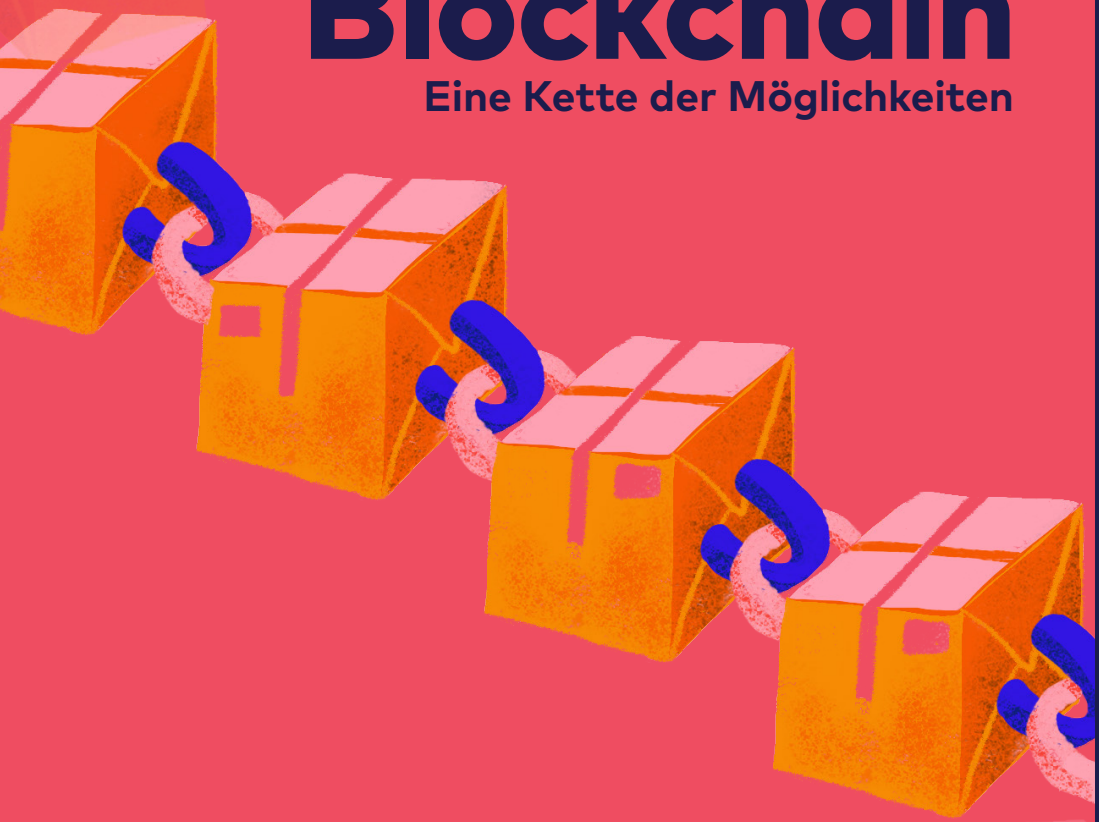
Lars Hupel · Stefan Tilkov

Blockchain

Eine Kette der Möglichkeiten



INNOQ

Blockchain

Eine Kette der Möglichkeiten

Lars Hupel
Stefan Tilkov

ISBN 978-3-9821126-5-7
innoQ Deutschland GmbH
Krischerstraße 100 · 40789 Monheim am Rhein · Germany
Phone +49 2173 33660 · WWW.INNOQ.COM

Layout: Tammo van Lessen with XeLaTeX
Design: Sonja Scheungrab
Print: Pinsker Druck und Medien GmbH, Mainburg, Germany

Blockchain – Eine Kette der Möglichkeiten
Published by innoQ Deutschland GmbH
1. Auflage · Januar 2020

Inhaltsverzeichnis

Vorwort

Die Blockchain-Technologie ist in vielerlei Hinsicht ungewöhnlich, ganz besonders aber auf einer Meta-Ebene: Sie ist ein Hype, der nicht den üblichen Gesetzen der Gartner'schen Hype-Kurve folgt, bei dem auf übersteigerte Erwartungen Desillusion und Ernüchterung folgt, bevor sich ein Plateau der Produktivität einstellt. Stattdessen ist bereits in der Phase, in der der Ansatz von einigen Vertretern als Lösung aller Menschheitsprobleme gepriesen wurde, auch häufig die genau gegenteilige Meinung anzutreffen gewesen: Blockchain als vollständig nutzlose, nahezu absurde Idee, die nur von ahnungslosen Spinnern ernst genommen wird. Und dass ganz besonders der prominenteste Vertreter der darauf basierenden Kryptowährungen, Bitcoin, durch den hohen Energieverbrauch vermeintlich zum Untergang der Menschheit beiträgt, wird dafür auch nur selten als hilfreich empfunden.

Beide Sichten werden dem Thema nicht gerecht. Wie für jeden anderen Architektur- und Technologieansatz gilt auch für die Blockchain, dass sie in sich weder gut noch schlecht ist, sondern für manche Anforderungen passt und für andere nicht. Und dabei gibt es auch nicht nur eine Variante, sondern eine Vielzahl von Varianten und Abstufungen, die für unterschiedliche Anwendungsszenarien Sinn ergeben. Für einige (wenige) Anwendungsszenarien gibt es aktuell überhaupt keine alternative Lösung, für andere haben die Alternativen Vor- und Nachteile, die entsprechend beurteilt werden müssen.

Mit diesem Primer versuchen wir, eine unaufgeregte, neutrale Sicht auf das Thema zu liefern. Dabei legen wir den Fokus auf die Technik, die dabei nicht Selbstzweck ist, sondern als Mittel zum Zweck dient, nämlich zum Erreichen bestimmter Ziele. Wir sind keinesfalls der Meinung, dass nun auf einmal jedes Problem mit einem Blockchain-Ansatz gelöst werden muss und damit auf einmal magisch perfekt erledigt ist, aber wir glauben auch nicht, dass es keinerlei Anwendungsfälle gibt. Als Software-Architekt*innen/-Entwickler*innen müssen wir entscheiden, wann welcher Ansatz aus unserem Werkzeugkasten angemessen ist, und dazu müssen wir die Ansätze verstehen und ihre Stärken und Schwächen beurteilen können. In bestimmten Fälllen können Blockchain-basierte Ansätze eine exzellente Lösung für ein (Teil-)Problem sein – und wir hoffen, mit diesem Primer dabei zu helfen, genau diese Fälle zu erkennen.

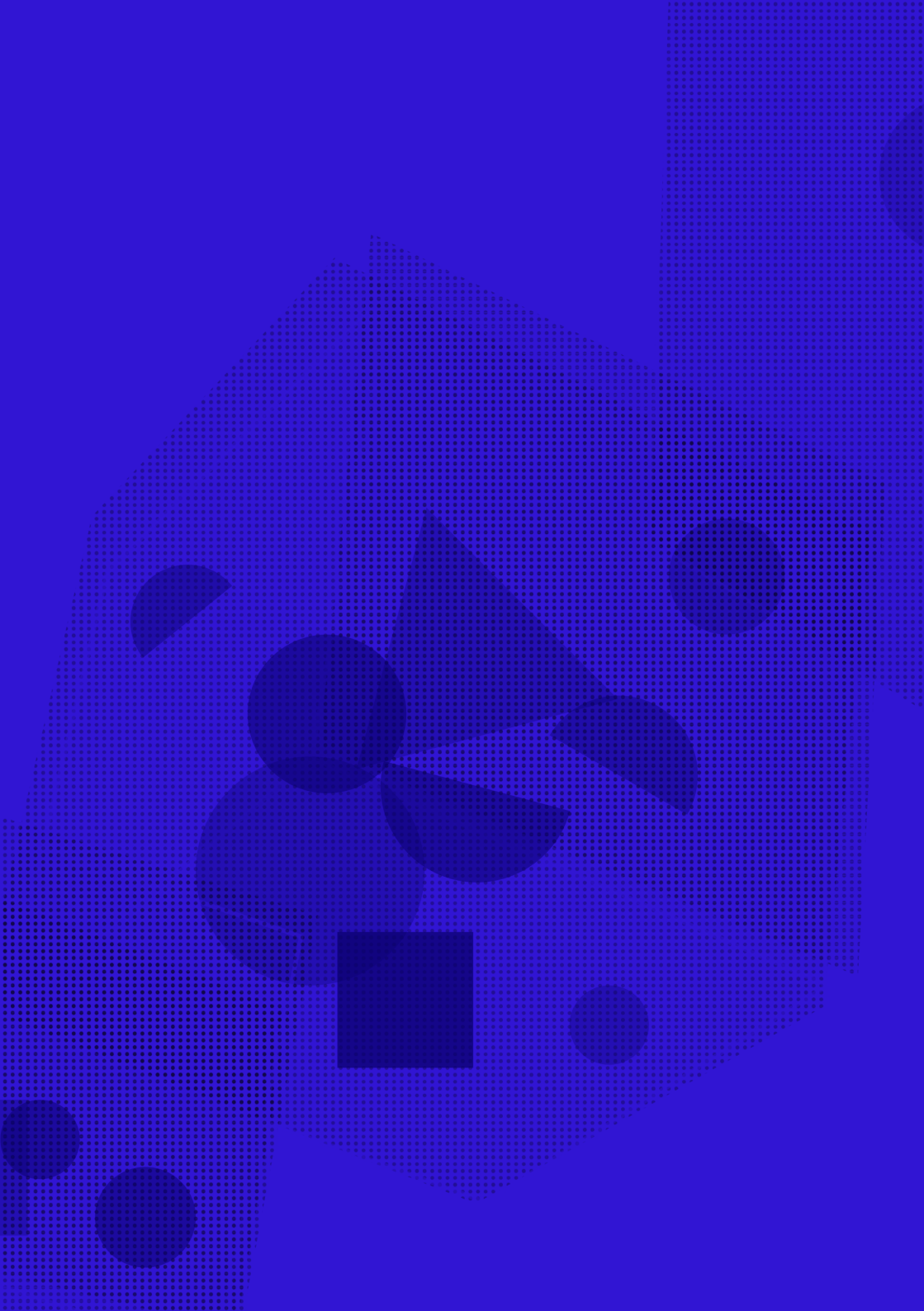
Über dieses Buch

In diesem Buch erläutern wir die Grundlagen von Blockchain-Technologien. Wir geben aber auch einen Ausblick auf bestehende und mögliche weitere Anwendungsfälle um das Potential der Technologie zu zeigen (Kapitel 5 und 6).

Auch wenn zum Verständnis der Anwendungen und zukünftigen Möglichkeiten keine Detailkenntnisse der Technik nötig sind, führen wir in den ersten Kapiteln zuerst die Grundlagen ein, auf denen Blockchain basiert (Kapitel 1 und 2) und erläutern dann die technischen Details von Bitcoin und Ethereum (Kapitel 3 und 4).

Danksagung

Zahlreiche Menschen haben uns bei der Entstehung dieses Buches unterstützt. Wir danken unseren Kollegen Marc Jansing, Joachim Praetorius und Gernot Starke für zahlreiche Beiträge zur Verbesserung der Texte. Die Illustrationen wurden gezeichnet von Olga Marques. Der Text über die Historie der Blockchain wurde von Christine Graf geschrieben.



1 Überblick

Oberflächlich betrachtet gleicht die Blockchain einer Datenbank, die Daten, Dokumente, Informationen und vor allem Transaktionen aller Art speichern kann. Das Besondere an der Blockchain ist die Art und Weise, wie diese Daten gesichert werden.

Wird eine Transaktion durchgeführt, also z.B. Ware gegen Geld getauscht, wird der gesamte Ablauf in einen Daten-*Block* eingeschlossen, der dann linear mit dem vorhergehenden Block verkettet wird. Diese Kette (*Chain*) reicht zurück bis zum Ursprung, also der allerersten Transaktion. Zusätzlich werden die Daten durch Kryptografie gesichert und mit einem Fingerabdruck versehen.

Dieser Vorgang wird allerdings nicht von einem zentralen Rechner durchgeführt (wie es im Internet typisch ist), sondern innerhalb eines großen, meist öffentlichen, und dezentralen Peer-to-Peer-Netzwerks aus Computern (*Nodes*). So entsteht eine durchgängige Nachweiskette, vervielfältigt und gespeichert auf jedem Computer innerhalb dieses Netzwerks.

Die Blockchain könnte man also auch – besonders anschaulich im Falle von *Kryptowährungen* wie Bitcoin – als dezentral verwaltetes Kassenbuch bezeichnen, das kaum gefälscht oder verändert werden kann. Eine Blockchain, besonders im Unternehmensumfeld, kann aber auch ganz ohne Währung betrieben werden.

