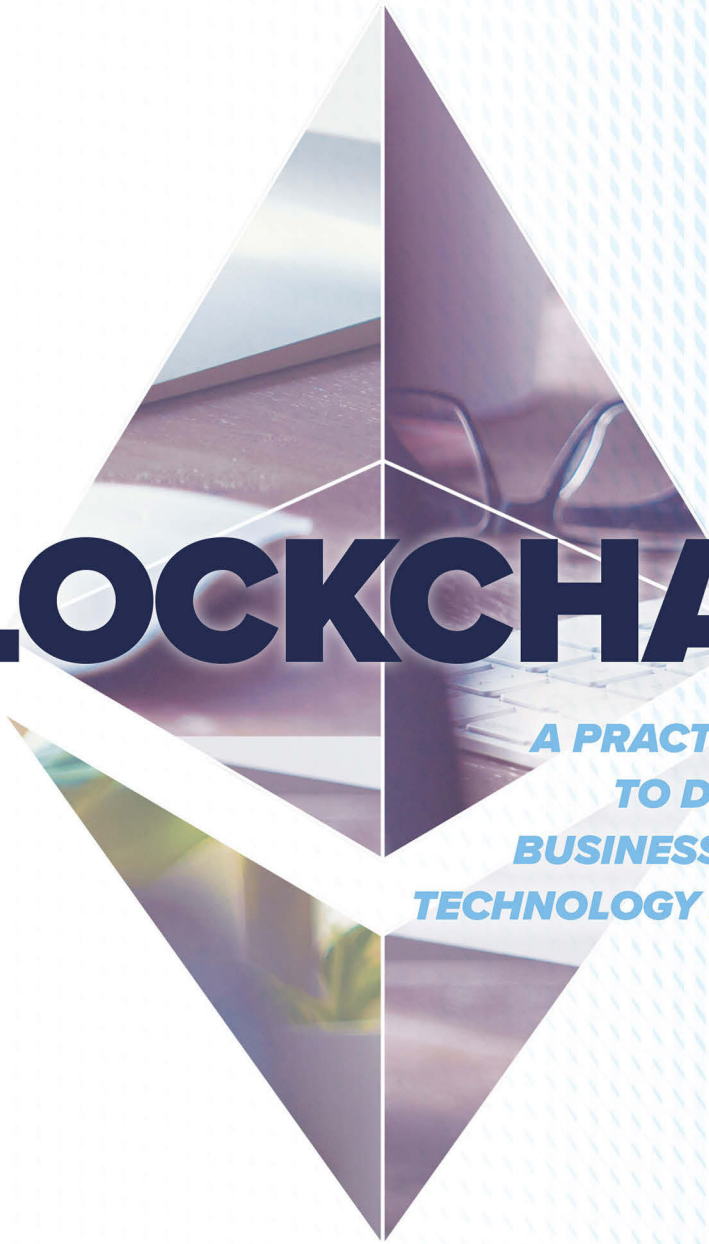

BLOCKCHAIN

A PRACTICAL GUIDE TO DEVELOPING BUSINESS, LAW, AND TECHNOLOGY SOLUTIONS

JOSEPH J. BAMBARA
PAUL R. ALLEN

KEDAR IYER | SOLOMON LEDERER
RENÉ MADSEN | MICHAEL WUEHLER

Mc Graw Hill Education

Blockchain

A Practical Guide to Developing Business, Law, and Technology Solutions

Joseph J. Bambara
Paul R. Allen

New York Chicago San Francisco
Athens London Madrid Mexico City
Milan New Delhi Singapore Sydney Toronto

ISBN: 978-1-26-011586-4
MHID: 1-26-011586-0

The material in this eBook also appears in the print version of this title: ISBN: 978-1-26-011587-1,
MHID: 1-26-011587-9.

eBook conversion by codeMantra
Version 1.0

I would like to dedicate this book to my family (Roseanne, Vanessa, and Michael) and friends (especially Hillary Brower and Rolando Marino) and to the hope that we in America can properly educate our young and lead the world in technology, innovation, and freedom.

—Joseph J. Bambara

I would like to dedicate this book to my children, Sophia and Terence.
My hope is that you will always do what you love, and therefore love what you do.

—Paul R. Allen

About the Authors

Joseph J. Bambara, CIPP/US, is an attorney/technologist. As an attorney with a private practice, he has counseled affiliate marketing, media, and technology firms in software and data licensing agreements, intellectual property, privacy, data security, and cybersecurity as well as analyzing the legal implications of new technologies like blockchain and smart contracts. As technologist/founder of UCNY, Inc., his experience includes 30 years of implementing computing and communications architecture for Wall Street, media and law enforcement including mobile, enterprise, database, cybersecurity, and most recently IoT and blockchain. He has taught courses in computing at CCNY School of Engineering in New York. He is the author of more than 10 internationally published books on software development covering Java, SQL, and related technologies for McGraw-Hill. As lecturer and co-chairman of the New York County Lawyers Association Law and Technology Group, he presents frequently on law and technology. He has a juris doctorate in law and a master's degree in computer science.

