boldsymbol{\xi}}^M\right)\propto\mathcal{L}\left(\hat{\boldsymbol{\xi}}^1,\ldots,\hat{\boldsymbol{\xi}}^M|\boldsymbol{\beta},\sigma^2\right)\pi\left(\sigma^2\right) \tag{4.18}$$

that is proportional to

$$\left(\sigma^2\right)^{-\left(\frac{M}{2}+1\right)}\exp\left\{-\sum_{k=1}^{M}\frac{\left(\ln\hat{y}^k-\ln f\left(\hat{\mathbf{x}}^k,\boldsymbol{\beta}\right)+\frac{\sigma^2}{2}\right)^2}{2\sigma^2}\right\} \tag{4.19}$$

This concludes the formulation of the BNN model.

4.3.5 Soil decontamination cost risk evaluation formulation

Risk evaluation methodology

Bayes' rule requires calculation of the normalizing constant, or evidence, as shown in (4.5). In many cases in Bayesian statistics, analytical calculation of the normalizing constant is infeasible and this fact became an obstacle in utilizing Bayesian statistics. Recently, with the introduction of MCMC methods in Bayesian statistics the calculation of the normalizing constant became unnecessary since direct sampling from the posterior distribution can be done instead. Typical sampling algorithms used in MCMC include the Metropolis-Hastings (MH) algorithm and the Gibbs sampler. The MH algorithm is a very general sampling algorithm and Gibbs sampling can be considered as a special case of MH. The Gibbs sampler can be implemented more easily in general and is more efficient to use than the MH algorithm but it requires that the full conditional densities of the parameters to be known. Calculating the full conditionals of the parameters in BNN is a very complicated task and, in most models, infeasible. In addition to that, in order to use the Gibbs sampler it is necessary that the log of the posterior density function be a concave function. In BNN models there is no guarantee that the posterior density of the parameters is concave so the Gibbs sampler is not a good choice for sampling from the posterior density of the parameters. As a result the more general MH algorithm is used to sample from the posterior density of the parameters of the BNN model.

The Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm is a method that allows sampling from probability densities that are very difficult to directly sample from. The posterior distribution of the parameters of BNN models, $\pi\left(\boldsymbol{\theta}|\boldsymbol{\xi}^1,\ldots,\boldsymbol{\xi}^M\right)$, falls in this category hence it is considered for the model developed in this research. The main requirement of the MH algorithm for calculating from a probability density function $f(x)$ is that the density can be calculated at point x. The starting point of the algorithm is to define the objective density f. A conditional density $q\left(y|x\right)$, that sampling can be easily done from it, is then chosen. If the conditional density, also called the proposal or instrumental distribution, is symmetric, i.e. $q\left(y|x\right) = q\left(x|y\right)$, then the algorithm simplifies considerably. If the proposal distribution is not symmetric then the ratio $f(y)/q(y|x)$ has to be known up to a constant independent of x. Following the steps in the algorithm, a Markov chain $\left(X^{(t)}\right)$ is produced that eventually converges to the invariant distribution, i.e. the chain becomes stationary. When this point is reached, sampling from the target distribution can commence. The generic form of the algorithm has the following steps:

- Start with $x^{(0)}$, then iterate
- Propose y from $q\left(x^{(t)}, y\right)$
- Calculate ratio $\alpha = \frac{\pi(y)q(y,x^{(t)})}{\pi(x^{(t)})q(x^{(t)},y)}$
- If $\alpha > 1$ accept $x^{(t+1)} = y$
  Else accept with probability α
  If rejected, $x^{(t+1)} = x^{(t)}$

If the proposal distribution, $q(y, x^{(t)})$, is symmetric, i.e. $q(y, x^{(t)}) = q(x^{(t)}, y)$, then the the ratio simplifies to $\alpha = \frac{\pi(y)}{\pi(x^{(t)})}$.

The MH algorithm for the BNN model is described as follows. First, initial values for the parameters $\boldsymbol{\beta}^{(0)} = \left(\beta_1^{(0)}, \ldots, \beta_s^{(0)}\right)$ and $\left(\sigma^2\right)^{(0)}$ are set. Next, samples for the parameters $\left(\boldsymbol{\beta}^{(i)}, \left(\sigma^2\right)^{(i)}\right)$, where $i = 1, 2, \ldots$, are obtained according to the following procedure. Let's assume that we are at iteration $(i-1)$ and that samples $\left(\boldsymbol{\beta}^{(i-1)}, \left(\sigma^2\right)^{(i-1)}\right)$ were obtained. The next step is to generate candidate samples $\bar{\boldsymbol{\beta}}, \bar{\sigma^2}$ from the proposal distributions q_p $(p = 1, \ldots, s), t$. A new candidate parameter sample $\bar{\beta}_p$ is generated at random from the proposal distribution $q_p\left(\beta_p^{(i)}\right)$ that can be used to generate the parameter sample $\beta_p^{(i)}$. The variates $\bar{\beta}_p$ are drawn from Gaussian distributions with mean $\beta_p^{(0)}$ and variance $\left(\alpha\beta_p^{(0)}\right)^2$

$$\bar{\beta}_p \sim q_p\left(\beta_p^{(0)}\right) = \frac{1}{\sqrt{2\pi}\alpha\beta_p^{(0)}} \exp\left\{-\frac{\left(\bar{\beta}_p - \beta_p^{(0)}\right)^2}{2\left(\alpha\beta_p^{(0)}\right)^2}\right\} \tag{4.20}$$

Because the proposed sample $\bar{\beta}_p$ is not directly from the target distribution, $\boldsymbol{\beta}^{(i)} = \bar{\boldsymbol{\beta}}$ is accepted with probability

$$\mathcal{A}\left(\boldsymbol{\beta}^{(i-1)}, \bar{\boldsymbol{\beta}} | \left(\sigma^2\right)^{(i-1)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right) = \min\left[\frac{\pi\left(\bar{\boldsymbol{\beta}} | \left(\sigma^2\right)^{(i-1)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right)}{\pi\left(\boldsymbol{\beta}^{(i-1)} | \left(\sigma^2\right)^{(i-1)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right)}, 1\right] \tag{4.21}$$

If the sample is rejected the new state stays in the current state, i.e. $\boldsymbol{\beta}^{(i)} = \boldsymbol{\beta}^{(i-1)}$. In reality a uniform variate from the uniform distribution $u \sim \mathscr{U}(0,1)$ is generated at the same time the proposal sample is generated and if $u \leq \mathcal{A}$ then the proposed sample $\bar{\boldsymbol{\beta}}$ is accepted. Next sampling from the variance posterior distribution is conducted. The proposal distribution for the variance is the exponential distribution with parameter $\left(\sigma^2\right)^{(0)}$

$$\bar{\sigma}^2 \sim \mathscr{E}xp\left(\left(\sigma^2\right)^{(0)}\right) = \frac{1}{\left(\sigma^2\right)^{(0)}} \exp\left\{-\frac{\bar{\sigma}^2}{\left(\sigma^2\right)^{(0)}}\right\} \tag{4.22}$$

Similar to the procedure followed earlier, because the proposed sample $\bar{\sigma}^2$ is not directly from the target distribution, $\left(\sigma^2\right)^{(i)} = \bar{\sigma}^2$ is accepted with probability

$$\mathcal{B}\left(\left(\sigma^2\right)^{(i-1)}, \bar{\sigma}^2 | \boldsymbol{\beta}^{(i)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right) = \min\left[\frac{\pi\left(\bar{\sigma}^2 | \boldsymbol{\beta}^{(i)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right)}{\pi\left(\left(\sigma^2\right)^{(i-1)} | \boldsymbol{\beta}^{(i)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right)}, 1\right] \tag{4.23}$$

If the sample is rejected the new state stays in the current state, i.e. $\left(\sigma^2\right)^{(i)} = \left(\sigma^2\right)^{(i-1)}$. In reality a uniform variate from the uniform distribution $u \sim \mathscr{U}(0,1)$ is generated at the same time the proposal sample is generated and if $u \leq \mathcal{B}$ then the proposed sample $\bar{\sigma}^2$ is accepted. Following is a summary of the procedure.

Step 1 Initial values for the parameters $\boldsymbol{\beta}^{(0)} = \left(\beta_1^{(0)}, \ldots, \beta_s^{(0)}\right)$ and $\left(\sigma^2\right)^{(0)}$ are set from the values of ANN parameters after training. Iteration counter is set to $i = 1$.

Step 2 Samples $\bar{\boldsymbol{\beta}} = \left(\bar{\beta}_1, \ldots, \bar{\beta}_s\right)$ are drawn from the proposal distribution, as described in (4.20), in order to calculate $\boldsymbol{\beta}^{(i-1)} = \left(\beta_1^{(i-1)}, \ldots, \beta_s^{(i-1)}\right)$.

Step 3 The probability of acceptance is calculated

$$\mathcal{A}\left(\boldsymbol{\beta}^{(i-1)}, \bar{\boldsymbol{\beta}} | \left(\sigma^2\right)^{(i-1)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right) = \min\left[\exp\left(\sum_{p=1}^{s}\left(\frac{\left(\beta_p^{(i-1)} - \tilde{\beta}_p\right)^2 - \left(\bar{\beta}_p - \tilde{\beta}_p\right)^2}{2\alpha^2\tilde{\beta}_p^2}\right) + \Phi\right), 1\right] \tag{4.24}$$

where Φ is

$$\sum_{k=1}^{M} \exp\left[\frac{\left(\ln \frac{\hat{y}^k}{f\left(\hat{\mathbf{x}}^k,\boldsymbol{\beta}^{(i-1)}\right)} + \frac{\left(\sigma^2\right)^{(i-1)}}{2}\right)^2 - \left(\ln \frac{\hat{y}^k}{f\left(\hat{\mathbf{x}}^k,\bar{\boldsymbol{\beta}}\right)} + \frac{\left(\sigma^2\right)^{(i-1)}}{2}\right)^2}{2\left(\sigma^2\right)^{(i-1)}}\right] \tag{4.25}$$

Step 4 A variate from the uniform distribution $u \sim \mathscr{U}(0,1)$ is generated and $\boldsymbol{\beta}^{(i)}$ is accepted according to the following rule

$$\boldsymbol{\beta}^{(i)} = \begin{cases} \bar{\boldsymbol{\beta}} & \text{if } u \leq \mathcal{A} \\ \boldsymbol{\beta}^{(i-1)} & \text{Otherwise} \end{cases} \tag{4.26}$$

Step 5 A new sample $\bar{\sigma}^2$ is drawn from the proposal distribution, as described in eq. (4.22), in order to calculate $\left(\sigma^2\right)^{(i-1)}$.

Step 6 The probability of acceptance is calculated

$$\mathcal{B}\left(\left(\sigma^2\right)^{(i-1)}, \bar{\sigma}^2 | \boldsymbol{\beta}^{(i)}, \hat{\boldsymbol{\xi}}^1, \ldots, \hat{\boldsymbol{\xi}}^M\right) = \min\left[\left(\frac{\bar{\sigma}^2}{\left(\sigma^2\right)^{(i-1)}}\right)^{-\left(\frac{M}{2}+2\right)} \exp\left\{\frac{\bar{\sigma}^2}{\left(\sigma^2\right)^{(i-1)}} - \frac{\left(\sigma^2\right)^{(i-1)}}{\bar{\sigma}^2} + \Theta\right\}, 1\right] \tag{4.27}$$

where Θ equals to

$$\sum_{k=1}^{M} \exp\left\{\frac{\left(\ln \frac{\hat{y}^k}{f\left(\hat{\mathbf{x}}^k,\boldsymbol{\beta}^{(i)}\right)}\right)^2}{2\left(\sigma^2\right)^{(i-1)}} - \frac{\left(\ln \frac{\hat{y}^k}{f\left(\hat{\mathbf{x}}^k,\boldsymbol{\beta}^{(i)}\right)}\right)^2}{2\bar{\sigma}^2}\right\} \tag{4.28}$$

Step 7 A variate from the uniform distribution $u \sim \mathscr{U}(0,1)$ is generated and $\left(\sigma^2\right)^{(i)}$ is accepted according to the following rule

$$\left(\sigma^2\right)^{(i)} = \begin{cases} \bar{\sigma}^2 & \text{if } u \leq \mathcal{B} \\ \left(\sigma^2\right)^{(i-1)} & \text{Otherwise} \end{cases} \tag{4.29}$$

Step 8 The process is repeated for a large number of iterations until enough samples are gathered, enough to make Monte Carlo estimates. For the estimates, usually the last 70% to 80% of accepted samples are used to make sure that the algorithm converged to the invariant distribution. Here we denote the starting point of the sample set used for the estimates as $\underline{N}$ and the ending point as $\overline{N}$.

The predictive distribution

Let's assume that new data from soil decontamination investigations, $\hat{\mathbf{x}}^{M+1}$, were collected from simple topsoil contamination investigations and previous land usage history of the land. From those data we would like to get the soil decontamination cost y^{M+1}. At this point we assume that we have successfully sampled from the posterior probability density of the parameters, $\pi\left(\boldsymbol{\theta}|\hat{\boldsymbol{\xi}}^{1},\ldots,\hat{\boldsymbol{\xi}}^{M}\right)$, using MH as described earlier. Let us assume that the sample we got for the posterior of the parameters is $\boldsymbol{\theta}^{(i)}=\left(\boldsymbol{\beta}^{(i)},\left(\sigma^{2}\right)^{(i)}\right)$, where $i=\underline{N}+1,\ldots,\overline{N}$. The sample we got from the MH process is a close representation of the true posterior of the parameters. The probability density of the soil decontamination cost given the new data inputs and the parameters is then Log-Normally distributed with mean equal to the output of the ANN, $f\left(\hat{\mathbf{x}}^{M+1},\boldsymbol{\beta}^{(i)}\right)$, given by

$$p\left(y^{M+1}|\hat{\mathbf{x}}^{M+1},\boldsymbol{\theta}^{(i)}\right)=\left\{2\pi\left(\sigma^{2}\right)^{(i)}\left(y^{M+1}\right)^{2}\right\}^{-\frac{1}{2}}\exp\left\{-\frac{\left(\ln y^{M+1}-\ln f\left(\hat{\mathbf{x}}^{M+1},\boldsymbol{\beta}^{(i)}\right)+\frac{\left(\sigma^{2}\right)^{(i)}}{2}\right)^{2}}{2\left(\sigma^{2}\right)^{(i)}}\right\} \tag{4.30}$$

Finally the cost predictive probability density, i.e. the soil decontamination cost risk, can be calculated through

$$\rho\left(y^{M+1}|\hat{\mathbf{x}}^{M+1},\hat{\boldsymbol{\xi}}^{1},\ldots,\hat{\boldsymbol{\xi}}^{M}\right)=\int p\left(y^{M+1}|\hat{\mathbf{x}}^{M+1},\boldsymbol{\theta}\right)\pi\left(\boldsymbol{\theta}|\hat{\boldsymbol{\xi}}^{1},\ldots,\hat{\boldsymbol{\xi}}^{M}\right)d\boldsymbol{\theta} \tag{4.31}$$

This concludes the formulation of the SD-CORE model.

4.4 Case Study

The model developed in this research was tested with data from soil decontamination projects conducted in Japan. Even though the decontamination and remediation of contaminated sites follows a rapid upward trend in Japan, especially after the passage of the SCCL, data from soil contamination projects related to cost are very sparse and difficult to get. The limited amount of data collected and used in this case study are from soil decontamination projects that utilize removal and out-of-site decontamination of contaminated soil. Following is the overview of the processes followed in the case study.

4.4.1 Setting up the database

In order to evaluate the soil decontamination cost risk of future soil decontamination projects, using the BNN model presented in this research, a database was set and populated with the collected data. The data used in the case study came from simple soil contamination investigations, i.e. from historical land usage and topsoil contamination investigations. Out of the dataset seven data categories were chosen for the network structure. Those categories were assigned to the input variables of the ANN and BNN models as shown in Table 4.2. The data

Table 4.2: Description of the explanatory variables

Variable	Description
x_1	Soil hardness indicator (Dummy Variable)
x_2	Land used for dry cleaning or gas station up to 1000 m^2 (DV)
x_3	Land used for housing, commercial or office (DV)
x_4	Land used for industrial purposes or for disposing construction material (DV)
x_5	Contaminated land area (m^2)
x_6	Density of VOC from topsoil investigation (mg/l)
x_7	Density of heavy metals from topsoil investigation (mg/l)

were then split into those to be used for the training of the ANN and those for testing and verifying the output of the model. The database also was designed to store the values of the trained ANN parameters and the posterior probability distributions of the parameters of the BNN model.

4.4.2 The ANN process

This process involved the creation of the network structure of the ANN model and its training and verification. A network structure was designed based on the dataset (i.e. input variables) which was utilized by both the ANN and BNN models. The network was structured as a single hidden layer MLP consisting of 8 inputs (including the input for the bias to the hidden layer), 3 units at the intermediate layer (including the input for the bias to the output layer) and one output. The network structure of the ANN model can be seen in Figure 4.4. The ANN model was then trained using data from the database that were set aside for this purpose. The result of the training process is shown in Figure 4.6. The vertical axis shows the total output error of the ANN and the horizontal the iteration number. The trained ANN was then tested

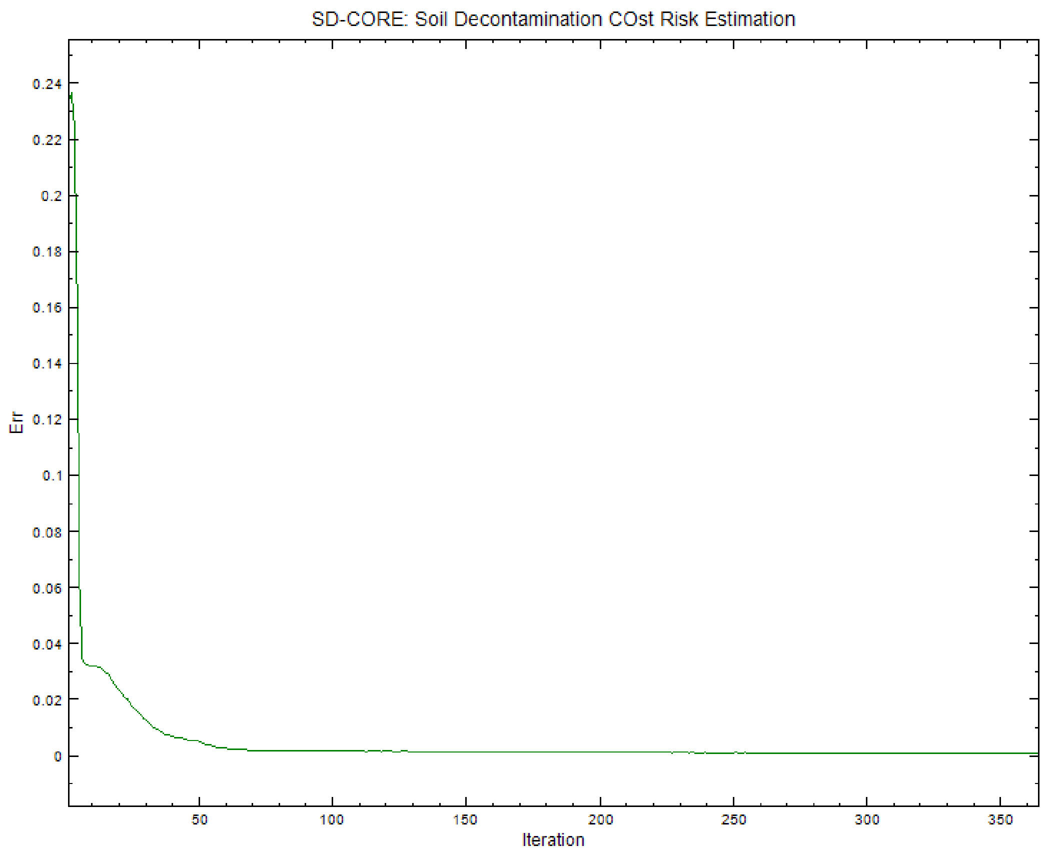


Figure 4.6: Training process of the ANN model

with the verification data from the database. Once the trained ANN was verified, the values of its parameters were saved in the database to be used in the BNN model.

4.4.3 The BNN process

First, the parameters of the trained ANN were used to set up the prior distributions of the BNN's parameters. The sampling process was then initiated and samples from the posteriors of the BNN's parameters were collected. The results of the sampling process for all the parameters of the BNN model are shown in Table 4.3. When the sampling from the posterior of the parameters concluded, the predictive probability distribution of the cost for a future soil decontamination project given previously unseen input data, shown in Figure 4.7, was calculated. The mean of the predictive probability density of the soil decontamination cost for that case was 28.9 million Yen, whereas the actual soil decontamination cost was 25 million Yen. A comparison of the model's outputs as compared with other actual soil decontamination cost values are shown in Table 4.4.