Vertrauen

In der gesamten Menschheitsgeschichte haben wir immer wieder nach sicheren Mitteln und Wegen gesucht, um Geschäfte abzuschließen. Die wichtigste Währung war dabei allerdings immer Vertrauen. Vertrauen in einen Handschlag, in den Wert des Geldes, in die Qualität der Ware, die Sicherheit des Bankensystems, in Staatsorgane, und so weiter. Doch der globale Handel wird immer unübersichtlicher. Im Internet kennen wir unser Gegenüber meist nicht. Wirtschaftskrisen und Bankencrashs haben unser Vertrauen in zentrale Systeme erschüttert.

Hier kommt die Blockchain ins Spiel. Sie steht für Unveränderbarkeit, Sicherheit und Dezentralisierung. Bei der Blockchain gibt es keine übergreifende Kontrollinstanz, keine zentrale Institution, der wir blind vertrauen müssen. Vielmehr handelt es sich um eine offene, transparente Infrastruktur, die die unterschiedlichsten Werte (Eigentum, Zertifikate, Verträge, persönliche Daten, Identitäten, Währungen, etc.) speichern und schützen kann. Was dieses System so sicher macht, sind Dezentralisierung, Vervielfältigung und Verschlüsselung. Vertrauen ist nicht mehr notwendig.

Konsens

Um die Blockchain vor Manipulationen zu schützen, übernimmt jeder Knoten die Aufgabe einer Kontrollinstanz. Je mehr Knoten es gibt, desto schwieriger ist es, Daten zu verändern. Eine entscheidende Rolle spielen dabei die *Konsens*-Mechanismen: Erst wenn bei der Mehrheit der Knoten Konsens über die Schaffung eines neuen Blocks, also die Richtigkeit der Daten herrscht, kann dieser an die zuvor erstellten Blöcke angehängt werden. Momentan gibt es dafür mehrere übliche Mechanismen:

- *Proof-of-Work* (*PoW*)
- *Proof-of-Stake* (*PoS*)
- *Proof-of-Authority* (*PoA*)

PoW ist das heute am weitesten verbreitete Konsensverfahren und ist auch bekannt als *Mining*. Dabei konkurrieren alle Rechner miteinander, die als Knoten an der Blockchain teilnehmen, um die Lösung eines mathematischen Rätsels. Der Node, der das Rätsel zuerst löst, darf den nächsten Datenblock schreiben und bekommt eine Belohnung. Bei Bitcoin werden beispielsweise für jeden neu gebildeten Block Bitcoins als Anreiz ausgegeben.

Diese Methode hat allerdings einen entscheidenden Nachteil: da die zu lösende Aufgabe kompliziert sein muss, benötigt sie extrem viel Rechenleistung und damit sehr viel Energie.

Deshalb entstehen nach und nach Alternativen wie z.B. Proof-of-Stake, die das Energieproblem lösen sollen.

Bekannte Konzepte

Viele der Konzepte, die in die Entwicklung der verschiedenen Blockchains geflossen sind, sind nicht neu. Zum Beispiel gab es mit *Hashcash* vor einiger Zeit den Versuch, Spam-Mails zu verhindern. Dabei haben Mail-Server für jede Mail eine individuelle *PoW*-Berechnung durchgeführt. Für Spammer hätte sich dieser Aufwand nicht gelohnt.

Auch die kryptografischen Algorithmen zur Absicherung der Blockchain sind schon länger bekannt. Die allermeisten Entwickler*innen nutzen verteilte Versionskontrollsysteme wie *Git*, bei denen jede Änderung eine Referenz auf die vorherigen Änderungen enthält. Nachträgliche Manipulation des Versionsverlaufs sind damit erschwert. Die mathematische Idee dahinter – sogenannte *Merkle*-Bäume – wurden bereits 1979 erstmalig beschrieben.

Trotzdem gilt das gesamte Gebiet nicht zu Unrecht als Treiber von Innovation. In hohem Rhythmus enstehen neue Blockchains, neue Programmiersprachen, neue Konsens-Verfahren. Den Überblick zu behalten ist daher nicht einfach.

2 Kryptografie

Die wichtigsten und zugleich mathematisch anspruchsvollsten Zutaten für Blockchains kommen aus der *Kryptografie*. In diesem Teil der Mathematik spricht man oft von *Primitiven*: Rechenvorschriften, die man als Blackbox auffassen kann und die nach außen hin bestimmte abstrakte Eigenschaften aufweisen. Welches konkrete kryptografisches Verfahren in der Implementierung letztendlich benutzt wird, ist für das Verständnis unerheblich, so lange die Eigenschaften der Primitiven geliefert werden.

Wir möchten in diesem Abschnitt abstrakt die wichtigsten dieser Primitiven erklären. Um bei der Konvention aus der Fachliteratur zu bleiben, verwenden wir als beispielhafte Kommunikationspartner*innen die Namen *Alice* und *Bob*.

2.1 Einwegfunktionen

Nach der mathematischen Definition ist eine *Funktion* eine Rechenvorschrift, die auf eine bestimmte, festgelegte Art Daten umwandelt. Oftmals können solche Funktionen umgekehrt werden, z.B. Kompressionsverfahren, die Dateien verkleinern und auch wiederherstellen können. Sicherheit in der Kryptografie basiert jedoch darauf, dass bestimmte Funktionen nur von befugten Personen umgekehrt werden können. Beispielsweise wird dann ein Geheimnis als zusätzliche Zutat zu den eigentlichen Daten verlangt. Manche Funktionen sollen aber auch gar nicht umgekehrt werden können. Dann spricht man von einer *Einwegfunktion*.

Bei solchen Funktionen liefert sich die Forschung einen Wettlauf mit der immer weiter steigenden Rechenkapazität. Manche Funktionen, die vor wenigen Jahrzehnten als unumkehrbar galten, können heute auf einen günstigen Smartphone umgedreht werden.

Ein konkretes Beispiel ist die Multiplikation zweier Primzahlen: Multiplikation ist vergleichsweise effizient zu berechnen. Die Umkehrung, nämlich die Zerlegung einer Zahl in ihre Faktoren, ist wiederum sehr aufwändig. Dennoch können durch schnellere und mehr Prozessoren heutzutage weit größere Zahlen faktorisiert werden. Jedoch gelten Verfahren auf dieser Basis trotzdem noch als sicher, da die Faktorisierung von

praktisch benutzten Zahlen noch in weiter Ferne liegt, selbst wenn man Fortschritte im Quantencomputing mit berücksichtigt.

Ein Telefonbuch ist eine passende Analogie hierfür. Es ist sehr einfach für Menschen, ausgehend von einem Namen die passende Telefonnummer zu finden. Wir brauchen bloß der Sortierung im Telefonbuch zu folgen und den Namen an der korrekten Stelle nachzuschlagen. Umgekehrt tun wir uns aber schwer, denn wenn wir nur die Telefonnummer kennen, müssen wir im schlimmsten Fall jeden einzelnen Eintrag nachprüfen.

Für Computer ist die „Rückwärtssuche" aber in Sekundenschnelle gelöst.

Selbstverständlich gibt es eine mathematisch schlüssige Definition von Einwegfunktionen, die die Rechenkomplexität der Umkehrung in Betracht zieht. Allerdings ist es in der Praxis sehr schwer zu beweisen, dass eine Rechenvorschrift die nötigen Eigenschaften erfüllt. Heutige kryptografische Verfahren müssen sich daher damit begnügen, dass noch niemand das Gegenteil für sie bewiesen hat.

2.2 Ver- und Entschlüsselung

Kommen wir nun zur ersten kryptografischen Primitive. Nehmen wir an, Alice möchte Nachrichten verschlüsseln und an Bob übermitteln.

Dazu erzeugen sowohl Alice als auch Bob jeweils ein *Schlüsselpaar* bestehend aus einem *öffentlichen* und einem *privaten* Schlüssel. Wie die Namen bereits andeuten, verbleiben die privaten Schlüssel bei der jeweiligen Person. Die öffentlichen Schlüssel können hingegen verbreitet werden, z.B. auf einer Webseite.

Alice kann sich nun Bobs öffentlichen Schlüssel besorgen. Doch wie kann Alice damit eine Nachricht an Bob verschlüsseln? Dazu kann sie *asymmetrische Verschlüsselung* benutzen. Damit können beliebige Menschen eine Nachricht an Bob verschlüsseln, ohne seinen privaten Schlüssel zu kennen.

Umgekehrt kann Bob dem zugehörigen Entschlüsselungs-Algorithmus seinen privaten Schlüssel und die verschlüsselte Nachricht füttern, woraufhin die ursprüngliche Nachricht von Alice herausspringt. Er ist auch der einzige, der das kann; es sei denn, er hat seinen privaten Schlüssel verloren.

Die Analogie aus der „realen" Welt ist ein Briefkasten: Jede*r darf Briefe einwerfen, denn der Einwurfschlitz ist *öffentlich*. Allerdings darf nur der oder die Empfänger*in die Briefe entnehmen, denn die Briefkastentür ist *privat*. Die Sicherheit basiert darauf, dass niemand Briefe über den Einwurfschlitz wieder entnehmen kann.

Asymmetrisch heißt dieses Verfahren, weil Ver- und Entschlüsselung jeweils andere Schlüssel benötigen, nämlich öffentlichen und privaten. Damit ist es auch ein Beispiel für eine Einwegfunktion, die nur mit Zusatzinformation (privater Schlüssel) umkehrbar ist.

2.3 Signaturen

Asymmetrische Kryptografie lässt sich auch „umgekehrt" benutzen: Statt ihre Nachricht mit dem öffentlichen Schlüssel von Bob zu verschlüsseln, könnte Alice die Nachricht mit ihrem privaten Schlüssel *signieren*. Bob kann dann diese Signatur mit dem öffentlichen Schlüssel von Alice nachvollziehen. Eine dritte Person könnte diese Signatur nicht angefertigt haben, da niemand außer Alice ihren privaten Schlüssel kennt.

Dieses Verfahren wird genutzt, um nachweisen zu können, dass niemand den Inhalt einer Nachricht nachträglich geändert hat. Denn wenn auch nur ein Bit in der Nachricht anders ist, wäre auch die Signatur anders. Somit wüsste zum Beispiel Bob, dass er zwar eine korrekt für ihn verschlüsselte Nachricht erhalten hat, sie aber sicher nicht von Alice stammt (oder Alice möchte ihn absichtlich täuschen, indem sie eine defekte Signatur erzeugt).

Im Idealfall funktionieren Unterschriften in der realen Welt ähnlich. Eine bestimmte Unterschrift kann nur von einer Person angefertigt, aber von allen anderen nachvollzogen werden. Natürlich ist das nicht gegeben, da sich Unterschriften technisch einfach replizieren lassen. Schwieriger wird das schon bei biometrischen Merkmalen wie zum Beispiel einem Fingerabdruck. Nicht umsonst werden kryptografische Signaturen oft als „digitaler Fingerabdruck" bezeichnet.

2.4 Hashes

Problematisch an asymmetrischer Kryptografie ist, dass die Algorithmen nicht besonders effizient sind. Die mathematischen Operationen, die durchgeführt werden müssen, sind aufwändig und sollten daher minimiert werden. Daher werden Methoden benutzt, um die Datenmengen zu reduzieren.

Für Signaturverfahren werden oft *Hashes* eingesetzt. Es handelt sich dabei um Algorithmen, die eine beliebige Datenmenge (z.B. ein großes Dokument) auf eine feste Menge an Bits reduzieren.

Eine gute Hashing-Funktion erfüllt folgende Anforderungen:

1. sie ist nicht umkehrbar (Einwegfunktion)
2. kleine Änderungen in den Eingabedaten führen zu großen (nicht vorhersagbaren) Änderungen in der Ausgabe
3. es ist sehr schwer, zwei verschiedene Eingabewerte zu konstruieren, die die gleiche Ausgabe produzieren

Statt also die gesamte Nachricht zu signieren, wird sie mit einer Hashfunktion z.B. auf 160 Bit eingedampft und diese dann signiert. Auch hier kann man Manipulation sehr leicht erkennen, denn eine Änderung an der Nachricht führt zur Änderung des Hashes und damit auch der Signatur.

