50 & 25 YEARS AGO

EDITOR **ERICH NEUHOLD**
University of Vienna
erich.neuhold@univie.ac.at

APRIL 1972

In the early years, *Computer* was only published bimonthly. Therefore, we will have to skip our interesting and/or informative extractions for April. The next one will appear in the May 2022 issue of *Computer*, and we hope you will eagerly wait for our next publication of this column.

APRIL 1997

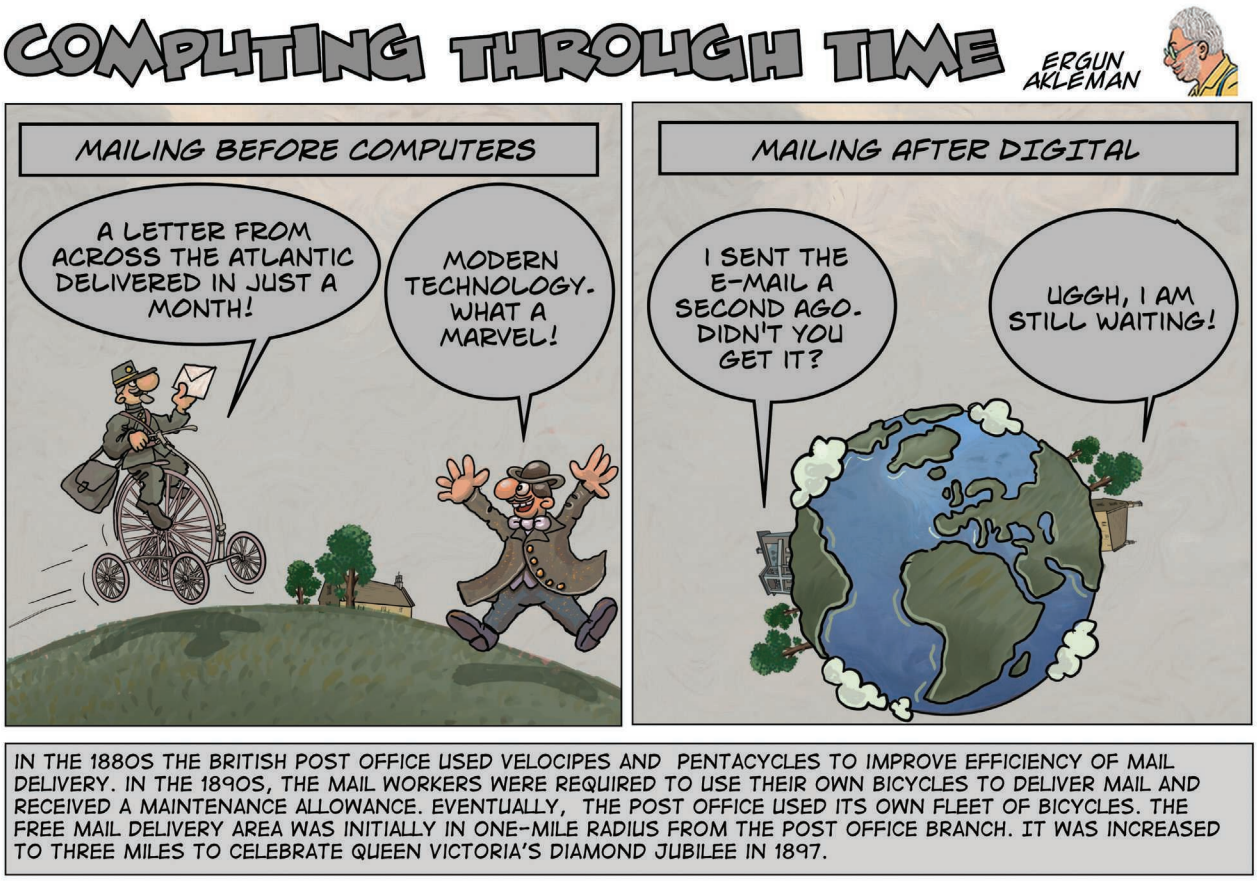
https://www.computer.org/csdl/magazine/co/1997/04

