Communications of the Association for Information Systems

Volume 36 Article 27

4-2015

Review: The Technical and Social History of Software Engineering

Frank Land
London School of Economics and Political Science, f.land@lse.ac.uk

Communications of the Association for Information Systems

CAIS

Review: The Technical and Social History of Software Engineering

Frank Land

London School of Economics

F.Land@lse.ac.uk

Abstract:

In his recent book The Technical and Social History of Software Engineering, Capers Jones promises to fill a long-standing need for a text that gives readers an understanding of the history of software engineering. In this paper, I examine the extent to which Jones succeeds in meeting that promise.

Keywords: Software Engineering, IT History.

Editor's note: The article was handled by Ping Zhang, the Department Editor for History of Information Systems

Volume 36, Article 27, pp. 577-581, April 2015

The manuscript was received 14/04/2014 and was with the authors 0 months for 0 revisions.

The author, Capers Jones, has a long record of making significant contributions to software metrics and the economics of software, which includes the publication of 15 books related to these topics. Hence, I eagerly received his new book on the technical and social history of software engineering. The book itself is long at over 452 pages, but the subject is large and, as Jones notes, if the book had been fully comprehensive, it would have required at least 1000 pages.

Jones has set out to chronicle the innovations, the entrepreneurs, and the organizations (private and public) associated with information technology's evolution and its software. He begins with a prelude that provides lists of innovations in mathematics, recording, and calculating from the beginning of the Stone Age to 1930. Subsequent chapters cover each decade from 1930 to 2019. In the book's final two chapters, he reviews the problems and failings of software engineering and reports on the threats posed by cybercrime and cyberwarfare. Throughout these chapters, although he focuses primarily on the American experience, he does make an attempt to include international contributions.

The story Jones tells is often interesting and illuminating, but, overall, I found the book disappointing. It is not the history of software engineering, technical or social, that a student of the subject might expect. Indeed, one expects a history of software engineering to provide an account of the ideas behind the way software engineering developed, including accounts of some of the debates such as the one triggered by Edsger Dijkstra's letter in the *Communications of the ACM* on the Go To statement[1]. However, Jones provides few such examples. Jones misses the opportunity to add clarity—and color—to his narrative, by omitting important episodes and not providing illustrative examples

Thus, he does not mention Vannever Bush's "invention" of the Memex[2], first descibed by Bush in the Atlantic Monthly in July 1945 (Bush, 1945). The Memex was an important milestone and surely ranks close to Turing's discoveries on computability and in his anticipation of hypertext and, thus, the World Wide Web. Nor does he mention Daniel Teichroew's[3] landmark program to devise a system that would automate the production of computer code directly from a requirements specification. The project (named information system design and optimization system (ISDOS)) developed a special program specification language (PSA) and an associated problem-statement analyzer (PSL) followed by a code-optimization process (SODA). The system was successfully deployed in industry but also had important impact in education and in triggering much fruitful research.

Further, Jones does not adequately address the critical role played by government and politics in the evolution of IT and its enabling software. While he does discuss developers in defense computing throughout the book, he neglects the larger role of governments. A good example is the Japanese Government-sponsored project known as the 5th Generation in 1982[4]. This US$850 million project was intended to push computing into new realms of knowledge-based AI systems built on improvements in both electronics and software engineering. Japan's success in this time period, with its increasingly dominant role in the automobile and electronic industries, led Governments and industries world-wide to fear Japan's supposed future dominance in computing. As a result, the US (under DARPA), the UK (Alvey)[5], France (Plan Calcul[6]), and the European Union (ESPRIT[7]) each sponsored major multi-million dollar research projects that brought academia, industry, and governments into collaborative research schemes. In the UK, for example, the Alvey Programme, launched in 1983 with funding of £350 million over 4 years for industry/academia projects included:

- Very large scale integration (VLSI) technology for microelectronics
- Intelligent knowledge-based systems (IKBS) or artificial intelligence (AI)
- Software engineering including the design and construction of computer-aided software engineering (CASE) tools

[1] Edsger Dijkstra *Communications of the ACM*, Vol. 11, No. 3, March 1968, pp. 147-148.
[2] http://en.wikipedia.org/wiki/Memex
[3] http://www.pslpsa.com/index.php/79-isdos-category/90-dt-article
[4] http://en.wikipedia.org/wiki/Fifth_generation_computer
[5] http://hansard.millbanksystems.com/written_answers/1988/jan/21/alvey-project
[6] http://en.wikipedia.org/wiki/Plan_calcul
[7] https://aclweb.org/anthology/H/H91/H91-1007.pdf

- Man-machine interface (included natural language processing), and
- Systems architecture (for parallel processing).