Auch in der Realität gibt es viele Vorgänge, die sich „chaotisch" verhalten. Nimmt man als Beispiel ein beliebiges Würfelspiel an, dann könnte man die erreichte Punktzahl jeder Spielerin als Hashfunktion auffassen. Schauen wir uns die Bedingungen für einen Hash an:

1. Aus der Gesamtpunktzahl ist der Spielverlauf nicht mehr zu rekonstruieren.
2. Ändert sich eine gewürfelte Zahl, dann kann auch das Spiel möglicherweise ganz anders ablaufen. Dadurch verändert sich auch die Punktzahl.
3. Spielt man mehrere Runden hintereinander, wäre es ein großer Zufall, zweimal genau die gleiche Kombination von Punktzahlen zu erzielen.

Ein kurzer historischer Abriss

von Christine Graf

Die Geschichte des Geldes und der Buchhaltung reicht lange zurück. Und auch die Idee der Blockchain ist gar nicht so jung wie man meint. Zumindest hat die Menschheit schon sehr früh Methoden ausgetüftelt, um Transaktionen so sicher wie möglich zu gestalten.

Das Kerbholz – Die Urform der Blockchain?

Aus der Zeit, aus der bereits Schuldverhältnisse dokumentiert worden sind, aber noch keine Banken existiert haben, kommt die Redewendung „etwas auf dem Kerbholz haben“. Das Kerbholz ist ein Stock oder Brett, auf dem durch Einritzungen Schulden festgehalten worden sind. Sozusagen der Prototyp des Schuldscheins. Erste Vorläufer wurden bereits auf prähistorischen Knochen gefunden, verbreitet war das Kerbholz jedoch vor allem im Mittelalter, als kaum einer schreiben konnte und auch Münzen knapp waren.

Versetzen wir uns für einen Moment in diese Zeit: Ich brauche Geld für eine Anschaffung und gehe zu einem Menschen, der mir Geld leihen kann. Wir beide stehen nun vor dem gleichen Problem: Wie können wir möglichst einfach und sicher festhalten, wie viel Geld ich zurückzuzahlen habe? Wie können wir beide verhindern, betrogen zu werden? Erschwerend kommt hinzu, dass mindestens einer von uns beiden wahrscheinlich weder lesen noch schreiben kann. Die Lösung ist bestechend einfach: Kerbhölzer. Wir legen zwei ähnliche Stöcke nebeneinander und ritzen den Betrag ein – jeweils ein Strich pro Währungseinheit.

Genial, denn als Schuldner habe ich keine Möglichkeit, einen Strich zu entfernen; im Gegenzug kann aber auch mein Verleiher keinen Strich hinzufügen, denn am Zahltag – wenn wir die Kerbhölzer wieder nebeneinanderlegen – würde das sofort auffallen. Auch das Entsorgen meines Kerbholzes wäre für mich nicht sinnvoll, denn dann könnte mein Gegenüber einfach beliebige Forderungen aufrufen. Dank der Verteilung der gleichen Information auf zwei Parteien und die Tatsache, dass man nur neue Information hinzufügen, aber niemals alte tilgen kann, können beide Parteien auf eines getrost verzichten: Vertrauen.

Bis ins 19. Jahrhundert gab es zahlreiche Verwendung dieser Technik. Der Nachweis von Schulden ist dabei nur ein Anwendungsfall. Auch Steuern, Fördermengen im Bergbau, Warenhandel und Arbeitslohn wurden damit verwaltet.

Oder doch die Sumerer?

Auf dem Weg zu den Ursprüngen können wir in der Geschichte durchaus noch ein weiteres Stück zurück gehen. Als die Menschen sesshaft wurden und es Ihnen gelang, Nahrung in Mengen zu erzeugen, die über den Eigenbedarf hinausgingen, begann die Geschichte des Handels – oder sagen wir besser: des Warenaustauschs.

So geht eine Urform der Buchhaltung auf die Sumerer zurück, die im 3. Jahrtausend v. Chr. in Mesopotamien lebten. Die einstige Hochkultur gilt nicht nur als Erfinderin der Keilschrift, die Sumerer entwickelten auch eine Art zentralisiertes Wirtschaftssystem. Wobei wohl beide Erfindungen untrennbar miteinander verbunden ist. Geld bzw. Besitz war schon immer ein geeigneter Ansporn für Fortschritt.

Begeben wir uns also 6000 Jahre zurück nach Uruk, eine sumerische Stadt in der Tausende Menschen lebten: Handwerker, Seeleute, Ärzte, Priester, Lehrer, Verwalter. Die Stadt befand sich inmitten eines „Garten Eden“, fruchtbares Land auf dem Getreide angebaut wurde, ebenso wie Datteln, Aprikosen und vieles mehr. Silber war ebenso Zahlungsmittel wie Milch, Datteln oder Öl.

Eine Stadt dieser Größe erforderte schon damals ein geeignetes System für sichere, wirtschaftliche Transaktionen. Das zentralisierte System ist heute bekannt als Tempelwirtschaft, mit dem Tempel als Hauptakteur.

Bei archäologischen Ausgrabungen fand man kleine, runde Gegenstände aus Ton, die mit abstrakten Zeichen verziert waren. Manchmal waren Sie in versiegelten Tonkugeln, den sogenannten Bullas, versteckt. Punkte, Linien und Siegel weisen auf ihre Eigentümer hin. Heute glaubt man zu wissen, dass diese versiegelten „Tokens“ für die Sumerer zuverlässige Werkzeuge zur Datenspeicherung waren – für die permanente Aufzeichnung realisierter Transaktionen. Das heißt, sie wurden auf Vertragsbasis geschlossen, versiegelt und zu Lagerzwecken in Tempeln aufbewahrt. Verwaltet von erfahrenen Buchhaltern und Priestern.

Ein vergleichsweise komplexes System, das die Grundlage der modernen zentralisierten Banken- und Wirtschaftswelt darstellt. Welches aber auch durchaus erste Merkmale der Blockchain aufweist: Die Ton-Token dienen als Sinnbild für Transaktionen, die Bullas stehen für die verschlüsselten Blöcke und die Siegel für Signaturen oder Smart Contracts. Und im Mittelpunkt der Tempel als zentrales System, in dem die Transaktionen aufgezeichnet und gespeichert werden und in dem das Vertrauen zwischen den Parteien aufgebaut wird.

Doppelte Buchführung – von Ägypten bis ins mittelalterliche Europa

Von Mesopotamien ist der Weg zu einer weiteren Hochkultur nicht weit: nach Ägypten im 4. Jahrhundert v. Chr. Es ist kaum verwunderlich, dass dort, wo die Handelsrouten der Karamelkarawanen endeten, der Zahlungsverkehr eine wichtige Rolle spielte. Die Tontafeln wurden von Papyrus abgelöst, die Keilschrift von Hieroglyphen. Und auch die alten Ägypter verfügten über Prüfsysteme, um die Bewegung in und aus den Lagerhäusern zu kontrollieren. So fand man auf Papyrusrollen erste Darstellungen von Soll und Haben.

Als eigentlicher Vater des Rechnungswesens und der Buchführung gilt der Italiener Luca Pacioli. Er veröffentliche Ende des 15. Jahrhunderts als erster eine Arbeit über die doppelte Buchführung. Es wird allerdings vermutet, dass jüdische Bankiers im frühmittelalterlichen Kairo (also wieder in Ägypten) die eigentliche Pionierarbeit für ein doppeltes Buchhaltungssystem geleistet haben. Italienische Kaufleute könnten diese Methode von ihren Kollegen aus dem Nahen Osten gelernt haben.

Seit sich im 13. Jahrhundert in Europa eine Geldwirtschaft entwickelte, waren die sesshaften Kaufleute auf die Buchhaltung angewiesen. Nur so konnten sie gleichzeitig mehrere durch Bankdarlehen finanzierte Transaktionen überwachen. Der Ursprung der englischen Entsprechung von „Soll" und „Haben" stammt übrigens aus dem Lateinischen: *Debit* in lateinischer Sprache bedeutet „er schuldet" und *Credit* bedeutet „er vertraut".

Apropos Vertrauen: Paciolis Abhandlung ermöglichte es den Händlern auch, ihre eigenen Bücher zu prüfen – vorausgesetzt, die Einträge in den Buchhaltungsunterlagen entsprachen der von ihm beschriebenen Methode. Ohne ein solches System waren alle Händler einem höheren Diebstahlrisiko durch ihre Mitarbeiter und Vertreter ausgesetzt. Doch fälschungssicher waren diese Methoden sicher nicht. Kein Wunder, dass das oben beschriebene Kerbholz noch im „Code Civil", dem napoleonischen Zivilrecht von 1804, als rechtsgültige Urkunde anerkannt war.

Blockchain aus Druckerschwärze

Und schon sind wir fast wieder in der Gegenwart, genauer gesagt im 20. Jahrhundert. 1991 entwickelten die Kryptografen Stuart Haber und Scott Stornetta eine Technologie, um digitale Dokumente mit einem Zeitstempel zu versehen und ihre Echtheit zu verifizieren. Ihre Lösung war ein fälschungssicherer Fingerabdruck für jedes Dokument. (Um genau zu sein: ein kryptographischer Hashing-Algorithmus, der für jedes Dokument eine einzigartige Identifikationsnummer bzw. einen Hashwert erstellt.) 1994 wurde daraus ein Geschäftsmodell – mit dem Zeitstempel-Service „Surety" bieten sie kryptografisch abgesicherte Siegel für digitale Dokumente. Das Besondere daran ist aber alles andere als digital: Jede Woche generiert Surety einen einzigartigen Hashwert für alle neuen Siegel und lässt diesen Wert in einer kleinen Annonce im Anzeigenteil der New York Times abdrucken. Somit betreiben sie eine Blockchain, bei der die New York Times als Autorität auftritt.

Doch es wird Zeit, über *die* Blockchain schlechthin zu sprechen: Am 9. Januar 2009 wurde die erste Version von Bitcoin gestartet. Am nächsten Tag erhielten hunderte Kryptographie-Experten und -Enthusiasten von Satoshi Nakamoto diese E-Mail:

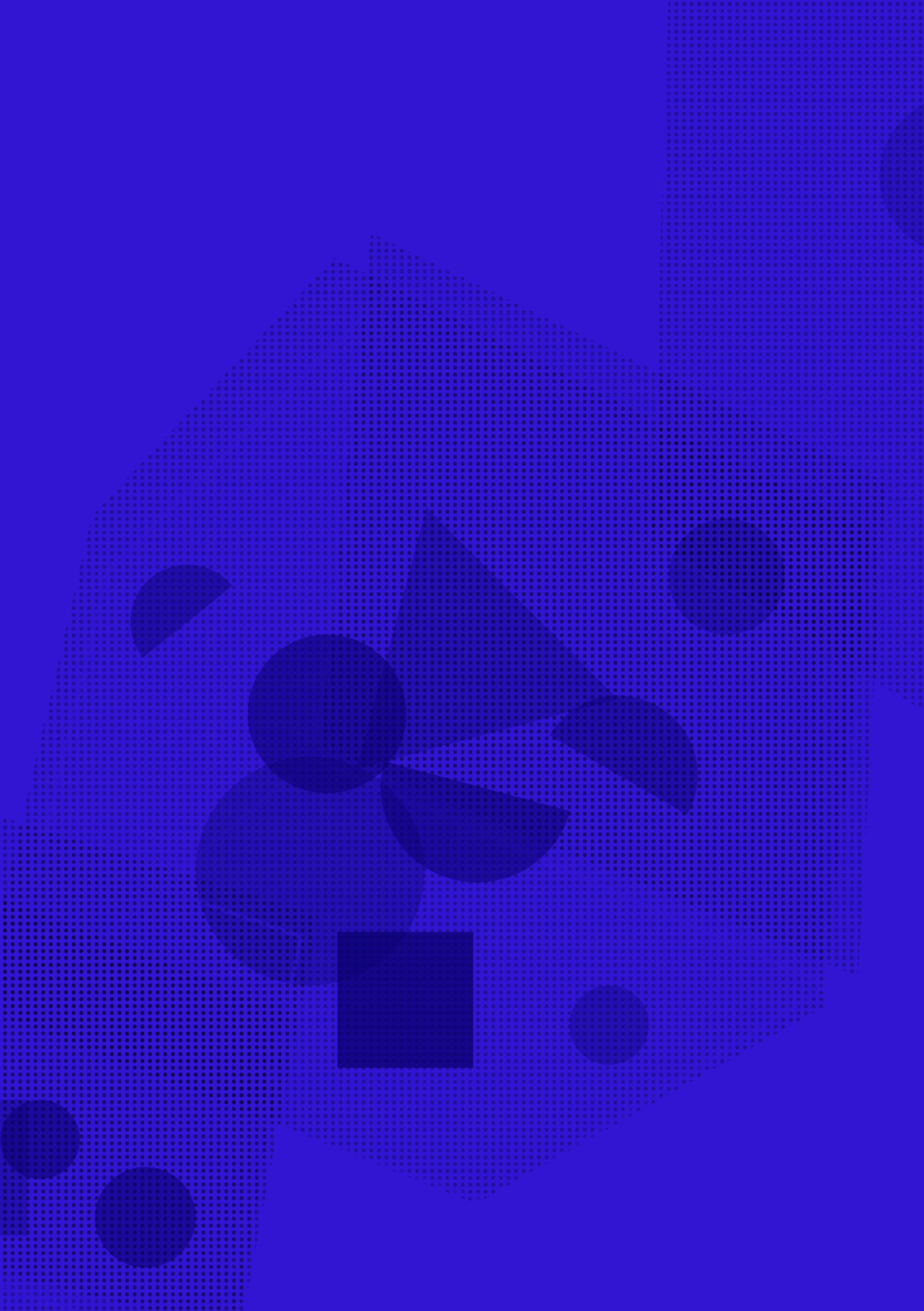
„Bitcoin v0.1 released

Announcing the first release of Bitcoin, a new electronic cash system that uses a peer-to-peer network to prevent double-spending. It's completely decentralized with no server or central authority. See bitcoin.org for screenshots.