Table 4.3: The model's parameters

Parameter	Mean Value	Dispersion
β_0	-0.697363508	0.037399577
β_1	0.117154314	0.000135268
β_2	-0.233727944	0.000599281
β_3	0.113015389	0.000145864
β_4	-0.573195554	0.003757479
β_5	4.061491064	0.135761751
β_6	1.353246711	0.017941547
β_7	-0.093457456	9.91E-05
β_8	-0.9868551	0.008731877
β_9	-1.581805744	0.030142949
β_{10}	-0.149048847	0.000241109
β_{11}	0.954447686	0.012013972
β_{12}	-0.849955223	0.010727311
β_{13}	7.562170522	0.251405088
β_{14}	-1.956174091	0.06739005
σ^2	3.877281877	0.25205143

Table 4.4: Comparison of the model's output with real cost

Sample ID	Actual Cost (million Yen)	Predicted Cost Mean (million Yen)
1	50	41.26841237
2	43	48.02913563
3	150	156.4315085
4	25	28.94503391
5	66	32.51719072
6	50	35.43150483
7	7	27.30256871
8	25	31.04073101
9	47	41.2881269
10	241	89.04181715

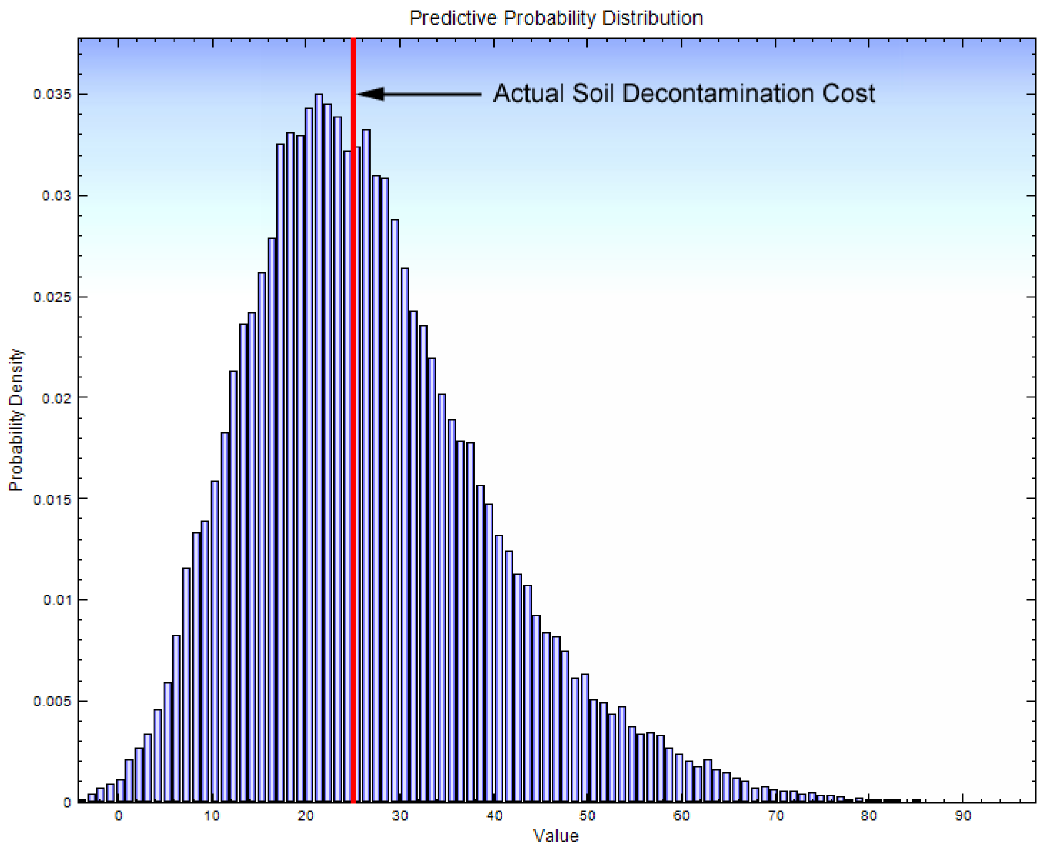


Figure 4.7: Soil decontamination cost risk output from SD-CORE

4.5 Conclusions

In this research a model was developed to be used at the conceptual stage of investing in contaminated land. The model was developed within the machine learning Bayesian framework, utilizing both ANN and BNN. Using historical land usage data as well as data from simple topsoil contamination investigations, the model generates the soil decontamination cost risk. The model was tested with a small dataset consisting of data collected from soil decontamination projects conducted in Japan. Even though the dataset used in the case study was small, fair results were obtained. This can be partially explained from the fact that the Bayesian approach does not require the model's structure to be dependent on the amount of data available. In the Bayesian framework prior probabilities are assumed first and when data arrive the posterior probabilities are calculated based on the priors and the likelihood given the data. The structure of a model developed in the Bayesian framework is indifferent on the amount of data available, it works for few data and for a lot of data the same way. On the other hand, machine learning depends on the amount and quality of data. Large datasets containing a wide range of data will typically enable machine learning models to generalize better (Bishop, 1995).

The model could possibly have applications in civil engineering as well as in the insurance industry e.g. for calculating the cost risk premium of the development of contaminated land. The methodology of calculating the soil decontamination cost risk developed in this research can be utilized in evaluating the risk of other phenomena, with some modifications in the model's structure. The model can be also utilized as a component of evaluating the risk of more complex phenomena such in the evaluation of the total risk of land development investments. For example, in land development investments on top of the soil decontamination cost risk several other types or risk exist, such as land price, housing value, construction cost and market risk. Some of the risks are interrelated and care should be taken in creating a risk valuation model that calculates the total risk of investment. It could be possible to include this model as a component of a real option model that can be used in evaluating the total risk of land development investments.

Topics of future research include the testing of the model with more complex network structures, e.g. with two hidden layers having several units and letting the BNN to adapt to the data using techniques such as the Automatic Relevance Determination (Neal, 1996). Numerous data from various soil decontamination methods could also be included in the database. Using data from a smorgasbord of methods of soil decontamination will enable the model to generalize better. Another topic of future research could be to re-evaluate the assumptions made and discover more efficient approaches. For example the soil decontamination cost risk was assumed to be Log-Normally distributed. Other assumption for the density of the cost could give better results. Also other sampling methods could be utilized. The model utilized the MH algorithm. The MH is considered to be a not so efficient sampler because it results in random walks in the parameter hyper-space. Other algorithms could be considered such the Hybrid-MH algorithm that uses a mixture of Gibbs sampler and MH algorithm for some of the parameters.

Chapter 5

Case Study: AGORA

5.1 Introduction

5.1.1 The Need for Open Standards in Risk Analysis

Losses to the built environment due to natural and technological hazards present a heavy burden on a global scale. The costs from the impacts of natural weather related disasters alone follow a dramatically upward path. Data from the Intergovernmental Panel on Climatic Change (IPCC) show that the global loss from climatic disasters has increased from US$8.9 billion (annual average of the period 1977-1986) to U.S.$45.1 (annual average of the period 1997-2006) (IPCC, 2007). Recent data from EM-DAT (2008) and Munich-Re NatCat-Service regarding the intensities and losses from natural disasters also confirm the upward trend (Hoeppe, 2008). As the world's population continues to grow and the population density of mega-cities, especially in Asia and the developing world, that are vulnerable to extreme natural disasters follow an irrevocable upward path, the need for a good understanding of the forces behind the natural hazards and how to well manage and mitigate their risk to the built environment becomes a necessity. Catastrophe risk modeling is central to understanding, managing, and mitigating the impacts of natural and technological hazards to the built environment. Research related to natural and technological catastrophes saw a remarkable growth in the last two decades attributed mainly to the technological advancement of computers. Researchers and practitioners are now able to use powerful supercomputers and sophisticated software to aid them in their research. Researchers use a variety of software in their daily job activities, either to control electronic and mechanical devices, to do simulations, experiments and to record, analyze and process data. Software typically used by researchers is either commercial pre-packaged or custom-made. Custom-made software represents in fact the lion's share of the software used in research. Custom-made software created in the academia, has been following the traditional rules of knowledge dissemination and peer review. That is, in order for the scientific community to validate the results of a specific research, its data, tools (e.g. software) and methodologies have to be transparent and accessible for review.

This approach was the norm in the early times of computing and software development. Recently, with the emergence and proliferation of commercial proprietary software, this norm has been pushed aside. This raised concerns among the research and software engineering and scientific communities that eventually resulted in the emergence of a new methodology to develop software that preserves the traditional ways of knowledge dissemination and provides transparency and accessibility. This new software development approach is referred to as Free/Open Source Software (F/OSS). This section analyses the current needs of researchers and practitioners in the disaster risk management field in view of models, methodologies and data in conjunction with the current trends in the field and the emergence of the F/OSS phenomenon. The section presents a new approach of dealing with those needs, called Open Risk Analysis (ORA). The ORA model was developed within AGORA, a global virtual organization based on the open collaboration paradigm initiated by the Open Source movement. The section discusses AGORA, the Open Risk model and an example of a software tool that is been developed within AGORA called the Mitigation Information and Risk Identification System (MIRISK).

5.1.2 Catastrophe Risk Modeling

Researchers of natural hazards risk currently have needs that can be well addressed by the Open Source paradigm. Needs such as a collaborative platform, open data and methodologies, rapid development of tools and software, re-use of software code and effective peer review are essential. It can be argued that all of those needs can be effectively addressed utilizing the F/OSS development model. The following subsection discusses in more detail the needs of the researchers and practitioners in the field and provides some suggestions on how those needs can be addressed by the Open Source development model.

Catastrophe risk modeling (cat-modeling) refers to the use of mathematical models used to estimate the performance of assets such as buildings subjected to various hazards, in terms of economic costs, human safety, and loss of use i.e. "dollars, deaths, and downtime" Scawthorn (2006). It is not unusual for researchers to develop computer software based on the mathematical models in order to aid them in their research. Practitioners and social planners are using cat-models predominately for decision-making. In general, a typical cat-model consists of four analytical stages:

- **An exposure model** describes the assets (e.g. buildings) that are vulnerable to some natural hazard (e.g. strong earthquake ground motion), given the assets' characteristics. The model assigns a geographic location and site soil classification to an asset based on its given recorded location (e.g. prefecture, city, street). For the case of a building asset, the model may estimate the building's replacement cost based on some measurable parameter (e.g. square footage). The model also could quantify and propagate uncertainties in an asset's value, location, etc.

- **A hazard model** describes probabilistically the hazard imposed on the assets, such as earthquake shaking intensities for earthquakes, wind-speeds for tropical cyclones, water height for floods, etc.

- **A vulnerability model** estimates the physical damage or loss to the asset as a result of the exposure to a natural hazard.

- **A financial loss model** relates the damage or loss to financial impacts.

Currently, cat-models consider earthquake, tropical cyclone, and flood as far as natural hazards are concerned. To a lesser extend landslide, tornado, hail, winter storms and volcanic activity hazards are also considered. In the technological and man-made hazards domain, blasts from impact and explosions are mainly modeled with a recent trend in modeling terrorist attacks to infrastructure. An excellent review of the history of computerized cat-models is offered by Scawthorn (2006).

Cat-models are extensively used in various industries. Applications of cat-models can be found in the finance, insurance and real estate industries. In those industries the cat-models are used to aid in the decision making process related to mortgage underwriting (e.g. whether a lender should require earthquake insurance), insurance and reinsurance transactions (e.g. whether and how much insurance or reinsurance to buy, and at what price), and the credit-worthiness of insurers and re-insurers (e.g. whether the investor in such a firm is likely to lose an investment because of the insurer's liability after a natural disaster). Cat-models are also used in the public sector to inform decisions about emergency planning and disaster response and mitigation. An example of a software tool developed in the public sector that is currently widely used is HAZUS.

Cat-modeling software tends to involve sophisticated sub-models from various fields of expertise. For example, earthquake models involve seismology, geotechnical and structural engineering, economics, etc., and integrate highly advanced and usually costly systems such as GIS and real time data gathering and monitoring. As a result, cat modeling software tends to be also costly to produce. Commercial cat-model licenses cost on the order of $1 million per year per seat. Because of its value as intellectual property, the source code for these models is closed, i.e., unavailable to users. A notable exception is the binaries of HAZUS which are available for free of charge. HAZUS, even though is available free of charge, its source code is closed and is not made available for studying it and potentially quickly improving it and adapting to various needs of its users. In addition to that the principles and data involved in the HAZUS software is largely public information and it seems odd that something built using public funds based on freely available public data is not made fully available back to the public. Commercial cat-modelers allow outsiders to view their source code only under carefully controlled conditions such as in response to regulatory requirements (e.g., the Florida Commission on Hurricane Loss Projection Methodologies) and are not allowed to reuse or modify the code.

5.1.3 Open Risk Analysis

Because of the closed-source nature of cat-models, researchers use them only under limited circumstances, usually without the ability to modify their underlying methods. Consumers of commercial cat-models likewise are unable to see or modify for themselves the underlying methods, which can lead to serious concerns about the models' dependability when different models produce dramatically different estimates of risk from the same input data. Kishi shows how the three principal cat-modeling firms estimated industry losses from hurricane Katrina that varied between them on the order of a factor of 3 at any given time and changed by up to a factor of 5 over the space of a few weeks (Kishi, 2007). The commercial cat-models also change over longer time spans, with periodic new releases that incorporate the modeler's perception of the best new science and data. The modifications sometimes result in dramatic changes to modeled risk for a given portfolio. For reasons of commercial competitiveness and perhaps because of the effort involved, modelers sometimes provide frustratingly limited explanation about these modifications. Furthermore the new science can take some time to find its way into the models. For all these reasons, there seems to be a need for cat models whose methods and software are open to inspection and rapid modification, referred to here as Open Risk Analysis. Open Risk Analysis was initiated by the Alliance for Global Open Risk Analysis or AGORA for short.

5.1.4 AGORA: The Platform for Open Risk Analysis

AGORA is a nonprofit organization that was established in early 2007 following the 1st International Workshop on Open-Source Risk Software, held at the California Institute of Technology in early 2007. AGORA was initiated by approximately 35 scholars and professionals in Japan, the US, and Europe who are themselves working on open-source risk software. AGORA is contributing to reducing the impacts from natural and technological hazards through open risk modeling. The main motivation behind forming AGORA was to enable and sustain the creation and development of open methods, tools (e.g. software) and data for the risk analysis of technological and natural hazards.

The mission of AGORA is to promote and coordinate Open Risk Analysis of natural and technological hazards, and the development of open-source risk software and methodologies to perform end-to-end risk modeling. End-to-end refers to the modeling of hazardous events and their impacts, from the event occurrence through site effects, physical damage to the built environment, to economic and human impacts. The AGORA framework states that to effectively manage the risk of natural and technological disasters open analyses and models are required. Technological advancements, especially in computers and electronics, are emerging at an increasing pace (*cf.* Moore's law) and the traditional modes of developing computer tools that integrate available data and models sadly lag in leveraging new work. In addition to that, researchers and practitioners increasingly lack risk-integration tools needed for understanding and mitigating risk.

The main novelty provided by AGORA is the formation of a specialized interdisciplinary platform for experts involved in the risk analysis and mitigation of natural disasters that are interested in sharing the fruits of their research, methodologies, data and tool in creating an open and transparent paradigm of collaborative content creation. The collaborative platform of AGORA offers a place where its members can effectively communicate and share their research utilizing the power of Internet and the methodologies of Open Source and the "bazaar" development approach. By participating in AGORA a researcher can associate with an open community sharing common researching goals. In addition to that, by being open and highly connected (i.e. via the Internet), AGORA receives a constant visiting flow of researchers from various disciplines and backgrounds resulting in a diverse community. Diversity is considered the cultivating ground of innovation. Additional fruits from participating at AGORA include the access to the methods, data and software tools that were developed collaboratively by the members of AGORA and to a larger extent, the Open Source community.

The AGORA platform has a layered access design. Each layer is distinguished by the type and level of access to the AGORA's content. The lowest layer, level 0, is publicly accessible. In that layer mostly general information, announcements explanatory and promotional documents can be found. The next layer, level 1, is accessible only by registered members of AGORA. In principle anybody can register with AGORA, given that he or she agrees with the terms of usage. Registered members can download all documents and executable of the software created within AGORA with the exception of the source code of software. In order for someone to download the source code of software developed within AGORA, he or she has to become a full member (level 2) by actively contributing content, e.g. software, data, methodologies, research papers, etc. Finally, the highest layer, level 3, is dedicated for the administration of AGORA. The layered structure of AGORA serves several purposes. First, the registration requirement acts as a filter allowing only those that are seriously interested in AGORA to join. As a result it lowers free riding. Second, in order to enlarge the knowledge base and social capital of AGORA access to the software source code and data is granted only to those members willing to contribute content to AGORA.

Within AGORA several Open Risk Analysis modeling software are being developed. This section will discuss one such software tool called Mitigation Information and Risk Identification System or MIRISK for short.

5.1.5 Open Risk Analysis Software Case Study: MIRISK

Introduction

MIRISK was initiated and developed at Kyoto University for the World Bank with assistance from the Government of Japan under Japan Consultant Trust Fund (JCTF). The project was organized as follows. The steering committee of the MIRISK provided the overall guidance and management of the project. The core development and programming of MIRISK was conducted by the methods/programming group. Finally, MIRISK was reviewed by natural hazard risk management specialists both from the academia and the World Bank.

MIRISK is a computer-based analytical guidance tool for infrastructure risk assessment and mitigation. It provides information on natural hazards design guidelines, norms and good practices by allowing users to identify the natural hazards related to a development project, the typical vulnerabilities of each infrastructure and to recommend a normal design and mitigation plan for each infrastructure asset. MIRISK is designed to aid the decision makers in reaching optimum decisions by providing means to consider natural hazards in the following ways: (i) identifying natural hazards affecting a region, (ii) defining the kinds of infrastructure assets that make up typical development projects, (iii) describing the vulnerability of these assets to natural hazards, and how vulnerability can be reduced and (iv) analyzing the natural hazards and vulnerability data, to assess whether projects should follow normal design practices, or whether the cost of some enhanced design for natural hazards is justified by the benefits (of avoided losses).

MIRISK System Overview

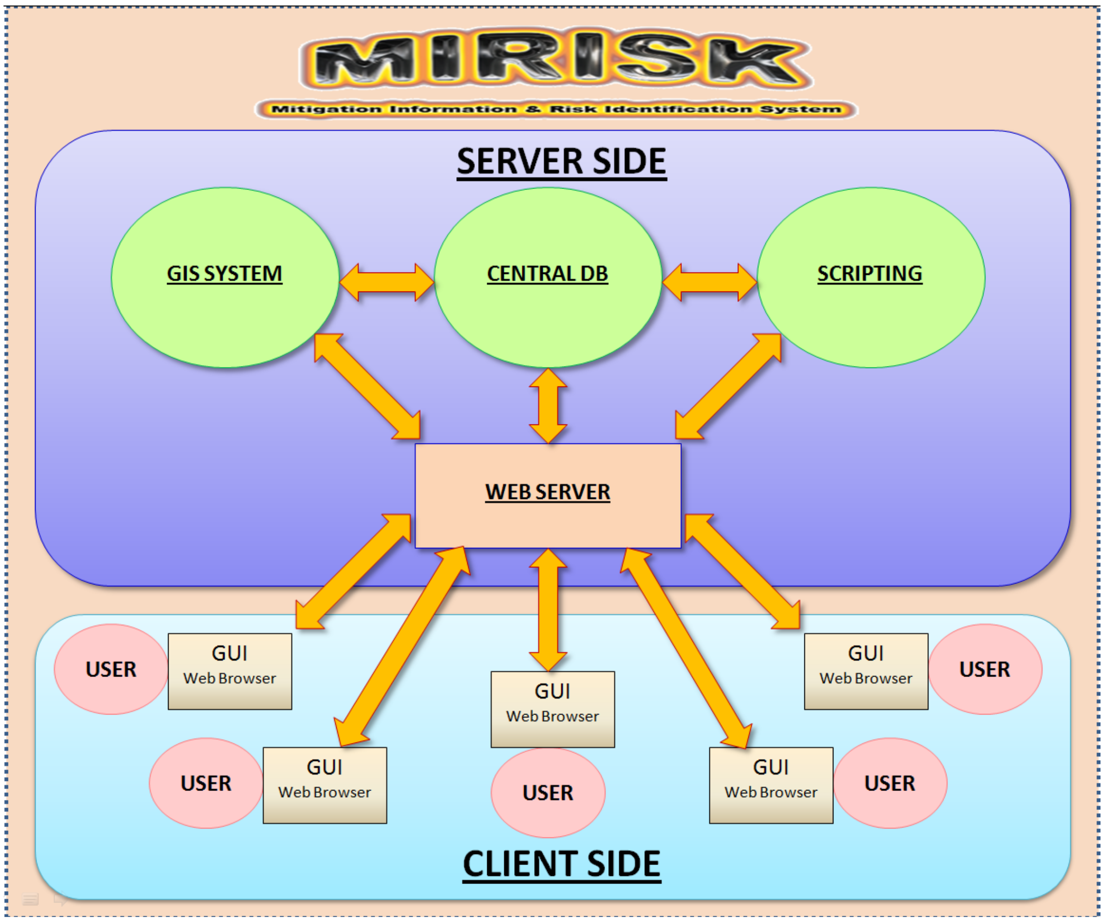
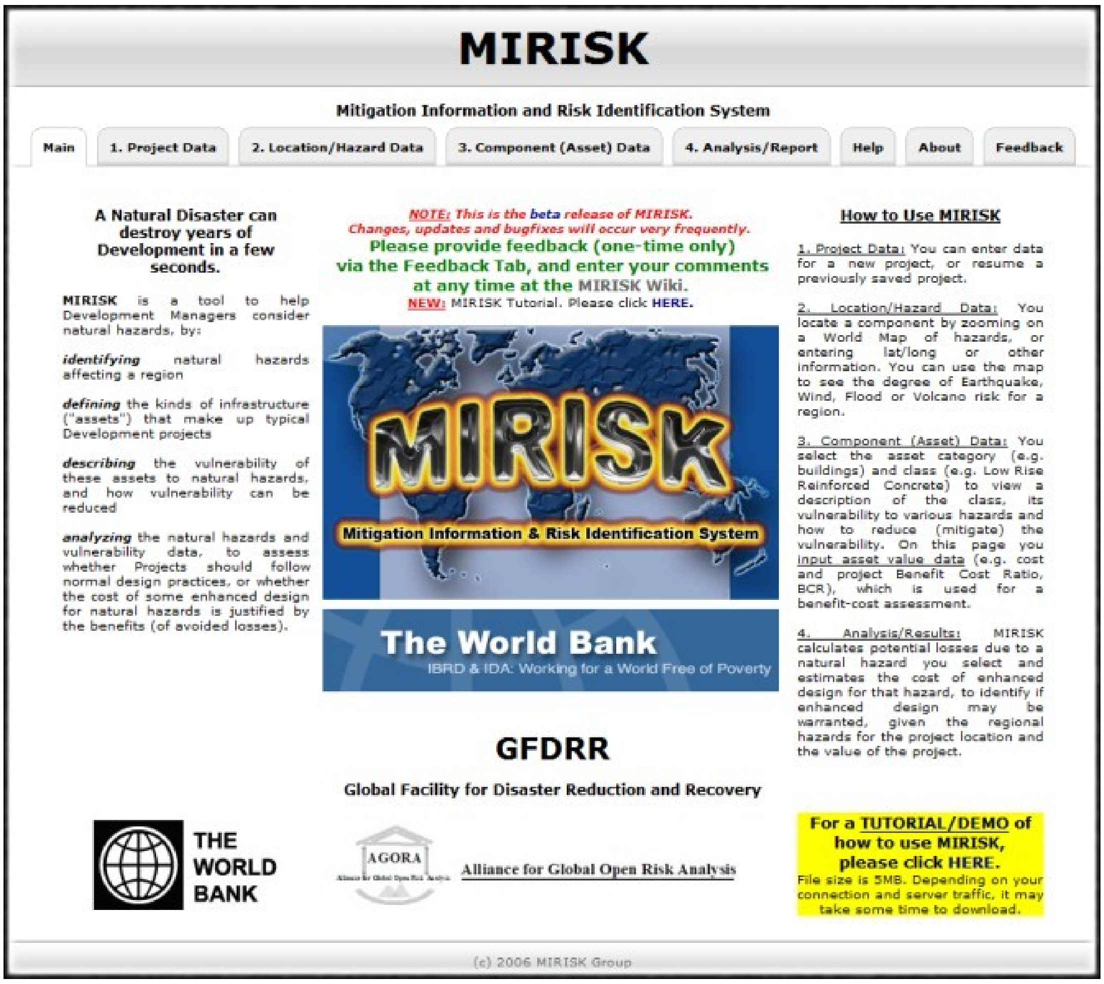


Figure 5.1: MIRISK System Overview

Figure 5.1 shows the overview of the MIRISK system. On the server side, MIRISK uses GNU/Linux as its base operating system, specifically a specialized GNU/Linux distribution with added GIS capabilities called HostGIS Linux (HostGIS, 2008). The MIRISK system consists of several components. Central to MIRISK is a relational database which stores all the data needed for the deployment of the tool including geographical data, hazard descriptions, asset vulnerabilities and mitigation data as well as data provided by the users to be used for a


MIRISK

Mitigation Information and Risk Identification System

Main | 1. Project Data | 2. Location/Hazard Data | 3. Component (Asset) Data | 4. Analysis/Report | Help | About | Feedback

A Natural Disaster can destroy years of Development in a few seconds.