Open Cannel: The Human Side of Architectural Styles; Dirk Riehle (p. 10) "I am slowly recovering from my recent transition from university to industry. ... Instead, I learned about human styles. Three styles I encountered have a negative effect on architecture: Spaghetti Code ... Bureaucracy ... Let's Go Meta. I also learned about styles that have a positive effect on project success: Worse is Better ... Lone Island ... Straightjacket. ... Software development can be interpreted as the application of architectural styles to software systems. ... The paper thereby concludes that architectural styles are by their very nature a human endeavor." *[Editor's note: This article provides a somewhat cynical view of human behavior and "architectural" styles in software development. It discusses the possible behavior of software teams, but in the end it unfortunately leaves any evaluation of these styles to the reader.]*

Students Stumble onto Internet Explorer Flaw; Lee Garber (p. 18) "Three undergraduate students uncovered a major flaw in Microsoft's Internet Explorer browser. Microsoft had to scramble to fix the flaw, which could have given a hacker access to and control over an Internet Explorer user's computer. ... The flaw could have allowed hackers to delete files and folders, steal data, move money out of online bank accounts, or even wipe out a hard drive." (p. 20) "Because of this, Balle said, Microsoft is exploring ways to let more users know about the bug ... The Internet is rife with these types of security problems, said Myron Cramer, a principal research scientist for Internet security issues at the Georgia Tech Research Institute." *[Editor's note: As we all know, this type of problem still plagues us today. Just think of the zero-day exploits that keep happening. But the user warning and update problem when such errors were found led to the automatic update approach mostly used today.]*

Sources of Failure in the Public Switched Telephone Network; D. Richard Kuhn (p. 28) "To account for the possible effects of seasonal fluctuations in call-processing volume, I analyzed failures over two years, from April 1992 to March 1994, beginning with the earliest FCC reports. ... Major sources of failure were human error (on the part of both telephone company personnel and others), acts of nature, and overloads. Overloads caused nearly half of all downtime (44 percent) in terms of outage minutes." (p. 35) "Despite its enormous size and complexity, the PSTN averaged an availability rate better than 99.999 per-cent in the time period studied. Why should perhaps the world's largest and most complex computerized distributed system also be among the most reliable? ... We can consider the PSTN a loosely coupled system because it can dynamically reroute calls along many paths." (p. 36) "Designers devote about half of the software in tele-phone switches to error detection and correction. Such a high percentage of self-checking is probably atypical for software systems. ... In addition to built-in self-test and recovery mechanisms, operators monitor telephone switches 24 hours a day and usually have the ability to modify switch software on the fly." *[Editor's note: This detailed and interesting analysis actually happened before the wide availability of the Internet and its protocols. It would be interesting to see a similar analysis of today's Internet and its failures. Vandalism played a small role in this study, and we might wonder how it would be today.]*

Digital Object Identifier 10.1109/MC.2022.3143627
Date of current version: 8 April 2022



Digital Object Identifier 10.1109/MC.2022.3150933
Date of current version: 8 April 2022

Understanding Fault Tolerance and Reliability; Arun K. Somani et al. (p. 45) "Future systems will be more complex and so more susceptible to failure. Despite many proposals in the past three decades, fault tolerance remains out of the reach of the average computer user. The industry needs techniques that add reliability without adding significant cost. ... Systems fail for many reasons. The system might have been specified erroneously, leading to an incorrect design. Or the system might contain a fault that manifests only under certain conditions that weren't tested. The environment may cause a system to fail. Finally, aging components may cease to work properly. It's relatively easy to visualize and understand random failures caused by aging hardware. It's much more difficult to grasp how failures might be caused by incorrect specifications, design flaws, substandard implementation, poor testing, and operator errors." (p. 50) "Once a fault-tolerant system is designed, it must be evaluated to determine if its architecture meets reliability and dependability objectives. There are two ways to evaluate dependability: using an analytical model or injecting faults." *[Editor's note: The five articles that follow this analysis of fault tolerance and reliability describe different approaches to increase reliability and availability in systems of different mission critical requirements. I will only briefly describe each of them as much of what is said there is still needed and used today.]*