Download link:

http://downloads.sourceforge.net/bitcoin/bitcoin-0.1.0.rar"

Damit nahm die rasante Entwicklung der Blockchain ihren Lauf. Seit 2009 wurden einerseits zahlreiche Änderungen, Verbesserungen und Varianten von Bitcoin veröffentlicht, andererseits alternative Blockchains von Grund auf neu entwickelt. Während es von diesen Alternativen je nach Zählung inzwischen mehrere Tausend gibt, hat Bitcoin selbst eine Gesamtbewertung von 100 Milliarden US-Dollar überschritten.



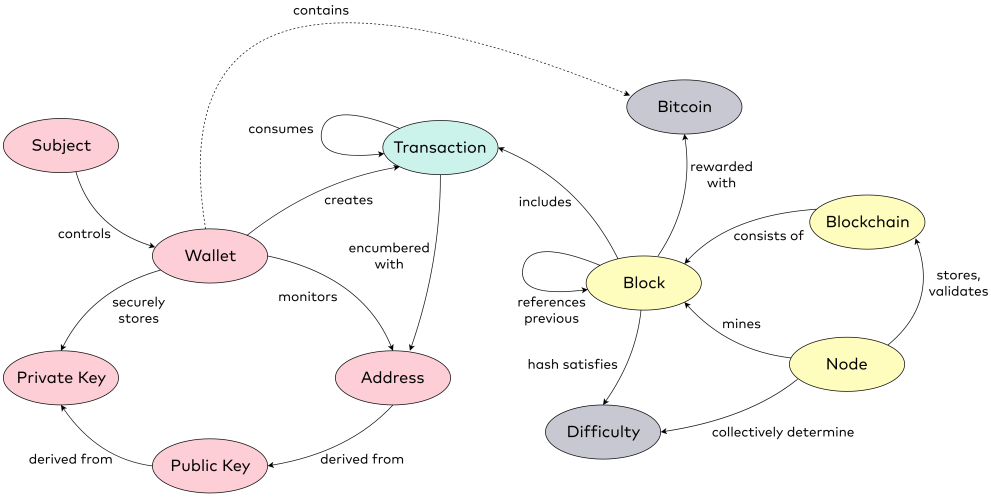
3 Bitcoin

Bitcoin ist die erste dezentrale Kryptowährung und wurde 2009 von einer Person (oder Personengruppe) mit dem Pseudonym *Satoshi Nakamoto* vorgestellt. Die allermeisten Konzepte, die Bitcoin benutzt, waren bereits bekannt, wurden aber von Nakamoto erstmalig kombiniert und ergeben dadurch mehr als die Summe ihrer Teile: ein Netzwerk, welches virtuelles Geld verwaltet und dazu weder eine zentrale Autorität noch gegenseitiges Vertrauen voraussetzt.

Die Selbstbeschreibung von Bitcoin lautet:

"Bitcoin is an innovative payment network and a new kind of money."

Wie die einzelnen Teile zusammenspielen, sehen wir uns in diesem Abschnitt genauer an. Das nötige kryptografische Handwerkszeug haben wir im vorherigen Kapitel gelernt.



Bitcoin-Konzepte und ihre Beziehungen zueinander

3.1 Wallets und Adressen

In einem klassischen Geldsystem gibt es zwei Objekte, mit denen man Geld verwalten kann. Zum einen sind das Geldbeutel, in denen Menschen Münzen und Geldscheine mit sich tragen. Eine Transaktion besteht dann darin, dass ein Mensch einem anderen Menschen physikalische Objekte überreicht. Geldbeutel als solche können im Prinzip beliebig viel Geld beinhalten und beliebig viele Transaktionen ausführen; können aber auch selbst den Besitz wechseln.

Zum anderen gibt es die Banken, die Konten anbieten. Ein Konto kann zu einem Menschen oder einer Organisation gehören. Der Einfachheit halber werden wir von *Subjekten* (*subjects*) sprechen, da es irrelevant ist, ob es sich um eine natürliche oder rechtliche Person handelt.

Die Aufgabe einer Bank ist es, Konten zu verwalten, die virtuelles Geld beinhalten. Ähnlich wie bei einer Geldbörse können Konten beliebig viel Geld beinhalten und beliebig viele Transaktionen ausführen. Konten werden über Kontonummern identifiziert. Überweisungen spezifizieren eine Absender- und eine Empfängerkontonummer. Die Bank, die das Absenderkonto führt, verlangt vom Subjekt eine Legitimation, z.B. durch ein PIN-TAN-Verfahren. Dieser Vorgang bleibt allerdings geheim. Für das Empfängerkonto ist es egal, wie die Legitimation abläuft. Das Geld als solches wird durch Einträge in einer Datenbank repräsentiert.

Bei Bitcoin funktionieren sowohl die Transaktionen als auch die „Münzen“ völlig anders, so dass Konten als Analogie nicht funktionieren.

Stattdessen besitzt jedes Subjekt ein kryptografisches Schlüsselpaar. Aus dem öffentlichen Teil des Schlüssels wird dann eine *Adresse* abgeleitet. Ausgehende Transaktionen müssen mit dem jeweiligen privaten Schlüssel des Absenders signiert sein.

Um eine Adresse zu bekommen, muss man bloß ein Schlüsselpaar erzeugen.

Verliert man seinen privaten Schlüssel, oder existierte vielleicht ein solcher Schlüssel nie, dann laufen Transaktionen an die zugehörige Adresse faktisch ins Leere, denn niemand kann jemals (mehr) ausgehende Transaktionen signieren.

Eine *Wallet*-Software ist dafür verantwortlich, die privaten Schlüssel des Subjekts zu verwalten und geeignet zu speichern. Sämtliche Interaktion eines Users mit dem Bitcoin-Netzwerk laufen daher über diese Software.

3.2 Transaktionen

Das Grundprinzip einer *Transaktion* besteht darin, eine (oder mehrere) Quellen auf eine (oder mehrere) Senken zu verteilen. Als Quelle kommt immer nur eine frühere Transaktion in Frage. Senken sind Adressen, denen eine bestimmte Menge an Bitcoin übertragen wird.

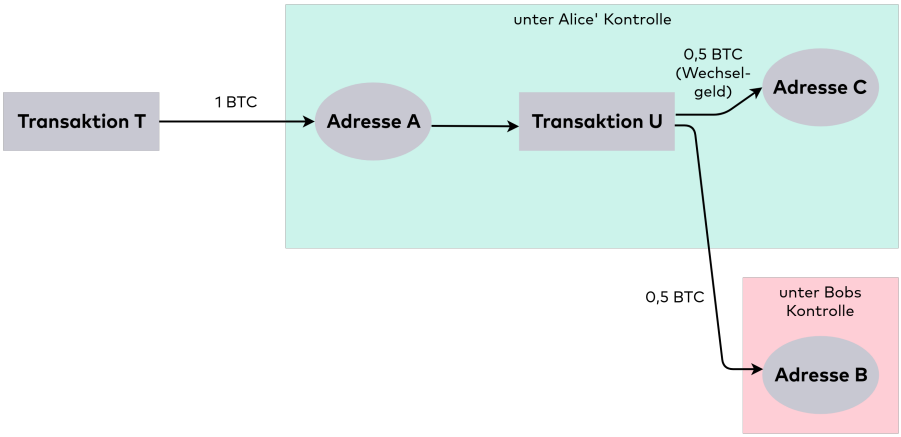
Bevor wir uns diesen Mechanismus anschauen können, müssen wir eine vereinfachende Annahme treffen. Wir gehen davon aus, dass im Bitcoin-Netzwerk bereits Geld (*Bitcoin*, abgekürzt als *BTC*) im Umlauf ist. Wie Geld ins System kommt (Wertschöpfung), klären wir später.

Angenommen, es gibt eine bereits abgeschlossene Transaktion T, die als Senke für 1 Bitcoin die Adresse von Alice angegeben hat. Nun kann Alice eine neue Transaktion U veranlassen, die den Gesamtbetrag von T in Höhe von 1 Bitcoin an weitere Senken verteilt. Die wichtigste Einschränkung ist, dass es nur eine einzige solche Transaktion geben kann, um das mehrfache Ausgeben des gleichen Geldes zu verhindern (*Double Spending*). Alice muss ihre ausgehende Transaktion mit ihrem privaten Schlüssel signieren, um nachzuweisen, dass sie sie wirklich selbst erstellt hat.

Wechselgeld

Was passiert aber nun, wenn Alice nicht den ganzen Bitcoin an Bob weitergeben möchte, sondern nur einen halben? Wenn sie diese Transaktion erstellt, bliebe der andere halbe Bitcoin unverteilt und ginge verloren. Denn wir erinnern uns: Alice darf nur ein einziges Mal über die eingehende Transaktion verfügen! Stattdessen muss Alice eine *Wechselgeld*-Adresse erzeugen, die unter ihrer eigenen Kontrolle steht, an die der Rest verteilt wird.

Damit ist der komplette Wert der Transaktion T verbraucht, die Alice erhalten hat. Bob und Alice können nun unabhängig voneinander über jeweils 0,5 Bitcoin verfügen.



Überweisung von Alice (A) an Bob (B) und Wechselgeld (Change, C)

In Bitcoin-Terminologie nennt man diese beiden noch ungenutzten Senken *Unspent Transaction Outputs* (*UTXOs*), die es irgendwann in Anspruch zu nehmen gilt.

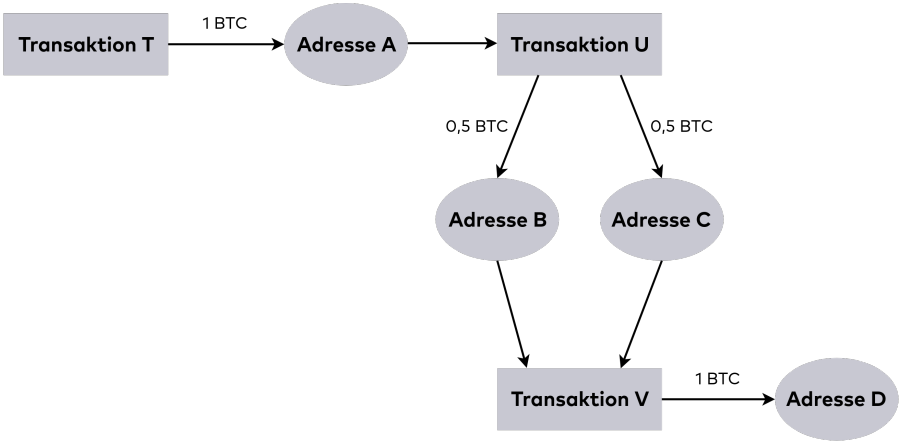
Diesen Mechanismus zu verstehen ist essentiell, um beim Handel mit Bitcoin keinen versehentlichen Verlust zu erleiden. Im Gegensatz zu einer Banküberweisung, wo der Betrag einfach vom Konto abgezogen wird, hört bei Bitcoin die Quelle auf gültig zu sein, nachdem eine einzige Transaktion auf ihrer Basis durchgeführt worden ist. Gängige Client-Software erzeugt automatisch eine frische Adresse, an die das Wechselgeld überwiesen wird.