Paul R. Allen is a director and product owner at Enterprise Engineering, Inc. Paul has been advising on, architecting, and developing applications systems for over 25 years. During this time, he has performed many strategic assessments of IT organizations, infrastructures, software development processes, and application architectures and helped companies and teams evaluate alternative technologies and products. He has developed systems for the financial, brokerage, pharmaceutical, and manufacturing industries, specializing in web-based, object-oriented technology and is now doing the same in the exciting world of blockchain, IoT, and smart contracts. He has taught numerous courses in computing at Columbia University in New York. He has authored more than a dozen books, including *OCM Java EE 6 Enterprise Architect Exam Guide* (Oracle Press, 2014), *Sun Certified Enterprise Architect for J2EE Study Guide* (McGraw-Hill, 2007), *J2EE Unleashed* (SAMS, 2001), *SQL Server Developer's Guide* (IDG, 2000), *Informix: Universal Data Option* (McGraw-Hill, 1998), and *PowerBuilder: A Guide to Developing Client/Server Applications* (McGraw-Hill, 1995). Paul has also given presentations on computing topics in cities around the globe, including London, Paris, Tokyo, Los Angeles, Vienna, Berlin, New York, Washington, D.C., Copenhagen, Oslo, and Stockholm.

About the Contributors

Kedar Iyer is a software engineer who has worked with satellite systems, autonomous robotics, and blockchain technologies. He was the co-founder of LetsChai, an India-based dating site. His most recent focus has been on blockchain technologies. He is the creator of PeerBet, a peer-to-peer sports betting platform on the Ethereum blockchain. He has a degree in mechanical engineering from UCLA and currently lives in Brooklyn, New York.

Solomon Lederer, PhD, is a founder of blockmatics.tech, a blockchain training and consulting firm, and the founder of Coinspace, a blockchain-focused co-working space. He is also partner and head of technology at Iterative Instinct, a private investment fund focused on crypto-assets. He has a doctorate in distributed and ad hoc sensor networks, where he developed novel ways for networks to self-organize. Before blockchain, he worked as a software engineer in the defense and finance industries. He has been working with/teaching blockchain technology and Ethereum since 2014.

René Madsen is an enterprise solution architect at Progressive A/S, specializing in blockchain development and big data for many enterprise organizations across western Europe. He is also an adjunct lecturer at Copenhagen Business School in Denmark. He has a master's degree in computer science from Copenhagen University and an MBA from Edinburgh Business School, Heriot-Watt University.

Michael Wuehler is a founder of Ethereum and INFURA. He is a blockchain evangelist at ConsenSys, a leading blockchain venture production studio. He leads a global team in building an ecosystem of consumer-centric products and enterprise solutions using blockchain technologies, primarily Ethereum. He is a business and information systems leader with 25 years of experience and a broad-based background that spans technical and business infrastructure, transformation, and operations. He attended the University of Chicago Booth School of Business. He lives in New York City.

About the Technical Editor

Sean T. McKeough is the co-founder of Blockmatics, a leader in the blockchain education space. He regularly meets with leaders from a variety of industries to help them understand this transformative technology and how they might apply it to their business or passion. Sean is a community organizer at heart and is a regular in the New York and Colorado blockchain event scene. You can get in touch with Sean at 21.co/mckeough.

Contents at a Glance

Contents

Acknowledgments

We would like to acknowledge all the incredibly hard-working folks at McGraw-Hill Education, especially Lisa McClain and Claire Yee. We would also like to thank Sean McKeough for his help in editing the technical material in this book and providing valuable and timely response to our work.

—Joseph J. Bambara and Paul R. Allen

A very special thanks to my lady, Hillary Brower, and my co-author, Paul R. Allen, especially for his friendship and for being a great partner no matter what we try. Thanks to my family, who are always there when I need them.

—Joseph J. Bambara
Port Washington, New York

As always, a very special thank you to my family for being a source of strength, support, and ideas. Evelyn, thank you for encouraging me to write "the latest last book." I won't be writing another . . . at least for a couple of years! A very special thank you to my co-author, Joseph J. Bambara, who first introduced me to blockchain and then correctly predicted that we would write about it one day!

—Paul R. Allen
New York, New York

This page intentionally left blank

Introduction

Blockchain technology is robust like the Internet, but unlike the web2 Internet of today, it stores identical blocks of information across its network. For this reason, a blockchain cannot be controlled by any single entity nor does it have a single point of failure. By storing data across its network, the blockchain eliminates the risks that come with data being held centrally. Blockchain networks lack centralized points of vulnerability that computer hackers can exploit easily.

Today's Internet has security problems that are familiar to everyone. We all rely on username and password credentials to access our assets online. Blockchain uses encryption technology to improve security. By allowing data and information to be widely distributed, blockchain technology has created the backbone of the new Internet, web3. Though it was originally devised for the digital currency Bitcoin, the business and technology communities are finding many uses for blockchain. Knowledge of this new technology will be required by not only programmers but by all businesses. In the next five to ten years, blockchain will change the business models in all types of industries—and perhaps change the way people work and live.

We have been involved in computing technology since its first practical use on Wall Street circa 1974. We have used and written about the evolving technology tools starting with IBM Assembler, Fortran, COBOL, and data access methods like QSAM, BDAM, and VSAM, all the way to present-day REST web services, Java, and SQL and everything in between (including client/server tools like PowerBuilder). We have been fortunate in that our passions for learning and becoming proficient in each new and emerging technology have served us well in the business community. We recognized blockchain technology as an ingenious invention, as it combines the best of what came before it in database design, cryptography, and virtual machine containers with the very capable distributed computing environment of today. Our passion for the technology was based on love at first sight. We have brought together a robust network of blockchain entrepreneurs, fellow blockchain technologists, and others who have made critical contributions to the coverage and text of this book.