MIRISK is a tool to help Development Managers consider natural hazards, by:

identifying natural hazards affecting a region

defining the kinds of infrastructure ("assets") that make up typical Development projects

describing the vulnerability of these assets to natural hazards, and how vulnerability can be reduced

analyzing the natural hazards and vulnerability data, to assess whether Projects should follow normal design practices, or whether the cost of some enhanced design for natural hazards is justified by the benefits (of avoided losses).

NOTE: This is the beta release of MIRISK.
Changes, updates and bugfixes will occur very frequently.
Please provide feedback (one-time only) via the Feedback Tab, and enter your comments at any time at the MIRISK Wiki.
NEW: MIRISK Tutorial. Please click HERE.

MIRISK
Mitigation Information & Risk Identification System
The World Bank
IBRD & IDA: Working for a World Free of Poverty

GFDRR

Global Facility for Disaster Reduction and Recovery

How to Use MIRISK

1. Project Data: You can enter data for a new project, or resume a previously saved project.

2. Location/Hazard Data: You locate a component by zooming on a World Map of hazards, or entering lat/long or other information. You can use the map to see the degree of Earthquake, Wind, Flood or Volcano risk for a region.

3. Component (Asset) Data: You select the asset category (e.g. buildings) and class (e.g. Low Rise Reinforced Concrete) to view a description of the class, its vulnerability to various hazards and how to reduce (mitigate) the vulnerability. On this page you input asset value data (e.g. cost and project Benefit Cost Ratio, BCR), which is used for a benefit-cost assessment.

4. Analysis/Results: MIRISK calculates potential losses due to a natural hazard you select and estimates the cost of enhanced design for that hazard, to identify if enhanced design may be warranted, given the regional hazards for the project location and the value of the project.

THE WORLD BANK

AGORA Alliance for Global Open Risk Analysis

Alliance for Global Open Risk Analysis

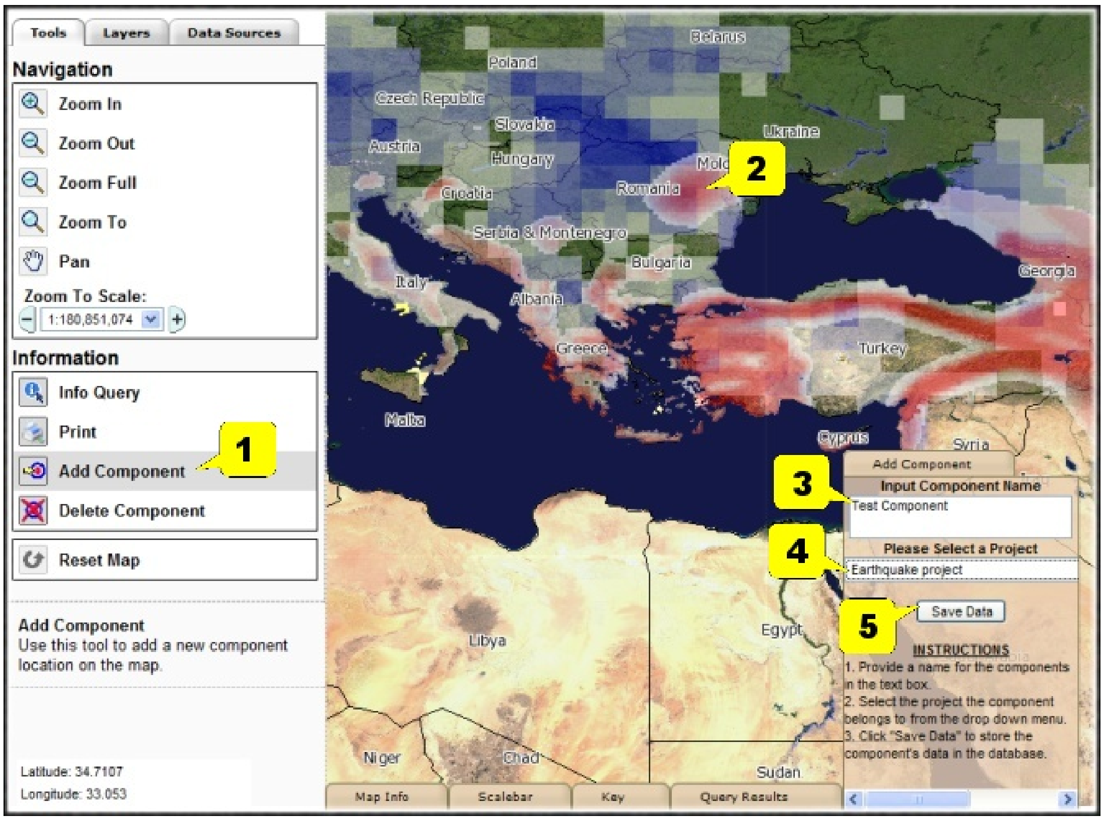
For a TUTORIAL/DEMO of how to use MIRISK, please click HERE.
File size is 5MB. Depending on your connection and server traffic, it may take some time to download.

(c) 2006 MIRISK Group

Figure 5.2: The Opening screen of MIRISK

simple cost benefits assessment. In order to effectively provide natural hazard maps and other geographical information to the users to assist them in locating the site of a future project and identifying the hazards that exist at the site, a GIS system was implemented which is closely integrated with the relational database and the other components of MIRISK. The graphical user interface (GUI) of MIRISK, the analysis module as well as data handling and calculation interfaces are deployed using F/OSS scripting languages such as PHP, JavaScript and HTML. MIRISK resides at a web server which is deployed using the widely used F/OSS Apache Web Server. On the client side, the users can access the MIRISK system by simply using their web browsers.

MIRISK was built based on client-server architecture in order to be able to serve its users at various locations simultaneously and to be easily accessible for updating and maintenance. In addition to that, the GUI of MIRISK comes with a comprehensible tabbed interface (see Figures 5.2 and 5.3) in order to facilitate ease of learning and convenience of use. For the making of MIRISK only F/OSS and open datasets are used. By using F/OSS and open datasets the positive characteristics of the Open Source approach, namely rapid development, lower costs, source code reuse, transparency and ease of customization were utilized. In addition to that, using F/OSS, we were able to connect to the large community of F/OSS developers and utilize its vast knowledge base.



Zoom To Scale: 1:180,851,074

Add Component
Use this tool to add a new component location on the map.

Latitude: 34.7107
Longitude: 33.053

Input Component Name: Test Component

Please Select a Project: Earthquake project

INSTRUCTIONS
1. Provide a name for the components in the text box.
2. Select the project the component belongs to from the drop down menu.
3. Click "Save Data" to store the component's data in the database.

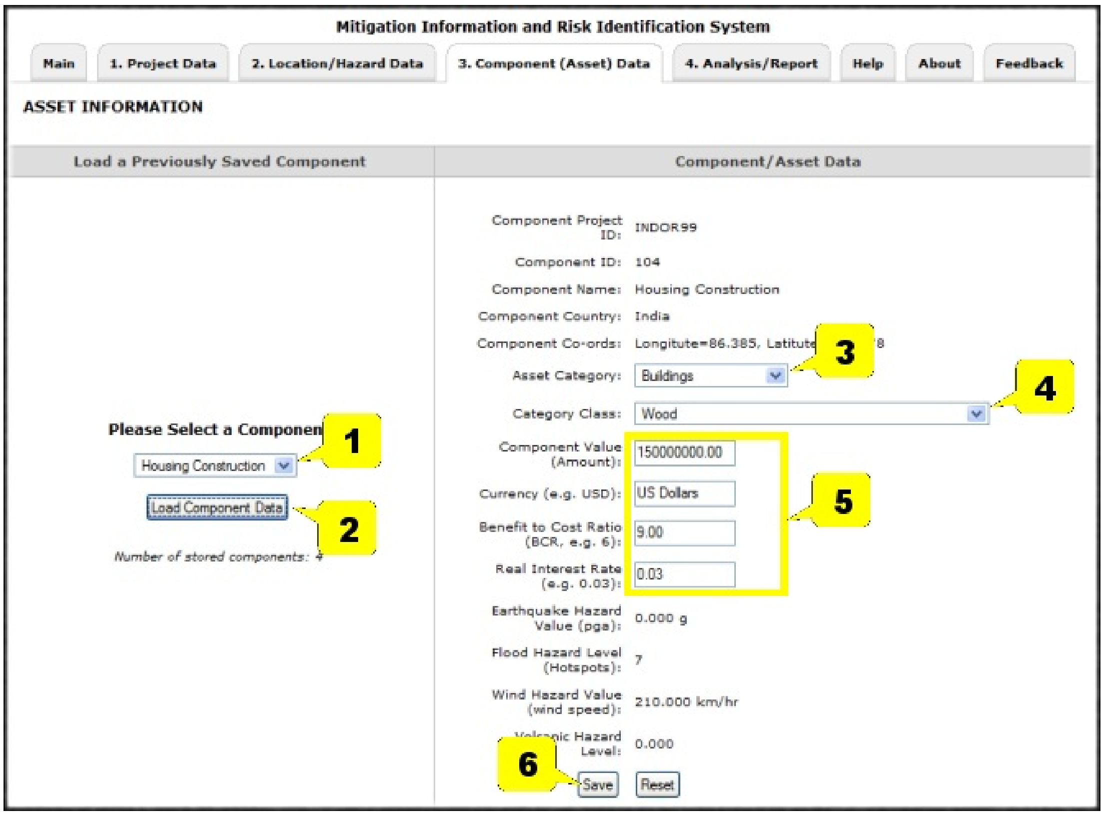
Figure 5.3: The GIS Graphic User Interface of MIRISK

MIRISK Datasets and Components

For the making of MIRISK several open and freely available datasets were used. Data from the Global HotSpots project was used as the basic dataset in the generation of the hazard maps of earthquake, flood, tropical cyclone and volcanic activity (Dilley et al., 2005). More detailed datasets such as the earthquake dataset from the GSHAP project are used for specific hazards (Giardini et al., 1999). For generating the cyclone hazard maps the Global Cyclone Hazard Frequency datasets from the UNEP/GRID-Geneva PreView were used (UNEP/GRID, 2008). The Global Cyclone Hazard Frequency and Distribution is a 2.5 minute grid based on more than 1,600 storm tracks for the period January 1st, 1980 through December 31st, 2000 for the Atlantic, Pacific, and Indian Oceans. The wind-speeds around storm tracks were modeled using Holland's model to assess the grid cells likely to have been exposed to high wind levels (Holland, 1997). This dataset is the result of collaboration among the Columbia University Center for Hazards and Risk Research (CHRR), International Bank for Reconstruction and Development and The World Bank, United Nations Environment Programme Global Resource Information Database Geneva (UNEP/GRID-Geneva), and Columbia University Center for International Earth Science Information Network (CIESIN). For generating the flood hazard maps datasets from the World Atlas of Flooded Lands of the Dartmouth Flood Observatory were used (DFO, 2008). In order to compile the volcanic activity hazard maps, datasets from Smithsonian's Global Volcanism Program were used (Smithsonian, 2008). Finally, raster map images were used from NOAA's National Geophysical Data Center (NGDC) (NOAA, 2008) and NASA's Earth Observatory (NASA, 2008).

MIRISK Methodologies

The analysis module of MIRISK aims to provide a quantitative estimate of incremental cost given project design level, cost of repair, duration of disruption, and benefit cost (see Figure 5.5). The module provides general information on the costs of such "normal" and "superior" design and tabulates the overall costs and benefits for direct and indirect impacts such that the output could be used in a project planning document. The methodology behind generating the output of MIRISK's analysis module is based on simple cost-benefit analysis. The main objective of the analysis was to determine the most cost-effective project design level of an asset considering at the same time the impacts of natural hazards on the asset. In order to achieve that, first a database that classifies assets into various taxonomies was generated. The taxonomy database and associated vulnerability data were based on the ATC-13 dataset (ATC-13, 1985). The ATC-13 dataset was developed specific to California construction. For MIRISK, some adjustments were made since most MIRISK applications are in developing countries. It should be noted that most MIRISK applications are for new construction per modern building codes. That is, MIRISK applications are not for existing construction. MIRISK's asset taxonomy database is divided into three general categories (buildings, transportation and utilities) which are subdivided into 32 classes such as reinforced concrete or masonry buildings, highways, bridges, electric substations and storage tanks. The vulnerability information provided by MIRISK is classified in two kinds: descriptive, consisting of narrative and photographs, so that users can understand the nature of the asset's vulnerability, and quantitative (see Figures 5.4 and 5.5). The quantitative vulnerability information is employed to estimate the present value of all future economic losses



ASSET INFORMATION

Please Select a Componen

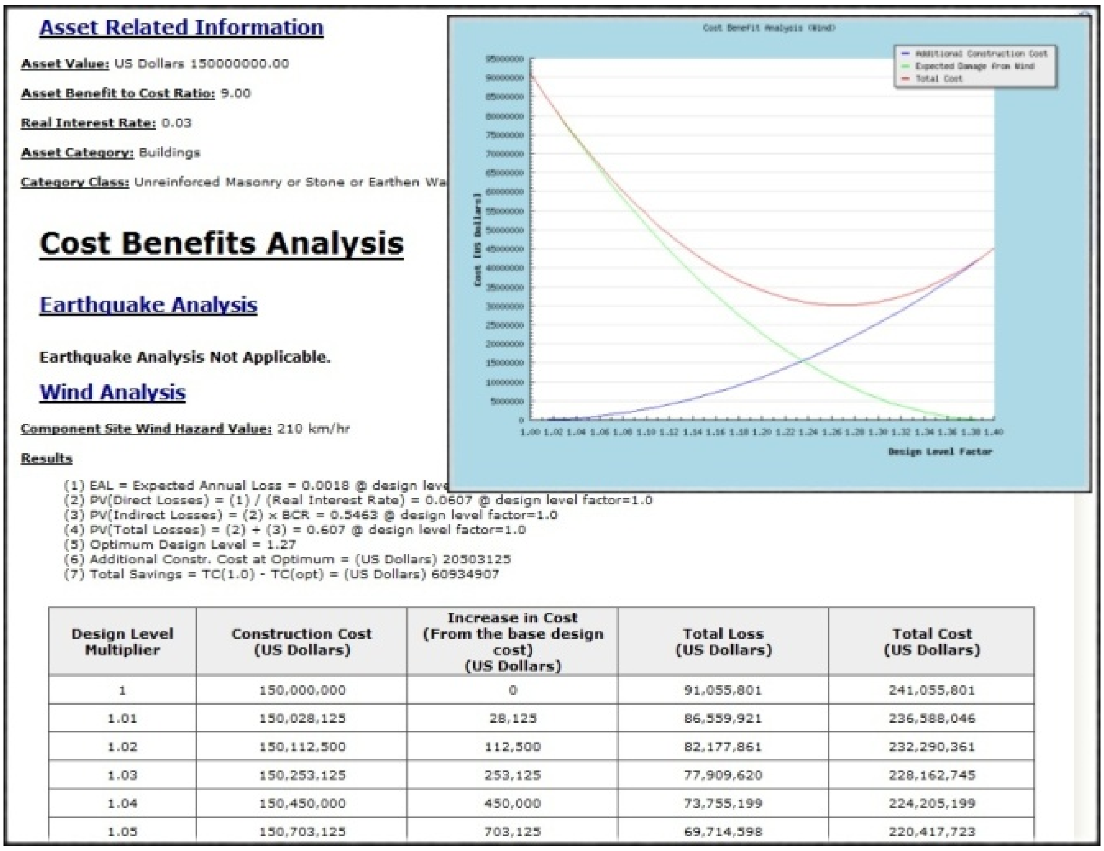
Housing Construction

Number of stored components: 4

Component Project ID:	INDOR99
Component ID:	104
Component Name:	Housing Construction
Component Country:	India
Component Co-ords:	Longitute=86.385, Latitute 8
Asset Category:	Buildings
Category Class:	Wood
Component Value (Amount):	150000000.00
Currency (e.g. USD):	US Dollars
Benefit to Cost Ratio (BCR, e.g. 6):	9.00
Real Interest Rate (e.g. 0.03):	0.03
Earthquake Hazard Value (pga):	0.000 g
Flood Hazard Level (Hotspots):	7
Wind Hazard Value (wind speed):	210.000 km/hr
Volcanic Hazard Level:	0.000

Figure 5.4: Providing Asset information to MIRISK

due to natural hazards. Economic losses refer to direct costs of repair to damage, and also to indirect costs due to loss of use. This present value of damage is then employed, together with data on associated initial investment, in a benefit-cost framework to estimate an optimum (least total cost) level of design for the asset. The output is presented both in tabulated and graph form with the optimum design level highlighted. The analysis methodology follows a standard benefit-cost framework, in which normal code design is taken as a given baseline. Normal building codes are written to assure life safety, and not to minimize property damage, so that significant property damage is likely to be sustained by even new construction designed per modern building codes, when subjected to higher intensity natural hazards. This is well-known (Hamburger, 2003), and has led to the recent emergence of performance-based design in the structural engineering field (SEAOC, 1996). Consequently, there is a substantial risk of economic loss given normal building code design. This loss is not only direct property loss (ie, cost of repair) but also the attendant loss of use and associated expenses (so-called indirect loss). Direct loss is accounted for by using the vulnerability functions of ATC-13 for the specific category class, and the hazard as determined for the site from the Hotspots data. The hazard is however only a point estimate, and the entire hazard curve is calculated based on the assumption



Asset Related Information

Asset Value: US Dollars 150000000.00

Asset Benefit to Cost Ratio: 9.00

Real Interest Rate: 0.03

Asset Category: Buildings

Category Class: Unreinforced Masonry or Stone or Earthen Wa

Cost Benefits Analysis

Earthquake Analysis

Earthquake Analysis Not Applicable.

Wind Analysis

Component Site Wind Hazard Value: 210 km/hr

Results

```
(1) EAL = Expected Annual Loss = 0.0018 @ design leve
(2) PV(Direct Losses) = (1) / (Real Interest Rate) = 0.0607 @ design level factor=1.0
(3) PV(Indirect Losses) = (2) x BCR = 0.5463 @ design level factor=1.0
(4) PV(Total Losses) = (2) + (3) = 0.607 @ design level factor=1.0
(5) Optimum Design Level = 1.27
(6) Additional Constr. Cost at Optimum = (US Dollars) 20503125
(7) Total Savings = TC(1.0) - TC(opt) = (US Dollars) 60934907
```

Design Level Multiplier	Construction Cost (US Dollars)	Increase in Cost (From the base design cost) (US Dollars)	Total Loss (US Dollars)	Total Cost (US Dollars)
1	150,000,000	0	91,055,801	241,055,801
1.01	150,028,125	28,125	86,559,921	236,588,046
1.02	150,112,500	112,500	82,177,861	232,290,361
1.03	150,253,125	253,125	77,909,620	228,162,745
1.04	150,450,000	450,000	73,755,199	224,205,199
1.05	150,703,125	703,125	69,714,598	220,417,723

Figure 5.5: Analysis Output of MIRISK

that the hazard frequency follows the Ishimoto-Iida law (i.e., log-log relation) with a slope of 1.0 (Ishimoto and Iida, 1939). Indirect loss is accounted for by multiplying the direct loss by the Benefit-Cost Ratio (BCR) for the component, as input by the user. The expected annual loss (EAL) for the component is then calculated by numerical integration of the vulnerability and hazard functions, over all values of hazard. The present value of all future losses is then calculated as the EAL divided by the real interest rate (this assumes an infinite economic life, which slightly overestimates the present value, but is not a bad assumption in general). The above algorithm can be applied for normal code design (denoted as DLF, or Design Level Factor, 1.0), or for any enhanced design. If the normal code requirement for a natural hazard is increased by 10% (e.g. earthquake design is for 0.22g, rather than for 0.20g, lateral force), this is denoted as a DLF=1.1. If a component is designed for a DLF=1.1, damage due to natural hazards will be significantly decreased, with only a marginally higher capital investment for the increased design. As DLF increases, damage drops sharply at first, and then diminishingly less, while capital investment is very modest at first, and then increases rapidly. For any given DLF, the sum of the present value of all future damages, and the increase in capital investment, is the total cost above normal building code design. The curve of total cost as a function of DLF has a typical U-shape, where the minimum total cost is the optimum DLF. MIRISK calculates the present value of all future damages, the increase in capital investment, the total cost above normal building code design, and presents these results in summary form for earthquake, tropical cyclone and flood, and in a detailed tabulation (with the optimum DLF highlighted), and graph. If any hazard is particularly low, the results are not presented, but simply it is noted that that particular hazard is “not applicable”. The optimum DLF is what makes economic sense. While the MIRISK data and methods are only preliminary, they provide guidance as to what hazards exist for a site, what the natural hazards performance is likely to be for the component to be constructed, what is an optimum DLF for the situation, what should be budgeted by the decision makers to achieve an economic optimum performance considering natural hazards, and what the savings are given this policy. Real interest rates vary, but an average value of 0.03 per annum is a reasonable value in many cases.