Obwohl die frischen Wechselgeld-Adressen oberflächlich seltsam klingen, sind wir das so ähnlich im Bargeldverkehr gewohnt. Wenn man ein Produkt für 5 € kaufen möchte und mit einem 10-€-Schein zahlt, erhält man als Wechselgeld einen 5-€-Schein. Dieser Wechselgeld-Schein hat eine andere Seriennummer als der Schein, mit dem wir bezahlt haben.

Bemüht man diesen Vergleich noch weiter, so müsste man eine „Bitcoin-Banknote" bei jeder Transaktion zerreißen und die entstandenen Teile jeweils einen eigenen Wert und Seriennummer versehen. Die ursprüngliche Banknote existiert dann nicht mehr, dafür zwei neue Banknoten, deren kombinierter Wert dem ursprünglichen Wert entspricht.

Anonymität

Der Vollständigkeit halber sei noch erwähnt, dass man nicht zwingend eine frische Adresse erzeugen müsste, da gängige Wallet-Software auch „Rücküberweisungen“ verarbeiten können. Aber allein aus Anonymitätsgründen wird allenthalben davon abgeraten. Denn von außen ist nicht einfach zu erkennen, welche der beiden (oder mehreren) Zieladressen nun Zahlung und welche Wechselgeld sind. Tatsächlich gibt es sogar Dienstleister, die absichtlich mehrere Transaktionen „mixen“, um Zahlungswege zu verschleiern.



Kette von Überweisungen, die 1 Bitcoin trennt und wieder zusammenführt

Hält man sich an diese Praktiken, dann führt das sehr schnell dazu, dass man eine ganze Reihe von reinen Wechselgeld-Adressen in der Wallet verwalten muss. Genau aus diesem Grund unterstützt Bitcoin Transaktionen mit mehreren Quellen. Flugs erzeugt man sich eine weitere Adresse und führt alle existierenden Wechselgelder zusammen.

3.3 Kontostand

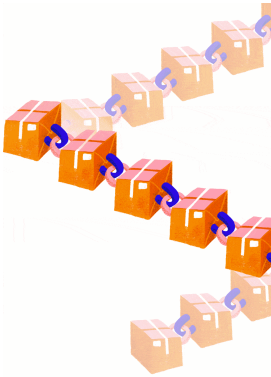
Wie weiter oben bereits erklärt kennt Bitcoin keinen klassischen Kontostand, sondern nur eingehende Transaktionen, die noch nicht verbraucht worden sind (UTXOs). Trotzdem ist ein schneller Überblick über die aktuellen Finanzen wünschenswert, auch wenn diese über viele Adressen verteilt sind.

Glücklicherweise bietet eine Bitcoin-Wallet genau diese Funktion. Sie überwacht im Hintergrund die ein- und ausgehenden Transaktionen und prüft, ob dort eine der eigenen Adressen vorkommt. Der Kontostand kann dann sehr einfach ausgerechnet werden: die Summe der eingehenden Transaktionen minus die Summe der ausgehenden Transaktionen.

Dieses Vorgehen ist dafür verantwortlich, dass man Blockchain-Technologien oft als *Decentralized Ledger (verteiltes Kassenbuch)* bezeichnet. Die Aufgabe eines Kassenbuchs ist für die Bilanzierung in Unternehmen wichtig: die Kontostände müssen sich aus den kumulierten ein- und ausgehenden Transaktionen ergeben. Diese Bilanz wird über alle Aktiva gebildet, so dass interne Überweisungen den Gesamtstand nicht verändern.

Der Unterschied einer Blockchain im Vergleich zum „altmodischen" Kassenbuch besteht in der Dezentralisierung. Statt einem einzigen Kassenbuch gibt es eine globale Liste, in dem sämtliche Transaktionen verzeichnet sind und von der jeder Node eine Kopie hat.

3.4 Blöcke



Da Bitcoin ein Zahlungsnetzwerk ist, muss sichergestellt werden, dass sich alle Beteiligten darüber einig sind, welche Transaktionen stattgefunden haben. Der entscheidende Unterschied zum klassischen Bankensystem ist, dass man aus Sicht des Gesamtsystems den beteiligten Computern – den *Knoten* (*Nodes*) – nicht vertrauen kann. Man muss davon ausgehen, dass jeder einzelne im Zweifelsfall das System so zu manipulieren versucht, dass er selbst einen Vorteil erhält. Eine solche Problemstellung bezeich-

net man als *byzantinisch*, nach einem wissenschaftlichen Artikel,[1] der sie anhand eines fiktiven Beispiels eines gemeinsames Feldzugs byzantinischer Generäle erläutert. Die Generäle können sich weder darauf verlassen, dass keiner den anderen verrät, noch den Nachrichten vertrauen, die sie erhalten, weil diese mit bösartiger Absicht auf dem Weg von Feinden verfälscht worden sein könnten.

Bitcoin löst dieses Problem auf Basis eines probabilistischen Verfahrens: Es ist nie zu 100% garantiert, dass man die „Wahrheit" kennt, aber man kann sich im Laufe der Zeit immer sicherer sein. In anderen Worten: Je mehr Zeit seit einer Transaktion vergangen ist, desto schwieriger wird es, diese zu fälschen, löschen oder anderweitig zu manipulieren.

Mining

Um das zu erreichen, bündeln Knoten mehrere Transaktionen zu einem *Block*. Diese Blöcke werden durch das *Proof-of-Work*-Verfahren (*PoW*) abgesichert. Das Verfahren ist mit Absicht sehr rechenintensiv und wird von den sogenannten *Miner*-Knoten durchgeführt. Die Grundidee dabei ist, dass ein Algorithmus mit Eingabedaten gefüttert wird, die sich aus den Transaktionen, einem Verweis auf den vorhergehenden Block, ein paar weiteren Metadaten und zusätzlich einem frei wählbaren Wert zusammensetzen. Da es sich beim verwendeten Verfahren um einen Hashing-Algorithmus handelt, ist das Ergebnis aus den Eingabedaten nicht vorherzusehen, es sei denn, man führt den Algorithmus aus. Das wiederum bedeutet, das der einzige Weg, ein gewünschtes Ergebnis zu erzielen, die *Brute-Force*-Methode ist, also aggressives Ausprobieren aller Möglichkeiten. Legt man nun nicht einen bestimmten Wert, sondern eine Menge von Werten fest, die als gültiges Ergebnis gelten, kann man bestimmen, wie schwierig die zu lösende Aufgabe ist. Bei Bitcoin ist dieser als *Difficulty Target* bezeichnete Parameter eine Anzahl von Nullen, mit denen das Ergebnis beginnen muss. Mit anderen Worten: Die Miner probieren solange andere Werte für das eine frei wählbare Feld im Block aus, bis sie ein Ergebnis bekommen, das mit der nötigen Menge führender Nullen beginnt. Dabei liefern sie sich ein Wettrennen mit allen anderen Miner-Knoten, die das im gleichen Zeitraum ebenfalls versuchen, bis einer von ihnen einen Block gefunden hat und diesem im Peer-to-Peer-Netzwerk verbreitet.

[1] Leslie Lamport, Robert Shostak, Marshall Pease: The Byzantine Generals Problem

Ein Miner, der einen solchen Block empfängt, prüft, ob dieser den gleichen Block als Vorgänger hat wie der Blockkandidat, für den er selbst gerade nach einer Lösung sucht. Ist dies der Fall, gibt er auf und bildet einen neuen Kandidaten mit dem neuen Block als Vorgänger (und ohne die darin enthaltenen Transaktionen). Warum? Könnte er nicht genau so gut den Block verwerfen, den er empfangen hat, und stattdessen einfach hoffen, schnell genug selbst einen Block zu finden, um diesen zum Teil der offiziellen Blockchain werden zu lassen? Könnte er und in der Praxis geschieht das sogar häufig, wenn auch meistens nicht mit Absicht, sondern weil schlicht nicht auszuschließen ist, dass ein bereits gefundener Block einen Miner, der ebenfalls einen gültigen Block findet, noch nicht erreicht hat. In solchen Fällen spricht man von einer Gabelung (*Fork*) in der Blockchain. Allerdings konvergiert der so entstandene Baum binnen kurzer Zeit wieder zu einer Kette, weil sich einer der Zweige gegenüber den anderen durchsetzt. Ein Block, der bereits zwei oder drei Schritte „voraus" ist, signalisiert einem Miner, dass er eventuell auf einem Zweig arbeitet, der kaum eine Chance hat, sich noch durchzusetzen, denn alle arbeiten nach demselben Prinzip: Die längste Kette gewinnt. Die Seitenstränge, die sich der Kette abgegabelt haben, werden verworfen.

Ressourcenverbrauch

Warum nun das aufwändige Rätselraten? Man könnte schließlich argumentieren, dass dadurch massiv Ressourcen verschwendet werden, weil all diejenigen Miner, die das Rennen nicht gewinnen, ihre Rechenleistung (und die dafür benötigte Energie und Hardwarekapazität) vergeblich eingesetzt haben.

Insbesondere Bitcoin steht massiv in der Kritik, weil sich der Energiebedarf der Bitcoin-Blockchain durch das PoW-Verfahren mittlerweile (Stand: Oktober 2019) dem Gesamtenergiebedarf der Niederlande nähert.

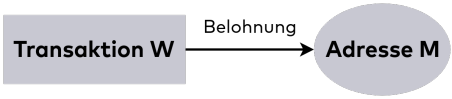
Der Grund dafür ist, dass das Produzieren eines Blocks unbedingt aufwändig sein muss, damit diese Sorte Blockchain funktioniert. Würde ein Miner versuchen, eine alternative Version der Realität zu erzeugen, z.B. durch Entfernen einer Transaktion aus der Historie, um Geld ein zweites Mal ausgeben zu können, müsste er eine alternative Version der Blockchain erzeugen. Weil die Blöcke aber verkettet sind, wird dieses

Fälschen um so schwieriger, je mehr Blöcke nach der bereits im System gespeicherten Transaktion, die man löschen oder verändern möchte, bereits entstanden sind. Das ist gerade der Effekt des byzantinischen Konsensverfahren.

Bei Bitcoin dauert das Finden eines Blocks ungefähr zehn Minuten. Nach sechs Blöcken, also nach etwa einer Stunde, kann man eine Transaktion als genügend abgesichert betrachten, dass sie nicht mehr gefälscht werden kann. Wenn jemand z.B. per Bitcoin online ein Auto bezahlt, könnte man nach einer Stunde davon ausgehen, dass man das Fahrzeug samt Brief und Schlüssel problemlos ausliefern kann. Eine Stunde mag dabei lang klingen, aber das ist ein Irrglaube: Im Gegensatz zu einer Kreditkarten-Transaktion gibt es keine Möglichkeit, Widerspruch einzulegen; als Empfänger*in des Geldes kann man sich also sehr sicher sein, dass der Transfer erfolgreich war.

3.5 Wertschöpfung

Kommen wir nun endlich zum Thema der Wertschöpfung: Wie gelangen Bitcoin in den Umlauf?



Coinbase-Transaktion: der Miner (M) überweist sich selbst die Belohnung

Jeder Block darf genau eine sogenannte *Coinbase*-Transaktion enthalten. Angenommen, es gibt eine bestimmte Menge an Transaktionen, die derzeit noch nicht in einem Block enthalten sind. Ein Miner mit der Adresse, der gerade auf der Suche nach einem Block ist, stellt dieser Transaktionsmenge eine eigene Transaktion ohne Quellen voran. Sie erzeugt Bitcoin in der Höhe der aktuellen Block-Belohnung für Miner (Stand Oktober 2019: 12,5 Bitcoin). Im Regelfall wird der Miner die eigene Adresse als Senke der Coinbase-Transaktion definieren.

Der entscheidende Punkt beim PoW-Verfahren ist, dass die Verschwendung – oder positiv formuliert: Investition – in die Rechenleistung und die damit verbundene

Energie es schwierig, und noch wichtiger, nicht lohnenswert macht, die Inhalte der Blockchain zu fälschen. Man müsste enorme Rechenleistung erbringen, um mit der gefälschten Historie gegen die Miner-Knoten, die sich an die Regeln halten, gewinnen zu können. Das wiederum lohnt sich nicht, denn die Miner erhalten durch die Coinbase-Transaktion einen Anreiz.

Unabhängig von der Wertschöpfung durch das Mining von Blöcken ist es üblich, in Transaktionen einen bestimmten, kleinen Betrag vom Wechselgeld abzuziehen, der dann als „Trinkgeld“ an den Miner fließt, der diese Transaktion in einen Block einschließt. Dadurch erreicht man eine natürliche Regulierung der Transaktionsmenge: Je mehr Transaktionen noch offen sind, desto höher müssen die Trinkgelder sein, denn Miner bevorzugen logischerweise die lukrativeren Transaktionen.