Target Audience

The target audience for this book includes anyone interested in blockchain technology as well as its use cases. It is also for anyone developing a blockchain application—what is required to build solutions in this space. Additionally, web and application developers of all levels as well as tech-savvy businesspeople and even attorneys who want to stay current with technology in general and blockchain specifically would find the discussions in this book of interest.

What This Book Covers

The book covers blockchain definition, use cases, distributed technology, and especially blockchain development, with a good deal of code snippets and best practices. It targets the Ethereum blockchain, introducing Solidity and other aspects of the Ethereum framework. Additionally, there are two chapters devoted to setup, coding, validation, and deployment of a complete and comprehensive blockchain betting application.

How to Use This Book

Each chapter in the book can stand alone, describing a particular aspect of blockchain technology and its use cases. That said, there is a sequence whereby each chapter builds on the previous chapters to provide a solid conceptual understanding of blockchain. This is coupled with a comprehensive treatment for getting started as a designer and developer of blockchain applications. This includes the Ethereum technology stack and code, and deployment techniques and examples, including an entire application from code start to deployment finish.

How This Book Is Organized

In **Chapter 1**, we provide an overview of everything blockchain. We introduce it as a distributed database technology with the capability to execute smart contracts. The expanding universe of blockchain application is covered. The efficiencies and cost savings provided by blockchain technologies—especially the private blockchains adopted by the financial community—are also briefly examined. In parallel, the use of blockchain is shown to affect global transactions, and this will push it forward toward maturity. Blockchain and its timing are critical to maintaining global transactions and providing trust in the integrity of those transactions. For these reasons and more, we argue that blockchain is here to stay, and the chapters herein will provide you with a comprehensive map and toolset with what you need as a designer or developer to successfully make the trip.

While blockchain technology is at the heart of cryptocurrencies like Bitcoin and Ethereum, it clearly is a technology with widespread applicability in many sectors of business. **Chapter 2** gives some depth to real-world use cases in the financial services industry, including sections on cryptocurrency and digital tokens. We then show how common business use cases ultimately lead to faster throughput, reduced costs, improved accuracy, greater transparency, and quality, reliability, simplicity, and traceability.

The chapter also discusses smart property and smart contracts and how they can be used in conjunction in the not-too-distant future of blockchain technology. Chapter 2 closes with how blockchain and the Internet of Things (IoT) will have a consortium of startups and companies to help define and refine security and interoperability, management, and coordination between connected devices. While IoT is still in its early stages of evolution and currently comprises mostly technologies that either collect data or allow remote monitoring and control, this will change as devices become smarter and artificial intelligence is added. This network of things will transition toward becoming a network of autonomous devices that talk to each other and (hopefully) make smart decisions without the need for human intervention or interpretation. In short, we live in exciting times for blockchain technology.

Chapter 3 introduces some of the components of the Web 3.0 architecture. Distributed networking and storage are happening now and promise solutions that could save the global economy trillions annually. The Web today needs a new security model and an architecture designed around contemporary use cases. The technology stack is just beginning to emerge. It includes Swarm, IPFS, Storj, Golem, and Whisper, just to mention a few of a growing number of components that represent the most ambitious solutions to this problem.

As the global infrastructure adapts to the new demands we are putting on it, unforeseen opportunities will open before us. New tools will change not only the way we work and use web conveniences but also the way we organize ourselves in groups. We are living in an interesting time in history, where the Web begins to bring more knowledge and action capacity for its users, resulting in considerable changes in several aspects of daily life. This new Web is moving fast toward a more dynamic environment, where the democratization of the capacity of action and knowledge can speed up business in almost all areas. Imagine a future with hundreds of real decentralized applications—for example, one in charge of registering land titles and mortgages and handling local taxes, and a more general application in charge of managing the supply chain of registered tenants, their monthly lease payments, as well as mortgage and expense payments. One could easily link information from properties registered in the first application with the tenants and their use of the properties registered in the second application. All this in an easy way using the whole stack of semantic web technology and—something that is not possible to date—ensuring that all data is 100 percent true, guaranteed by the smart contracts. The real questions we must ask are: How will the world of Web 3.0 differ from the world of Web 2.0? How will this technology penetrate beyond the cultures that created it? One thing is for sure: blockchain will be at the center of this new world.

Chapter 4 is primarily about blockchain and the law. We show that smart contracts may be the most transformative current blockchain component, especially for lawyers. We explore why Professors Marco Iansiti and Karim R. Lakhani of the Harvard Business School said:

> "The implications are fascinating. . . . If contracts are automated, then what will happen to traditional firm structures, processes, and intermediaries like lawyers and accountants? . . . Their roles would all radically change. . . . [W]e are decades away from the widespread adoption of smart contracts. . . . A tremendous degree of coordination and clarity on how smart contracts are designed, verified, implemented, and enforced will be required. We believe the institutions responsible for those daunting tasks will take a long time to evolve. And the technology challenges—especially security—are daunting. . . . [L]aw firms will have to change to make smart contracts viable. They'll need to develop new expertise in software and blockchain programming."

The chapter also explores how blockchain provides new possibilities for the way we interact and exchange information, and as such brings forth challenging and complex legal issues and pushes the boundaries of existing laws. We see that blockchain technology is something that our laws will have to adapt to, just as they adapted to the Internet, medical technology, e-discovery, and social media. There is a huge change before us as lawyers and as developers to embrace it and be part of its evolution.

We show how regulation around blockchain is still up in the air, not only globally but also in each state here in the United States. Businesses operating in regulated industries should seek guidance from their regulators before integrating critical, customer-facing, or data-handling

processes with platforms like Ethereum. We examine the large strides made in the financial services arena with private and consortium varieties of blockchain. We see clearly that this is an indication that financial institutions are playing in and watching this space very closely.