The analysis module of MIRISK aims to provide a quantitative estimate of incremental cost given project design level, cost of repair, duration of disruption, and benefit cost. The module provides general information on the costs of such “normal” and “superior” design and tabulates the overall costs and benefits for direct and indirect impacts such that the output could be used in a project planning document. In order to estimate the loss (damage) to a specific type of asset (structure) the following two assumptions are made:

1. Loss is estimated based on a specified base design level.

2. Hazards are analyzed independent of each other.

The metric of damage used is the Expected Annual Loss (EAL) which is defined as the average loss per year due to the occurrence of a natural hazard. In order to estimate EAL the following three assumptions are made:

1. Loss (damage) is calculated using the ATC-13 Mean Damage Factor methodology: $L = f(A)$, where L is the loss function and A is the measure of the hazard's intensity.

2. The hazard curve follows the Ishimoto-Iida relationship, i.e. the natural logarithm of the probability of exceedance of the hazard's intensity, P_E, is linearly dependent on the hazard intensity, A,: $\ln(P_E) = a - bA$.

3. Total damage is defined as direct damage plus indirect damage where direct damage is given by the construction cost times the EAL and indirect cost is given by the direct

damage times the benefits to cost ratio (BCR) of the project.

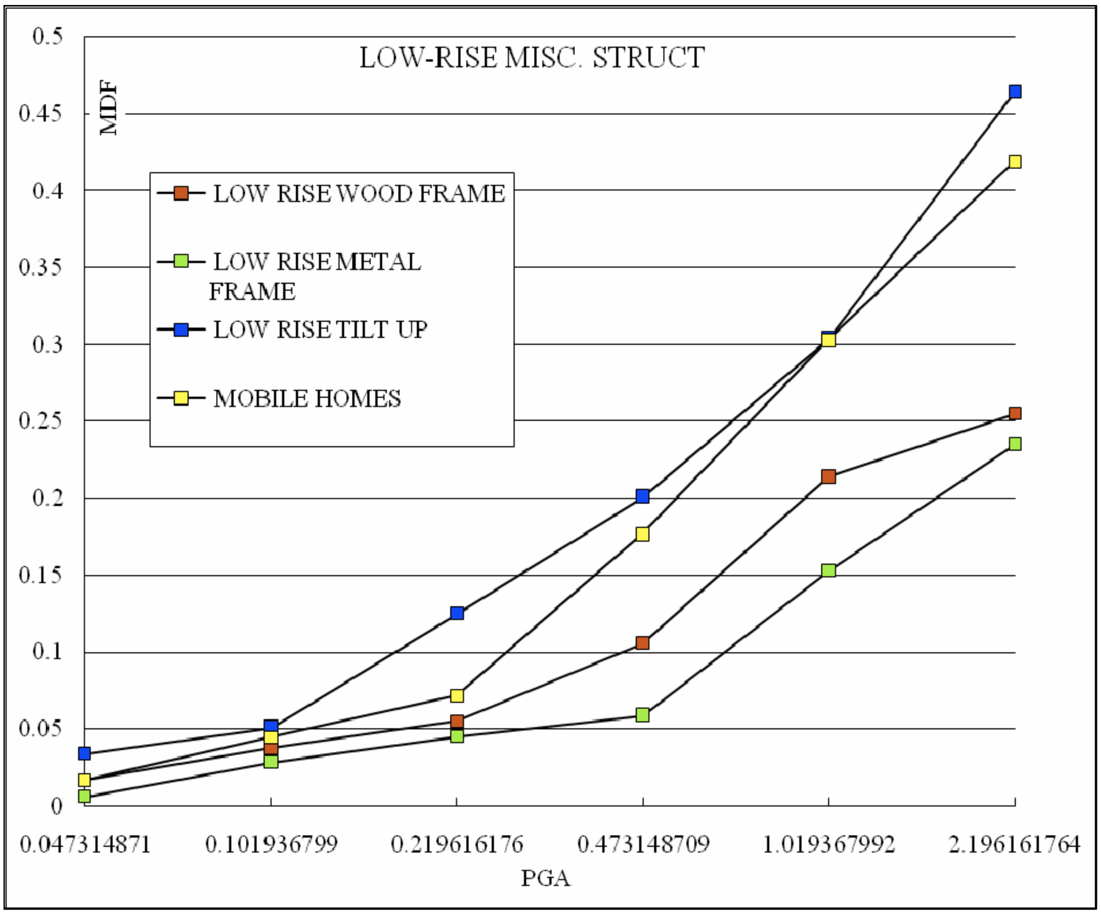


Figure 5.6: Typical damage curve

A typical damage curve for a given hazard (earthquake) assuming a specific structure constructed at a specified design level is shown in Figure 5.6. The general form of EAL is expressed by the equation

$$EAL = \sum_{D} \sum_{H} P(L|D)P(D|H)P(H) \tag{5.1}$$

where H, D and L are the hazard measure, damage and loss respectively and $P(L|D)$ is the conditional probability of loss given damage, $P(D|H)$ is the conditional probability of damage given hazard and $P(H)$ is the probability of hazard. Using the assumptions stated earlier, EAL is given by

$$EAL = \int_{-\infty}^{+\infty} L(A)p(A)dA \tag{5.2}$$

where L is the loss function, A is the hazard intensity and $p(A)$ is the probability density function of the hazard's intensity. The value of $p(A)$ can be derived from the hazard curve. The equation of the hazard curve is given by

$$\ln(P_E) = a - bA \tag{5.3}$$

where a and b are constants. Solving for the probability of exceedance, P_E, yields

$$P_E = e^{a-bA} \tag{5.4}$$

The probability density of hazard, $p(A)$, is then given by

$$p(A) = \frac{d}{dA}(1 - P_E) = \frac{d}{dA}\left(1 - e^{a-bA}\right) = be^{a-ba} \tag{5.5}$$

EAL can then be calculated using numerical integration.

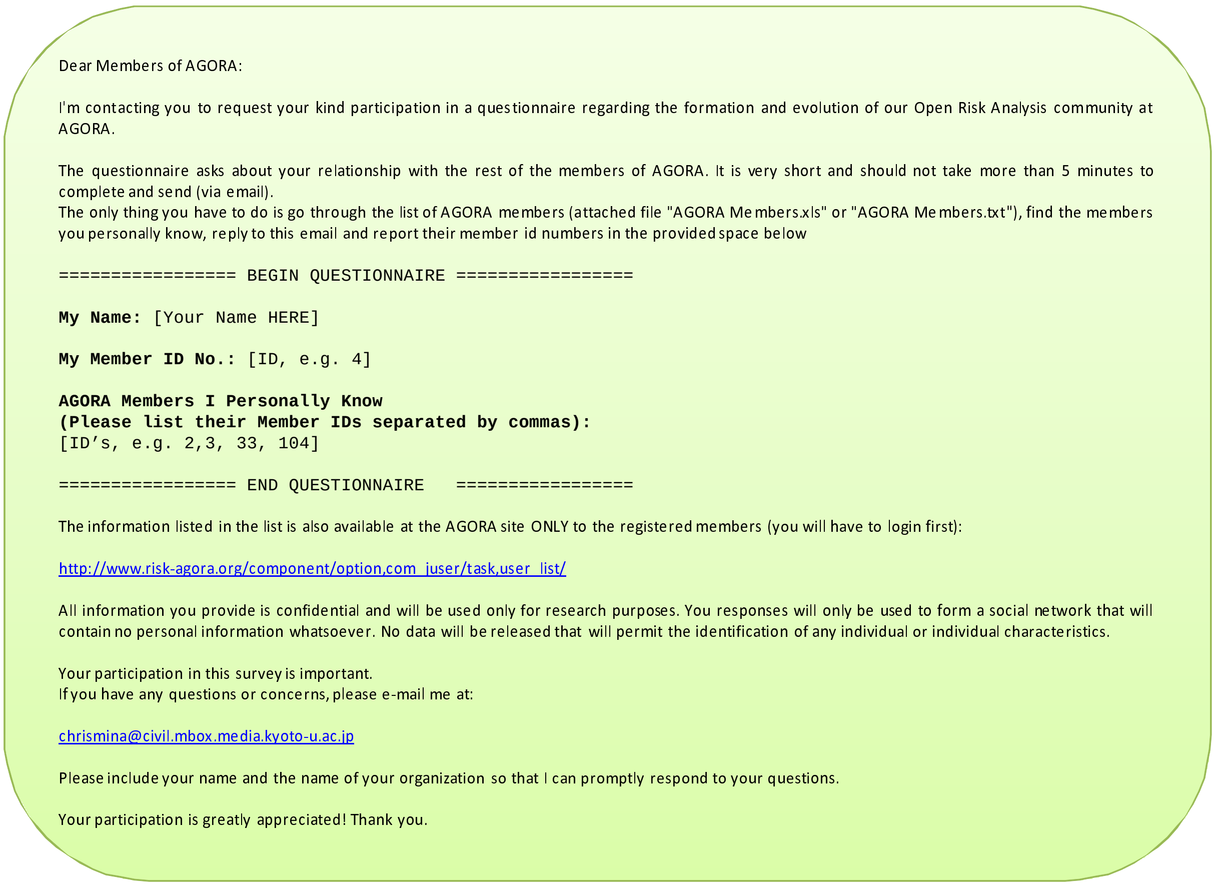
5.2 Outline of the case study

The collaboration network of AGORA presents a paradigm of specialized scientific collaboration based on the F/OSS collaboration model. Typically, members of AGORA meet and collaborate online. The website of AGORA, utilizing a F/OSS content management system (CMS), acts as the virtual platform[1] of all activity. The website provides among others a software repository, scientific papers and software documentation, announcements of Open Risk related conferences and an online forum where members can communicate with each other on various topics related of risk analysis. A very important motivation behind creating AGORA is to bring together researchers, practitioners and experts in the field of risk analysis interested in open standardization and allow them to get to know each other and collaborate. Registered members of AGORA have the ability to see all the public information of the rest of the members and contact them in the forum. All the data utilized by the CMS of AGORA are stored in a relational database on the server of AGORA. For this case study we used the user data from the AGORA website along with data collected from questionnaires to create and analyze the social collaboration network of AGORA from over its lifetime. To analyze the collaboration network of AGORA we utilized complex social network methodologies similar to those used in analyzing the SourceForge.net collaboration network presented in Chapter 3.

The AGORA database contains data about the AGORA's registered users, including the time stamps of their registration. This allowed us to investigate the evolution of AGORA's collaboration network. In order to conduct complex SNA on AGORA's collaboration we needed additional information about the relationships of its registered users, i.e. we needed data to

[1]As stated on the AGORA website, AGORA means a "place of congregation", in this case, an international forum for open-source software, data and methodologies for multihazard risk modeling.

construct the adjacency matrix. In order to achieve that we conducted a questionnaire survey by contacting all the registered members of AGORA via e-mail and requested them to identify the members of AGORA they know in person. Along with the email questionnaire a list of all registered members of AGORA with their affiliations was also attached. Figure 5.7 shows the questionnaire e-mail that was sent to the members of AGORA. Following the e-mailing



Dear Members of AGORA:

I'm contacting you to request your kind participation in a questionnaire regarding the formation and evolution of our Open Risk Analysis community at AGORA.

The questionnaire asks about your relationship with the rest of the members of AGORA. It is very short and should not take more than 5 minutes to complete and send (via email).
The only thing you have to do is go through the list of AGORA members (attached file "AGORA Members.xls" or "AGORA Members.txt"), find the members you personally know, reply to this email and report their member id numbers in the provided space below

```
================= BEGIN QUESTIONNAIRE =================

My Name: [Your Name HERE]

My Member ID No.: [ID, e.g. 4]

AGORA Members I Personally Know
(Please list their Member IDs separated by commas):
[ID's, e.g. 2,3, 33, 104]

================= END QUESTIONNAIRE   =================
```

The information listed in the list is also available at the AGORA site ONLY to the registered members (you will have to login first):

http://www.risk-agora.org/component/option,com_juser/task,user_list/

All information you provide is confidential and will be used only for research purposes. You responses will only be used to form a social network that will contain no personal information whatsoever. No data will be released that will permit the identification of any individual or individual characteristics.

Your participation in this survey is important.
If you have any questions or concerns, please e-mail me at:

chrismina@civil.mbox.media.kyoto-u.ac.jp

Please include your name and the name of your organization so that I can promptly respond to your questions.

Your participation is greatly appreciated! Thank you.

Figure 5.7: Questionnaire sent to the members of AGORA

of the questionnaire, a reminder was sent 2 months later. The response to the questionnaire was rather low. Out of 180 members only 50 responded, a 28% response rate. There were also 13 unreachable (7%) and 10 bogus (6%) e-mail addresses. Nevertheless, because of the assumption that the AGORA collaboration network is undirected, even with a 28% response rate it was possible to construct and analyze a representative social network of AGORA. Figure 5.8 shows the number of registered members of AGORA from its formation until the present, roughly a two year timespan. Figure 5.9 shows the cumulative data of member registrations over the same period. Using the responses from the questionnaire and the membership data from AGORA's database we constructed the adjacent matrix of AGORA's collaboration network,

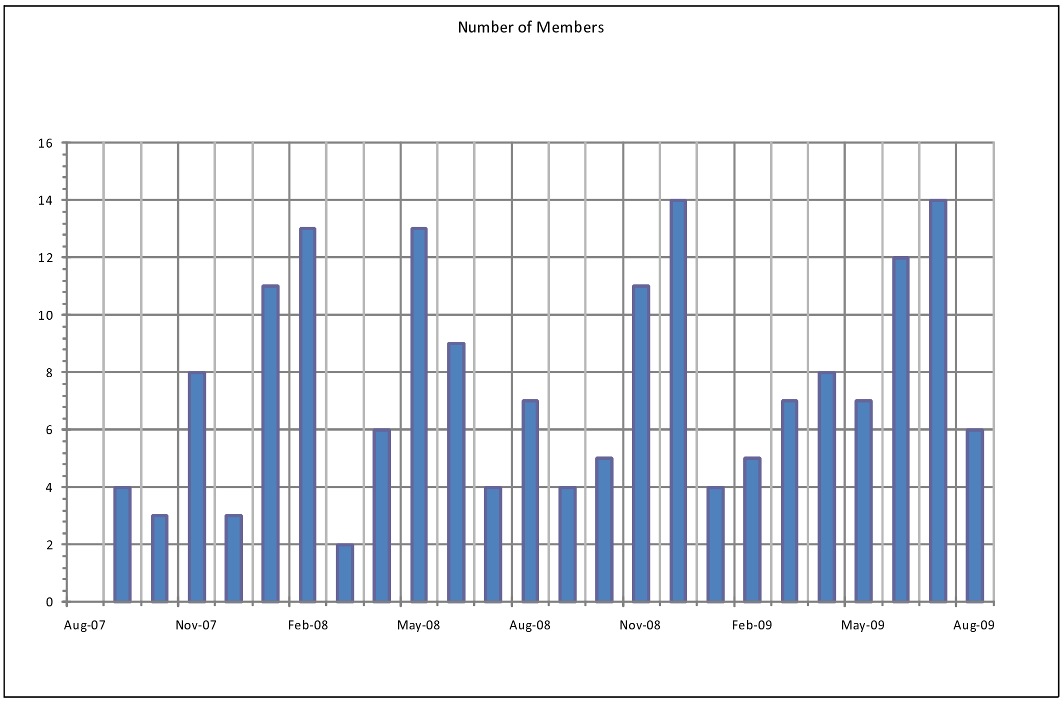


Figure 5.8: Registered members of AGORA

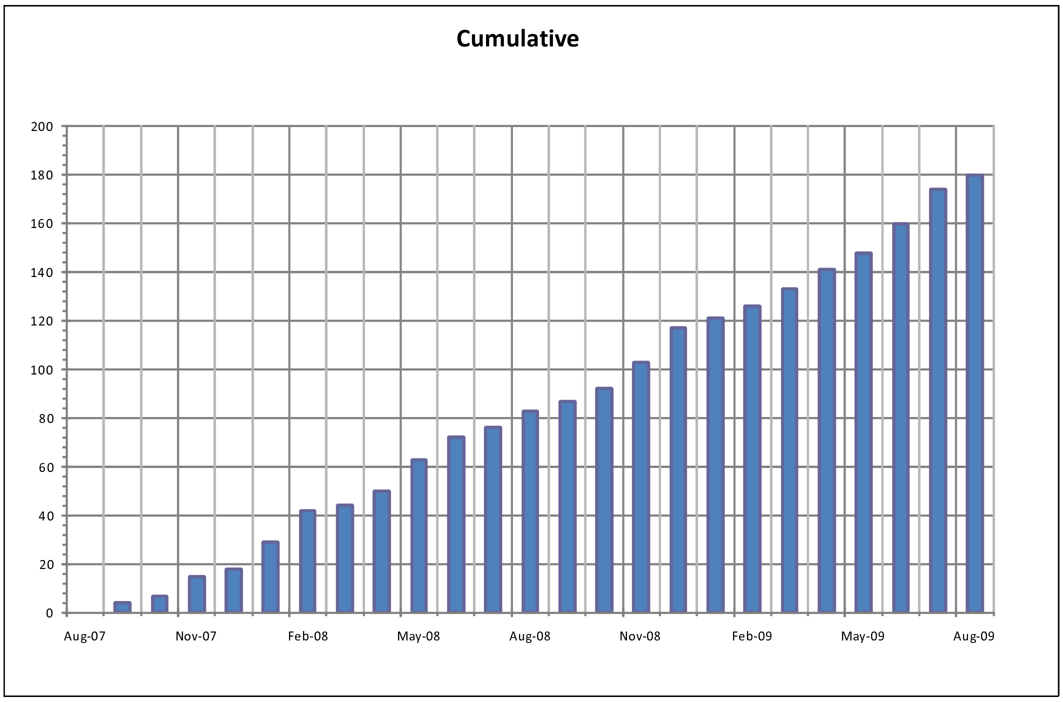


Figure 5.9: Cumulative data of member registrations of AGORA

shown in Figure 5.10. In Figure 5.10 only the lower part of the matrix shown to contain data for clarity. The adjacency matrix of undirected social networks are symmetric about their diagonals. Cells with dark green indicate the existence of a link. Highlighted on the matrix are

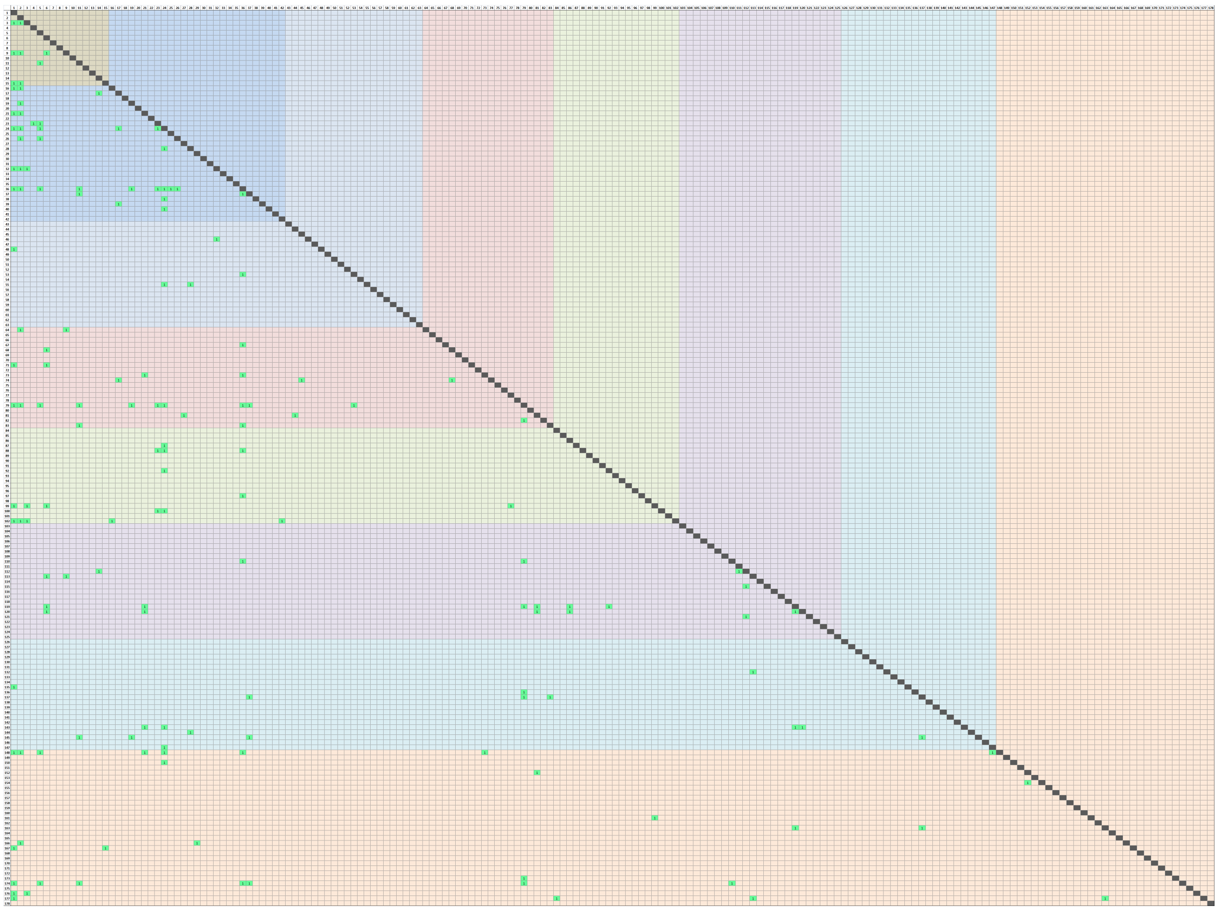


Figure 5.10: Adjacency Matrix of AGORA's Collaboration Network

also the time evolution of AGORA's collaboration network. Each color block represents three months (a quarter), starting from August 2007 until the end of July 2009. The analysis of the collaboration network of AGORA was conducted based on the complex social network analysis framework presented in Chapter 3 using Pajek. The results were verified with the Network Workbench Tool (NWB-Team, 2006). The following results were obtained. The maximum network (cumulative data on end of July 2009) had 178 vertices. Among them 97 were isolated, i.e. not connected to any other vertices. The network had 138 edges. The average degree of the network was 1.5505617977528083. The largest component of the network was made out of 79 vertices. The density of the network was calculated to be 0.00876. The diameter of the network was 9 while the average shortest path between a pair of nodes of the network was calculated to be 3.5798183. The average clustering coefficient of the network was calculated to be 0.36993643611290666. The degree distribution of the network is shown in Figure 5.11. Finally Figures 5.12 and 5.13 show visualizations of the network's evolution for the first year (August 2007 to July 2008) and for the second year (August 2008 to July 2009) respectively.

