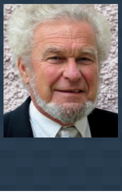
50 & 25 YEARS AGO



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

APRIL 1970

In the early years, *Computer* was only published bimonthly. Therefore, we will have to skip our interesting and/or informative extractions for this month and continue in an upcoming issue of *Computer.*

APRIL 1995

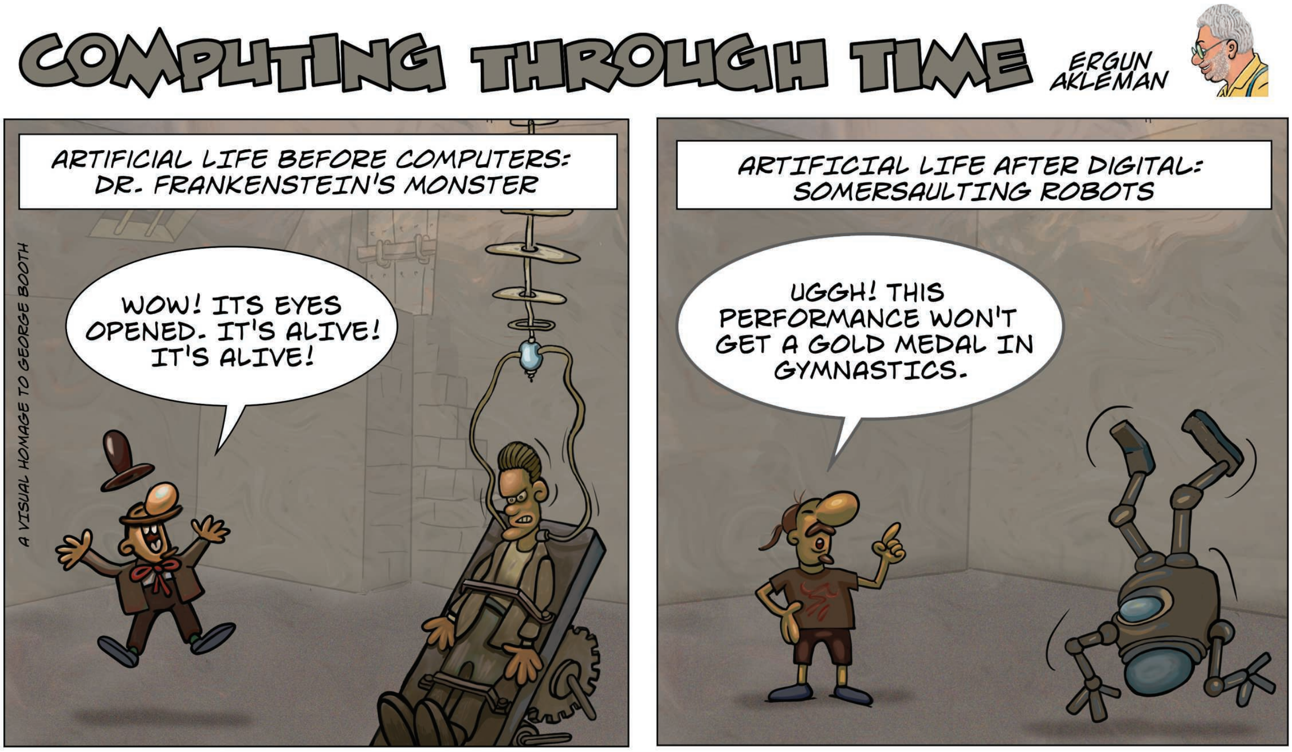
www.computer.org/csdl/mags/co/1995/04/index.html