Chapter 5 covers terminology (including block attributes) and concepts as well as the technology stack, blockchain development platforms, and APIs. It also covers the Ethereum Virtual Machine and Ethereum dapps, DAOs, and autonomous smart contracts.

In **Chapter 6**, we introduce the Solidity smart contract programming language and the tools that make it simple and easy to fast-track deploy a smart contract to the Ethereum blockchain. In this chapter, you get to create your first simple smart contract.

In **Chapter 7**, we introduce tools and techniques that are a little more complicated and support a workflow to handle the development of more complex solutions.

In **Chapter 8**, we look into use cases for private and consortium blockchain solutions. We review private blockchain technology such as Hyperledger, Monax, Ethereum, Hyperledger Fabric, Quorum, and the Hyperledger tools Cello, Composer, and Explorer. We explore the many options for permissioned private blockchains and show why the list is likely to grow. In many cases, government/industry regulation dictates that private control will be needed. That being said, the freedom, neutrality, and openness that started Bitcoin on the public blockchain are important to keep in mind. The focus of decentralizing control and consensus on the public platform is clearly something to think about. There is a great deal of chatter and concern around privacy, identity, speed, and cost of the public blockchain solutions. It is important to note that by creating privately administered smart contracts on public blockchains, or cross-chain exchange layers that sit in between public and private blockchains, it is possible to deliver some degree of the properties of private blockchains on the public platforms. Time will tell if these types of capabilities and properties ever get built into the public blockchain.

While blockchains are heralded as a technological breakthrough that will solve many problems, it's clear that they face a large number of unique challenges. **Chapter 9** reviews these challenges and why they are not insurmountable, though they require a lot of work to develop infrastructure and safety mechanisms to overcome them.

In **Chapter 10**, we introduce the development life cycle of a fully functioning betting application built on Ethereum. While the focus was primarily on coding with Solidity, we made sure to provide the reader with an entire application and explain each line of code and every setup move required to build the application.

In **Chapter 11**, we show the reader how to deploy the application built in Chapter 10. We also step through the development of a simple front end to run the application. If you have read, understood, and tried some of the code in this chapter, you can now write new scripts to deploy and test your own smart contracts. You can create a contract, and you can create a front end to interact with it. In short, once you get through this whole book, you will be ready to design, code, test, and deploy an Ethereum blockchain application.

1 Introduction to Blockchain

For the world of technology users, blockchain represents a dramatic improvement to the landscape of information collection, distribution, and governance. That point has been espoused these past few years in the books and presentations that hype and imagine this new world. This book is one of the first to address the development of blockchain applications. As such, we will present a development road map to the emerging options and trends. That said, this is the first edition of what will be a series following the blockchain development evolution. This book is aimed at all levels of developers, software engineers, and anyone interested in the basics of blockchain technology, as well as the languages and tools required to build decentralized applications. We will introduce everything needed to understand the technology, write "smart contracts," build applications that interact with them, and deploy and maintain these applications on a host of emerging platforms.

So, let's begin. Simply put, a blockchain is a database encompassing a physical chain of fixed-length blocks that include 1 to N transactions, where each transaction added to a new block is validated and then inserted into the block. When the block is completed, it is added to the end of the existing chain of blocks. Moreover, the only two operations—as opposed to the classic CRUD—are add transaction and view transaction. So the basic blockchain processing consists of the following steps, which are numbered 3, 4, and 5 in Figure 1-1:

1. Add new and undeletable transactions and organize them into blocks.
2. Cryptographically verify each transaction in the block.
3. Append the new block to the end of the existing immutable blockchain.

More comprehensively, a blockchain is also a distributed database that maintains a doubly linked list of ordered blocks. Each block averages 1 megabyte (see https://blockchain.info/charts/avg-block-size) and contains control data of approximately 200 bytes, such as a timestamp, a link to a previous block, some other fields (as depicted in Figure 1-2, to be discussed later), and 1 to N transactions as can fit in the remaining space.