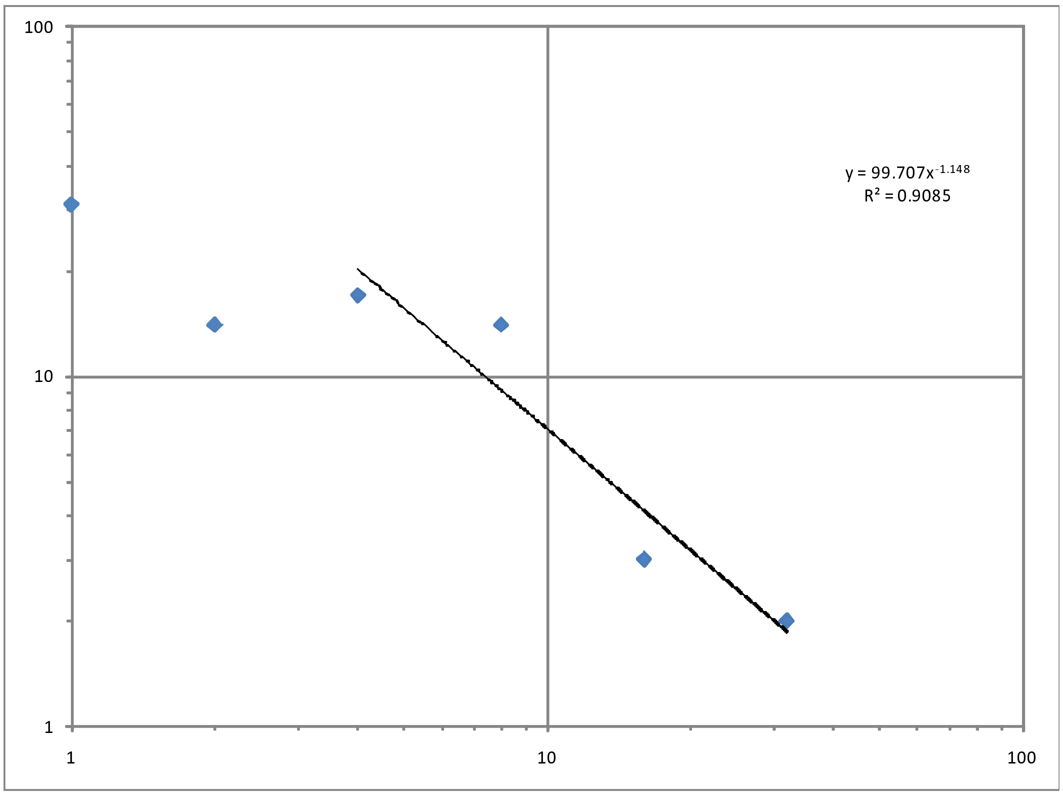
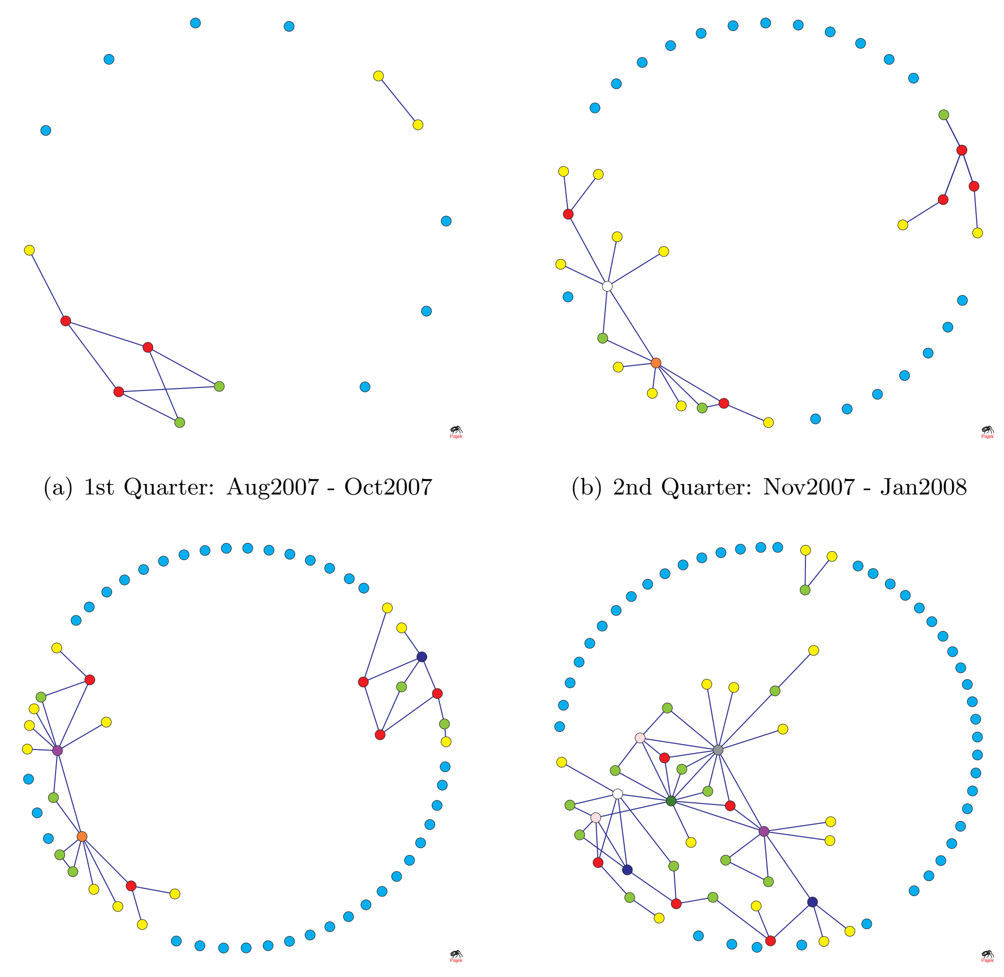


Figure 5.11: Degree Distribution of AGORA's Collaboration Network

5.3 Case Study Conclusions

The Open Risk Analysis model and AGORA, even though they have their roots in the rather well developed and well known concept of open source, are new approaches and as a result not widely known and adopted. AGORA, being a platform, depends heavily on the number and quality of its comprising members. A metric of the quality of such a platform is its output and contributions, in the case of AGORA the software tools, methodologies and data it disseminates. The infrastructure of the AGORA platform is in place, i.e. inexpensive means of communication, virtual community and forum and channels of dissemination. What is necessary for the acceptance and wide adoption of AGORA and the ORA model is firstly, to attract and motivate more volunteer researchers and practitioners and second the accumulation of open software tools and data in order to create a strong knowledge base. The two work in synergy, that is, more contributors create a bigger and better knowledge base which in turn attracts more contributors. In order to achieve that, research is necessary to understand the motives and behavior of the stakeholders of AGORA.

The analysis of the collaboration network of AGORA even though it was based on

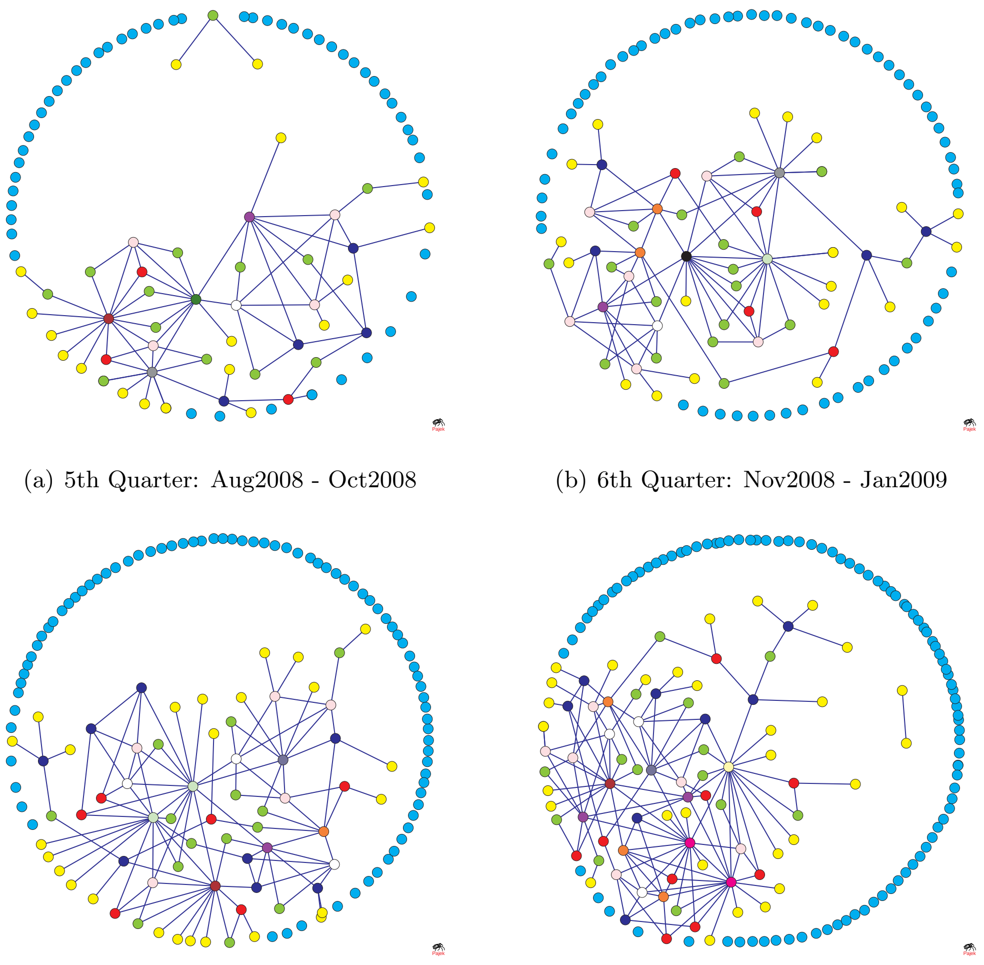

(a) 1st Quarter: Aug2007 - Oct2007 (b) 2nd Quarter: Nov2007 - Jan2008

(c) 3rd Quarter: Feb2008 - Apr2008 (d) 4th Quarter: May2008 - Jul2008

Figure 5.12: AGORA's Network Evolution for the First Year

minimal data it revealed that the AGORA presents typical characteristics of scientific collaboration social networks such as large components, power-law degree distributions and small diameters. Those characteristics are evident in Figure 5.11 which shows the degree distribution of the AGORA's collaboration network following a typical power-law distribution and from Figures 5.12 and 5.13 that show visualizations of the network's evolution. In the visualizations the creation of a large component is evident, typically appearing in complex social networks.

Regarding the future perspectives of MIRISK in conjunction with AGORA, it is expected that as soon as the testing and validation of MIRISK completes it will be disseminated to the members of AGORA. The dissemination of MIRISK will have several impacts on the future of AGORA, its members, the Open Source community and to a greater extend the scientific community. MIRISK being one of its kind open global multi-natural hazard risk analysis



(a) 5th Quarter: Aug2008 - Oct2008

(b) 6th Quarter: Nov2008 - Jan2009

(c) 7th Quarter: Feb2009 - Apr2009

(d) 8th Quarter: May2009 - Jul2009

Figure 5.13: AGORA's Network Evolution for the Second Year

software will hopefully attract the interest of experts in joining AGORA and thus increasing its social capital and knowledge base. MIRISK has several limitations in its methods and data that can be effectively address by experts in the risk management and software engineering fields. In addition to that, MIRISK is build to be flexible and easily updated/modified a fact that will attract the attention of researchers from various fields. For instance, the GIS engine of MIRISK can be used to create a visual tool that can enable city planners and economists to calculate various economic impacts due to policy changes regarding the impacts of natural hazards, infrastructure asset management and so on. Finally, it is expected that MIRISK will provide an excellent case study where experts can study the collaborative creation of a scientific tool using open methodologies and data.

In the paper it was mentioned that one of the most important problems faced by

AGORA and open collaboration platforms in general is the motivation of their members/collaborators to participate and contribute. A potential way to deal with this problem is to offer some kind of sustainable business models and or rewards to the collaborators. As a result, the study and creation of viable business models that take into consideration the special characteristics of AGORA is a very promising topic for future research.

Something not mentioned so far in the paper is the problem of free riding and "hijacking" of the collaborative content of AGORA. This problem is well known in the open source community and attracts a lot of scholarly attention. Licenses such as the GNU GPL are in place to prevent such collaborative content "hijacking" but in reality they are not effective in preventing it. A preventive measure could be the rate of change and dissemination of content for example. Research into finding ways to minimize free riding and prevent collaborative content "hijacking" is also a very promising topic for future research.

Chapter 6

Case Study: The Open Collaboration Book Project

6.1 Introduction

In early 2008, Kyoto University of Japan and Tongji University of China jointly initiated a project with the objective to produce and disseminate a body of knowledge in the form of a scientific digital anthology. The project, dubbed the Open Collaboration Book Project or OCBP for short, is an effort that aims to create a specialized electronic body of knowledge available on the Internet, utilizing production and dissemination methodologies derived from the Open Source paradigm. The digital anthology will initially consist of contributions from experts in the fields of civil engineering, transportation engineering, asset management and urban planning with a future plan to expand its scope to include other disciplines. The project also aims to strengthen the ties between the two universities, aid in the exchange of research ideas and results, enhance the knowledge base and increase the social capital of the participating organizations. The project also aims to provide an open knowledge management system. In order to meet its objectives, OCBP is build upon 3 pillars: "Dynamic Knowledge Creation and Dissemination", "Perpetual Peer-Review" and "Creation of Content at the Frontiers of Science."

The main novelty provided by the project lies in the production methodologies and dissemination of the project's deliverables. The digital anthology is made out of the collaborative effort of members of the academia and experts from various engineering disciplines residing at different geographical locations. The collaborators communicate and contribute to the project via the Internet. The constituent blocks of the digital anthology (i.e. its "source files", using software engineering jargon), reside on a web-server which is accessible by all contributors.

The source files of the articles of the anthology are made open to all contributors to view, provide suggestions and feedback and to modify as necessary. Selected project deliverables such as general information regarding the contents of the project and its contributors are made available to the public. Access to specific areas of the system follows a layered access structure. For example, access to the e-text (user generated anthology) compilation area is allowed only to registered users.

The project offers a unique chance to study the production and dissemination of specialized open content. In this chapter we focus on the economic and legal aspects of open collaborative content production and dissemination. Issues such as open content intellectual property, copyright and licensing are also explored. The chapter is organized as follows. The following section explores intellectual property, copyright and licensing of open collaborative content. The next section presents the OCBP system. Finally the chapter closes with conclusions and discussion of topics for future research.

6.2 Intellectual Property, Copyright and Licensing

Intellectual Property (IP) is a very broad term that has several interpretations. For example, the United Nations World Intellectual Property Organization (WIPO) defines IP as:

> *"The legal rights which result from intellectual activity in the industrial, scientific,literary and artistic fields."* (WIPO, 2009)

IP can be also understood as any intangible asset that consists of human knowledge and ideas including those products of the intellect that are of commercial value. Of great interest to this study is the legal interpretation of IP. Within the legal framework, IP is considered to be a type of property[1]. It worths noting here that "property" is a legal term that describes something that a legal entity has legally granted control over. Unlike a "good" which is an item that exists regardless of law, IP is a pure legal concept. Lindberg describes IP as a "hybrid good" made up of both knowledge and law (Lindberg, 2008). To grasp IP is necessary to view it as a bundle of separate and independent rights given to the IP creator by law. IP rights can be given away, assigned or sold to third parties one by one or in combinations. For example copyright, one of the four fundamental systems of IP, grants the creator of copyrighted works the rights to reproduce the copyrighted work in copies or phonorecords and to prepare derivative works based on the copyrighted work (U.S.C., 2009). The main rights given by copyright can be further

[1] In this chapter we are mostly referring to the US legal system. Countries that have active IP legal frameworks and abide to international standards have legal frameworks very similar to USA's system.

broken down. For example, the right to reproduce can be broken down further to the rights to reproduce in electronic format or in paper format in the case of a book.

Within the U.S. legal system, IP is considered to be a set of four systems. The four systems are patents, copyrights, trademarks and trade secrets. Each system has its own unique characteristics. What they all have in common is the fact that they were created to address the problem of knowledge market failure. Knowledge creation is a very important activity for society but very costly and time consuming. On the other hand, knowledge once created can be shared with little or no cost. This, simply put, means that to create knowledge and ideas vast amounts of money and effort are required but once knowledge is created it costs very little to recreate and distribute. These characteristics of knowledge made it being classified by economists as a public good, i.e. a non-rivalrous, non-excludable good. Some economists argue that knowledge and technology in general are partially excludable goods rather than pure public goods (Romer, 1990). Whichever the case, non-excludable or partially excludable, knowledge creation suffers from the important market failure of "free riding". Because of this fact people are not motivated to create new ideas and knowledge. As a result ways to motivate people to create new knowledge were devised. IP is the most common way to motivate people in free markets to create new knowledge by providing a temporary monopoly to the IP creators in exchange for their IP. For example, patents provide a type of temporary "monopoly" of 20 years to the IP creator by giving her the right to keep others from making, using, offering for sale, selling, and importing the claimed IP (e.g. an invention.) When the patent expires the invention enters the public domain where everybody has legal access to it. Open content advocates argue that the waiting time of 20 years is a long period to wait, especially in the case of technological inventions and software. Open content goes one step further by enabling sharing of IP at the time of dissemination, based on a licensing scheme. For that reason, open content including F/OSS became a very attractive approach in spreading knowledge. Nevertheless, the problems of content "hijacking" and "free riding" in open collaboration still linger and currently their studies are considered hot research topics in the fields of engineering, economics and social sciences.

6.3 Free/Open Source Software Licensing

Free/Open Source Software (F/OSS) collaboration is a representative case of open collaboration. In the case of F/OSS there is always risk of a third party incorporating the source of F/OSS into a proprietary product without the knowledge (and agreement) of its creators and thus profiting from the efforts of F/OSS developers. This is possible due to the fact that proprietary (closed source) software comes only in the form of binary (machine readable, executable) files that do not include the actual source code. Reverse engineering of executables is sometimes possible but extremely time consuming. Proprietary software licenses strictly prohibit reverse engineering of the software so the actual source code used in producing the executable programs remains hidden and thus inaccessible to the users of the software. On the other hand, F/OSS makes the source code of software available to all by default, thus the risk of a third party free-riding by "hijacking" its source code is possible. The "hijacking" of open collaborative content comes in direct contrast with the beliefs and philosophies of open collaboration communities such as the Free Software and Open Source communities. It also disrupts the efforts of the collaborators and discourages new participants in joining the community. As a result, open collaboration communities developed mechanisms that aim to protect their open collaborative content from "hijacking" and exploitation by third parties.

The most common protection mechanism of open collaborative content comes in the form of licensing. Licensing of F/OSS appeared first in the mid 1980's with the dissemination of "free software" created by the Free Software Foundation (FSF). The software created by FSF has been traditionally distributed under the GNU General Public License (GPL). Since the inception of GPL, a variety of licenses that can be used with F/OSS were developed. Currently the terms and conditions of F/OSS licenses vary substantially as it can be seen in Table 6.1. Some licenses impose restrictions on the modifications of licensed software, e.g. require that modified versions of the software be available under the same conditions of the original licensed software. Some licenses do not allow commercialization of the software while others allow it but require fees for the use of the source code. Following the footsteps of FSF, the Open Source Initiative (OSI) was established in 1998. The OSI is responsible for the definition of Open Source Software and provides guidance on the structure and contents of Open Source licenses. The most important requirement, which is coincidentally common for all F/OSS licenses, is the provision of access to the source code of the software. This requirement is very unique in the case of software since the source code is at the same time the means to create the final product and the final product itself; source code contains the necessary information and the means to create the executable program.

Table 6.1 presents some of the most common Open Source licenses categorized by their characteristics. Excluding the availability of source code characteristic which is common for all F/OSS licenses, the licenses that require derivative works to be made available (if derivative

Table 6.1: F/OSS Licenses

License Name	Source Code[1]	Derivative Works[2]	Reciprocal[3]	No Mixing[4]
GNU GPL	O	O	O	O
GNU LGPL	O	O	O	X
Mozilla PL 1.1	O	O	O	X
Qt PL	O	O	O	X
Apple Public Source	O	O	O	X
Sun PL	O	O	O	X
Common PL	O	O	X	X
IBM PL V1.0	O	O	X	X
Eiffel Forum L	O	O	X	X
The BSD L	O	X	X	X
Apache Software L	O	X	X	X
The MIT/X11 L	O	X	X	X
Zlib/Libpng L	O	X	X	X

Sources: Commonwealth of Massachusetts Website and Lerner and Tirole (2005)

Notes:

L=License, PL=Public License, GPL= General Public License

[1] Source code MUST be made available.

[2] Derivative works MUST be made available.

[3] Derivative works MUST be released under the same license terms.

