Lecture Notes in Networks and Systems

Volume 27

Series editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland
e-mail: kacprzyk@ibspan.waw.pl

The series "Lecture Notes in Networks and Systems" publishes the latest developments in Networks and Systems—quickly, informally and with high quality. Original research reported in proceedings and post-proceedings represents the core of LNNS.

Volumes published in LNNS embrace all aspects and subfields of, as well as new challenges in, Networks and Systems.

The series contains proceedings and edited volumes in systems and networks, spanning the areas of Cyber-Physical Systems, Autonomous Systems, Sensor Networks, Control Systems, Energy Systems, Automotive Systems, Biological Systems, Vehicular Networking and Connected Vehicles, Aerospace Systems, Automation, Manufacturing, Smart Grids, Nonlinear Systems, Power Systems, Robotics, Social Systems, Economic Systems and other. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution and exposure which enable both a wide and rapid dissemination of research output.

The series covers the theory, applications, and perspectives on the state of the art and future developments relevant to systems and networks, decision making, control, complex processes and related areas, as embedded in the fields of interdisciplinary and applied sciences, engineering, computer science, physics, economics, social, and life sciences, as well as the paradigms and methodologies behind them.

Advisory Board

More information about this series at http://www.springer.com/series/15179

Ludwik Czaja

Introduction to Distributed Computer Systems

Principles and Features

Springer

Ludwik Czaja
Vistula University
Warsaw
Poland

and

Institute of Informatics
University of Warsaw
Warsaw
Poland

ISSN 2367-3370 ISSN 2367-3389 (electronic)
Lecture Notes in Networks and Systems
ISBN 978-3-319-72022-7 ISBN 978-3-319-72023-4 (eBook)
https://doi.org/10.1007/978-3-319-72023-4

Library of Congress Control Number: 2017959902

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To Nina and Mateusz

Acknowledgement

I am indebted to my colleagues for encouragement to collect notes of my 12 lectures into a book form, but my special thanks go to Prof. Andrzej Skowron and Prof. Janusz Kacprzyk for initiation and promoting idea of publication of the book in Springer-Verlag. Also, Mr. Viju Falgon, a project manager at Scientific Publishing Services of Springer who handled the editorial production of the book, deserves special thanks, being bothered with numerous corrections of my errors and misprints I noticed during the production process.

Ludwik Czaja

Contents

About the Author

Ludwik Czaja obtained his M.Sc., Ph.D., and habilitation degrees from the University of Warsaw, where he was a professor of informatics at the Faculty of Mathematics, Informatics, and Mechanics. He spent some years in other universities, like Carnegie-Mellon, Oxford, Ben-Gurion, Humboldt, as visiting professor or a research fellow. Now he is a full professor of the Vistula University and professor emeritus of the University of Warsaw. His work encompasses formal and programming languages, compilers, theory of computation, parallel, and distributed processing.

Introduction

In the most general meaning, a distributed computer system is identified with computer network. There are two problems with this identification. One concerns the word "computer". Even the origins of such systems encompassed not only computers but also devices far from being computers, like missile launchers and other military objects (a brief historical outline is in Chap. 2). Let alone today's networks that connect things of professional and everyday usage: one can hardly say that the so-called intelligent refrigerator is a computer. The other problem concerns the word "distributed". The main feature of such architectures is lack of shared physical memory, where processors intercommunicate by message passing through data links (channels) of arbitrary length and bandwidth. So, in this meaning, a motherboard of transputers (Sect. 5.3 in Chap. 5), processors interacting by data packet exchange inside one machine, all are distributed systems too. But on the other hand, a computer surrounded by simple terminals for data input and output, multi-access computer or workstations with direct access to memory of a mainframe, are systems categorized as not distributed in the aforesaid meaning, though distributed in the common parlance, because separated spatially. Nonetheless, the distinguishing characteristic of distributed computer systems as the research and engineering domain, is inter-computer communication by message passing based on networks. However, the design, technical solutions, and application of distributed systems and general ("generic") networks, are not identical. The conceptual difference lies in their destination. A distributed system is being often constructed as a computation environment for specific class of applications, for a company of certain activity profile, for a corporation or education center, whereas a network is a universal tool for data transmission not limited to particular applications. Thus, the constructional difference consists in software: in (distributed) operating system and programs on top of it—compilers of programming languages and diverse applications. But dividing functions of a system into hardware and software is not constitutive from the information processing point of view. For distributed systems, hardware and software of a network constitute a communication infrastructure. So, presentation of their principles takes into account some issues of network technology, because design and implementation of