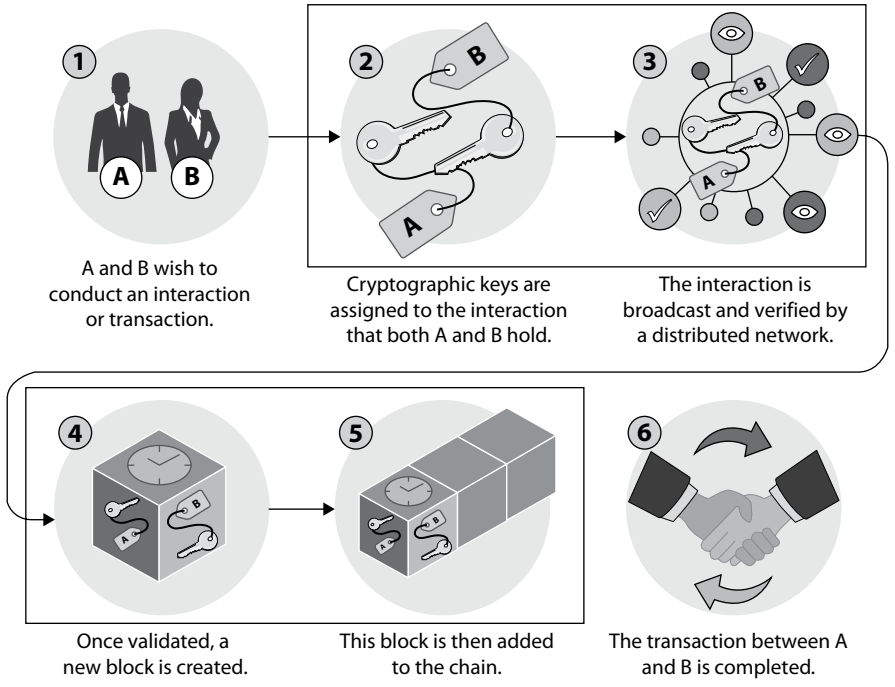

FIGURE 1-1 **Public blockchain transaction flow**

The blocks once recorded are designed to be resistant to modification; the data in a block cannot be altered retroactively. Through the use of a peer-to-peer network and a distributed timestamping server, a public blockchain database is managed autonomously. Blockchains are an open, distributed ledger that can record transactions between two parties efficiently and in a verifiable and permanent way as depicted in Figure 1-1.

The ledger itself can also be programmed to trigger transactions automatically. Blockchains are secure by design and an example of a distributed computing system with high byzantine fault tolerance. Decentralized consensus can therefore be achieved with a public blockchain. As we shall discuss in detail later, these features make blockchains ideal for recording events, medical records and other records management activities, identity management, transaction processing, and a host of emerging applications. Moreover, blockchain technologies allow us to achieve large-scale and systematic cooperation in an entirely distributed and decentralized manner. This can be considered and implemented as a global governance tool, capable of managing social interactions on a large scale and dismissing traditional central authorities. For example, in 2015, libertarian political activist Vit Jedlička declared Gornja Siga—a seven-square-kilometer patch of uninhabited forest between Croatia and Serbia—to be the "Free Republic of Liberland." He used the Bitcoin blockchain as a provisional government and released a constitutional document setting out how this new country would be governed: voluntary taxation, an almost nonexistent government, and zero restrictions whatsoever on speech and information.

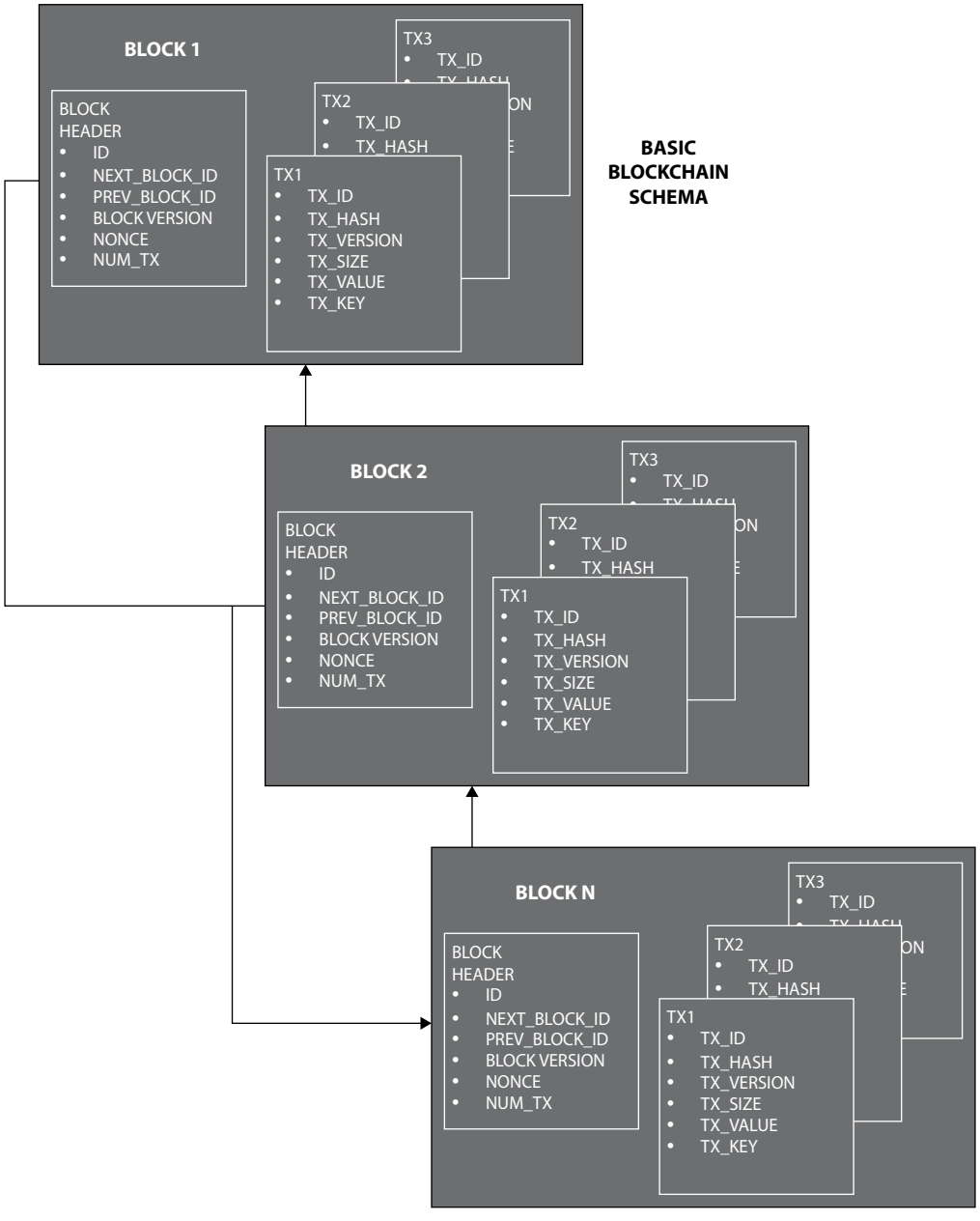

FIGURE 1-2 Blockchain data layout

Let's look at some analogies that illustrate what is different about the public blockchain. It's both a database and the software that envelops it. As software, it is like BitTorrent, a program that allows you to upload and download files directly with others also running the BitTorrent software. So instead of uploading a file to a file-sharing service, such as Dropbox, and then

sending your friend a link to download the file, you just upload the file directly to your friend's computer. This is what we mean by a peer-to-peer (p2p) program (see Figure 1-3).