Other governments also created initiatives that were concerned with structuring and restructuring their countries' computer industries. This included the attempt in the 1970s to bring together the major players in the Dutch, German, French, and U.K. computer industries as one company, UNIDATA[8]. However, the attempt ultimately failed, partly because of the reluctance of the U.K.'s major player ICL to join the consortium.

In 1966, Soviet Union authorities, noting the IBM 360 series' worldwide success, took the momentous, but heavily criticized, step to abandon the Soviet's indigenous computer designs and replace them with IBM clones[9]. The Russians managed to clone the hardware quite well, but had more problems with the software engineering in that they failed to properly re-engineer IBM systems software. At that time, an embargo (CoCom[10]) initiated by the U.S. Government and its allies as part of its cold-war strategy prevented American computer technology from being exported to the Soviets and its allies.

Although most of the initiatives, including the Japanese 5th generation project, failed to achieve their major targets, they left a lasting legacy through increased collaboration across borders and between universities, research establishments, and industry.

One further government-sponsored initiative that Jones should have noted was the launching in France and the UK of the Minitel[11], Ceefax[12], and Prestel[13] teletext services, which provided subscribers with an information service direct to their television screen. This anticipated much of the Web-based services we now rely on. The French system Minitel had considerable success with millions of subscribers, in part due to the subsidies provided for subscribers. It lasted well into the 21st century. The uptake of Prestel services was more limited and faltered earlier, which resulted in the British Government abandoning the service, but Ceefax ran well into the 21st century.

In Jones' final chapter (Chapter 12), he discusses Cybercrime's rise and briefly notes cyberwarfare's future. Unfortunately, he wrote the book too early to catch Edward Snowden's revelations of the U.S.'s and U.K.'s government agencies' joint mass surveillance of their citizens. The advance of computing and communication capabilities enabled by software engineering have provided the means by which suitably equipped agents can capture details of mobile phone calls, email messages, stored data, searches made via browsers such as Google, Facebook and Twitter messages. These agents can even decrypt coded information. Snowden's revelations indicate that Government agencies capture significant proportion of all Internet traffic. Ostensibly, the purpose of this mass surveillance is security against terrorist attack and criminal intent. In any case, any further editions of the book will need to discuss the privacy and security security aspects of Snowden's revelations, including the security of the surveillance agencies themselves[14].

Jones omits several other events that a history such as his might be expected to cover: these include Isaac Auerbach's[15] establishment of the International Federation of Information Processing (IFIP). IFIP has several technical committees that comprise international working groups covering a range of topics including computer languages, software engineering, different types of systems, all aspects of computer education, and computing history (working group 9.7). Some of these working groups have unquestionably made major contributions to their fields of study, including the software engineering field.

Jones only lightly covers the role played by international standards organizations[16] such as ISO and the influence they have wielded in the evolution of software practices (both positive and negative), and he does so in an unsystematic manner. Other examples of important events and milestones in the evolution of IT include is the much-discussed productivity paradox, which emerged as a concept in the 1990s, though the economic value of investing in business computer systems had been listed as a top-ranking concern of business leaders in survey after survey since the 1960s[17].

[8] http://www.feb-patrimoine.com/histoire/english/chronoa8.htm
[9] http://en.wikipedia.org/wiki/ES_EVM
[10] http://en.wikipedia.org/wiki/CoCom
[11] http://en.wikipedia.org/wiki/Minitel
[12] http://en.wikipedia.org/wiki/Ceefax
[13] http://en.wikipedia.org/wiki/Prestel
[14] http://en.wikipedia.org/wiki/Edward_Snowden
[15] http://en.wikipedia.org/wiki/International_Federation_for_Information_Processing
[16] http://www.stanhopecentre.org/cotswolds/IT-Standardisation_Jakobs.pdf
[17] http://ccs.mit.edu/papers/CCSWP130/ccswp130.html

Communications of the Association for Information Systems

Too often, Jones poorly balances how he treats topics. For instance, he sparingly covers the important topic of enterprise resource planning (ERP)[18], which has transformed the way many companies arrange their information processing by, for example, changing the role of systems designers to implementing the design created by an external vendor. However, the Norden bombsight, which had had little impact on software engineering, is discussed at some lengths (pages 59-61). I could include many other examples that demonstrate Jones' poor judgment on how what topics to include and how much attention to give them. The problem is exacerbated by Jones' use of long lists: for example, list of beneficial tools and applications (page 23) covers more than 9 pages. Jones repeatedly argues the value of an analysis of program complexity by using function points as opposed to counting lines of code; however, his long lists resemble a simplistic form of description rather like lines of code.