distributed systems exploits solutions elaborated there. The principles concern, for instance, physical and logical time, coordination of processes (synchronization in particular), communication, exception handling, fault tolerance, and security of system's behavior. More specific are distributed transactions, scheduling joint actions of computers, remote procedure call and simulation of distributed shared memory, giving the user impression of having direct access to huge memory space. Presentation of these issues, limited to principles, in order not to overshadow them with technical details, is contained in successive chapters. The details change as the research and technology develops, the principles remain longer. Therefore, great many dedicated distributed systems and several of general usage, are not discussed here. Their presentation would exceed capacity of any book, a few of such projects are mentioned only in Chap. 2. Emphasis in this book is on objectives of distributed systems. The objectives determine properties. As the most important is regarded *transparency*, that is, hiding the joint usage of resources by multiple users—making every user imagine to be exclusive one. He/she is to be unaware how and where the resources and mechanisms of their usage are located: in hardware, operating system kernels, application libraries, compilers, or runtime systems of programming languages. From the principles point of view, this is immaterial. In existent distributed systems, the transparency has been accomplished only to a certain degree, as the problem of balance of user's convenience against system's efficiency becomes of prime importance. However, along with the development of communication infrastructure and increase of large data sets' transmission rate, attainment of full transparency seems realistic. To keep up with the progress in materializing such ideas, a look through professional writings is indispensable. Their short selection, quoted in the following chapters, is in the Bibliography.

A presentation of the wide subject matter in a small book requires making decision regarding its content. Apart from some issues belonging to the mainstream of network technology and distributed systems outlined here, such topics like construction of distributed operating systems (including resource naming and mapping names onto addresses) and security of communication (cryptography in particular), are beyond this book. They are broad and individual research and technology branches, enjoying rich and easily available literature. Chapter 9, however, contains a fairly well-known pattern-protocol "in action" (expressed by a cause/effect structure diagram; the calculus of cause/effect structures is outlined in Chap. 10) as a sequence of states which the protocol passes during execution. This is the so-called *alternating bit protocol*, which reveals basic features of a certain category of protocols (its enhanced versions like "*sliding window*", "*abracadabra*", became a basis for some existent protocols design).

The book is inspired by a 30-h lecture course on distributed computer systems, given on the elementary level by the author. It is a handbook, although containing some monographic elements not published elsewhere, like a protocol described in Sect. 4.4, formal proofs of some important facts and a method of presentation of algorithms in action: as sequences of states. Experience shows that students, especially not professional programmers, easier assimilate essence of algorithmic constructs presented in action rather than in the form of static descriptions as

programs. Hence, the method of presenting: a short explanation of a construct is followed by a sequence of states it passes. If these states are regarded as animated film frames, then projection of the film illustrates process of execution of the construct. In accordance with such metaphor, description of a construct (a program) is a screenplay, while the process—realization of this screenplay on the film tape. Furthermore, using computer for the animated show of the process gives rise to see it as a live scenery in a slow motion.

Understanding of the following chapters does not require advanced knowledge in informatics. Sufficient will be familiarity with topics provided in an introductory course to computer science accompanied by laboratory exercises, and to algorithms and data structures. The participants acquire there a certain skill in design and analysis of algorithms, to write them as programs and run on the computer. Beneficial will also be some knowledge in foundations of operating systems and computer networks, where pretty much time is devoted to standard communication protocols, being omitted here as technical realization of principles. Of great importance is experience in a laboratory teamwork—a practical contact with a kind of distributed system. However, a certain problem may be noticed. Despite having acquired these foundations, the students of computer science or engineering, who begin programming in a high-level language, often exhibit ignorance in knowledge of structure and activity of a processor on the level of internal machine language, what is the instruction execution cycle in particular. Yet, many functions of distributed system are found in ordinary, stand-alone machine, especially with multiprogram operating system. For instance, communication mechanisms in networks play a part similar to input/output channels (serviced by protocols too), management of concurrent transactions requires actions similar to synchronizing processes in a sequential multiprogrammed machine, etc. That is why, the first chapter contains an outline of sequential machines activity with internal control, where many mechanisms have counterparts, though significantly more sophisticated, in network systems. The activity of a machine model (also multiprogrammed) with a short set of typical instructions, as well as a collection of such machines interconnected by communication channels, has been demonstrated by means of abovementioned "film method". This visual demonstration should clearly explain principle of sequential processor's activity. In Chap. 1, the reader will also find example of a synchronous vector machine "in action". The presented material is, as usually based on the principle of von Neumann's stored program sequential machines. A few sentences about other principles of information processing, though not yet materialized in architectures for the public use, are in the final remarks of the book.

The question of mathematical description of phenomena appearing in distributed systems remains. Excessive formalization has been avoided, yet clear presentation of some issues demanded a formal notation, especially in a few proofs of properties. To such issues belong, for instance, time (physical and logical), correctness of some algorithms of distributed mutual exclusion, chances of system failure, unattainable simultaneity of certain actions, definition of distributed memory consistency models and its consequences. Therefore, to express some concepts, facts and proofs, mathematical language has been used, though in a very modest extent: operations

on sets and relations, partial and linear order, functions, sequences and series, probability, elementary propositional and predicate calculus. Some of them are outlined in Chap. 10. Such topics are familiar to everybody who attended introductory course at mathematical or engineering studies.