Infobusiness Meets Neuromancer (p. 7) "Freeside is many things, not all of them evident to the tourists who shuffle up and down the well. Freeside is brothel and banking nexus; pleasure dome and free port, border town and spa. … Infobahn is many things, not all of them evident to Net cruisers who shuttle up and down the web. Infobahn is social lubricant to the verbose, heir apparent to TV's wasteland, and magnet to intellectuals in hot pursuit of the world's 'knowledge at my fingertips.' Infobahn the new shopping mall in multimedia colors, venture capitalist dance craze of the 1990s, political mantra of the ultraleft, right, and middle-of-the-road. … The stampede to the Internet is an entirely new phenomenon that cannot be compared with existing technology. … You can't match the interactivity and instant gratification with a static mail-order catalog." *[Editor's note: This is a short article that very correctly predicts the development of the intervening 25 years.]*

Distributed Coordination Models for Client/Server Computing (p. 14) "Distributed architectures let users of individual, networked computers share programs and data resources, often between geographically remote sites. … One coordination model widely used in distributed systems is the client/server architecture. A program, the client, requests an operation or service that some other application, the server, provides." (p. 15) "A major limitation in the basic client/server model is its focus on clients requesting individual services. Clients often need to invoke multiple services, coordinated to reflect how those services interrelate and contribute to the overall application. … This article identifies and examines extensions to the basic client/server model that provide explicit support for coordinating multiserver interactions." (p. 16) "NetWorks is a tool based on object-oriented message-passing technology. The Networks Messaging Facility (NMF) provides its core distributed-communication capabilities. The NMF kernel contains message queues, queue management services, message transport services (network drivers), and a scheduler." (p. 22) "The object-oriented architecture presented here integrates diverse client/server coordination models. Developers capture the control logic for managing complex combinations of application services using high-level APIs." *[Editor's note: Over the years, distributed systems and their architectures have undergone many convolutions, mostly by renaming old problems. Be it parallel computing, client-server computing, networked computers, grid, cloud, or lately, edge and fog computing, the problems mostly remain the same but, of course, renaming leads to the perception that things are "new" and need special attention.]*

Processing in Memory: The Terasys Massively Parallel PIM Array (p. 23) "The PIM prototype in a workstation environment delivers supercomputer performance at a fraction of the cost. The next step is to incorporate PIM chips into Cray-3 memory. … Thus, the economics of purchasing a high-performance computer often dictate giving up peak performance on a small application set (massively parallel SIMD) in favor of more modest improvement over a larger range of applications (general-purpose MIMD)" (p. 24) "Processor-in-memory chip: The PIM integrated circuit, with slightly over one million transistors on 1-μm technology, contains 2K×64 bits of SRAM, 64 custom-designed single-bit processors, plus control and error detection circuitry." (p. 26) "Parallel Prefix Network (PPN). The PPN consists of 15 levels, settable by the programmer. It can be used for nearest-neighbor communication and for linear scan operations, which are useful for accumulating partial results such as sums or other associative operations." *[Editor's note: The architecture explained in this interesting article*

Digital Object Identifier 10.1109/MC.2020.2970818
Date of current version: 9 April 2020


COMPUTING THROUGH TIME
ERGUN AKLEMAN
ARTIFICIAL LIFE BEFORE COMPUTERS: DR. FRANKENSTEIN'S MONSTER
A VISUAL HOMAGE TO GEORGE BOOTH
WOW! ITS EYES OPENED. IT'S ALIVE! IT'S ALIVE!
ARTIFICIAL LIFE AFTER DIGITAL: SOMERSAULTING ROBOTS
UGGH! THIS PERFORMANCE WON'T GET A GOLD MEDAL IN GYMNASTICS.

MARY SHELLEY WROTE FRANKENSTEIN IN 1815 DURING A FRIENDLY COMPETITION WITH LORD BYRON AND PERCY SHELLEY. THE BOOK MIGHT HAVE BEEN INFLUENCED BY MUSCLE-TWITCHING EXPERIMENTS USING ELECTRICTY AROUND THAT TIME. IT IS ALSO INTERESTING TO NOTE THAT LORD BYRON'S DAUGHTER ADA LOVELACE, WHO IS CONSIDERED ONE OF THE FIRST COMPUTER PROGRAMMERS, WAS ALSO BORN IN 1815.