The public blockchain is also a peer-to-peer program with one very important difference: not only does it move files (data) from peer to peer, it also ensures that all the peers have the same exact data. It enforces this. If the data changes on one machine, it changes on all the machines. There are rules specifying exactly how a change can be made, and if someone doesn't follow them and modifies their copy illegally, they're ignored. It's no different from an email program trying to send an email without the proper SMTP headers—it won't be recognized by other email programs. By the same token, if your version gets deleted or corrupted, it's not a problem, just re-sync with your peers and you get a fresh valid copy.

As noted, the way current public blockchains like Bitcoin and Ethereum work is that instead of changing data within the dataset, new data is just appended onto the old. In other words, data is only written, never deleted. This is how it gets the name *blockchain*, because new data is added in batches, or blocks, and appended to the existing blocks, forming a chain of blocks. Not only does everyone have the same database (blockchain), but everyone gets a locker within the blockchain that only they can access. Normally, exclusive access to something is managed with usernames and passwords. The public blockchain has no central authority to manage usernames and passwords, so instead it uses cryptography. Each user is able to generate a locker address and a private key code that allows them to unlock the locker. The locker is only an analogy, of course. In reality it's just an ID number (referred to as an address), which is tagged to a user's data. The private key is a code that allows the user to prove they're the creator (or owner) of that *address*. Only the person who generated the address would have the private key, and no one can ever determine what the private key is from the address alone.

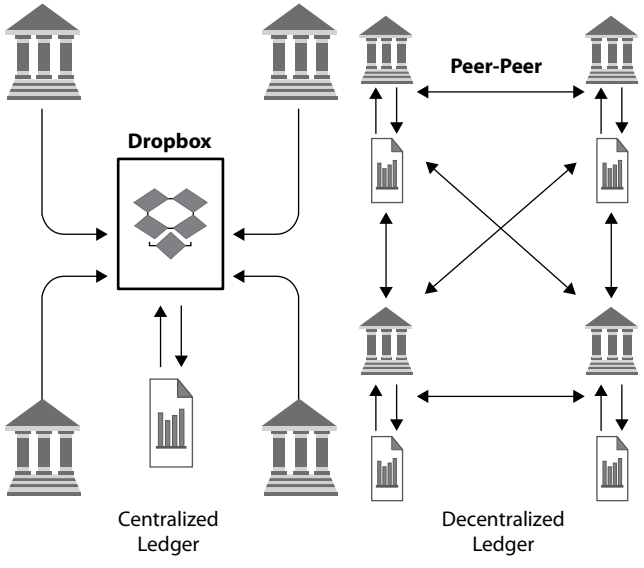

FIGURE 1-3 Decentralized versus centralized data stores

So while everyone can see the data tagged with your address in the blockchain, no one is allowed to modify it. It can only be modified by the person who can prove they're the owner via the private key. For example, if bitcoins are tagged with your address, they cannot be moved (i.e., tagged with another address) unless the private key is used.

What's also amazing about this system is that anyone can generate an address by themselves, in isolation, without concern that it will clash with anyone else's address. The reason for this is that there are so many possible addresses, it is essentially impossible to clash with another address, even if you tried.

It gets better. Not only is static data stored in the dataset, but you can store executable code in it too. Say you have a piece of code written in JavaScript-type language, such as Ethereum Solidity, sitting there on everyone's machine waiting to be executed. Remember that data is only written to the blockchain, never erased, so you now have a piece of code no one can change. Everyone can be certain the way it's written is the way it will always run.

This code is also tagged with someone's address. The owner of that address gets to decide what operations are open to the public and what only they can execute. They only get to make this decision at the time the code is written. Once written, it cannot be changed. Everyone will still be able to see the code and what it's doing, but can only interact with it in the ways specified by the owner.

Let's start with the original motivation to create a blockchain: money. Our current monetary system is based on records of how much money is out there and who has how much of it. We rely on our governments and banks to maintain these records. But a blockchain allows us to keep these records ourselves, since it guarantees that the record is the same for everyone. We each keep this dataset—that is, the blockchain that contains a record of every single transaction that happens in the particular monetary system. Since everyone's copy is synchronized with everyone else's, no one has to worry about fraudulent or conflicting entries. There's now no need for a bank to manage our records. The blockchain does it instead. As far as how money gets created and distributed in the first place, that is another story, but the Bitcoin network as well as other cryptocurrencies handle that as well.

