Concurrent And Distributed Computing In Java

On Concurrent Programming
Concurrent and Distributed Computing in Java
Distributed Systems
Elements of Distributed Computing
Concurrency
The SR Programming Language
Concurrent Systems
Learning Concurrent Programming in Scala
Designing Data-Intensive Applications
Nonsequential and Distributed Programming with Go
Parallel and Concurrent Programming in Haskell
Parallel and Distributed Computation: Numerical Methods
Concurrent Programming: Algorithms, Principles, and Foundations
Principles of Concurrent and Distributed Programming
Creating Components
Designing Concurrent, Distributed, and Real-time Applications with UML
Distributed Algorithms
Concurrent Programming
Programming Distributed Computing Systems
The Origin of Concurrent Programming
Concurrent, Parallel and Distributed Computing
The Art of Concurrency
Concurrent Data Processing in Elixir
Distributed Computing with Python
Foundations of Multithreaded, Parallel, and Distributed Programming
Parallel and Distributed Programming Using C++
Distributed Systems
Parallel and Distributed Simulation Systems
Synchronization Algorithms and Concurrent Programming
Distributed Services with Go
Introduction to Reliable Distributed Programming
Introduction to Reliable and Secure Distributed Programming
Actors
Atomic Transactions: In Concurrent and Distributed Systems
Distributed Computing with Go
Topics in Parallel and Distributed Computing
Concurrent Systems
Creating Components
Operating Systems

TORRES HOGAN

On Concurrent Programming Addison Wesley Publishing Company
This book is a must-have tutorial for software developers aiming to write concurrent programs in Scala, or broaden their existing knowledge of concurrency. This book is intended for Scala programmers that have no prior knowledge about concurrent programming, as well as those seeking to broaden their existing knowledge about concurrency. Basic knowledge of the Scala programming language will be helpful. Readers with a solid knowledge in another programming language, such as Java, should find this book easily accessible.
Concurrent and Distributed Computing in Java CRC Press
This text takes complicated and almost unapproachable parallel programming techniques and presents them in a simple, understandable manner. It covers the fundamentals of programming for distributed environments like Internets and Intranets as well as the topic of Web Based Agents.
Distributed Systems Springer Science & Business Media
Mathematics of Computing -- Parallelism.
Packt Publishing Ltd
A text intended as a modern replacement for a first course in operating systems modern in the sense that concurrency is a central focus throughout; distributed systems are treated as the norm rather than single-processor systems, and effective links are provided to other systems courses. It is also
Elements of Distributed Computing MIT Press
Learn different ways of writing concurrent code in Elixir and increase your application's performance, without sacrificing scalability or fault-tolerance. Most projects benefit from running background tasks and processing data concurrently, but the world of OTP and various libraries can be challenging. Which Supervisor and what strategy to use? What about GenServer? Maybe you need back-pressure, but is GenStage, Flow, or Broadway a better choice? You will learn everything you need to know to answer these questions, start building highly concurrent applications in no time, and write code that's not only fast, but also resilient to errors and easy to scale. Whether you are building a high-frequency stock trading application or a consumer web app, you need to know how to leverage concurrency to build applications