4 Ethereum

Genau wie Bitcoin ist *Ethereum* eine öffentliche Blockchain mit Kryptowährung. Konzeptuell funktioniert vieles ähnlich, wie z.B. die Adressen und das Mining. Doch es gibt einen gewichtigen Unterschied: Ethereum bringt die Möglichkeit mit, (fast) beliebigen Code als Teil von Transaktionen auszuführen. Diese sogenannten *Smart Contracts* sind genauso manipulationssicher wie die eigentliche Währung, wie die Ethereum-Webseite verkündet:

> **"Ethereum is a decentralized platform that runs smart contracts: applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third-party interference."**

In diesem Kapitel möchten wir Ethereum gegenüber Bitcoin abgrenzen und erläutern, wie Transaktionen in Ethereum funktionieren. Doch kurz vorab: bei Ethereum gibt es begrifflich einen Unterschied zwischen der Blockchain und der eigentlichen Kryptowährung, worauf wir am Ende noch eingehen werden.

4.1 Transaktionen

Wir erinnern uns: In Bitcoin können Transaktionen mehrere Quellen und Senken haben. Als Quellen sind bisher unverbrauchte Senken zulässig.

Ethereum hingegen funktioniert viel eher wie eine klassische Bank. Sogenannte *Externally Owned Accounts* (*EOAs*) sind von einem Schlüsselpaar gedeckt. Transaktionen haben genau ein Zielkonto und spezifizieren optional *Betrag* (*Value*) und *Daten*. Fürs erste können wir die Daten ignorieren. Der Betrag gibt die Menge an *Ether* an – die Währung von Ethereum –, die das Zielkonto erhalten soll. Das Quellkonto ergibt sich aus der Signatur der Transaktion.

Ethereum unterscheidet mehrere Arten von Transaktionen, je nachdem, ob Daten oder Betrag vorhanden sind:

- Eine Transaktion, die einen Betrag enthält, wird *Zahlung* genannt.

- Eine Transaktion, die Daten enthält, wird hingegen *Aufruf* genannt.

Beides lässt sich kombinieren. Transaktionen, denen beides fehlt, sind zwar möglich, aber nutzlos.

Der Begriff „Aufruf“ deutet schon an, dass Ethereum in irgendeiner Form Routinen unterstützt.

4.2 Verträge

Neben EOAs gibt es in Ethereum auch Vertragskonten, für die sich der Begriff *Smart Contract* etabliert hat. Verträge werden im Gegensatz zu EOAs nicht von einem Schlüsselpaar repräsentiert; stattdessen werden sie von einem anderen Konto aus erzeugt.

Ähnlich wie ein EOA kann auch ein Vertrag über einen Kontostand (*Balance*) verfügen. Darüber hinaus enthält ein Vertrag auch Programmcode sowie quasi beliebigen Zustand.

Übertragen auf Java entspricht ein Contract einer Instanz einer Klasse mit öffentlichen und privaten Methoden. Von einer Klasse kann es mehrere Instanzen geben und auch ein Vererbungskonzept existiert. Üblicherweise schreibt man Klassen nicht direkt in JVM-Bytecode, sondern benutzt einen Compiler, der Hochsprache in Bytecode übersetzt. In Ethereum heißt die populärste Sprache *Solidity*, die aus einer objektorientierten Syntax in den Bytecode der *Ethereum Virtual Machine* (*EVM*) übersetzt.

Verträge werden niemals von selbst tätig, sondern können nur auf eingehende Aufrufe reagieren und dabei auch mit anderen Verträgen interagieren oder Beträge an EOAs versenden.

Trotz Ethereums Natur als verteiltes System sind Verträge nie nebenläufig: Wenn eine Transaktion einen Vertrag erzeugt oder aufruft, werden alle Programmschritte sequenziell ausgeführt. Wichtig ist, dass alle diese Verarbeitungsschritte von einer einzigen Transaktion ausgehen: ein Vertragsaufruf, der zu weiteren Vertragsaufrufen führt, erzeugt dabei keine neuen Transaktionen. Eine Transaktion kann immer nur von einem EOA ausgehen.

Erzeugt wird ein Vertrag, in dem man eine Transaktion generiert, die in den Daten den Bytecode enthält und als Empfängeradresse die spezielle Adresse `0x0` definiert hat.

Ein einmal definierter Vertrag ist unveränderlich und kann prinzipiell für alle Ewigkeit von beliebigen Transaktionen aufgerufen werden.

Ein Ethereum-Vertrag ist ähnlich zu einer Kapitalgesellschaft im Wirtschaftsrecht: Er existiert unabhängig von Personen (EOAs) und kann selbst Verträge aufrufen und aufgerufen werden sowie Geld versenden und empfangen. Wenn das Subjekt, das den Vertrag erzeugt hat, die Kontrolle über seinen privaten Schlüssel verloren hat, existiert der Vertrag trotzdem weiter. Die einzige Möglichkeit, einen Vertrag wieder aufzulösen, besteht darin, dass der Vertrag die SELFDESTRUCT-Operation ausführt. Diese Möglichkeit muss der Vertrag selbst vorgesehen haben; der Erzeuger darf nicht „einfach so" seinen Vertrag auflösen.

4.3 Währung

Genau wie in Bitcoin gibt es auch in Ethereum eine eingebaute Währung: *Ether*. Über diese Währung werden Transfers und Gebühren abgewickelt. Mittels spezieller Verträge können bei Ethereum aber auch virtuelle Währungen angelegt werden (sogenannte *Tokens*). Für eine Einbindung dieser Tokens in Wallets und andere Software gibt es gar einen Standard. Stand Oktober 2019 gibt es im öffentlichen Ethereum-Netzwerk bereits weit über 200.000 solcher Token-Verträge. Davon haben 23 eine Marktkapitalisierung von über 100 Millionen US-Dollar.

Dank der Möglichkeiten der Smart Contracts ist Ethereum nicht nur eine Kryptowährung, sondern auch eine *Plattform* für Kryptowährungen.

4.4 Programmierung

Die Lingua Franca von Ethereum ist die Ethereum Virtual Machine. Wie der Name bereits andeutet, handelt es sich dabei um ein Low-Level-Bytecode-Format. Ein Vertrag könnte daher in etwa so aussehen:

```
[1] PUSH1 0x80
[3] PUSH1 0x40
[4] MSTORE
[5] CALLVALUE
[6] DUP1
[7] ISZERO
[10] PUSH2 0x0010
...
```

Die Struktur erinnert sehr deutlich an Assembler, der auch auf einer realen CPU laufen könnte.

Solidity

Natürlich möchte man Verträge nicht auf solch einer niedrigen Ebene programmieren. Deswegen gibt es im Ethereum-Ökosystem einige Hochsprachen, die eher an moderne Programmiersprachen erinnern. Die gängigste heißt *Solidity*, ist an JavaScript-Syntax angelehnt und objektorientiert. Ein Solidity-Contract ist deutlich leichter zu lesen als obiges Bytecode-Pendant:

```
pragma solidity >=0.4.22 <0.6;

contract Wallet {
    uint256 balance;
    address payable owner;
    constructor () public {
        balance = 0;
        owner = msg.sender;
    }
    function addfund() payable public returns (uint256) {
        require (msg.sender == owner);
        balance += msg.value;
        return balance;
```

```
    }
    // ...
}
```

Dieser Vertrag definiert eine Art *Online-Wallet*. Alice könnte diesen Vertrag anlegen und anschließend Ether überweisen. Diese Möglichkeit bleibt Bob verwehrt, weil er sich dem Vertrag gegenüber nicht als Eigentümer authentifizieren kann.

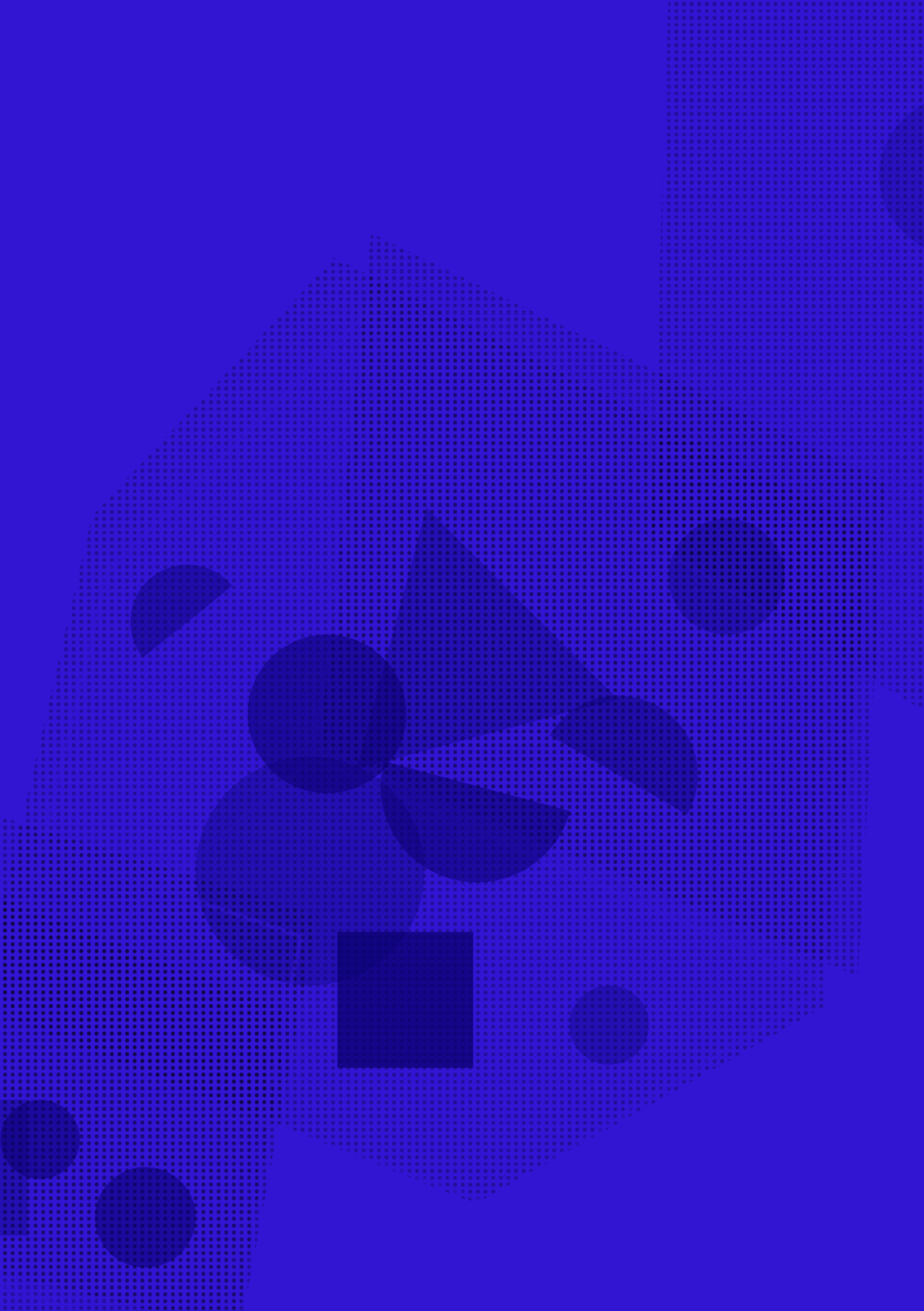
Testing

Gutes Software Engineering erfordert das systematische Testen von Code. Auf Smart Contracts trifft das sogar noch stärker zu: Ein Vertrag, der einmal angelegt worden ist (und vielleicht sogar schon Geld verwaltet), kann nachträglich nicht geändert werden.

Für Solidity gibt es verschiedene Frameworks und Testumgebungen, mit denen Verträge auf Herz und Nieren geprüft werden können.

Doch die Frage stellt sich: Wie kann man Bugs verhindern? In der vergleichsweise kurzen Geschichte von Ethereum gab es schon verschiedene Vorfälle von Diebstahl oder Totalverlust von hohen Werten. Ein besonders schwerer Programmierfehler führte 2017 dazu, dass Ether im Gegenwert von 360 Millionen USD unwiederbringlich zerstört worden sind.