[4] Mixing of software with non-compatible licenses is not allowed.

works are to be distributed) under the same license conditions as the original work and additionally disallow mixing of software with incompatible licenses are considered restrictive. By restrictive here we mean that the creators of derivative works have restrictions on the distribution of works. A restrictive license such as the GPL, requires that derivative works be made publicly available, be distributed under the same license terms as the original work[2] and does not allow incorporation (i.e. mixing) of software released under incompatible licenses (e.g. proprietary software or libraries) with the original work. Such kind of licenses are referred to as "copyleft" licenses, a term coined by the FSF to show their fundamental philosophical differences from the system of copyright. In the words of Richard Stallman, the founder of FSF:

> *"To copyleft a program, we first state that it is copyrighted; then we add distribution terms, which are a legal instrument that gives everyone the rights to use, modify, and redistribute the program's code or any program derived from it but only if the distribution terms are unchanged. Thus, the code and the freedoms become legally inseparable. Proprietary software developers use copyright to take away the users' freedom; we use copyright to guarantee their freedom. That's why we reverse the name, changing "copyright" into "copyleft." Copyleft is a way of using of the copyright on the program. It doesn't mean abandoning the copyright; in fact, doing so would make copyleft impossible. The word "left" in "copyleft" is not a reference to the verb "to leave" - only to the direction which is the inverse of "right"."* (Stallman, 1984a).

Unrestrictive F/OSS licenses are more flexible than copyleft licenses. Unrestrictive licenses are also called "Academic Licenses" because they originated in the academia. A typical example of an academic license is the BSD license. Such a license allows derivative works to be released under any license, even a proprietary license.

6.4 Open Content Licensing

Recently, the global unprecedented interest generated by the Open Source revolution sparked an effort to define and support what is termed "Open Content". Open content is a very broad term that includes all creative work that can be disseminated in a format that explicitly allows copying and modification (e.g. digital format) which anyone is free to use, re-use and redistribute without legal, social or technological restriction. Examples of content that can be potentially disseminated as open content include music, publications in digital form (e.g. books, journal papers), scientific data, engineering and architectural drawings and movies. Table 6.2 shows a small sample of the diversity found in open content projects. Currently there are open collaborative efforts to build open source cars, open architecture, open source movies and open prosthetics among others.

An effort to create a general definition of open content was undertaken by the Open Knowledge Foundation (OKF)[3]. The OKF created a generalized definition of open content in

[2]This type of requirement in a license is sometimes referred to as a "share-alike" or "viral" clause, in the sense that it "infects" future derivative works to be released under the same terms.
[3]http://www.okfn.org/

Table 6.2: Open Content Projects

Project Name	Description
c,mm,n	An open source car design. The vehicle's technical drawings and blueprints are freely available on-line, and everyone is invited to add their own ideas and modifications, provided of course that these are shared again with the community.
OSCar	An effort to develop a car according to Open Source principles.
Free Beer	The recipe and branding elements of FREE BEER is published under an Open Content License, which means that anyone can use the recipe to brew their own FREE BEER or create a derivative of the recipe.
Free Model Foundry	The project advances the development and free distribution of open source models of electronic components for system and IC design around the world.
Open Architecture Network	An online, open source community dedicated to improving living conditions through innovative and sustainable design.
OpenProsthetics	An open source collaboration between users, designers and funders with the goal of making prosthetics designs freely available for anyone to use and build upon.
Open source cola	A brand of cola unique in that the instructions for making it are freely available and modifiable. Anybody can make the drink, and anyone can modify and improve on the recipe as long as they, too, license their recipe under the GNU General Public License.
A swarm of angels	A collaboratively funded and scripted feature film.
BurdaStyle	Open Source sewing patterns.
FreeSound	The project aims to create a collaborative database of audio recordings released under an Open Content License.
OpenClipArt	This project aims to create an archive of clip art that can be used for free for any use.
GuitarWeek	An organization dedicated to teaching guitar online, with a large archive of free online guitar lessons.
OpenContext	A free, open access resource for the electronic publication of primary field research from archaeology and related disciplines.

Source: OpenTTT (http://www.openttt.eu/)

its Open Knowledge Definition[4] (provided *ad verbatim* in the Appendices). In parallel, several organizations created open content licenses with the Creative Commons and the FSF leading the way. Several open content projects were also created at the same time, among them the Wikipedia project[5] which is probably the most successful and well known open content project worldwide.

Currently, copyright law is defined by international conventions and its format is very similar in countries that have active copyright laws. Basically, copyright laws state that a legal entity cannot copy, reproduce, communicate or transmit copyrighted content, including music, movies, literary and other artistic work, without the permission of the copyright owner (Fitzgerald, 2005). The law generally provides for exceptions when an "insubstantial" part of the work is used e.g. for educational purposes and for review and criticism (*cf.* fair use and fair dealing). New digital technologies and global digital networks such as the Internet and mobile phone networks that allow easy, inexpensive and fast transmission of data impose a great challenge on copyright laws. It is now easier and less expensive than anytime in human history to copy and transmit digital content. As a result, protecting copyrighted digital content became a great challenge that requires substantial resources. Most of the protection mechanisms of digital content aim to block duplication and distribution. Open content advocates argue that this policy has a very adverse social effect by slowing the spread of knowledge and hampering creativity. To put it in Fidgerald's words:

> *"In the fast paced and serendipitous world of the Internet the traditional notion of obtaining permission before re-use is out of place. The key to seamless access to knowledge - through open access, new business models or e-commerce mechanisms - is to work out how that permission process can be automated."* (Fitzgerald, 2005)

Researchers worldwide supporting the open content movement initiated efforts to address the problem. Those effort resulted in the creation of interest groups and non-profit organizations. Currently, non-profit organizations such as Creative Commons (CC) is leading the way in providing legal mechanisms that can be used to promote and protect open collaborative content. The concept of CC, a virtual space on the Internet where sharing and reuse of copyrighted material is possible without the fear of legal complications, was conceived by a group of scholars at Stanford and MIT Universities, lead by Professor Lawrence Lessig (CreativeCommons, 2001). Frustrated by the fact that technology has a great potential in spreading knowledge and provide access to content to many people but only slowed down by the cumbersomeness of copyright

[4]`http://www.opendefinition.org/1.0/annotated`
[5]`http://www.wikipedia.org/`

laws in negotiating access, Lessig came with the idea of CC. Vital to making the vision of CC becoming real, Lessig and his colleagues created an open content licensing model, based on the FSF's model, which is simple to understand and implement. The licensing model of CC also tries to achieve compatibility with existing open content licenses and to create a standard in licensing open collaborative content. All CC licenses have in common the following features[6]:

- Licensees are granted the right to copy, distribute, display, digitally perform and make verbatim copies of the work into another format
- The licenses have worldwide application that lasts for the entire duration of copyright and are irrevocable
- Licensees cannot use technological protection measures to restrict access to the work
- Copyright notices should not be removed from all copies of the work
- Every copy of the work should maintain a link to the license
- Attribution must be given to the creator of the copyright work (BY).

A typical CC license is created by selecting from a menu of optional features that attach to the baseline features of the license, as described in detail below:

- **Non-commercial (NC):** others are permitted to copy, distribute, display and perform the copyright work - and any derivative works based upon it - but for non-commercial purposes only
- **No derivative works (ND):** others are permitted to copy, distribute, display and perform exact copies of the work only and cannot make derivative works based upon it
- **Share alike (SA):** others may distribute derivative works only under a license identical to that covering the original work

The optional features can be mixed with some exceptions (e.g. ND cannot be used along SA) resulting in six core licenses[7]:

- **Attribution (BY):** This is the most accommodating of the licenses offered, in terms of what others can do with your work. It lets others copy, distribute, re-use and build upon your work, even commercially, as long as they credit you for the original creation.

[6]http://creativecommons.org/about/licenses/fullrights
[7]http://creativecommons.org/about/licenses/meet-the-licenses

- **Attribution-Non-commercial (BY-NC):** This license lets others copy, distribute, re-use and build upon your work, as long as it is not for commercial purposes and they credit you as the original author.
- **Attribution-Share alike (BY-SA):** This license lets others re-use and build upon your work even for commercial purposes, as long as they credit you and license any derivative works under identical terms.
- **Attribution-Non-commercial-Share alike (BY-NC-SA):** This license lets others re-use and build upon your work, as long as it is for non-commercial purposes, they credit you and they license their new creations under identical terms.
- **Attribution-No derivatives (BY-ND):** This license allows use of a work in its current form for both commercial and non-commercial purposes, as long as it is not changed in any way or used to make derivative works, and credit is given to the original author.
- **Attribution-Non-commercial-No derivatives (BY-NC-ND):** This is the most restrictive of the six core licenses. It is often called the "advertising" license because it only allows a work to be copied and shared with others in its original form, and only for non-commercial purposes and where credit is provided to the original author. This license does not allow the creation of derivative works, or the use of the work for commercial purposes.

6.5 Outline of the OCBP System

Based on the three pillars of "Dynamic Knowledge Creation and Dissemination", "Perpetual Peer-Review" and "Creation of Content at the Frontiers of Science", a computerized system was designed and built to meet the requirements of the project. The system, based on client-server architecture, allows its users to conduct several activities simultaneously in real time. The collaborators can create, edit and administer content from anywhere in the world, given they have Internet access. OCBP, being a system that supports the creation of open content, naturally adopts open standards and F/OSS. The OCBP system is made out of various subsystems. For example there are subsystems for content version control, content administration and user administration. The collective body of knowledge that resides in the system can be visualized as a collection of individual articles (e.g. essays, papers) which are the basic building blocks of the digital anthology. The building blocks dynamically change via the addition and editing of content. This renders the project a collective dynamic body of knowledge that evolves over time. Registered users are capable of compiling their own digital collections (referred to as "e-texts") by selecting the content that is of most interest to them.

The outline of the OCBP system is shown in Figure 6.1. Each article is made out of

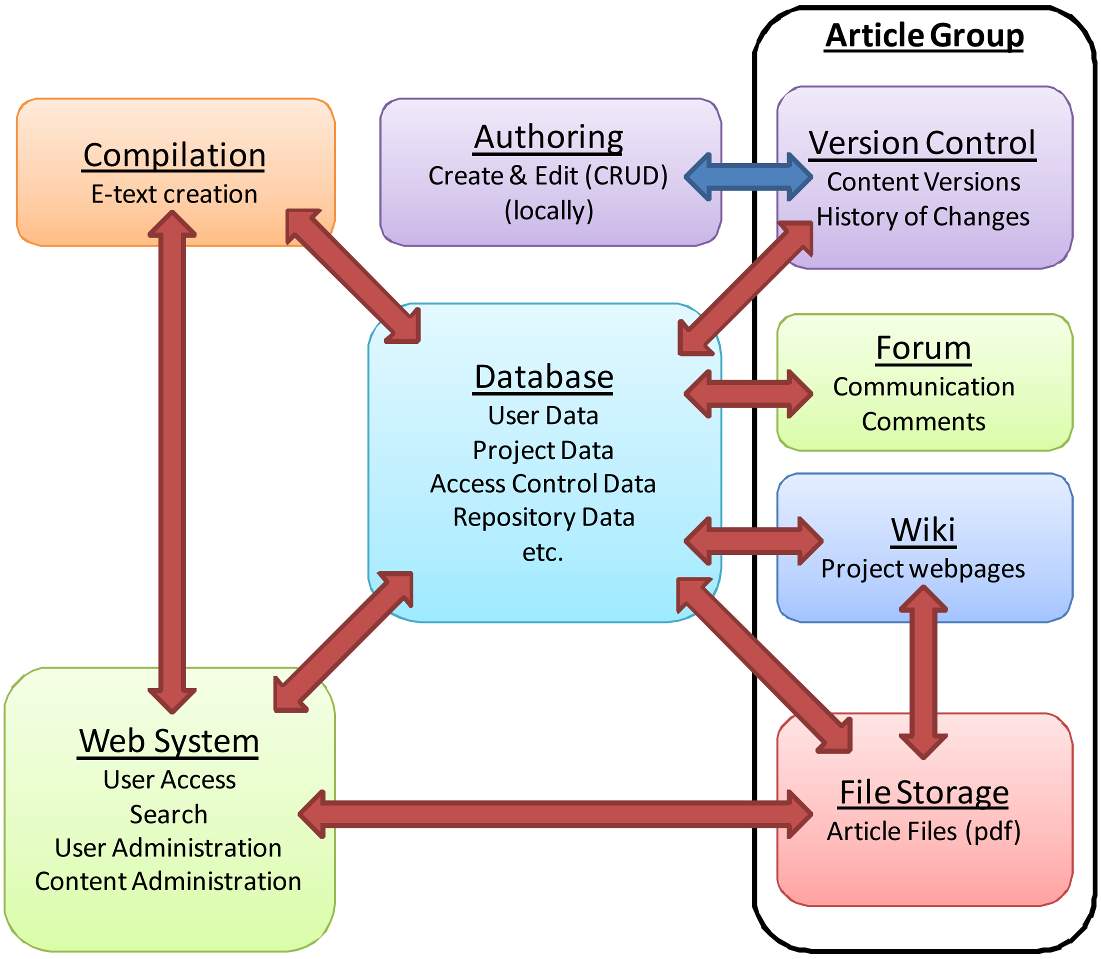


Figure 6.1: Outline of the OCBP System

4 systems: a file storage system where all the versions of the completed article in PDF form reside, a dynamic wiki website system that enables fast, easy creation and editing of webpages, a version control repository and a simple forum system. The systems aim to support the 3 main pillars of the OCBP. For example the version control system subscribes to the "Dynamic Knowledge Creation and Dissemination" pillar whereas the forum and Wiki subscribe to the "Perpetual Peer-Review" pillar.

6.5.1 The Main Database

Central to the OCBP system is its main database. The database system, uses the well known F/OSS relational database software MySQL. The main purpose of the system's main database is to hold all the project's data and allow the other systems of OCBP to access and manipulate the data. The database plays a very important role in generating dynamic content by allowing CRUD[8] operations on the project's data. The OCBP data are organized in tables and include user related information, data regarding the individual articles, article collection (e-texts) data, administrative data, system access control data and version control information.

6.5.2 Authoring and Version Control

The system utilizes a client-server architecture in order to allow decentralized access, i.e. allow many contributors from various locations to access the system at the same time, while keeping the core system at a central location for easiness of troubleshooting, updating and maintenance. A very important component of the OCBP system is the version control system. OCBP uses the widely used F/OSS Subversion (SVN) version control software. Figure 6.2 shows the SVN system layout (Collins-Sussman et al., 2008). Versioning systems such as Subversion, CVS and Bazaar have being used successfully in the management and versioning control of software projects. Typically a versioning system can be used for:

> *"...the management of multiple revisions of the same unit of information. It is most commonly used in engineering and software development to manage ongoing development of digital documents like application source code, art resources such as blueprints or electronic models, and other critical information that may be worked on by a team of people."*[9]

One of the merits of version control systems (including SVN) is that they allow "off-line" revision control. Off-line here means that collaborators do their article editing locally, without having to be constantly connected to the system's server. That eases the server's workload and allow collaborators to work at their convenience on their own computers. Collaborators have to only connect to the OCBP server when they download the latest version of the article's source or when they commit (i.e. upload) chances back to the server. The source files of all the articles, as well as their history of changes, reside in a file repository, stored in the database of the system and controlled by the SVN versioning system. All registered contributors of the project have secure access to the OCBP system via the user authentication system. Once a contributor accesses the system, he can download the latest version of the source files, make version comparisons and upload file changes back to the system. Figure 6.3 shows the authoring process of the system. The typical authoring procedure is as follows. First the contributor logs in to the OCBP system

[8]Create, Read, Update and Delete are the four basic functions of data storage and manipulation.
[9]`http://en.wikipedia.org/wiki/Revision_control`

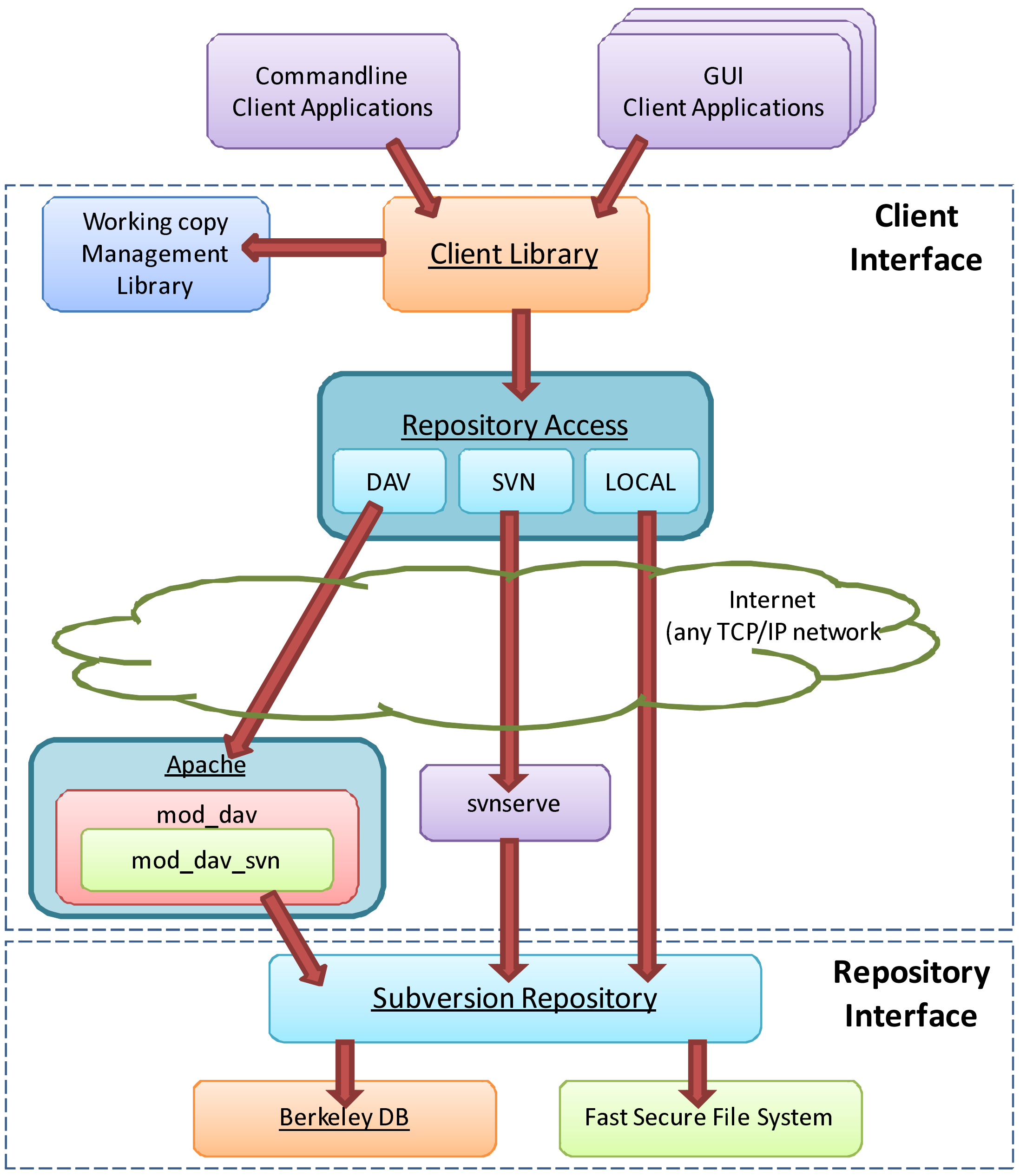


Figure 6.2: SVN System Layout

and updates her local working article source files to the the latest version from the OCBP server's file repository. She then proceeds to work on the source files, i.e. to add, edit and review content. Once the editing work is completed, the collaborator once again logs in to the system and "commits" the changes, i.e. uploads the revised files back to the file repository. The versioning system tracks the changes, i.e. who made the changes, what was changed and when it was changed, records any additional comments the contributor submitted along and resolve any conflicts if any.