That is just on the data, or ledger, side of things. It gets far more interesting when computer code is managed in that way too. Let's imagine a legal contract: Certain actions are taken under certain conditions. Even after the parties sign, they must still rely on the good faith of the other or our justice system to carry out their side of the agreement. Let's take an example. Donald hates flight delays. AIGore Insurance tells him if he pays them $5 and his flight is delayed by more than an hour, AIGore will return his $5 and pay him an additional $20. A simple insurance scheme, or perhaps it's a bet. In any case, when Donald gives AIGore $5 he has to trust them that they will carry out their end of the bargain. However, by using a blockchain he can eliminate this risk. Collectively, they write the conditions of their agreement in computer code, and initialize the contract with enough funds to make good on either side: Donald sends $5 worth of cryptocurrency, and AIGore Insurance sends $20. Then an hour after Donald's flight is scheduled to arrive, the computer code contract will do the following:

1. Look up Donald's flight on www.flightstats.com.
2. If it was delayed more than an hour, send Donald $25.
3. Otherwise, send AIGore Insurance $25.

This code, once it's written to the blockchain, cannot be removed or altered. Neither party can unilaterally remove the money. Donald and AIGore are guaranteed that the terms of the contract will be executed. In the Ethereum blockchain, this is termed a *smart contract*; much more on that as we proceed. We will examine the code for an application that does just that. See http://fdd.etherisc.com/ for details on how this type of insurance is implemented using the blockchain.

Blockchain: An Information Technology

As mentioned, a blockchain is a distributed ledger of transactions implemented as data batched into blocks that use cryptographic validation to link the blocks together. Each block references and identifies the previous block using a hashing function which forms an unbroken chain (i.e., blockchain).

A public blockchain is not stored in one central computer. Nor is it managed by any central entity. Instead, it is distributed and maintained by multiple computers or nodes that compete to validate the newest block entries before the other nodes to gain a reward for doing so.

The block validation system is designed to be immutable. That is to say, all transactions old and new are preserved forever with no ability to delete. Anyone on the network can browse via a designated website and see the ledger. This provides a way for all participants to have an up-to-date ledger that reflects the most recent transactions or changes. In this way, blockchain establishes trust, which as we shall see facilitates transactions and brings many cost-saving efficiencies to all types of transactional interactions.

A Distributed Trusted Information Technology

From a technical point of view, the blockchain is a distributed, transparent, immutable, validated, secured, and pseudo-anonymous database existing as multiple nodes such that if 51 percent of the nodes agree then trust of the chain is guaranteed. The blockchain is distributed because a complete copy lives on as many nodes as there are in the system. The blockchain is immutable because none of the transactions can be changed. The blockchain is validated (e.g., in the Bitcoin space) by the miners who are compensated for building the next secure block. The blockchain is pseudo-anonymous because the identity of those involved in the transaction is represented by an address key in the form of a random string.

That said, this is an evolving space and, like the cloud computing paradigm, there are public, private, and even hybrid blockchains, which we will explore in detail later in this chapter. These blockchain variations on the basic theme are the result of enterprise architects looking to implement blockchain applications to save time and fees. Enterprise requirements around scaling, performance, the need to know the identity of those involved in the transaction, and other things provide its emerging variations. Blockchain evangelists reckon distributed ledger technology has the potential to upend centralized database practices in institutional finance and most other transaction-based technology. In 2017, the technology shifted from hype to commercial reality. For blockchain to succeed, the application development life cycle, which facilitated large web applications using tools like HTML, CSS, JavaScript, REST web services, Java, SQL, and NOSQL data stores, will have to be amended to integrate the blockchain

onto that stack. We will need integrated development environments (IDEs) and continuous integration and testing procedures to move applications and their attendant code from development to QA and ultimately reliable production implementations. Additionally, because blockchains cannot access data outside their network on their own, third-party services (known as oracles, agents, or data feeds) will also need to be integrated. These oracles or agents typically access and verify real-world occurrences and submit this information to a blockchain to be used by smart contracts. They provide external data when needed and push it onto the blockchain. Such conditions could be anything, such as the flight delay information we saw in the insurance example. The blockchain will have to operate efficiently, scale well, handle the know-your-client (KYC) process, create the aforementioned oracles or APIs that produce and consume off-chain services to verify events and data and handle/convert real-world money to and from cryptocurrencies, and integrate well with different chains. This is all in progress, and we will explore some of these IDEs and development processes in detail as we proceed.

Implementation Trends

A lot has changed since blockchain first emerged as the technology underpinning the cryptocurrency Bitcoin as a distributed ledger of transactions and asset ownership that is maintained by a network of computers over the Internet. More proof of its ability to reduce costs and speed up post-trade processes has emerged in the past year. We will explore this in detail.

A key factor to its rapid growth is the backdrop against which it has launched. Financial institutions and infrastructures are under pressure both to comply with regulation and to reduce cost. That pressure coincided with this technology coming into existence. It's the intersection of requirement and opportunity that's causing the rapid growth. The technology will make moves from "proof-of-concept" technology into production, especially in cross-border payments and trade finance. In this book, we will examine the development life cycle that is emerging with the blockchain. Having been developers for over 30 years, we have witnessed lots of changes in technology starting from the IBM Assembler/COBOL days. We like to kid our colleagues with the fact that we may have worked on the first blockchain.

