Final Remarks

The book concerns main principles and features of distributed systems, composed of computers internally controlled, that is, the so-called von Neuman's architectures (von Neumann 1945). Presenting basic idea behind activity of such machine in Chap. 1, details of its physical construction have been omitted. All the nowadays realizations of the instruction execution cycle are electronic, but one may imagine, for instance, mechanical realization, involving cogwheels, levers, etc., with program and data supplied on punched cards as memory—as in analytical machine built by Babbage (1864). The principle is similar, but execution duration of arithmetic and other operations in case of present-day electronic processors is several billions times shorter. Along with physical realizations, the von Neumann's machine is the human concept. However this is not the only principle of activity of devices performing computation operations—arithmetical, logical and control. Inspiration for searching for different principles, called computational paradigms, are natural phenomena. Nowadays, the intensive research is being carried out on mechanisms of biological particles activity, in particular information flow among molecular structures and on making usage of their multitude for mass of simple operations, executed in parallel. Some computational architectures have already been created, that operate on such principle, though not yet carried into widespread, public usage. Such constructs are capable of solving tasks exceeding (owing to time complexity) capability of classical sequential processors, even interconnected into distributed systems. Computations in such experimental architectures, variously named: molecular, biomolecular, biochemical or DNA computing, have so far efficiently applied to solving some combinatorial problems, the so-called computationally hard, like finding Hamiltonian cycle in a graph (closed path crossing every vertex exactly ones), verification of satisfiability of propositional formuli of very many variables, modelling of the so-called self-organizing systems (e.g. evolution of organisms, weather phenomena, etc.) and many more problems of high complexity. Another natural phenomenon inspiring non classic work principle of computational devices comes from quantum mechanics. The information unit is there the so-called quantum bit, *qubit* in short, mathematically represented as a linear combination of two states of polarization of a quantum object, e.g. photon, where the coefficients of the combination are the so-called amplitudes of probability

L. Czaja, *Introduction to Distributed Computer Systems*, Lecture Notes in Networks and Systems 27, https://doi.org/10.1007/978-3-319-72023-4

of a certain state occurrence. Thus, the qubit is, in a sense, a superposition (or composition) of two binary states, whose sequence may be simultaneously in many states, due to the so-called quantum entanglement. This phenomenon has been used for simultaneous execution of mass computations. Each computation is a sequence of quantum states and a state is a sequence of n qubits, where n is their number which the so-called quantum computer can contain. The computer may be **simultaneously** in 2^n states, whereas its classic, sequential counterpart—in one only from among 2^n states **at a time**. The quantum architecture (Deutsch 1985), still in experimental phase, alike the molecular one, is capable of overcoming the complexity barriers of classic processors, also in distributed systems. Typical problems of high complexity come from number theory, especially their application in cryptography. Both aforesaid non classic paradigms of computation inspired by natural phenomena, along with usage of natural objects—biochemical and quantum—in their physical realization (hardware), are expected to be applied in the massively parallel computing. The principles belong to a research domain called natural computation. This domain encompasses also imitation of phenomena and processes encountered in the nature as problem solving techniques, but by creating algorithmic models executed in the classic computer system, or in grids of many simple and cheap sequential processors. Examples are neural computation, cellular automata, swarm intelligence (imitating behaviour of a "swarm" of objects striving to a joint objective), evolutionary computation (imitating Darwinian evolution), membrane computing (imitating of cell membrane functions), cf. Rozenberg et al. (2012). These and other models of computation—a result of observation of natural phenomena—are being devised for special tasks, but there are also research attempts to create the Universal Quantum Turing Machine (Turing 1937; Deutsch 1985; Penrose 1989), a system for all the effective computations, like a classic computer with arbitrarily large memory needed for a given class of problems. One may also mention other systems, where parallelization and distribution of great number of cooperating elements takes place, like the cyber-physical systems (Lamnabhi-Lagarrigue et al. 2014) (for control of technical devices) or multi agent systems (Russell and Norvig 2010). All them are dynamically developing scientific and engineering explorations. This book was concerned with distributed systems composed of computers of the classic von Neuman's architecture, leaving to the interested reader a closer familiarity with endeavours leading to practical application of the above mentioned contemporary computational principles.