that are fast and efficient. Elixir and the OTP offer a range of powerful tools, and this guide will show you how to choose the best tool for each job, and use it effectively to quickly start building highly concurrent applications. Learn about Tasks, supervision trees, and the different types of Supervisors available to you. Understand why processes and process linking are the building blocks of concurrency in Elixir. Get comfortable with the OTP and use the GenServer behaviour to maintain process state for long-running jobs. Easily scale the number of running processes using the Registry. Handle large volumes of data and traffic spikes with GenStage, using back-pressure to your advantage. Create your first multi-stage data processing pipeline using producer, consumer, and producer-consumer stages. Process large collections with Flow, using MapReduce and more in parallel. Thanks to Broadway, you will see how easy it is to integrate with popular message broker systems, or even existing GenStage producers. Start building the high-performance and fault-tolerant applications Elixir is famous for today. What You Need: You'll need Elixir 1.9+ and Erlang/OTP 22+ installed on a Mac OS X, Linux, or Windows machine.
Concurrency Maarten Van Steen
Principles of Concurrent and Distributed Programming provides an introduction to concurrent programming focusing on general principles and not on specific systems. Software today is inherently concurrent or distributed - from event-based GUI designs to operating and real-time systems to Internet applications. This edition is an introduction to concurrency and examines the growing importance of concurrency constructs embedded in programming languages and of formal methods such as model checking.
The SR Programming Language Springer Science & Business Media
Concurrent and Distributed Computing in Java addresses fundamental concepts in concurrent computing with Java examples. The book consists of two parts. The first part deals with techniques for programming in shared-memory based systems. The book covers concepts in Java such as threads, synchronized methods, waits, and notify to expose students to basic concepts for multi-threaded programming. It also includes algorithms for mutual exclusion, consensus, atomic objects, and wait-free data structures. The second part of the book deals with programming in a message-passing system. This part covers resource allocation problems, logical clocks, global property detection, leader election, message ordering, agreement algorithms, checkpointing, and message logging. Primarily a textbook for upper-level undergraduates and graduate students, this thorough treatment will also be of interest to professional programmers.
Concurrent Systems Morgan & Claypool
Harness the power of multiple computers using Python through this fast-paced informative guide About This Book You'll learn to write data processing programs in Python that are highly available, reliable, and fault tolerant Make use of Amazon Web Services along with Python to establish a powerful remote computation system Train Python to handle data-intensive and resource hungry applications Who This Book Is For This book is for Python developers who have developed Python programs for data processing and now want to learn how to write fast, efficient programs that perform CPU-intensive data processing tasks. What You Will Learn Get an introduction to parallel and distributed computing See synchronous and asynchronous programming Explore parallelism in Python Distributed application with Celery Python in the Cloud Python on an HPC cluster Test and debug distributed applications In Detail CPU-intensive data processing tasks have become crucial considering the complexity of the various big data applications that are used today. Reducing the CPU utilization per process is very important to improve the overall speed of applications. This book will teach you how to perform parallel execution of computations by distributing them across multiple processors in a single machine, thus improving the overall performance of a big data processing task. We will cover synchronous and asynchronous models, shared memory and file systems, communication between various processes, synchronization, and more. Style and Approach This example based, step-by-step guide will show you how to make the best of your hardware configuration using Python for distributing applications.
Learning Concurrent Programming in Scala Athena Scientific
A tutorial leading the aspiring Go developer to full mastery of Golang's distributed features. Key Features This book provides enough concurrency theory to give you a contextual understanding of Go concurrency It gives weight to synchronous and asynchronous data streams in Golang web applications It makes Goroutines and Channels completely familiar and natural to Go developers Book Description Distributed Computing with Go gives developers with a good idea how basic Go development works the tools to fulfill the true potential of Golang development in a world of concurrent web and cloud applications. Nikhil starts out by setting up a professional Go development environment. Then you'll learn the basic concepts and practices of Golang concurrent and parallel development. You'll find out in the new few chapters how to balance resources and data with REST and standard web approaches while keeping concurrency in mind. Most Go applications these days will run in a data center or on the cloud, which is a condition upon which the next chapter depends. There, you'll expand your skills considerably by writing a distributed document indexing system during the next two chapters. This system has to balance a large corpus of documents with considerable analytical demands. Another use case is the way in which a web application written in Go can be consciously redesigned to take distributed features into account. The chapter is rather interesting for Go developers who have to migrate existing Go applications to computationally and memory-intensive environments. The final chapter relates to the rather onerous task of testing parallel and distributed applications, something that is not usually taught in standard computer science curricula. What you will learn Gain proficiency with concurrency and parallelism in Go Learn how to test your application using Go's standard library Learn industry best practices with technologies such as REST, OpenAPI, Docker, and so on Design and build a distributed search engine Learn strategies on how to design a system for web scale Who this book is for This book is for developers who are familiar with the Golang syntax and have a good idea of how basic Go development works. It would be advantageous if you have been through a web application product cycle, although it's not necessary.
Designing Data-Intensive Applications MIT Press
Concurrency is a powerful technique for developing efficient and lightning- fast software. For instance, concurrency can be used in common applications such as online order processing to speed processing and ensure transaction reliability. However, mastering concurrency is one of the greatest challenges for both new and veteran programmers. Softwar
Nonsequential and Distributed Programming with Go Wiley-Interscience
This book is devoted to the most difficult

