Dan Bricklin Proposes 200-Year Software

by M. E. Kabay, PhD, CISSP
Associate Professor, Information Assurance
Norwich University, Northfield VT

Dan Bricklin and Bob Frankston created VisiCalc in 1978, which some of my older readers will recall was the first electronic spreadsheet (it ran on the Apple II). The Computer Desktop Encyclopedia <http://www.computerlanguage.com> states, “Thousands of $3,000 Apples were bought to run the $150 VisiCalc.” In the decades since that early success, Mr. Bricklin has contributed in many ways to software development. He has worked with Software Garden Inc., Slate Corp., Trellix Corp., and Interland, Inc. and has won many awards and honors <http://www.bricklin.com/pressinfo.htm>.

I was interested to read an abstract of Mr. Bricklin’s stimulating essay, “Software That Lasts 200 Years,” <http://www.bricklin.com/200yearsoftware.htm> last year in the INNOVATION newsletter <http://www.newsscan.com/>. The author points out that many aspects of our society are created with relatively long expected lifespans; e.g., buildings, roads, bridges and so on. In contrast, most computer software has been written under the assumption that it will last only a few years. The Y2K debacle was a result of assuming that software written in the 1970s could not possibly still be in use 25 years later. Bricklin points to accounting standards as further evidence of the short-term expected lifespan of software: “In accounting, common depreciation terms for software are 3 to 5 years; 10 at most. Contrast this to residential rental property which is depreciated over 27.5 years and water mains and brick walls which are depreciated over 60 years or more.”

Bricklin makes the point that computers are increasingly responsible for storing important societal documents which until recently were stored on relatively stable paper. As I pointed out in my 1995 paper, “Eternity in Cyberspace” (available in HTM or PDF from <http://www.mekabay.com/overviews/>), there are serious issues of long-term readability of computer-based records due to changes in application software, operating systems, and hardware.

Bricklin goes on to discuss factors that are conducive to short time horizons as the norm in software development. He proposes interesting changes in the ways that society manages software development, including shifting from private corporate sources towards more public efforts that include explicit emphasis on longevity and portability. He brings in lessons from civil engineering, where

- a common body of knowledge is the basis of professional education in the field,
- standards bodies collate and publish best practices,
- publicly funded or industry based inspections are normal, and
- failures lead to public investigations and published reports (think of what we read in the “Risks Forum Digest” only even more thorough – see <http://catless.ncl.ac.uk/Risks/> for countless examples of analysis of system failures).
It seems to me that taking a longer-term perspective on software engineering must also involve integrating security considerations in all aspects of systems development from the very first discussions of functional requirements all the way to long-term maintenance and evolution of our systems. The same principles should apply to network design and implementation.

I hope you will read Mr Bricklin’s entire analysis and think about it. We need to fix these underlying problems before we work ourselves into yearly equivalents of Y2K disasters.

* * *


M. E. Kabay, PhD, CISSP is Associate Professor in the Division of Business and Management at Norwich University in Northfield, VT. Mich can be reached by e-mail at <mailto:mkabay@norwich.edu>; Web site at <http://www.mekabay.com/index.htm>.

Copyright © 2005 M. E. Kabay. All rights reserved.

Permission is hereby granted to Network World to distribute this article at will, to post it without limit on any Web site, and to republish it in any way they see fit.