References

Babbage, C. (1864). *The life of a philosopher*. London.

Deutsch, D. (1985). Quantum theory, the church-turing principle and the universal quantum computer. *Proceedings of the Royal Society of London, A400*, 97–117.

Lamnabhi-Lagarrigue, F., Di Benedetto, M. D., & Schoitsch, E. (2014). Introduction to the special theme cyber-physical systems. *Ercim News, 94*, 6–7.

Penrose, R. (1989). *The emperor's new mind*. UK: Oxford University Press.

Rozenberg, G., Baeck, T., & Kok, J. (2012). *Handbook of natural computing*. Heidelberg.

Russell, S., & Norvig, P. (2010). *Artificial intelligence. A modern approach*. Englewood Cliffs, NJ: Prentice Hall.

Turing, A. M. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society Series 2, 42*, 230–265.

von Neumann, J. (1945). *First draft of a report on the EDVAC*.

Bibliography

Akl, S. G. (1997). *Parallel computation: Models and methods*. USA: Prentice Hall.

Bedrouni, A., Mittu, R., Abdeslem Boukhtouta, A., & Berger, J. (2009). *Distributed intelligent systems. A coordination perspective*. Berlin: Springer.

Belapurkar, A., Chakrabarti, A., Ponnapalli, H., Varadarajan, N., Padmanabhuni, S., & Sundarrajan, S. (2009). *Distributed systems security. Issues, processes and solutions*. USA: Wiley.

Ben-Ari, M. (1996). *Podstawy programowania współbieżnego i rozproszonego*. Warszawie: Wydawnictwa Naukowo-Techniczne 1996 (Polish translation of 1990).

Boykin, J., Kirschen, D., Langerman, A., & Loverso, S. (1993). *Programming under Mach*. Reading, MA: Addison-Wesley.

Broy, M., & Stølen, K. (2001). *Abracadabra protocol*. In: *Specification and development of interactive systems. Monograps in Computer Science*. NY: Springer.

Comer, D. E. (1995). *Internetworking with TCP/IP, Volume 1: Principles, Protocols, and Architecture*. USA: Prentice Hall.

Coulouris, G., Dollimore, J., & Kindberg, T. (1998). *Systemy rozproszone, podstawy i projektowanie*. Warszawa: WNT (Polish translation of 1994).

Czaja, L. (1973). Niektóre aspekty implementacji ALGOL-u 60 dla maszyn ZAM/21 ALFA. In *Materiały Sympozjum XV-rocznicy Instytutu Maszyn Matematycznych i Roku Nauki Polskiej*, Warszawa (Some aspects of implementation of ALGOL 60, in Polish).

Czaja, L., & Szorc, P. (1967). Implementation of ALGOL for ZAM computers. *Algorytmy, 4*(7), 91–111.

Dahl, O.-J., Dijkstra, E. W., & Hoare, C. A. R. (1972). *Structured programming*. London and New York: Academic Press.

Dasgupta, P., LeBlanc, R. J., Jr., Ahamad, M., & Ramachandran, U. (1991). The clouds distributed operating systems. *IEEE Computer, 24*(11), 34–44.

Davies, D. W., Holler, E., Jensen, E. D., Kimbleton, S. R., Lampson, B. W., LeLann, G., et al. (1983). In B. W. Lampson, M. Paul, & H. J. Siegert (Eds.), *Distributed systems—Architecture and implementation, an advanced course*. Berlin: Springer.

Gabassi, M., & Dupouy, B. (1995). *Przetwarzanie rozproszone w systemie UNIX*. Warszawa: Lupus (Distributed processing in UNIX, in Polish).