Toward Systematic Design of Fault-Tolerant Systems; Algirdas Avizienis (p. 51) "The greater the benefits these systems bring to our well-being and quality of life, the greater the potential for harm when they fail to perform their functions or perform them incorrectly. ... At the same time, threats to dependable operation are growing in scope and severity." (p. 52) "Here I summarize the most mature version of the guidelines for bottom-up fault tolerance. An abstraction of observed design processes in which steps often overlap, it is offered as a way to minimize the probability of oversights, mistakes, and inconsistencies that may occur during the implementation of fault tolerance. The first three steps—specification, implementation, and evaluation—deal with the building of a new system. Implementation and evaluation are concurrent. Step four—modification—addresses the repair or

augmentation of an existing design." *[Editor's note: In this article, a detailed analysis of the fault tolerant aspects of these four steps is provided. Most of these principles are still relevant and practiced today.]*

Piranha: A CORBA Tool for High Availability; Silvano Maffeis (p. 59) "A widely used distributed system standard is the Object Management Group's Common Object Request Broker Architecture. ... First, Piranha acts as a network monitor that reports failures through a graphical user interface. Second, Piranha acts as a manager: It automatically restarts failed CORBA objects, replicates stateful objects (objects that maintain an internal set of values) on the fly, migrates objects from one host to another, and enforces predefined replication degrees—numbers of copies—on groups of objects." *[Editor's note: The remainder of the article describes in detail the prototypical implementation of the system and shows the help it offers to the system administrator to detect earlier possible availability problems.]*

Software-Based Replication for Fault Tolerance; Rachid Guerraoui et al. (p. 68) "To discuss replicated servers, we must first explain the correctness criterion linearizability. Sometime called one-copy equivalence, linearizability gives clients the illusion of non-replicated servers." (p. 74) "The relationships we have discussed between, for example, primary-backup replication and view-synchronous multicast, illustrate the convergence of replication techniques and group communications." *[Editor's note: The detailed analysis of various replication techniques and their behavior in case of failure is interesting but repeats quite a bit of what is known about multitask and multithread behavior in software only systems.]*

Fault Injection Techniques and Tools; Mei-Chen Hsueh et al. (p. 75) "Fault injection is important to evaluating the dependability of computer systems." (p. 76) "Engineers use fault injection to test fault-tolerant systems or components. Fault injection tests fault detection, fault isolation, and reconfiguration and recovery capabilities. ... Choosing between hardware and software fault injection depends on the type of faults you are interested in and the effort required to create them." *[Editor's note: The article then discusses various techniques of using fault injection both in hardware and software to increase failure tolerance but also discusses the problems that appear if injected faults are not detected.]*

Fault-Tolerant, Real-Time Communication in FDDI-Based Networks; Biao Chen et al. (p. 83) "FAULT-TOLERANT, REAL-TIME SYSTEM MODEL—An FBRN consists of FDDI trunk rings, each composed of two fiber loops. The two loops transmit messages in opposite directions, one clockwise, the other counterclockwise. An FBRN can consist of many such trunk rings." (p. 85) "The objective of fault-tolerant real-time management in FBRN is to manipulate network resources and messages so that all communication requirements are met. Our approach deals with message groups rather than individual messages. This greatly reduces online management overhead. ... There are three basic approaches to fault tolerance: temporal redundancy, spatial redundancy, and integrated temporal/spatial redundancy." *[Editor's note: A detailed analysis of these three approaches follows in the article. However, it is noteworthy that the Fiber Distributed Data Interface protocol was soon replaced by the Fast Internet and later Gigabit Ethernet, as they have been easier to handle.]*

Pitfalls and Strategies in Automated Testing; Cem Kaner (p. 114) "Though some efforts to use these tools have been successful, several have failed miserably. ... Maintenance requirements don't go away just because your friendly automated tool vendor forgot to mention them. ... When the UI language changes ... When the program's UI changes ... We need strategies we can count on to deal with these issues. Capture-based test-case creation and test scripting done in a tester's "spare time" are ad hoc approaches, not working strategies." *[Editor's note: The short article investigates some of the strategies needed for successful testing but is far from recommending "automated" testing as an exclusive tool. We should note, those recommendations still hold in today's testing field.]*

Comparing Internet Search Engines; Andrew Kingoff (p. 117) "With more than 150 search engines available, choosing the right one (or ones) is important. ... I narrowed the field down to the four I found most useful: Alta Vista, Deja News, Excite, and Yahoo." *[Editor's note: This article is not very deep as it uses a few simple examples to demonstrate the search engines. It is more interesting that of the four search engines discussed, only Yahoo is still around; Google only appeared in 1998 and still was able to take over the search engine world.]* C