IRM PRESS
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.irm-press.com
ITB7697

Chapter 10

Ethical Issues in Software Engineering Revisited

Ali Salehnia
South Dakota State University

Hassan Pournaghshband
Southern Polytechnic State University

The process of software development is usually described in terms of a progression from the project planning to the final code, passing through intermediate stages such as requirement analysis, system design, coding, system testing, and maintenance. One important aspect of these requirements concerns the reliability of the software. The use of computers for life-critical systems demands extremely high reliability of the computing functions as a whole. The consequences of negative results from unreliable systems and software are becoming public knowledge every day. Since these situations create a negative image for computer professionals and since these episodes create an environment of nontrust for the discipline, a good look at the ethical issues in software engineering is necessary. In this chapter, we look at each of the software engineering steps and the important aspect of their reliability and safety in the analysis, design, and implementation of software. We also examine the ethical aspects of the software and system development.

INTRODUCTION

As software become more complex and sophisticated, so too must the methods of writing these programs. Failure to take responsibility for errors will only mean more catastrophes. The price for these failures is rising even today and it will

Previously Published in *Managing Information Technology in a Global Economy*, edited by Mehdi Khosrow-Pour, Copyright © 2001, Idea Group Publishing.
This chapter appears in the book, Ethical Issues of Information Systems by Ali Salehnia.

be paid in the future by the computer professionals' through loose of dignity and trust (Lee, 1992). What follows are some examples of problems raised when ethical issues in software engineering were ignored.

In a local newspaper on May 20, 1996, we found an article with the following headline: "Bank error produces 800 near-billionaires." The story was about a programming error that increased the account balance by $924.8 million dollars for each of the bank's 800 customers. This is a total of $763.9 billion, which are more than six times the bank's assets.

On May 7, 1996 in another local newspaper we found an article with the title: "Error causes 2 jets to occupy same runway." This story explains how two passenger jets came within 1,500 feet of each other on the same runway because both were assigned the same flight number.

Another article titled, " Planes in Northwest lose link with air traffic control center," appeared on January 7, 1996 and the story explains how a regional center (part of a $1.4 billion computerized system) lost communications with an aircraft for a few seconds because of a software problem.

The following quotes were taken from a mug acquired at a 1982 ACM Computer Conference (Art101, 1982) in order to indicate to the reader how far we have come with software engineering steps and processes:

- Weinberger Law: "If builders built buildings the way programmers write programs then first woodpecker that came along would destroy civilization."
- Troutman's Programming Laws: "If a test installation functions perfectly all subsequent systems will malfunction; not until a program has been on production for at least six months will the most harmful error then be discovered; any program will expand to fill any available memory."
- Gioub's Laws of Computerdome: "The effort required to correct the error increases geometrically with time."
- Hare's Law of Large Programs: "Inside every large program is a small program struggling to get out."

The question is: What has changed or improved since 1982? Can we say that "the more thing change the more they stay the same?"

We believe that some of the problems in software development can be dealt with by computing professional if they are trained to explicitly practice ethical guidelines and accept their social responsibilities. Software developers are held responsible for the outcome of their software. Hence, they should also be held responsible if their design is at fault. Furthermore, they should assume total legal liability for their faulty and unreliable programs and they should be required to let their clients know when their systems fail to deliver something that is required of them, such as a missing function when the clients need it. Basili and Musa (1991) consider such an event as a reliability problem and therefore a failure.