Digital Object Identifier 10.1109/MC.2020.2970820
Date of current version: 9 April 2020

describes and implements many of the issues that also had to be solved again in today's massive computer farms.]

Tuning Memory Performance of Sequential and Parallel Programs (p. 32) "When programmers know where and why memory bottlenecks occur, they can make the appropriate program transformations to enhance performance, with the help of this tool's detailed statistics. … MemSpy uses cache simulations to gather detailed memory statistics. Since efficiency is a key concern in simulation-based performance monitoring, we have evaluated two performance optimizations—hit bypassing and reference-trace sampling—that reduce the execution time overhead required to gather such information. Together, these techniques reduce simulation time by nearly an order of magnitude. For a simple memory simulator and sequential applications, the time to run a program with MemSpy is 3 to 10 times as long as the time to run the program normally. For parallel applications, the overhead increases to factors of 8 to 25. Our experience in using MemSpy to tune several sequential and parallel applications demonstrates that it effectively profiles memory performance at speeds that make it an attractive alternative to other approaches." (p. 34) "For case study purposes, we ran MemSpy with a program called MatMul, which performs a blocked-matrix multiply. Although MatMul is a simple application, it is also a case where the common intuitions about the code's behavior are incorrect; namely, choosing block sizes to fit into the cache is not necessarily sufficient for good performance." *[Editor's note: The article examines a tool that allows programmers to view and analyze program behavior, both through data- and code-oriented statistics. It helps to better understand the causes of cache misses and supports the programmer in his or her tuning task.]*

Predicting Client/Server Availability (p. 41) "The classic availability models that considered only hardware are no longer relevant, since software, operations, and environmental failures are all at least as important as hardware failures. Scheduled maintenance periods are being eliminated, as customers demand continuous application availability. … In this article, I present a methodology and data to help design engineers evaluate client/server availability. I evaluate client/server outage data, indicate the most important outage causes in a client/server environment, and discuss methods for improving client/server availability." (p. 42) "Surprisingly, we were able to find very few companies that kept good client/server outage data. Of those we approached, most did not keep outage data of any type. Only a few had data on network outages or on server outages, and none kept data on single-client outages." (p. 43) "We used the client/server outage data to determine the number of outage minutes that should be ascribed to each outage cause. We used the typical client/server environment to determine the number of users affected by an outage." *[Editor's note: The article investigates the downtime caused by each outage and the number of systems and users affected in terms of detail outage causes. Using those data architectural improvements can be prioritized to increase system availability for the users.]*

Where Is Client/Server Software Headed? (p. 49) "Computers moved to desktops; now, the underlying software is itself evolving. The direction computing will take is clear, but the route it will take is a matter for some speculation. … In this article, we'll examine several promising software ideas, especially distributed software and its underlying infrastructure called middleware. Industrial software is being driven by document-centric design, reusable component implementation based on industry standards, and end-user programming. … By 1991 the stage was set for a major new scene in the epic called 'where industrial software is headed.' … Standards in this context are those that enhance, rather than retard, the creation of a distributed computing architecture that will evolve through the following stages: server-centric, client-centric, peer–peer, and fully distributed peer-peer collaborative computing." (p. 55) "The core technology to do this is middleware; hence, the clamor for supremacy deepens among vendors of middleware products. Along the way, these competitors will advocate a flurry of standards and transform object technology from a language concept into a system concept. The main benefactors will be the new breed of so-called end-user programmers." *[Editor's note: Interestingly enough, despite its claim for end-user orientation, the detailed article does not mention important client-server application systems such as SAP or PeopleSoft of that time, which still maintain aspects of client-server architectures instead of fully distributed systems, as predicted by this article. Of course, other applications have become fully distributed, for example, social networks of all kinds.]*