Gegenstand aktiver Forschung sind daher neue, modernere Hochsprachen für Ethereum, mit denen sich Smart Contracts mit an Sicherheit grenzender Wahrscheinlichkeit rigoros analysieren und formal verifizieren lassen. Einerseits macht das die Entwicklung von Verträgen komplizierter, andererseits erscheint es angesichts der potenziellen Schäden wie eine sinnvolle Investition.



5 Anwendungsfälle

Wenn wir heute ein Produkt kaufen oder einen Service in Anspruch nehmen wollen, suchen wir nach Sicherheiten. Oft behelfen wir uns mit Bewertungen und Ratings. Doch Fünf-Sterne-Bewertungen können uns zwar ein sicheres Gefühl geben, aber eben auch leicht gefälscht werden. Ebenso wie viele andere Daten und Werte, vor allem im Internet.

Hier kommt die Blockchain ins Spiel: Durch die Verkettung der Transaktion wird Manipulation schier unmöglich. So kann beispielsweise, wie mit Bitcoin, eine nahezu fälschungssichere weltweite Währung erschaffen werden.

Doch die Blockchain funktioniert auch im wesentlich kleineren Rahmen, z.B. bei der zweifelsfreien Nachverfolgung von Waren oder eben bei Smart Contracts zwischen einander unbekannten Partnern.

Auf den nächsten Seiten haben wir ein paar Anwendungsfälle zusammengestellt, für die die Blockchain eine technologische Grundlage bilden kann.

5.1 Gemeinnützige Organisationen

„Wir sind eine gemeinnützige Organisation. Wie beweise ich, dass die Spenden auch wirklich ankommen?“

Um nachzuweisen, dass eine Spendenorganisation mit den ihr anvertrauten Geldern sorgfältig und verantwortungsvoll umgeht, kann der gesamte Geldfluss und der sinnvolle Einsatz der Spenden in der Blockchain transparent nachgewiesen werden. Je nach Organisation kann auch der Spendenbedarf offengelegt werden, z.B. die Anträge kleiner gemeinnütziger Vereine auf bestimmte Geldsummen.

Gerade der Vorwurf, dass Spendengelder durch einen zu großen „Wasserkopf“ verschwendet werden oder überhaupt nicht bei den Hilfsbedürftigen ankommen, kann durch eine Blockchain wirkungsvoll entkräftet werden. Es gibt allerdings bereits anerkannte Spenden-Siegel (z.B. DZI), die von den Organisationen das Offenlegen genau dieser Daten verlangen und großes Vertrauen genießen.

5.2 Ausbildung

„Ich biete Schulungen für Software-Architekt*innen an. Wie werden meine Zertifikate fälschungssicher?"

Die Ausbildung zur zertifizierten Software-Architekt*in umfasst das Advanced Level und das Foundation Level, das durch die Absolvierung verschiedener Schulungsmodule erreicht werden kann. Da die Teilnehmer*innen an den iSAQB-Schulungen für das Foundation Level und das Advanced Level nicht an einzelne Schulungsunternehmen gebunden sind, bekommen sie eine Identifikationsnummer, die sicher in der Blockchain gespeichert wird. Genauso wie die Teilnahmebestätigung an jeder Schulung und die damit erreichten Credit-Points. Die bisher nur in Papierform ausgestellten Zertifikate können nun fälschungssicher nachgewiesen werden.

Bisher gibt es kein einheitliches System, welches die abgeschlossenen Module erfasst. Hinzu kommt, dass die zertifizierenden Unternehmen in den meisten Fällen direkte Konkurrenten sind. Das Vertrauen ist also eher gering, die Vorteile einer Blockchain in diesem Fall umso größer. Auch weil der rechtliche Rahmen feststeht, da bereits alle zertifizierenden Unternehmen Mitglied im iSAQB-Verein sind.

5.3 Eigentum

„Ich möchte ein gebrauchtes Auto kaufen. Wer garantiert mir, dass der Kilometerstand stimmt?"

Das Fahrzeug bekommt eine – von einem Gutachter geprüfte – digitale Identität mit allen wichtigen Daten, beispielsweise Kilometerstand, Vorbesitzer oder Unfallstatistik. Diese Identität wird manipulationssicher in einer Blockchain gespeichert. Jetzt kann ein Smart Contract erstellt werden, der den gesamten Kauf- und Ummeldeprozess automatisiert übernimmt. Dafür werden noch die Identitäten von Käufer und Verkäufer benötigt sowie Kfz-Papiere.

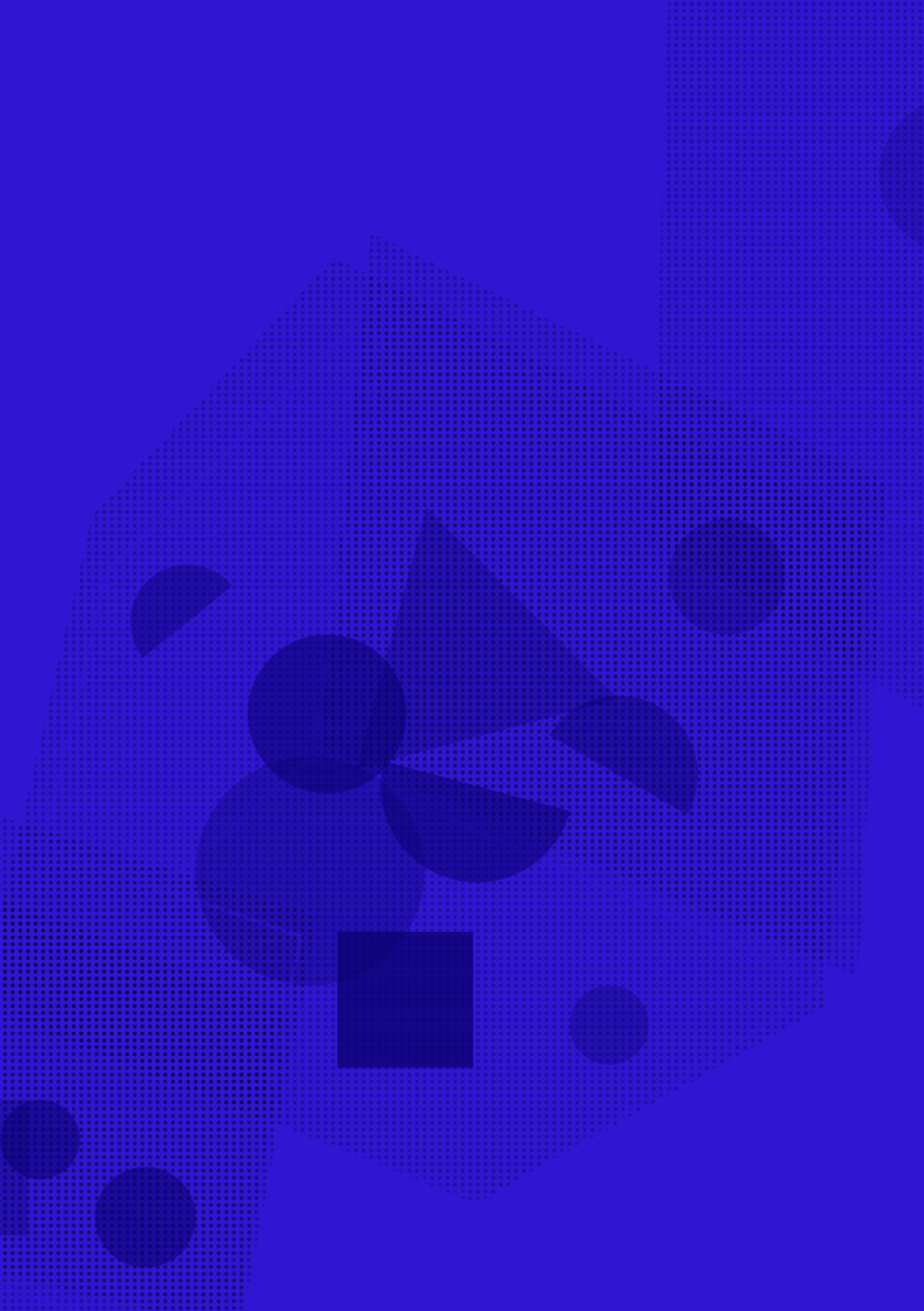
Gerade der Gebrauchtwagenkauf ist ein Musterbeispiel für einen sinnvollen Einsatz der Blockchain: Kilometerstände werden manipuliert, Unfälle verschwiegen. Als Laie hat man bisher kaum eine Chance, Manipulationen zu erkennen. Und als Verkäufer will man sich von den schwarzen Schafen der Branche absetzen.

5.4 Handel

„In meinen Filialen wird häufig bar bezahlt. Wie beweise ich dem Finanzamt, dass alles ordnungsgemäß abgerechnet wird?"

Der Gesetzgeber schafft die notwendigen Rahmenbedingungen für eine Regulierung, z.B. eine Quittungspflicht. Dann müssen alle Transaktionen umgehend digitalisiert und in kurzen Abständen in eine Blockchain eingespeist werden: z.B. Einzel- und Tagesabrechnungen direkt über die Kassensysteme, Wochen- und Monatsabrechnungen über die Buchhaltung. Des weiteren kann die Berechnung der monatlichen oder quartalsweisen Umsatzsteuerschuld mit Hilfe eines Smart Contracts automatisiert werden.

Restaurants und Einzelhändlern wird oft vorgeworfen, zu wenig Steuern zu zahlen, weil nicht alle Barbeträge ordnungsgemäß dem Finanzamt gemeldet werden bzw. weil Kassensysteme manipuliert werden. Eine Blockchain kann zwar Abhilfe verschaffen, ist aber anfällig für Fehler, z.B. wenn Transaktionen gar nicht erfasst werden oder wenn Kassensysteme beispielsweise durch eine schlechte Internetanbindung ausfallen.



6 Ausblick

In letzter Zeit kam es zu einer regelrechten kambrischen Explosion von Technologien, die Bitcoin mehr oder weniger ähneln. Manche verabschieden sich gar von einigen fundamentalen Konzepten der Blockchain, lösen aber ähnliche Probleme. Aus dieser Strömung ist oft der Begriff *Distributed Ledger Technology* zu hören; sozusagen die Essenz der Blockchain.

Zum Abschluss möchten wir noch einige dieser Konzepte in Kürze vorstellen.

6.1 Mining oder nicht?

Das PoW-Verfahren, welches zum Mining von Bitcoin, Ethereum und vielen anderen Kryptowährungen eingesetzt wird, ist heftig umstritten. Zum einen liegt das am erheblichen Energiebedarf, der – je nach Fraktion – als „Versündigung an folgenden Generationen“ oder als „erheblich sinnvoller als das bestehende Finanzsystem und das für weniger Energie als die jährliche Weihnachtsbeleuchtung“ betrachtet wird. Die Lösungen lassen sich in unterschiedliche Kategorien einsortieren.

Zunächst gibt es dabei die Kategorie, die das Problem adressiert, indem sie einige der Grundanforderungen in Frage stellt: Vor allem die Dezentralisierung, die Offenheit gegenüber beliebigen Teilnehmern (Knoten) und den Grad des Vertrauens, das man ihnen entgegenbringt. Lässt man die Anforderung nach Dezentralisierung komplett weg, kann man sich auf einen einzigen zentralen Leistungserbringer beschränken und ist damit bei einer klassischen Lösung, die mit Blockchain oder Distributed Ledger-Ansätzen gar nichts zu tun haben braucht. Auch wenn man die Teilnehmer auf wenige bekannte Knoten beschränken kann, denen man vollständig vertraut, ist die Lösung gradlinig. Es handelt sich dann schlicht um eine verteilte Datenbank. Auch in diesem Fall ist der Begriff „Blockchain“ bestenfalls als Marketing-Gimmick zu verstehen.

Proof-of-Authority

Interessanter sind Szenarien, bei denen die Menge der Teilnehmer beschränkt ist, aber keiner von ihnen Vertrauen genießt. Solche Umgebungen können mit byzantinischen

Protokollen oder optimierten Verfahren unterstützt werden, die oft eine bessere Skalierbarkeit bzw. eine geringere Komplexität versprechen. Ein prominentes Beispiele dafür ist *Ripple,* das den Konsens in die Verantwortung besonders privilegierter Knoten legt, die nicht einfach jeder betreiben kann, sondern die bewusst ausgewählt werden. Diejenigen, die dazu berechtigt sind, beweisen dies durch ein geeignetes Verfahren wie z.B. ein Zertifikat. In diesem Fall spricht man von einem *Proof-of-Authority*-Ansatz (*PoA*).