Gien, M. (1995). Evolution of the CHORUS open microkernel architecture: The STREAM project. In *FTDCS '95 Proceedings of the 5th IEEE Workshop on Future Trends of Distributed Computing Systems*. USA: IEEE Computer Society.

Gościński, A. (1991). *Distributed operating systems, the logical design*. USA: Addison Wesley.

Haddad, S., Kordon, F., Pautet, L., & Petrucci, L. (2011). *Distributed systems, design and algorithms*. USA: Wiley.

Holenderski, L., & Szałas, A. (1988). Propositional description of finite cause-effect structures. *Information Processing Letters, 27,* 111–117.

L. Czaja, *Introduction to Distributed Computer Systems*, Lecture Notes in Networks and Systems 27, https://doi.org/10.1007/978-3-319-72023-4

Kshemkalyani, A. D., & Singhal, M. (2011). *Distributed computing: Principles, algorithms, and systems*. Cambridge: Cambridge University Press.

Milner, R. (1989). *Communication and concurrency*. In C. A. R. Hoare (Series Ed.). USA: Prentice-Hall.

Shapiro, E. Y. (1987). *Concurrent PROLOG: Collected papers*. Cambridge MA, USA: MIT Press.

Sinha, P. K. (1997). *Distributed operating systems—Concepts and design*. Piscataway: IEEE Press.

Sorin, D. J., Hill, M. D., & Wood, D. A. (2011). A premier on memory consistency and cache coherence. In M. D. Hill (Ed.), *Synthesis lectures on computer architecture*. USA: Morgan and Calypool Publishers.

Spector, A., & Gifford, D. (1984). The space shuttle primary computer system. *Communications of the ACM 27*(i), 874–900.

Sportack, M. (1998). *Networking essentials unleashed* (1st ed.). USA: Sams Publishing.

Starke, P. (1987). *Sieci petri.* Warsaw: PWN (Polish translation of the author's book in German: *Petri Netze*, 1980).

Stevens, W. R. (1999). *UNIX Programowanie usług sieciowych*. Warszawa: WNT.

SYSTEM AUTOMATYCZNEGO KODOWANIA SAKO. (1961). *Prace Zakładu Aparatów Matematycznych PAN*. Warszawa: PAN (SAKO—A system of automatic coding, in Polish).

Tanenbaum, A. S., Van Renesse, R., Van Staveren, H., Sharp, G. J., Mullender, S. J., Jansen, J., et al. (1990). Experience with the amoeba distributed operating system. *Communication of the ACM, 33*(12), 46–63.

Tanenbaum, A. S. (1997). *Rozproszone systemy operacyjne*. Warsaw: PWN (Polish translation of 1995).

Tanenbaum, A. S. (2004). *Sieci komputerowe*. Gliwice: Wydawnictwo Helion (Polish translation of 2003).

Tanenbaum, A. S., & Wetherall, D. J. (2011). *Computer networks*. USA: Prentice Hall.

Tanenbaum, A. S., & van Steen, M. (2002). *Distributed systems: Principles and paradigms* (2nd ed.) UK: Pearson.

Tanenbaum, A. S., & van Steen, M. (2003). *Computer networks*. USA: Prentice Hall.

Tanenbaum, A. S., & van Steen, M. (2006). *Systemy rozproszone, zasady i paradygmaty*. Warszawa: WNT (Polish translation of 2002).

Wierzbicki, A. (2010). *Trust and fairness in open, distributed systems*. Berlin: Springer.

Wirth, N. (1988). *Programming in modula-2* (4th ed.), *Texts and Monographs in Computer Science*. Berlin: Springer.

Wulf, W. A., & Bell, C. G. (1972). *C.mmp—A multi-mini processor*. In *Proceedings of AFIPS, FJCC* (Vol. 41, pp. 765–777). Montvale, N.J.: AFIPS-Press.

Index

L. Czaja, *Introduction to Distributed Computer Systems*, Lecture Notes in Networks and Systems 27, https://doi.org/10.1007/978-3-319-72023-4