Back in 1974, no Wall Street firm had its own computer. All processing was done by the famous payroll firm Automatic Data Processing (ADP), located a few blocks away from the heart of Wall Street. The licensed securities exchange firms would drop their stack of punch cards containing the transactions into a designated dropbox. They also dropped off one of their master data magnetic tapes containing the sort of "blockchain"—that is, all transactions to date for that particular organization in sequential order stored in IBM's QSAM format. The programs written in Fortran, COBOL, or IBM assembler could only read data sequentially front to back. The cards representing the transactions would be added to the end of the existing chain of QSAM records, creating/writing a new tape file and hence a new state of the database or simple blockchain. Next came the emergence of early database technology like IBM IMS, IDMS, and ADABAS, followed by the dawn of the ever-enduring SQL in the mid-1980s. Then the open source revolution led to the Linux/Python/Java/SQL/NOSQL/HTML/JavaScript technology stack where web development and database development have matured to create the big data/web-enabled world we live in. Blockchain will further disrupt this evolution to bring trust firmly back into the picture. That said, it will

have to be integrated into this existing development paradigm. These changes are emerging. Early adopters of blockchain used first-generation IDEs to develop applications and, as we shall see, JavaScript-type languages for things like smart contracts, to be discussed in a later chapter. Moreover, integrating blockchain with existing applications will also present challenges that we will examine.

As further impetus to the rise of blockchain, we expect 2018 will be dominated by last-minute preparations for the incoming revised European Union Revised Payment Services Directive (PSD2) effective in May of that year. As PSD2 becomes implemented, banks' monopoly on their customers' account information and payment services will disappear. PSD2 enables bank customers—both consumers and businesses—to use third-party providers to manage their finances. That means you may be using Google to pay your bills, make transfers, and analyze your spending. Banks will provide these third parties access to their customers' accounts through open APIs (application program interfaces). The EU directive opens the door to any interested company, with provisions that will make it easier for startups to access data from banks. This will allow the startups to use blockchain to better penetrate some functions currently performed by banks. With the creation of open banking platforms, there will be opportunities for financial technology (fintech) firms to partner with banks to create new wave customer experiences and provide increased transparency on performance and fee structures. We believe that 2018 is the year that fintech catches up with the hype: validating blockchain-based technologies with promise, scale, customers, and adoption. We are all seeing signs that are consistent with a technology going mainstream in the next five years and changing the transactional landscape for many more years to follow.

Trust: The Byzantine Generals Problem

Back in early days of business computing, circa 1980, computer scientists began to examine reliability and computers. It was determined that a reliable computer system must be able to cope with the failure of one or more of its components. A failed component may exhibit a type of behavior that is overlooked and problematic, namely, sending conflicting information to different parts of the system. The problem of coping with this type of failure is expressed abstractly as the *Byzantine Generals Problem*, from the 1982 scholarly paper by Leslie Lamport, Robert Shostak, and Marshall Pease (www-inst.eecs.berkeley.edu/~cs162/fa12/hand-outs/Original_Byzantine.pdf). This abstract problem and the solutions thereof are used in developing highly reliable and trusted blockchain implementations.

The Byzantine Generals Problem Explained: Why Trust Is So Important

The Byzantine Generals Problem (BGP) is one of many in the field of agreement protocols. Lamport framed his paper around a story problem. This was the style of the day as evidenced by the attention received by another computer scientist, Edsger Dijkstra and his dining philosophers problem, based on a classic operating system synchronization problem.

To set the table for this problem—and as the musical *Hamilton* will be on Broadway long after this book is out of print—we see that the BGP problem was relevant in the American Revolution. At the start of the conflict, the Continental Congress ordered an oath of allegiance to be administered to all army officers. George Washington, the commander-in-chief, administered

it to the general officers. When Washington began to read the oath to Major General Charles Lee, Lee withdrew his hand from the Bible. When Washington demanded a reason for the strange conduct, according to *The Writings of George Washington*, Lee replied, "As to King George, I am ready enough to absolve myself from all allegiance to him; but I have some scruples about the Prince of Wales." This odd reply elicited much laughter. Lee was then playing a desperate game of treason, and probably had some problems with his conscience about taking such an oath which he (and later Major General Benedict Arnold) would violate.

The BGP is built around a similar story line: the commanding general who makes a decision to attack or retreat, and must communicate the decision to his lieutenant generals. A given number of these actors are traitors (possibly including the general). Traitors cannot be relied upon to properly communicate orders; worse yet, they may actively alter messages in an attempt to subvert the process.

In the analogy, the generals are collectively known as processes, the general who initiates the order is the source process, and the orders sent to the other processes are messages. Traitorous generals and lieutenant generals are faulty processes, and loyal generals and lieutenant generals are correct processes. The order to retreat or attack is a message with a single bit of information: a one or a zero.

A solution to an agreement problem must pass three tests: termination, agreement, and validity. As applied to the Byzantine Generals Problem, these three tests are:

1. A solution has to guarantee that all correct processes eventually reach a decision regarding the value of the order they have been given.
2. All correct processes have to decide on the same value of the order they have been given.
3. If the source process is a correct process, all processes must decide on the value that was original given by the source process.

The best way we know to implement a reliable trustworthy computer system (e.g., blockchain) is to use many different processors to compute the same result, and then to perform a majority vote on their outputs to obtain a single value. See Figure 1-4, which views the BGP and blockchain issues in a side-by-side analogy.

GENERALS		BLOCKCHAIN
Agree on a Strategy	Objective	Agree on Valid Transactions
Separated Camps	Spacial Distribution	Distributed Nodes in the Network
Loyal Troop and Generals	The Good Ones	Truthful Nodes
Traitors	The Bad Ones	Evil Nodes
Corrupt a Message	The Attack	Add an Invalid Transaction to the Blockchain
How to know which Message is True	The Problem	How to know which Transaction is Valid
Don't have	A Solution	Proof-of-Work
Don't have	Consensus	Blockchain with More Difficulty

FIGURE 1-4 BGP and blockchain compared