Ein Hinweis allerdings: Bei solchen Systemen ist die Dezentralisierung, also die Verteilung der Verantwortung auf eine Menge wirklich unabhängiger Instanzen, oft erst für die Zukunft geplant; wird das auf dezentralen Betrieb ausgerichtete System jedoch zentral betrieben, kann man die berechtigte Frage stellen, ob nicht auch eine zentrale Lösung ausgereicht hätte.

Proof-of-Stake

Eine weitere Variante ist *Proof-of-Stake* (*PoS*). Dieser liegt die Annahme zugrunde, dass niemand Interesse daran hat, ein System zu destabilisieren, auf dessen korrekter Funktion sie selbst angewiesen ist. Während beim PoW-Verfahren ein Miner „beweist", dass er bereit war, eine Menge Energie zu investieren, würde in einem Währungssystem, das auf PoS aufsetzt, stattdessen ein bestimmter Betrag der Währung im System selbst eingesetzt. Das kann direkt oder indirekt geschehen: Bei der direkten Variante „setzen" ein oder mehrere Partner einen Betrag ein, indem sie ihn während der Validierung sperren und nicht anderweitig ausgeben können. Bei der indirekten Variante werden der oder die Partner, die über valide Blöcke entscheiden, mit einer Wahrscheinlichkeit ausgewählt, die im Verhältnis zu der Größe ihres Anteils (ihres Besitzes) steht.

Der große Vorteil des PoS-Verfahrens gegenüber PoW ist, dass es zwar für den gleichen Anwendungsfall geeignet ist – öffentliche, für jeden zugängliche Blockchains –, aber ohne den massiven Energiebedarf auskommt. Aus diesem Grund hat sich z.B. Ethereum schon sehr früh darauf festgelegt, das PoW- durch ein PoS-Verfahren abzulösen (ein Prozess, der aktuell mit einer hybriden Lösung vorbereitet wird). Der Nachteil ist zum einen, dass diejenigen, die innerhalb des Systems über viel „Reichtum" verfügen, nicht außerhalb, sondern nur innerhalb des Systems motiviert sind, sich an die Regeln zu halten, zum anderen die schlichte Tatsache, dass ein solches System bislang noch in

keiner öffentlichen Blockchain-Implementierung verwendet wird und die Reife daher bezweifelt werden muss.

6.2 Anonym oder nicht?

Das Bitcoin eine vollständig anonyme Währung ist, ist nur eine Legende. Tatsächlich ist sie nur *pseudonym*, denn Geldflüsse oder manchmal gar Zusammenhänge mit realen Personen lassen sich mit genügend Rechenpower aus den Daten nachvollziehen. Eine echte anonyme Währung muss aber verhindern, dass Dritte solche Zusammenhänge rekonstruieren können.

Monero (und einige andere Kryptowährungen) schicken sich an, dieses Problem zu lösen. Die Basis dafür bilden sogenannte Zero-Knowledge-Protokolle: Ein Beweiser möchte einer einer Prüferin beweisen, dass er über bestimmtes Wissen verfügt, aber keine Informationen verraten, die darüber hinaus Rückschluss auf das geheime Wissen zulassen. Im Falle von Kryptowährungen besteht das darin, dass ein Subjekt allen anderen Subjekten nachweisen können muss, eine bestimmte Transaktionen durchführen zu dürfen, ohne Informationen über diese Transaktion offenzulegen.

Die dafür nötige Kryptografie ist sehr komplex und basiert darauf, zwei verschiedene Schlüsselpaare zu benutzen. Das eine Schlüsselpaar ist ein Sichtschlüssel, das andere ein Ausgabenschlüssel. Der Sichtschlüssel wird dazu benutzt, um Transaktionen in der Blockchain zu erkennen, die an einen selbst gerichtet sind. Im Gegensatz dazu wird der Ausgabeschlüssel benutzt, um ausgehende Transaktionen zu signieren. Ferner werden für die Senken von Transaktionen Einmaladressen gebildet.

Auch die eigentlichen Beträge sind geheim. Die besondere Herausforderung hierbei ist es, dass man bei einer Transaktion nachweisen können muss, dass die Summe der Senken kleiner oder gleich der Summe der Quellen ist. Übrig bleibt wie bei Bitcoin eine Gebühr für Miner. Für den Beweis der Summengleichheit sind weitere kryptografische Mechanismen notwendig. Monero schreibt wegen der Komplexität der kryptografischen Operationen eine Mindestgebühr per Größeneinheit von Transaktionen vor.

Auf Grund der Anonymität von Monero wird es gerne für dubiose Geschäfte benutzt. Ob und wie diese Anonymität aufgehoben werden kann und ob dies in der Zukunft rückwirkend geschehen kann, ist aktueller Forschungsgegenstand.

6.3 Öffentlich oder nicht?

Eigentlich führt die *private Blockchain* das Blockchain-Prinzip ad absurdum. Denn es braucht wieder eine Autorität, die darüber bestimmt, wer an dieser Blockchain teilnehmen kann. Es lässt sich also trefflich darüber streiten, ob ein Netzwerk mit zentral Verantwortlichen diesen Namen tragen darf. Doch rein technisch gesehen ist die private Variante eine Blockchain und hat durchaus ihre Daseinsberechtigung, vor allem für Unternehmen und Gruppen, die eine geschützte Umgebung benötigen.

Dieser Markt von privaten (oder *permissioned*) Blockchains ist hart umkämpft. Die zwei Platzhirsche, *Corda* und *Hyperledger Fabric* möchten wir allerdings nicht unerwähnt lassen. Beide unterscheiden sich sehr stark und haben auch verschiedene Anwendungsgebiete.

Corda

Corda ist eine eher untypische Implementierung einer Blockchain, zumal es gar keine Blöcke gibt. Stattdessen handelt es sich eher um eine Distributed-Ledger-Lösung. Das liegt daran, dass Corda sogenannte *Shared Facts* verwaltet, die an einen bestimmten Kreis von Knoten verteilt werden.

Verträge können in einer beliebigen JVM-Sprache implementiert werden. Diese werden auf den einzelnen Knoten ausgeführt und von einem zentralen *Notary Service* validiert. Verträgen können Fakten verändern, wodurch der alte Zustand des Fakts historisiert ist und ein neuer Zustand entsteht. Damit entsprchen Fakten den UTXOs aus Bitcoin. Im Unterschied zu Bitcoin müssen allerdings alle beteiligten Vertragsparteien der Evolution eines Faktes zustimmen. Ist das geschehen, so wird der neue Fakt unmittelbar gültig, ohne in einen Block aufgenommen zu werden.

Hyperledger Fabric

Fabric entspricht folgt eher dem klassischen Blockchain-Konzept kombiniert mit PoA. Für die Nachrichtenverteilung im System wird Kafka benutzt; eine Standardkomponente, die mittlerweile in vielen verteilten Systemen zum Einsatz kommt. Das Datenmodell von Fabric ist eine Datenbank von Key-Value-Paaren, die von *Chaincode* – Fabrics Bezeichnung für Smart Contracts – gelesen und geschrieben werden. Chaincodes

laufen in Docker-Containern, daher ist man in der Wahl der Implementierungssprache flexibel.

Ruft ein berechtigter Client einen Chaincode auf, werden die gelesenen und geschriebenen Keys protokolliert. Mehrere Chaincodes dürfen parallel ablaufen. Ein spezieller Knoten ist dafür zuständig, die Schreibzugriffe in die korrekte Reihenfolge zu bringen und in Blöcken zusammenzufassen. Transaktionen, die einen Konflikt hervorrufen (z.B. zwei widersprüchliche Schreiboperationen) werden zurückgewiesen und müssen vom Client neu gestartet werden.

7 Fazit

Die schier unüberschaubare Menge an Blockchain-Lösungen macht es für Entwickler*innen schwierig, zu entscheiden, ob und wenn ja welche Blockchain-Variante passen könnte. Anwender*innen, denen eine Lösung präsentiert wird, fällt es noch schwerer, zu entscheiden, ob sie gerade innovative Technologie oder nur heiße Luft vor sich haben.

Als ersten Filter sollte man prüfen, ob für den bestehenden Anwendungsfall eine Blockchain überhaupt sinnvoll sein kann.

In vielen Fällen reicht eine klassische, datenbankzentrierte, zentrale Lösung völlig aus. Handelt es sich um eine verteilte Lösung, die nur ausgewählten Teilnehmern Zugang gewährt, kann ein klassisches Konsens-Verfahren wie z.B. PBFT (Practical Byzantine Fault Tolerance) in Kombination mit Proof-of-Authority gewählt werden. Vorsicht ist geboten, wenn die Lösung dabei ausgerechnet für den Konsens, ein Informatikproblem, das seit Jahrzehnten Material für Dissertationen liefert, eine selbst gestrickte Lösung liefert. Die Mindestanforderung wäre, dass der verwendete Algorithmus in einem wissenschaftlichen Paper beschrieben wurde und bereits signifikantes Feedback dazu existiert.

Für öffentliche Blockchains gilt: Trotz der sehr bitteren Pille des Energieverbrauchs ist Proof-of-Work zum aktuellen Zeitpunkt die einzige ausgereifte, erprobte und im Echtbetrieb gehärtete Variante.

Alternativen wie zum Beispiel Proof-of-Stake sind zwar sehr interessant, sollten aber aktuell noch mit Vorsicht eingesetzt werden. Eine gute Zwischenlösung kann sein, eine öffentliche, durch PoW abgesicherte Blockchain wie Ethereum oder Bitcoin nicht für jede Transaktion zu verwenden, sondern immer nur an bestimmten Kontrollpunkten wie z.B. nach jeder tausendsten Transaktion. Dies hat entsprechende Auswirkungen auf den Grad, zu dem man dem Gesamtsystem vertrauen kann.

Wenig überraschend ist die Blockchain nur ein Mittel zum Zweck, das auf vielfältige Arten eingesetzt werden kann.

Entscheidend ist, dass der gewählte Lösungsansatz zum Problem passt.

Quellen

- https://finanzblog.lgt.com/kerbholz/
- https://medium.com/menapay/from-sumerian-tokens-to-distributed-ledger-19b1216243d9
- https://medium.com/@COTInetwork/ledgers-over-the-years-from-ancient-egypt-to-blockchain-dagand-beyond-47924175cb97
- https://en.wikipedia.org/wiki/History_of_accounting
- https://www.vice.com/de/article/j5nzx4/aelteste-blockchain-der-welt-new-york-times
- https://bitcoinbook.info/
- https://ethereum.org/
- https://github.com/ethereumbook/ethereumbook/blob/develop/06transactions.asciidoc
- https://github.com/ethereumbook/ethereumbook/blob/develop/04keys-addresses.asciidoc
- https://solidity.readthedocs.io/en/latest/
- https://etherscan.io/tokens
- https://www.corda.net/
- https://www.hyperledger.org/

Noch mehr Blockchain …

Wenn Sie jetzt noch viele Fragen haben und tiefer ins Thema einsteigen wollen, sind unsere Workshops vielleicht etwas für Sie:

Im **Blockchain Reality Check** erfahren Sie, ob Blockchain zu Ihrem Geschäftsmodell und Ihrem Unternehmen passt. Und wenn ja: welche Art und Technologie sind empfehlenswert? Mehr Infos & Anfrage auf blockchain.innoq.com

Das iSAQB-Modul **Blockchain: Konsens in dezentralisierten Applikationen** vermittelt Entwickler*innen und Architekt*innen eine ausgewogene, technisch fundierte Sicht auf Blockchain-Technologien – von öffentlichen Kryptowährungen bis zum unternehmensfokussierten Distributed Ledger – ganz ohne Hype.
Buchung unter innoq.com/isaqb-blockchain

Über die Autoren

Lars Hupel

@larsr_h

Lars ist Consultant bei INNOQ in München und ist bekannt als einer der Gründer der Typelevel-Initiative, die sich der Entwicklung von typgetriebenen Scala-Bibliotheken in einer einsteigerfreundlichen Umgebung verschrieben hat. Er spricht oft auf Konferenzen und ist im Open-Source-Umfeld in Scala unterwegs. Außerdem programmiert und redet er gern über Haskell, Prolog und Rust.

Stefan Tilkov

@stilkov

Stefan Tilkov ist Geschäftsführer und Principal Consultant bei der INNOQ, wo er sich vorwiegend mit der strategischen Beratung von Kunden im Umfeld von Softwarearchitekturen beschäftigt. Er ist Autor des Buchs „REST und HTTP“, Mitherausgeber von „SOA-Expertenwissen“ (beide dpunkt.verlag), Autor zahlreicher Fachartikel und häufiger Sprecher auf internationalen Konferenzen.