Chapter II
The Social Study of Computer Science

Matti Tedre
Tumaini University, Tanzania

ABSTRACT

This chapter introduces the reader to some social research characteristics that are central to the social study of computer science. It introduces research studies that focus on the sociocultural aspects of computing and computer science, explains some of the central characteristics of those studies, and discusses their implications for the computer science discipline. Furthermore, this chapter is aimed at giving the reader a basic understanding of why social studies are important for the discipline of computing, as well as some broad guidelines and pointers towards carrying out such studies in computer science.

Our objective ... is to state precisely and clearly where and why sociological analysis is necessary in the understanding of scientific knowledge. Our main method is to present historical case studies. We then show how sociological analysis applies in these cases, and how it is an essential complement to even the most insightful interpretations derived from other perspectives.

—Barnes, Bloor, & Henry (1996)

INTRODUCTION

Computer science is a relatively new discipline, and it spans across traditional disciplinary boundaries, covering mathematical, engineering-oriented, and scientific traditions (Denning et al., 1989). From the birth of modern (digital, Turing-complete, electronic) automatic computing in the 1940s, those traditions have been essential to the development of the discipline. Modern computer science was born in the 1940s as a result of a number of organizations, a number of top people, many coincidences, a variety of disciplines, an uncommon political situation, a certain culture, unusually liberal funding, and

convergence of a number of technical and scientific breakthroughs (Tedre, 2006:passim).

Since the 1940s, modern computer science has been surrounded and shaped by a vastly complex conjunction of affairs. Due to their rich and colorful history, computer science and computer technologies include plenty of phenomena, the form and functioning of which cannot be explained in terms internal to those phenomena. For instance, one cannot explain the design and the (non-)diffusion of any programming language by referring solely to its technical characteristics (Sammet, 1991). Understanding the design and diffusion of any programming language requires understanding its history and the original motivations for its development in the first place (e.g., Denning, 2003; Rosenblatt, 1984). Similar, one cannot explain the development of GNU/Linux in solely technological terms—several non-technological motives, such as economic, political, ideological, and cultural motives, can be attributed to the development of GNU/Linux (cf. Tedre et al., 2006). Technical characteristics of GNU/Linux that stem from non-technological motives are perhaps better explained in other terms, such as in psychological, sociological, or anthropological terms.

So it is implausible that one could understand the current state, a static snapshot, of knowledge in computer science without understanding the history of computer science. Moreover, one cannot understand *why* knowledge in computer science is what it is without understanding the history of computer science. In addition to history, one must also understand how society and culture today shape computer science. As computer science is a product of an array of sociocultural forces, any portrayal of computer science is a historically, culturally, and societally specific image. Especially computer science as *human activity* always happens in some philosophical, historical, and sociocultural framework. That is, of course, not to say that computer science that is situated in a historical, cultural, and societal framework could not be *objective*. *Objectivity* can be defined in a number of ways that permit comparisons of socially constructed knowledge (e.g., Searle, 1996:p.8). For example, objectivity can refer to how strong consensus there is concerning a specific statement.

The importance of historical, cultural, and societal self-understanding of computer science are explicitly noted in the ACM/IEEE computing curricula CC1991 and CC2001 (Tucker et al., 1991: p.73; Denning et al., 2001:p.141). Those curricula emphasize the importance of understanding cultural, social, legal and ethical issues; and stress the appreciation of philosophical questions, technical problems, and aesthetic values. It is, however, uncertain how exactly should philosophical questions, technical problems, and aesthetic values be studied. Neither is it certain how the cultural, social, legal, and ethical issues in computing should be approached. One approach that originates from science and technology studies is *social studies of computer science*—that is, research of computer science itself in its sociocultural context. The focus of social studies of computer science is different from that of social studies of computing as the former is focused on the discipline, whereas the latter is focused on the activity. Social studies of computer science aims at enriching disciplinary self-understanding of computer science by producing meta-knowledge about computer science. That knowledge helps computer scientists to delineate between brute facts (like the laws of nature) and socially constructed facts (like standards and models).

THE CONTRIBUTION OF SOCIAL STUDIES OF COMPUTER SCIENCE

Researchers of social studies of computer science often adopt different conceptual and theoretical frameworks, and start from different sets of assumptions. Often those assumptions are in line with the constructionist, contingent, non-relativist, and nominalist viewpoints of science. In other words, social studies of computer science often entails the assumptions that much of people's knowledge is constructed (rather than absolute), that the history and development of current computer science is one out of an infinite number of possible routes (rather