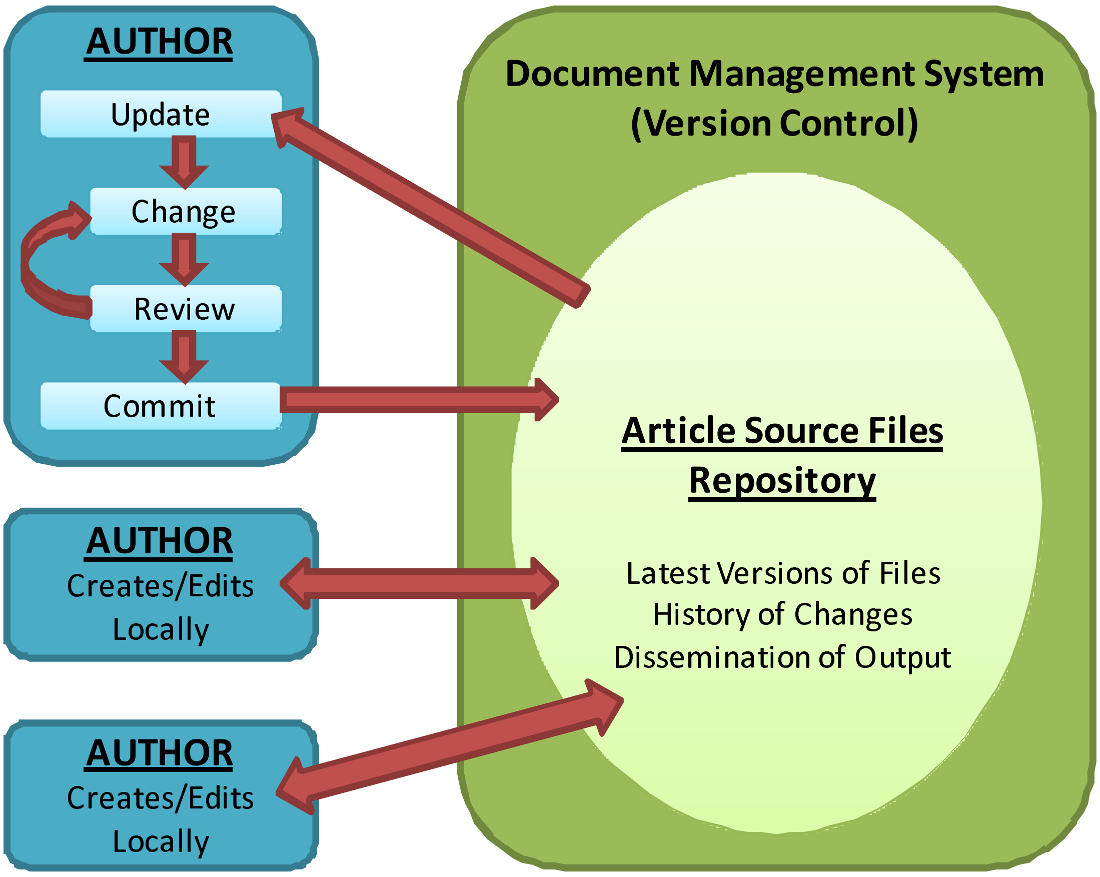


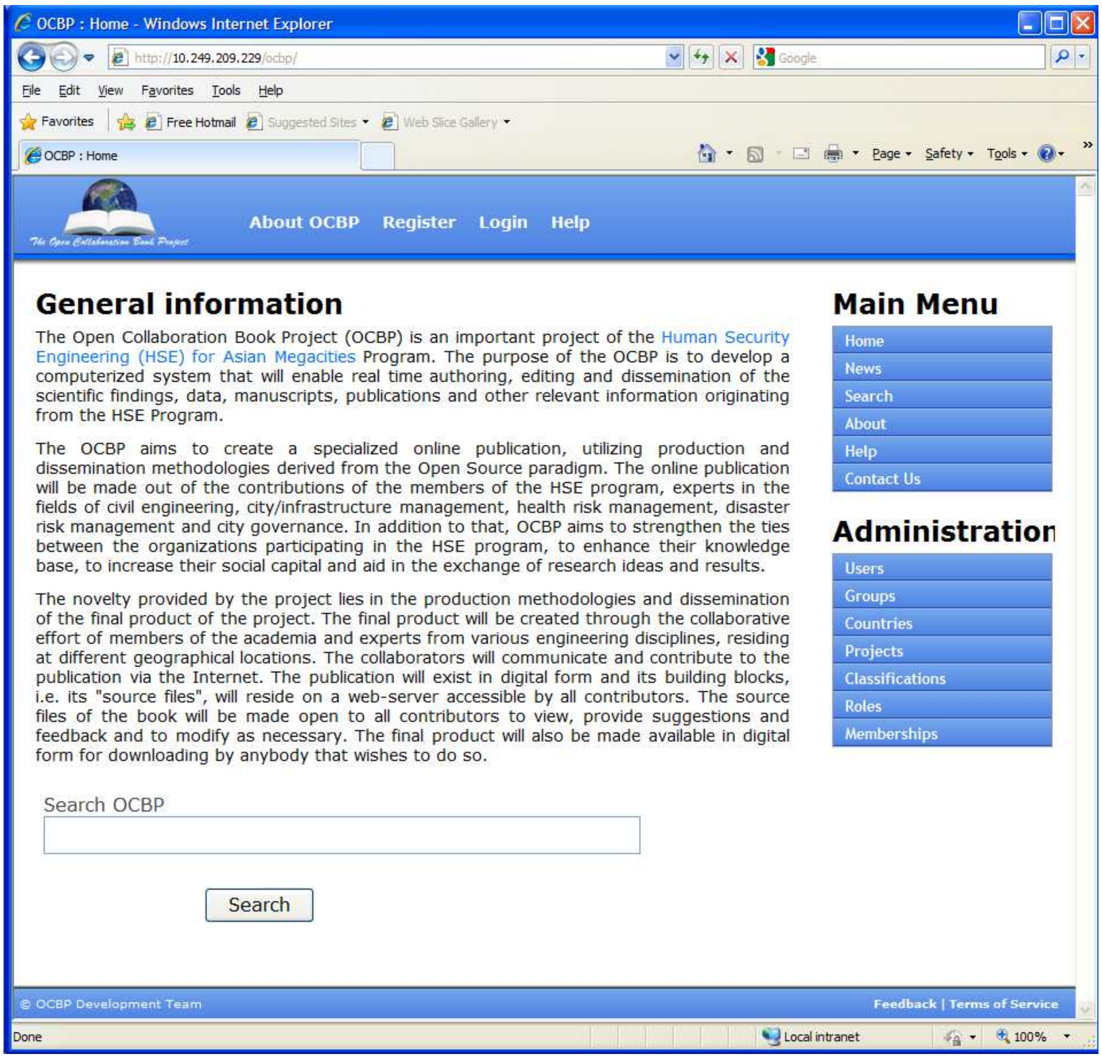
Figure 6.3: The OCBP Authoring Process.

6.5.3 The Web System

The web system of OCBP provides the main public graphical user interface (GUI) to the system. Via the web system users can access project information, search the database, administer their profiles and their articles. The web system also provides a GUI to other OCBP systems such the article file storage system and the e-text compilation system. Figure 6.4 shows the prototype of the web system's GUI.

In order to control and effectively administer the access to the OCBP system the web system is design to provide access control. System access is divided into two categories, public access and registered access. Public access refers to the accessing of the system without user authentication, i.e. guest access. Registered access refers to the accessing of the system by a registered user who is successfully authenticated. Based on the system's access levels the following roles are implemented:

- **Unregistered Users:** They have access to the public content of the system, e.g. the OCBP website. Unregistered users also have limited access to the project's contents,



General information

The Open Collaboration Book Project (OCBP) is an important project of the Human Security Engineering (HSE) for Asian Megacities Program. The purpose of the OCBP is to develop a computerized system that will enable real time authoring, editing and dissemination of the scientific findings, data, manuscripts, publications and other relevant information originating from the HSE Program.

The OCBP aims to create a specialized online publication, utilizing production and dissemination methodologies derived from the Open Source paradigm. The online publication will be made out of the contributions of the members of the HSE program, experts in the fields of civil engineering, city/infrastructure management, health risk management, disaster risk management and city governance. In addition to that, OCBP aims to strengthen the ties between the organizations participating in the HSE program, to enhance their knowledge base, to increase their social capital and aid in the exchange of research ideas and results.

The novelty provided by the project lies in the production methodologies and dissemination of the final product of the project. The final product will be created through the collaborative effort of members of the academia and experts from various engineering disciplines, residing at different geographical locations. The collaborators will communicate and contribute to the publication via the Internet. The publication will exist in digital form and its building blocks, i.e. its "source files", will reside on a web-server accessible by all contributors. The source files of the book will be made open to all contributors to view, provide suggestions and feedback and to modify as necessary. The final product will also be made available in digital form for downloading by anybody that wishes to do so.

Figure 6.4: The OCBP Prototype Web System GUI.

such as the papers' titles, abstracts and contributors' public information. They can use the system's search engine. Unregistered users can not compile e-texts or download any contents of the OCBP other than what is mentioned above.

- **Registered Users:** They can download full versions of articles in pdf format, compile e-texts and access content reserved only for registered access.

- **Authors:** They can contribute and edit articles. Authors belong to specific articles. It is possible for an author to belong to more that one article. This role is assigned by a Article Editor.

- **Article Editors:** They administer articles and can assign or remove authors to articles. Article editors belong to specific articles. It is possible for an article editor to belong to more that one article. This role is assigned by the Book Editor.

- **Book Editor:** Administers all articles (i.e. all the project's content.) The Book Editor can assign or remove Article Editors.

The access layers of OCBP can be seen in Figure 6.5. A role inherits the properties of the roles

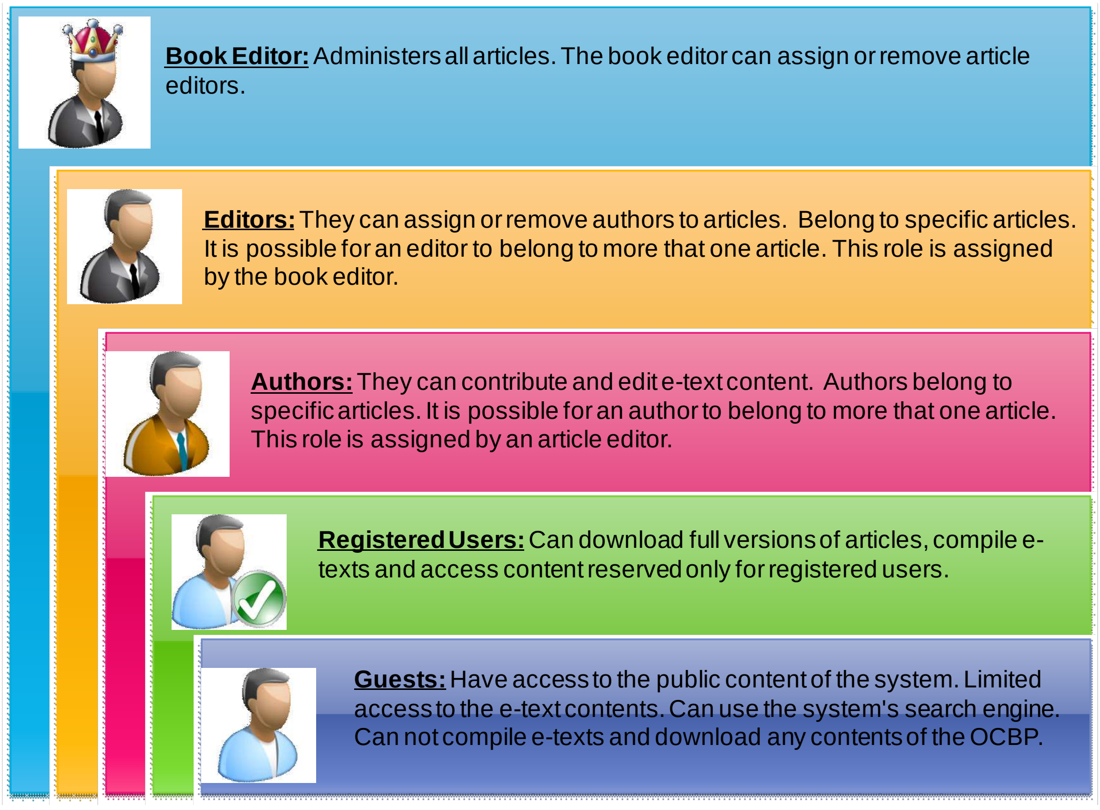

Book Editor: Administers all articles. The book editor can assign or remove article editors.
Editors: They can assign or remove authors to articles. Belong to specific articles. It is possible for an editor to belong to more that one article. This role is assigned by the book editor.
Authors: They can contribute and edit e-text content. Authors belong to specific articles. It is possible for an author to belong to more that one article. This role is assigned by an article editor.
Registered Users: Can download full versions of articles, compile e-texts and access content reserved only for registered users.
Guests: Have access to the public content of the system. Limited access to the e-text contents. Can use the system's search engine. Can not compile e-texts and download any contents of the OCBP.

Figure 6.5: The OCBP Access Layer

before it, e.g. authors inherit the properties of the registered users and the unregistered users groups whereas article editors inherit the properties of authors, registered users and unregistered users.

6.5.4 Other OCBP Systems

The OCBP system includes three more systems, mainly utilized at the article administration level. Each article is given a total of four systems: a version control system explained earlier, a file storage system, a wiki system and a simple online communication forum system. The file storage system is responsible for storing and providing access to the compiled (pdf) versions of the article. The Wiki system provides an easy and fast web page development system available to the contributors of each article. Using the Wiki system the article's contributors can disseminate information about their article, biographies of each contributor, announcements and any other information pertinent to their article. Finally, the online communication forum system can be used for communication among the contributors of each article and the public. The forum can enable online discussions related to the article, comments, feedback and any other pertinent content. Among the four systems, the communication forum is optional.

6.6 Conclusions and Future Research

This chapter presented a new effort in creating open collaborative scientific content on the Internet via an online knowledge management system. The computerized system that was developed to support the effort was also presented and discussed. One of the major difficulties in creating and disseminating open collaborative content is the issues related to intellectual property. Those issues are the most challenging and need to be addressed properly in order for the project to be successful. For example, there are copyright issues to be considered at various times and activities. During the authoring of articles the copyright belongs to the authors. When articles are updated by new contributors derivative works are created. That requires licensing from the original authors. The updated article is also copyrighted by all the authors (including the new contributors). When the OCBP users compile e-texts they create new (derivative) copyrighted works, hence they own the copyright of those works. Before doing so it is necessary for them to get proper permission from all the original authors of the articles used in the compilation. It is not hard to see that without a proper licensing framework legal chaos will be soon emerge. For that reason it is necessary to plan and implement a framework that will allow the proper handling of intellectual property issues of the project.

In the future, in order to deal with the copyright issues, it is suggested that a non-profit virtual organization be established in order to administer and protect the project. A licensing scheme should then be designed and implemented that will allow the transfer of copyrights of the project's contents from the contributors to the organization and from the organization to the OCBP users.

In the chapter we presented a common legal intellectual property framework and highlighted some of the difficulties involved in open content creation, e.g. the "free-riding" problem. Open content licensing is one way to deal with this problem. Another suggestion is to have frequent updates and prompt dissemination of open content. That way open content "hijackers" will be discouraged in using content that they know soon will be rendered obsolete by a new update. This could be a promising topic for future research.

Chapter 7

Conclusions and Future Research

The main objective of the research presented in this dissertation was to investigate open standardization processes via complex social networks and learning networks. The paradigm open standardization process phenomenon researched was the collaboration network of F/OSS development. In addition to that, a formal model that investigates learning networks and their applications in engineering, specifically in risk analysis and management, via the Bayesian Neural Network (BNN) framework was also developed. The research presented in this dissertation spanned several disciplines that included engineering, computer science, statistical physics, social sciences and law. The main research questions were: "why open standardization is emerging", "why open standardization phenomena such as F/OSS are currently thriving?" and "are those phenomena sustainable in the long run?".

The main model of open standardization process formation and evolution was presented in Chapter 3. The model was build within the complex social network analysis framework, utilizing data from SourceForge.net. The SourceForge.net collaboration network was analyzed from its inception to the present. In answering the research questions, the main findings of the chapter were:

- F/OSS collaboration has the main characteristic of other complex social networks such as the scientific co-authorship collaboration network, the patent collaboration network and the Internet. Those characteristics include power-law degree distributions, small network diameters and "small-world" properties.
- The SourceForge.net collaboration network is expanding over time and the rates of growth of the number of developers and projects follow similar trends.
- The SourceForge.net collaboration network's level of completion is low and the network

has structural holes.

- If the SourceForge.net collaboration network's characteristics remain unchanged it will be sustainable over time.

In the future, the network evolution parameters calculated in this study can be used to run simulations in order to forecast the future growth and sustainability of the F/OSS phenomenon. In addition to that, our analysis model can be updated by gathering new data over time and verify our sustainability hypothesis.

In Chapter 4 learning processes within networks were investigated. The model presented in the chapter was developed within the Bayesian Neural Network framework as a paradigm of learning process within a complex network. In order to present a practical application, the BNN model was used to evaluate the cost risk of soil decontamination projects. In building the application we utilized data from actual soil decontamination projects executed in Japan. Even though the dataset used in the case study was small, fair results were obtained. This can be partially explained from the fact that the Bayesian approach does not require the model's structure to be dependent on the amount of data available. In the Bayesian framework prior probabilities are assumed first and when data arrive the posterior probabilities are calculated based on the priors and the likelihood given the data. The structure of a model developed in the Bayesian framework is indifferent on the amount of data available, it works for few data and for a lot of data the same way. On the other hand, machine learning depends on the amount and quality of data. Large datasets containing a wide range of data will typically enable machine learning models to generalize better (Bishop, 1995). Topics of future research include the testing of the model with more complex network structures, e.g. with two hidden layers having several units and letting the BNN to adapt to the data using techniques such as the Automatic Relevance Determination (Neal, 1996). Numerous data from various soil decontamination methods could also be included in the database. Another topic of future research could be to consider other sampling methods in the model. The MH method utilized in the model is considered to be a not so efficient sampler because it results in random walks in the parameter hyper-space. Other algorithms could be considered such the Hybrid-MH algorithm that uses a mixture of Gibbs sampler and MH algorithm for some of the parameters.

Chapters 5 and 6 presented two case studies related to the research. In Chapter 5 the Open Risk Analysis model and AGORA were investigated. The Open Risk Analysis AGORA model is a case of a specialized open standardization process. The process is investigated utilizing the analysis model developed in Chapter 3. The analysis of the collaboration network of AGORA even though it was based on minimal data it revealed that AGORA has the typical characteristics of scientific collaboration social networks such as large components, power-law degree distributions and small diameters. The investigation of AGORA also revealed organizational problems that can be expected in the creation of platforms that support open standardization processes such Open Risk Analysis. The Open Risk Analysis model and AGORA, even though they have their roots in the rather well developed and well known concept of open source, are new approaches and as a result not widely known and adopted. AGORA, depends heavily on the number and quality of its comprising members. A metric of the quality of a platform like AGORA is its output and contributions, i.e. the open software tools, methodologies and data it disseminates. One of the most important problems faced by AGORA and open collaboration platforms in general is the motivation of their members/collaborators to participate and contribute. A potential way to deal with this problem is to offer some kind of sustainable business models and or rewards to the collaborators. As a result, the study and creation of viable business models that take into consideration the special characteristics of AGORA is a very promising topic for future research. An additional problem encountered in open collaboration is the problem of free riding and "hijacking" of the collaborative content. This problem is well known in the open source community and attracts a lot of scholarly attention. Licenses such as the GNU GPL are in place to prevent such collaborative content "hijacking" but in reality they are not effective in preventing it. A preventive measure could be the rate of change and dissemination of content for example. Research into finding ways to minimize free riding and prevent collaborative content "hijacking" is also a very promising topic for future research.

Chapter 6 presented a new approach in creating open collaborative scientific content on the Internet via an online knowledge management system. One of the major difficulties in creating and disseminating open collaborative content is the issues related to intellectual property. Those issues are the most challenging and need to be addressed properly. For example, copyright issues must be considered at various times and activities such as during the authoring of articles by the first author, its editing by other collaborators and finally during the article's dissemination. Additionally, when users compile articles into anthologies (e-texts), they effectively create new (derivative) copyrighted works. Before doing so it is imperative to have a licensing (rights transfer) mechanism in place to handle all of the intellectual property issues. As a topic of future research, an investigation of how a non-profit virtual organization can be utilized in order to administer and protect the project and to help in dealing with the intellectual property issues. Additionally, various licensing schemes could be investigated that will allow the transfer of copyrights of the project's contents from the contributors to the organization and from the organization to the OCBP users. In the chapter a common legal intellectual property framework was presented. Within the intellectual property framework, the difficulties involved in open content creation, e.g. the "free-riding" problem were also presented. Open content licensing is one way to deal with this problem. Another suggestion is to have frequent updates and prompt dissemination of open content. That way open content "hijackers" will be discouraged in using content that they know soon will be rendered obsolete by a new update. This could also be a promising topic for future research.

Bibliography

R. Albert, H. Jeong, and A. L. Barabási. Diameter of the world-wide web. *Nature (London)*, 401(6749):130, 1999.

ATC-13. *Applied Technology Council, Earthquake Damage Evaluation Data for California ATC-13*. Applied Technology Council, 1985.

A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science (New York, N.Y.)*, 286(5439):509–512, 1999.

A. L. Barabási, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications*, 311(3-4):590 – 614, 2002. ISSN 0378-4371. doi: DOI:10.1016/S0378-4371(02)00736-7. URL http://www.sciencedirect.com/science/article/B6TVG-45S9HG2-1/2/dff30ba73ddd8820aca3e7f072aa7885.

P. Baran. On distributed communications networks. *IEEE transactions on communications*, 12 (1), 1964.

V. Batagelj and A. Mrvar. Pajek - analysis and visualization of large networks. In M. Jüngen and P. Mutzel, editors, *Graph Drawing Software*, Series Mathematics and Visualization, pages 77–103. Springer, 2003. ISBN 3-540-00881-0.

L. Benussi. Analysing the technological history of the open source phenomenon. stories from the free software evolution (working paper), 2005. Available at: http://pascal.case.unibz.it/retrieve/2312/benussi.pdf.

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

J. Bitzer and P. J. Schröder. Bug-fixing and code-writing: The private provision of open source software. *Information Economics and Policy*, 17:389–406, July 2005. doi: 10.

1016/j.infoecopol.2005.01.001. URL http://www.sciencedirect.com/science/article/B6V8J-4FGX828-1/1/52b9bcd7d45505a3697d70765f00db5b.

J. Bitzer and P. J. Schröder. The economics of open source software development: An introduction. In J. Bitzer and P. J. Schröder, editors, *The Economics of Open Source Software Development*, chapter 1, pages 1–13. Elsevier, 2006.

Y. Bramoullé and R. Kranton. Public goods in networks. *Journal of Economic Theory*, 135: 478–494, July 2007. doi: 10.1016/j.jet.2006.06.006. URL http://www.sciencedirect.com/science/article/B6WJ3-4M0J4YN-1/1/ef867ba44503251af609ffda3abaf56c.

W. L. Buntine and A. S. Weigend. Bayesian back-propagation. *Complex Systems*, 5(6):603–643, 1991.

P. J. Carrington, J. Scott, and S. Wasserman. *Models and Methods in Social Network Analysis*. Cambridge University Press, 2005. ISBN 0521809592.

B. Collins-Sussman, B. Fitzpatrick, and M. Pilato. *Version Control with Subversion*. O'Reilly, 2008.

K. Coyle. Open source, open standards. *Information Technology and Libraries*, 21(1):33–36, 2002.

CreativeCommons. Creative commons history, 2001. Available at: http://creativecommons.org/about/history.

DFO. Dartmouth flood observatory, world atlas of flooded lands, 2008. Available at: http://www.dartmouth.edu/~floods/Atlas.html.

C. DiBona, S. Ockman, and M. Stone, editors. *Open Sources: Voices from the Open Source Revolution*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1999. ISBN 1565925823.

M. Dilley, R. S. Chen, U. Deichmann, A. L. Lerner-Lam, and M. Arnold. *Natural Disaster Hotspots: A Global Risk Analysis*. The World Bank, 2005.

EM-DAT. Em-dat: The international disaster database, 2008. Available at: http://www.emdat.be/Database/Trends/trends.html.

D. C. Engelbart and W. K. English. A research centre for augmenting human intellect. In *Fall Joint Computer Conference*, 1968.

P. Erdös and A. Rényi. On random graphs i. *Publicationes Mathematicae Debrecen*, 6:290, 1959.

M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pages 251–262, New York, NY, USA, 1999. ACM. ISBN 1-58113-135-6. doi: http://doi.acm.org/10.1145/316188.316229. URL http://portal.acm.org/citation.cfm?id=316229.

R. M. Fano and F. J. Corbato. Time-sharing on computers. *Scientific American*, 215(3):128–140, 1966.

B. F. Fitzgerald. Open content licencing (ocl) for open educational resources. In *Proceedings OECD Expert Meeting on Open Educational Resources*, 2005. URL http://eprints.qut.edu.au/archive/00003621/.