Jones lists the fascinating Antikythera mechanism (which was possibly invented by Archimedes, had (as far as we can determine) the capability of predicting solar and lunar eclipses, and whose identification resulted in a complete reappraisal of the ancient Greeks' mathematical and engineering expertise) in Table 1.1 without any explanation. Nor does he mention or comment on inventions such as double-entry book keeping or the re-invention of movable type by Gutenberg around 1450[19], which was perhaps as important in the late Middle Ages as the invention of electronic computers in the 20th century. Further, missing from his discussion on software engineer's early history is the sophistication reached by automata in the 16th, 17th, and 18th centuries, including the importance of arranging the figures' complicated movements via a fixed program determined by the shaping of a cam and later by the pattern of holes in a card driving musical instruments and industrial machinery as in the Jacquard Loom. He also does not mention Maurice Wilkes'[20] 1951 invention of micro-programming, even though the technique became central to the IBM 360 range's architecture by permitting compatibility across the range.

Jones often lists the names of important contributors with only the soubriquet "famous" before them and without explaining what a person's fame is based on. Reader who have not come across that name but are interested have to search elsewhere to find out what the named person is famous for.

More importantly, each chapter has a table (see, for example, Table 8.3 on page 243) that summarizes the growth of software applications in the US in the decade the chapter covers. Jones does not provide any information about how he has constructed these tables or the tables' information sources. Additionally, the individual items are often ambiguous. What, for example, are information technology applications? What counts as an application? How accurate is the breakdown and the numbers?

Generally speaking, Jones does not make clear who the book is written for, and reading it does not make it any clearer. The book provides little to help the software engineering student understand the ideas or practices that underlie software engineering. Moreover, scholars will be concerned at the lack of citations and the many omissions. The layman reader might be irritated by the long lists and the lack of explanations.

Despite these criticisms, Jones' book does have its virtues. It provides a readable account of the way the IT industry has grown from very small beginnings both in range of application and in the depth of its penetration into every aspect of our lives worldwide. In addition, the later chapters (those on the problems and failings of software engineering and on the threats posed by cybercrime and cyberwarfare) are excellent. I liked Capers Jones' reminiscences of his early days as a programmer at IBM in part because they mirrored my own experience with LEO computers a decade earlier.

Perhaps it would require at least two volumes to provide the kind of history of software engineering I would like to see. Is this a challenge Capers Jones is willing to meet?

REFERENCES

Editor's Note: The following reference list contains hyperlinks to World Wide Web pages. Readers who have the ability to access the Web directly from their word processor or are reading the paper on the Web, can gain direct access to these linked references. Readers are warned, however, that:

1. These links existed as of the date of publication but are not guaranteed to be working thereafter.
2. The contents of Web pages may change over time. Where version information is provided in the References, different versions may not contain the information or the conclusions referenced.
3. The author(s) of the Web pages, not AIS, is (are) responsible for the accuracy of their content.

The author(s) of this article, not AIS, is (are) responsible for the accuracy of the URL and version information.

[18] http://en.wikipedia.org/wiki/Enterprise_resource_planning
[19] http://en.wikipedia.org/wiki/Johannes_Gutenberg
[20] http://en.wikipedia.org/wiki/Microcode

Bush, V. (1945). As we may think. *Atlantic Monthly.*

Jones, C. (2014). *The technical and social history of social engineering*. Upper Saddle River, NJ: Addison Wesley.

ABOUT THE AUTHOR

Frank Land started his career in computing with J. Lyons, in 1953, working on the pioneering LEO Computer first as a programmer and then as a systems analyst. In 1967, he left industry to join the London School of Economics on National Computing Centre grant to establish teaching and research in systems analysis. This led to the formation of the Department of Information Systems of which he became convener. The department developed teaching programs at graduate level (Masters and PhD) and a lively research activity, the first university in the UK to establish such programs. In 1982, he was promoted to Professor of Systems Analysis. In 1996, he joined the London Business School as Professor of Information Management. He retired in 1991 and was appointed Emeritus Professor at the LSE in the Department of Information Systems in 2000. Frank Land has been Visiting Professor at the Wharton School, the University of Sydney, the University of Cairo, Bond University, Curtain University, and is currently Visiting Professor at Leeds Metropolitan University. He is past chairman of IFIP WG 8.2 and on the editorial board of several academic journals. He is a Fellow of the British Computer Society, was awarded a Fellowship of the AIS in 2001, and received the AIS LEO Award in 2003.

Communications of the Association for Information Systems

ISSN: 1529-3181