Ten Commandments of Formal Methods (p. 56) "Formal methods permit more precise specifications and earlier error detection. Software developers who want to benefit from formal methods would be wise to heed these ten guidelines. … Unfortunately, although projects based on formal methods are proliferating, the use of these methods is still more the exception than the rule, which results from many misconceptions regarding their costs, difficulties, and payoffs. … The subjective question 'What makes a formal methods project successful?' cannot be definitively answered. However, through observations of many recently completed and in-progress projects—successful and otherwise—we've come up with ten 'commandments' that, if adhered to, will greatly increase a project's chances for success." *[Editor's note: This interesting article, despite being 25 years old, holds many observations, such as the ten commandments, which will help in all formal, semiformal, or even informal approaches in today's software system development. Many references to formal techniques and applications can be found at https://formalmethods.wikia.org/wiki/Formal_methods and https://en.wikipedia.org/wiki/Formal_methods.]*

Multimedia networks: Issues and challenges (p. 68) "Video on demand, teleconferencing, digital libraries, and remote tutoring systems are just a few of the many distributed multimedia applications that have emerged in recent years. Widespread use of such applications will require networks that can offer transport services to both discrete media (such as text and digital images) and continuous media (such as audio and video). … While the need for QoS in multimedia networks is obvious, there is little agreement on the types of performance guarantees needed by multimedia applications. A network service without any QoS commitments is called *a best-effort service*. Most current data networks, including those that makeup the Internet, offer only best-effort service." (p. 69) "The transition from traditional data networks to QoS networks will introduce a major shift in our communication environment. QoS networks that can reserve network resources to individual multimedia connections will shatter the familiar 'Internet paradigm,' where networks are shared resources that can be accessed by all users at all times. However, the transition to resource reservation in multimedia networks appears to be inevitable." *[Editor's note: Despite this prediction for today's networks—wired and wireless—best effort is still the principle applied. Quality-of-service commitments are offered only in specialized applications; however, we were saved by the tremendous increase now offered by 3G, 4G, and 5G.]*

Determining Software Schedules (p. 73) "But now we can measure these factors with reasonable accuracy and collect empirical data on both 'average' and 'best-in-class' results. We are particularly interested in the wide performance gaps between laggards, average enterprises, and industry leaders, as well as differences among the various software domains. … Although many commercial software cost-estimating tools can predict schedules with fairly good accuracy, as of 1994 only about 15% of US software managers—and even less abroad—were using them. … Not only are both ends of software projects ambiguous and difficult to determine, but the middle can get messy, too. Even with the 'waterfall model' of development, there is always overlap and parallelism between adjacent activities, so that a project's end-to-end schedule is never the same as the duration of those activities. Software requirements are usually only about 75% defined when design starts. Design is often little more than 50% complete when coding starts. Integration and testing can begin when coding is less than 20% complete. And user documentation usually starts when coding is about 50% finished." (p. 74) "But the function-point metric provides a convenient, quick estimator for schedule durations that can be applied early in a software project's development cycle." *[Editor's note: The article claims that a simple metric based on function points will accurately provide the project's duration. However, it does not explain how the "power level" numbers are derived, which are instrumental for this calculation, especially because the latter part of the article clearly distinguishes between the four chosen application fields and their requirements.]*

Open Channel: Orbiting the Metaworld (p. 96) "In a land (not so) far away, there lived a clan of artificially intelligent beings. Loosely based on a quantum mix of carbon and silicon, these intrepid beings comprised a virtual reality. They spoke in pseudocode, shared simulated emotion, and possessed a remarkable metaknowledge. It was a perfectly homogeneous society (excluding the exceptions), which was governed by an absolutely constant set of standards that constantly evolved. … A fuzzy subset of the populace conceived the possibility of simulating actual reality! Fuzzy fear spread throughout virtually the entire colony. … The band of outcasts prepared to leave their metaworld. … No more would they be virtual, pseudo, or quasi; no longer were they children of the metaworld. A great synthesis had taken place—actual reality had been achieved. The fuzzy colony of artificial intelligence had become well-defined, natural stupidity." *[Editor's note: The article reflects well the ups and downs of artificial intelligence, from hype to disappointment and back.]*

Digital Object Identifier 10.1109/MC.2020.2981747