L. C. Freeman. *The Development Of Social Network Analysis: A Study in the Sociology of Science*. Booksurge, 2004. ISBN 1594577145.

FSFE. Free software foundation europe: Open standards definition, 2009. Available at: http://fsfe.org/projects/os/def.

R. A. Ghosh. An economic basis for open standards. Technical report, FLOSSPOLS Project, 2005. Available at: http://flosspols.org/deliverables.php.

D. Giardini, G. Grünthal, K. M. Shedlock, and P. Zhang. The gshap global seismic hazard map. *Annals of Geophysics*, 42, 1999.

R. W. Hahn, editor. *Government Policy toward Open Source Software*. AEI Brooking Joint Center for Regulatory Studies, Washington DC, USA, 2002.

R. O. Hamburger. Building code provisions for seismic resistance. In W. F. Chen and C. Scawthorn, editors, *Earthquake Engineering Handbook*. CRC Press, 2003.

G. Hardin. The tragedy of the commons. *Science*, 162(3859):1243–1248, December 1968. doi: 10.1126/science.162.3859.1243. URL http://dx.doi.org/10.1126/science.162.3859.1243.

S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

C. Haythornthwaite. Social network analysis: An approach and technique for the study of information exchange. *Library & Information Science Research*, 18(4):323–342, 1996. URL http://dx.doi.org/10.1016/S0740-8188(96)90003-1.

P. Hoeppe. The munich climate insurance initiative (mcii) report, 2008. Available at: http://www.climate-insurance.org/upload/pdf/COP_11_Hoeppe.pdf.

G. J. Holland. The maximum potential intensity of tropical cyclones. *Journal of the Atmospheric Sciences*, 54, 1997.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:356–366, 1989.

HostGIS. Hostgis linux, 2008. Available at: http://www.hostgis.com/linux.

J. Howison, M. Conklin, and K. Crowston. Flossmole: A collaborative repository for floss research data and analyses. *International Journal of Information Technology and Web Engineering*, 1:17–26, July 2006.

IPCC. Intergovernmental panel on climate change (ipcc), climate change 2007, 2007. Available at: http://www.ipcc.ch/.

M. Ishimoto and K. Iida. Seismological observation by tremometer, 1. magnitude and distribution pattern. *Bull. Earthquake Res. Inst. Univ. Tokyo Univ.*, 17:443–478, 1939.

H. Jeffreys. *The Theory of Probability*. Oxford University Press, 1961.

H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L. Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

J. P. Johnson. Open source software: Private provision of a public good. *Journal of Economics & Management Strategy*, 11:637–662, 2002. doi: 10.1111/j.1430-9134.2002.00637.x. URL http://dx.doi.org/10.1111/j.1430-9134.2002.00637.x.

S. A. Kauffman. *The origins of order: self-organization and selection in evolution*. Oxford University Press, New York, 1993.

N. Kishi. Insurance industry perspectives, 2007. 1st International Workshop on Open-Source, Risk Software, California Institute of Technology; February 27th and 28th, 2007.

C. Lattemann and S. Stieglitz. Coworker governance in open-source projects. In J. Bitzer and P. J. Schröder, editors, *The Economics of Open Source Software Development*, chapter 7, pages 149–164. Elsevier, 2006.

J. Lerner and J. Tirole. The open source movement: Key research questions. *European Economic Review*, 45:819–826, May 2001. URL http://www.sciencedirect.com/science/article/B6V64-430XMS9-N/1/8266f6c9acd2625ecd3161eba7d6b515.

J. Lerner and J. Tirole. Some simple economics of open source. *Journal of Industrial Economics*, 50:197–234, 2002. doi: 10.1111/1467-6451.00174. URL http://dx.doi.org/10.1111/1467-6451.00174.

J. Lerner and J. Tirole. The scope of open source licensing. *Journal of Law, Economics and Organization*, 21(1):20–56, 2005. doi: 10.1093/jleo/ewi002.

J. C. R. Licklider. Man-computer symbiosis. *IRE Transactions on Human Factors in Electronics*, HFE-1:4–11, 1960.

V. Lindberg. *Intellectual property and open source*. O'Reilly, 2008. ISBN 9780596517960.

D. J. C. MacKay. *Bayesian Methods for Adaptive Models*. PhD thesis, California Institute of Technology, 1992a.

D. J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992b.

D. J. C. MacKay. Probable networks and plausible predictions - a review of practical bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6(3): 469–505, 1995.

J. Maguire. The sourceforge story, 2007. Available at: http://itmanagement.earthweb.com/cnews/article.php/3705731.

R. Miller, D. Clark, B. White, and W. Knottenbelt. *An Introduction to the Imperative Part of C++*. Online, 1996-2007. Available at: http://www.doc.ic.ac.uk/~wjk/C++Intro/.

K. Muraoka. Present condition of soil pollution and the soil contamination countermeasures law. *APEC-VC*, 2005. Available at: http://www.apec-vc.or.jp/e/modules/tinyd00/index.php?id=28&kh_open_cid_00=5.

NASA. National aeronautics and space administration: Earth observatory, 2008. Available at: http://earthobservatory.nasa.gov/.

R. M. Neal. Bayesian training of backpropagation networks by the hybrid monte carlo method. CRG-TR-92-1. Technical report, Department of Computer science, University of Toronto, 1992.

R. M. Neal. *Bayesian Learning for Neural Networks, Vol. 118 of Lecture Notes in Statistics*. Springer-Verlag, 1996.

R. R. Nelson. Recent evolutionary theorizing about economic change. *Journal of Economic Litterature*, XXXIII:48–90, 1995.

M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America*, 98(2):404–409, January 2001. ISSN 0027-8424. doi: 10.1073/pnas.021544898. URL http://dx.doi.org/10.1073/pnas.021544898.

NOAA. National oceanic and atmospheric administration, united states department of commerce: National geophysical data center, 2008. Available at: http://www.ngdc.noaa.gov/.

NWB-Team. Network workbench tool. indiana university, northeastern university, and university of michigan, 2006. Available at: http://nwb.slis.indiana.edu.

B. Perens. The open source definition, 1998. Available at: http://www.opensource.org/docs/definition.php.

D. Pountain. *The Penguin Dictionary of Computing*. Penguin Putnam, 2003.

J. C. Principe, N. R. Euliano, and W. C. Lefebvre. *Neural and Adaptive Systems: Fundamentals Through Simulations*. John Wiley and Sons, 2000.

E. S. Raymond. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly, 2001.

R. D. Reed and R. J. Marks. *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. MIT Press, 1999.

P. M. Romer. Endogenous technological change. *The Journal of Political Economy*, 98(5): S71–S102, 1990. ISSN 00223808. doi: 10.2307/2937632. URL http://dx.doi.org/10.2307/2937632.

M. A. Rossi. Decoding the free/open source software puzzle: A survey of theoretical and empirical contributions. In J. Bitzer and P. J. Schröder, editors, *The Economics of Open Source Software Development*, chapter 2, pages 15–55. Elsevier, 2006.

D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, 1986.

A. Scala, L. A. N. Amaral, and M. Barthélémy. Small-world networks and the conformation space of a lattice polymer chain. *Europhysics Letters*, 55:594–600, 2001.

C. R. Scawthorn. History of seismic risk assessment, 2006. Workshop on Strategic Directions for (Seismic) Risk Modeling and Decision Support, Boulder CO, July 2006. Mid-America Earthquake Center.

J. Scott. *Social Network Analysis: A Handbook*. Sage Publications, second. edition, 2000. ISBN 0761963391. URL http://www.amazon.com/Social-Network-Analysis-Professor-Scott/dp/0761963383/ref=sr_1_1?ie=UTF8&s=books&qid=1256622319&sr=1-1.

SEAOC. *Vision 2000, a Framework for Performance-Based Seismic Design*. Structural Engineers Association of California, 1996.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.

Smithsonian. Smithsonian institution, global volcanism program, 2008. Available at: http://www.volcano.si.edu/.

R. Stallman. Copyleft, 1984a. Available at: http://www.gnu.org/copyleft/.

R. Stallman. The free software definition, 1984b. Available at: http://www.gnu.org/philosophy/free-sw.html.

UNEP/GRID. Unep/grid-geneva preview, global cyclones tracks datasets, 2008. Available at: http://www.grid.unep.ch/data/gnv199.php.

U.S.C. U.s. code: Title 17, chapter 1, par. 106. exclusive rights in copyrighted works, 2009. Available at: http://www.law.cornell.edu/uscode/html/uscode17/usc_sec_17_00000106----000-.html.

M. Van Antwerp and G. Madey. Advances in the sourceforge research data archive (srda). In *Fourth International Conference on Open Source Systems, IFIP 2.13 (WoPDaSD 2008)*, Milan, Italy, September 2008.

F. Vega-Redondo. *Complex social networks*. Number 44 in Econometric Society monographs. Cambridge Univiversity Press, 2007. ISBN 978-0-521-85740-6.

S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1 edition, 11 1994. ISBN 0521387078.

J. White, E. Southgate, J. N. Thomson, and S. Brenner. The structure of the nervous system of the nematode c. elegans. *Philosophical transactions Royal Society London*, 314:1–340, 1986.

WIPO. Intellectual property handbook, 2009. Available at: http://www.wipo.int/about-ip/en/iprm/.

Appendix A

Program Listings

A.1 Program: Create Pajek conectivity input file

This program creates the connectivity input file for Pajek, based on Algorithm 1.

```
#include <iostream>
#include <algorithm>
#include <string>
#include <fstream>
#include <iterator>
#include <sstream> //stringstream library
#include <vector>

using namespace std;

int countLines (istream& in);
void StringExplode(string str, string separator, vector<string>* ↩
    results);
void StringSplit(string tmpstr, string tmparr[2]);

int main ()
{
  int z=0;
  int count;
  int num;
  vector<int> a;

  string str;
  string tempArray[2];
  string oldArray[2];
  vector<string> results;

  // count number of lines of all files passed as argument
  ifstream in("input.csv", ios::in);
  count = 0;
  if (!in) {
    cerr << "Failed to open file\n";
  }
  else {
    count += countLines(in);
    cout << "Number of lines in input file= " << count << endl;
  }
  in.close();

  // Pass data from file to vector
  ifstream in2("input.csv", ios::in);
  if (!in2) {
    cerr << "Failed to open file\n";
  }
  else {
    //cout << "Size of vector a is " << a.size() << endl;
    while (! in2.eof() )
      {
    cout << "Reading line " << z << "\r";
    getline (in2, str);
  //cout << " with content = " << str << endl;
  // Split String

  if ( z == 0 ) {
    // First line of input file
    StringSplit(str, tempArray);
    stringstream myStream(tempArray[0]); //create the stringstream
    myStream >> num; //convert the string to an integer
    a.push_back(num);
      oldArray[0] = tempArray[0];
      oldArray[1] = tempArray[1];
  }
  else {
          StringSplit(str, tempArray);
    if (tempArray[1] == oldArray[1]) {
      stringstream myStream(tempArray[0]);
      myStream >> num;
      a.push_back(num);
      oldArray[0] = tempArray[0];
      oldArray[1] = tempArray[1];
    }
    else {
      // Write Pairs to Output File

      // First sort the vector contents
      sort( a.begin(), a.end() );

      // Append data to output file
      //cout << "Vector size=" << a.size() << endl;
      fstream File;
      File.open("clustout.csv", ios::out | ios::app);
      //File.open("clustout.txt", ios::out | ios::app);
      if (File.is_open ()) {
        for ( int j = 0 ; j < a.size()-1 ; j++ )
        {
          for ( int i = j+1 ; i < a.size() ; i++ )
          {
            File <<  a.at(j) << "," << a.at(i) << endl;
            //File <<  a.at(j) << " " << a.at(i) << endl;
            //cout << "(" << a.at(j) << "," << a.at(i) << ")" << " ↩
                ";
          }
          //cout << "\n";
        }
      }
      File.close();
      // Clear Contents of vector a
      a.clear();
        //cout << "Vector size=" << a.size() << endl;

      StringSplit(str, tempArray);
    stringstream myStream(tempArray[0]); //create the stringstream
    myStream >> num; //convert the string to an integer
    a.push_back(num);
      oldArray[0] = tempArray[0];
      oldArray[1] = tempArray[1];
    }

  }
  z++;
  //cout << "z=" << z << endl;
      }
    in2.close();
  }
  return 0;
}

int countLines (istream& in)
{
  return count(istreambuf_iterator<char>(in),
         istreambuf_iterator<char>(),
         '\n');
}


void StringExplode(string tmpstr, string separator, vector<string>* ↩
    results)
{
  int found;
  found = tmpstr.find_first_of(separator);
  while(found != string::npos){
    if(found > 0){
      results->push_back(tmpstr.substr(0,found));
    }
    tmpstr = tmpstr.substr(found+1);
    found = tmpstr.find_first_of(separator);
  }
  if(tmpstr.length() > 0){
    results->push_back(tmpstr);
  }
}

void StringSplit(string tmpstr, string tmparr[2])
{
  int cutAt;
  string delim = ",";
  cutAt = tmpstr.find_first_of(delim);
  tmparr[0] = tmpstr.substr(0,cutAt);
  tmparr[1] = tmpstr.substr(cutAt+1);
}

//StringSplit(string str, string delim, vector<string> results)
//{
//int cutAt;
//while( (cutAt = str.find_first_of(delim)) != str.npos )
//  {
//    if(cutAt > 0)
//  {
//    results.push_back(str.substr(0,cutAt));
//  }
//    str = str.substr(cutAt+1);
//  }
//if(str.length() > 0)
//  {
//    results.push_back(str);
//  }
//}
```

A.2 Program: Pajek input file generator

This program creates an input file for Pajek.

```
#include <iostream>
#include <algorithm>
#include <string>
#include <fstream>
#include <iterator>
#include <sstream> //stringstream library
#include <vector>

using namespace std;

int countLines (istream& in);
void StringExplode(string str, string separator, vector<string>* ↩
    results);
void StringSplit(string tmpstr, string tmparr[2]);

int main ()
{
    int z=0;
    int i=0;
    int num, num1, num2;
    vector<int> a;

    string str;
    string str_usr;
    string tempArray[2];
    vector<string> results;
    //string user_group = "";
    //string user = "";
    //string outfile = "";
    char user_group[255];
    char user[255];
    char outfile[255];

    // Pass data from file to vector
    cout << "user_group file name (e.g. 2000.csv): ";
    cin >> user_group;
    cout << "user file name (e.g. users2000.csv): ";
    //getline(cin, user);
    cin >> user;
    cout << "Output file name (e.g. input2000.csv): ";
    cin >> outfile;

    fstream File;
    File.open(outfile, ios::out | ios::app);

    ifstream in2(user_group, ios::in);
    if (!in2) {
        cerr << "Failed to open file\n";
    }
    else {
        //cout << "Size of vector a is " << a.size() << endl;
        while (! in2.eof() ) {
            cout << "Reading line " << z << "\r";
            getline (in2, str);
            //cout << endl;
            //cout << " with content = " << str << endl;
            // Split String
            StringSplit(str, tempArray);
            stringstream myStream1(tempArray[0]); //create the ↩
               stringstream
            myStream1 >> num1; //convert the string to an integer
            stringstream myStream2(tempArray[1]);
            myStream2 >> num2;
            //cout << "num1=" << num1 << ", num2=" << num2 << endl;

            ifstream in3(user, ios::in);
            if (!in3) {
                cerr << "Failed to open file\n";
            }
            else {
                // COMMENT
                i = 1;
                while (! in3.eof() ) {
                    getline (in3, str_usr);
                    stringstream myStream(str_usr);
                    myStream >> num;
                    //cout << "num=" << num << endl;
                    if ( num1 == num) {
                        //fstream File;
                        //File.open(outfile, ios::out | ios::app);
                        if (File.is_open ()) {
                                    File <<  i << "," << num2 << endl;
                                    //cout << "Wrote to file: " << i << ", " << num2 << endl;
                        }
                        //File.close();
                        break;
                    }
                    i++;
                    //cout << "i=" << i << endl;
                    num = 0;
                }
                in3.close();
            }
            z++;
            num1 = -1;
            num2 = -1;
        }
        in2.close();
        File.close();
    }
    return 0;
}

void StringSplit(string tmpstr, string tmparr[2])
{
  int cutAt;
  string delim = ",";
  cutAt = tmpstr.find_first_of(delim);
  tmparr[0] = tmpstr.substr(0,cutAt);
  tmparr[1] = tmpstr.substr(cutAt+1);
}
```

Appendix B

The Open Source Definition (Annotated)

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. **Free Redistribution**

   The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

   > *Rationale: By constraining the license to require free redistribution, we eliminate the temptation to throw away many long-term gains in order to make a few short-term sales dollars. If we didn't do this, there would be lots of pressure for cooperators to defect.*

2. **Source Code**

   The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

   > *Rationale: We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.*

3. **Derived Works**

   The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

   Rationale: The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.

4. **Integrity of The Author's Source Code**

   The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

   Rationale: Encouraging lots of improvement is a good thing, but users have a right to know who is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations.
   Accordingly, an open-source license must guarantee that source be readily available, but may require that it be distributed as pristine base sources plus patches. In this way, "unofficial" changes can be made available but readily distinguished from the base source.

5. **No Discrimination Against Persons or Groups**

   The license must not discriminate against any person or group of persons.

   Rationale: In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process.
   Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.

6. **No Discrimination Against Fields of Endeavor**

   The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

   Rationale: The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.

7. **Distribution of License**

   The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

   Rationale: This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.

8. **License Must Not Be Specific to a Product**

   The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

   Rationale: This clause forecloses yet another class of license traps.

9. **License Must Not Restrict Other Software**

   The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

   Rationale: Distributors of open-source software have the right to make their own choices about their own software.
   Yes, the GPL is conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.

10. **License Must Be Technology-Neutral**

    No provision of the license may be predicated on any individual technology or style of interface.

    Rationale: This provision is aimed specifically at licenses which require an explicit gesture of assent in order to establish a contract between licensor and licensee. Provisions mandating so-called "click-wrap" may conflict with important methods of software distribution such as FTP download, CD-ROM anthologies, and web mirroring; such provisions may also hinder code re-use. Conformant licenses must allow for the possibility that (a) redistribution of the software will take place over non-Web channels that do not support click-wrapping of the download, and that (b) the covered code (or re-used portions of covered code) may run in a non-GUI environment that cannot support popup dialogues.

Appendix C

Torvalds message to comp.os.minix

```
Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones.  This has been brewing
since april, and is starting to get ready.  I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work.
This implies that I'll get something practical within a few months, and
I'd like to know what features most people would want.  Any suggestions
are welcome, but I won't promise I'll implement them :-)

                Linus (torvalds@kruuna.helsinki.fi)

PS.  Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT protable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's all I have :-(.
```

SOURCE: http://groups.google.com/group/comp.os.minix/msg/b813d52cbc5a044b

Appendix D

Open Knowledge Definition v1.0

D.1 Terminology

The term knowledge is taken to include:

1. Content such as music, films, books
2. Data be it scientific, historical, geographic or otherwise
3. Government and other administrative information

Software is excluded despite its obvious centrality because it is already adequately addressed by previous work.

The term work will be used to denote the item or piece of knowledge which is being transferred.

The term package may also be used to denote a collection of works. Of course such a package may be considered a work in itself.

The term license refers to the legal license under which the work is made available. Where no license has been made this should be interpreted as referring to the resulting default legal conditions under which the work is available (for example copyright).

D.2 The Definition

A work is open if its manner of distribution satisfies the following conditions:

1. **Access**

   The work shall be available as a whole and at no more than a reasonable reproduction cost, preferably downloading via the Internet without charge. The work must also be available in a convenient and modifiable form.

Comment: This can be summarized as 'social' openness - not only are you allowed to get the work but you can get it. 'As a whole' prevents the limitation of access by indirect means, for example by only allowing access to a few items of a database at a time.

2. **Redistribution**

   The license shall not restrict any party from selling or giving away the work either on its own or as part of a package made from works from many different sources. The license shall not require a royalty or other fee for such sale or distribution.

3. **Reuse**

   The license must allow for modifications and derivative works and must allow them to be distributed under the terms of the original work.

   Comment: Note that this clause does not prevent the use of 'viral' or share-alike licenses that require redistribution of modifications under the same terms as the original.

4. **Absence of Technological Restriction**

   The work must be provided in such a form that there are no technological obstacles to the performance of the above activities. This can be achieved by the provision of the work in an open data format, i.e. one whose specification is publicly and freely available and which places no restrictions monetary or otherwise upon its use.

5. **Attribution**

   The license may require as a condition for redistribution and re-use the attribution of the contributors and creators to the work. If this condition is imposed it must not be onerous. For example if attribution is required a list of those requiring attribution should accompany the work.

6. **Integrity**

   The license may require as a condition for the work being distributed in modified form that the resulting work carry a different name or version number from the original work.

7. **No Discrimination Against Persons or Groups**

   The license must not discriminate against any person or group of persons.

Comment: In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open knowledge. Therefore we forbid any open-knowledge license from locking anybody out of the process.

Comment: this is taken directly from item 5 of the OSD.

8. **No Discrimination Against Fields of Endeavor**

   The license must not restrict anyone from making use of the work in a specific field of endeavor. For example, it may not restrict the work from being used in a business, or from being used for genetic research.

   Comment: The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.

   Comment: this is taken directly from item 6 of the OSD.

9. **Distribution of License**

   The rights attached to the work must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

   Comment: This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.

   Comment: this is taken directly from item 7 of the OSD.

10. **License Must Not Be Specific to a Package**

    The rights attached to the work must not depend on the work being part of a particular package. If the work is extracted from that package and used or distributed within the terms of the work's license, all parties to whom the work is redistributed should have the same rights as those that are granted in conjunction with the original package.

    Comment: this is taken directly from item 8 of the OSD.

11. **License Must Not Restrict the Distribution of Other Works**

    The license must not place restrictions on other works that are distributed along with the licensed work. For example, the license must not insist that all other works distributed on the same medium are open.

    Comment: Distributors of open knowledge have the right to make their own choices. Note that 'share-alike' licenses are conformant since those provisions only apply if the whole forms a single work.

    Comment: this is taken directly from item 9 of the OSD.