part of concurrent programming, namely synchronization concepts, techniques and principles when the cooperating entities are asynchronous, communicate through a shared memory, and may experience failures. Synchronization is no longer a set of tricks but, due to research results in recent decades, it relies today on sane scientific foundations as explained in this book. In this book the author explains synchronization and the implementation of concurrent objects, presenting in a uniform and comprehensive way the major theoretical and practical results of the past 30 years. Among the key features of the book are a new look at lock-based synchronization (mutual exclusion, semaphores, monitors, path expressions); an introduction to the atomicity consistency criterion and its properties and a specific chapter on transactional memory; an introduction to mutex-freedom and associated progress conditions such as obstruction-freedom and wait-freedom; a presentation of Lamport's hierarchy of safe, regular and atomic registers and associated wait-free constructions; a description of numerous wait-free constructions of concurrent objects (queues, stacks, weak counters, snapshot objects, renaming objects, etc.); a presentation of the computability power of concurrent objects including the notions of universal construction, consensus number and the associated Herlihy's hierarchy; and a survey of failure detector-based constructions of consensus objects. The book is suitable for advanced undergraduate students and graduate students in computer science or computer engineering, graduate students in mathematics interested in the foundations of process synchronization, and practitioners and engineers who need to produce correct concurrent software. The reader should have a basic knowledge of algorithms and operating systems.

Parallel and Concurrent Programming in Haskell Addison Wesley

Distributed Systems: Concurrency and Consistency explores the gray area of distributed systems and draws a map of weak consistency criteria, identifying several families and demonstrating how these may be implemented into a programming language. Unlike their sequential counterparts, distributed systems are much more difficult to design, and are therefore prone to problems. On a large scale, usability reminiscent of sequential consistency, which would provide the same global view to all users, is very expensive or impossible to achieve. This book investigates the best ways to specify the objects that are still possible to implement in these systems. - Explores the gray area of distributed systems and draws a map of weak consistency criteria - Investigates the best ways to specify the objects that are still possible to implement in these systems - Presents a description of existing memory models and consistency criteria

Parallel and Distributed Computation: Numerical Methods MIT Press

An essential reader containing 19 important papers on the invention and early development of concurrent programming and its relevance to computer science and computer engineering. All of them are written by the pioneers in concurrent programming, including Brinch Hansen himself, and have introductions added that summarize the papers and put them in perspective. The editor provides an overview chapter and neatly places all developments in perspective with chapter introductions and expository apparatus. Essential resource for graduates, professionals, and researchers in CS with an interest in concurrent programming principles. A familiarity with operating system principles is assumed.

Concurrent Programming: Algorithms, Principles, and Foundations "O'Reilly Media, Inc."

Suitable for real-world systems that deal with complex issues such as concurrency and real-time constraints. Providing detailed guidelines, this book is useful for software engineers.

Principles of Concurrent and Distributed Programming Elsevier

The transition from sequential to parallel computation is an area of critical concern in today's computer technology, particularly in architecture, programming languages, systems, and artificial intelligence. This book addresses central issues in concurrency, and by producing both a syntactic definition and a denotational model of Hewitt's actor paradigm—a model of computation specifically aimed at constructing and analyzing distributed large-scale parallel systems—it substantially advances the understanding of parallel computation. Contents Introduction • General Design Decisions • Computation in ACTOR Systems • A More Expressive Language • A Model for ACTOR Systems • Concurrency Issues • Abstraction and Compositionality • Conclusions

Creating Components Pragmatic Bookshelf

Data is at the center of many challenges in system design today. Difficult issues need to be figured out, such as scalability, consistency, reliability, efficiency, and maintainability. In addition, we have an overwhelming variety of tools, including relational databases, NoSQL datastores, stream or batch processors, and message brokers. What are the right choices for your application? How do you make sense of all these buzzwords? In this practical and comprehensive guide, author Martin Kleppmann helps you navigate this diverse landscape by examining the pros and cons of various technologies for processing and storing data. Software keeps changing, but the fundamental principles remain the same. With this book, software engineers and architects will learn how to apply those ideas in practice, and how to make full use of data in modern applications. Peer under the hood of the systems you already use, and learn how to use and operate them more effectively Make informed decisions by identifying the strengths and weaknesses of different tools Navigate the trade-offs around consistency, scalability, fault tolerance, and complexity Understand the distributed systems research upon which modern databases are built Peek behind the scenes of major online services, and learn from their architectures

Designing Concurrent, Distributed, and Real-time Applications with UML Morgan Kaufmann

This is the fourth edition of "Distributed Systems." We have stayed close to the setup of the third edition, including examples of (part of) existing distributed systems close to where general principles are discussed. For example, we have included material on blockchain systems, and discuss their various components throughout the book. We have, again, used special boxed sections for material that can be skipped at first reading. The text has been thoroughly reviewed, revised, and updated. In particular, all the Python code has been updated to Python3, while at the same time the channel package has been almost completely revised and simplified. Additional material, including coding examples, figures, and slides, are available at www.distributed-systems.net.

Distributed Algorithms Springer Science & Business Media

You know the basics of Go and are eager to put your knowledge to work. This book is just what you need to apply Go to real-world situations. You'll build a distributed service that's highly available, resilient, and scalable. Along the way you'll master the techniques, tools, and tricks that skilled Go programmers use every day to build quality applications. Level up your Go skills today. Take your Go skills to the next level by learning how to design, develop, and deploy a distributed service.

Start from the bare essentials of storage handling, then work your way through networking a client and server, and finally to distributing server instances, deployment, and testing. All this will make coding in your day job or side projects easier, faster, and more fun. Lay out your applications and libraries to be modular and easy to maintain. Build networked, secure clients and servers with gRPC. Monitor your applications with metrics, logs, and traces to make them debuggable and reliable. Test and benchmark your applications to ensure they're correct and fast. Build your own distributed services with service discovery and consensus. Write CLIs to configure your applications. Deploy applications to the cloud with Kubernetes and manage them with your own Kubernetes Operator. Dive into writing Go and join the hundreds of thousands who are using it to build software for the real world. What You Need: Go 1.11 and Kubernetes 1.12.

Concurrent Programming Addison-Wesley Professional

The book "Concurrent, Parallel, and Distributed Computing" offers an excellent overview of the various areas of the computing field. There is a lot of overlap between the words "concurrent computing," "parallel computing," and "distributed computing," and there is no obvious differentiation between them. The same system can be described as "parallel" and "distributed"; in a typical distributed system, the processors run concurrently in parallel.The content in the book is presented in such a way that even a reader with no prior knowledge of computers may understand it and become acquainted with the fundamental concepts of computing. It offers numerous small examples, demonstration materials, and sample exercises that teachers can use to teach parallel programming principles to students who have just recently been introduced to basic programming concepts. It focuses on Python multiprocessing features like fork/join threading, message passing, sharing resources between threads, and using locks. Parallelism's utility can be seen in applications like searching, sorting, and simulations. Students and researchers can get an accessible and comprehensive explanation of the concepts, guidelines, and, in particular, the complex instrumentation techniques used in computing.

Programming Distributed Computing Systems Springer Science & Business Media

If you're looking to take full advantage of multi-core processors with concurrent programming, this practical book provides the knowledge and hands-on experience you need. The Art of Concurrency is one of the few resources to focus on implementing algorithms in the shared-memory model of multi-core processors, rather than just theoretical models or distributed-memory architectures. The book provides detailed explanations and usable samples to help you transform algorithms from serial to parallel code, along with advice and analysis for avoiding mistakes that programmers typically make when first attempting these computations. Written by an Intel engineer with over two decades of parallel and concurrent programming experience, this book will help you: Understand parallelism and concurrency Explore differences between programming for shared-memory and distributed-memory Learn guidelines for designing multithreaded applications, including testing and tuning Discover how to make best use of different threading libraries, including Windows threads, POSIX threads, OpenMP, and Intel Threading Building Blocks Explore how to implement concurrent algorithms that involve sorting, searching, graphs, and other practical computations The Art of Concurrency shows you how to keep algorithms scalable to take advantage of new processors with even more cores. For developing parallel code algorithms for concurrent programming, this book is a must.