Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorthesis

Andreas-Marko Dymek

Dezentrale Blockchain Applikation für den OTC Energiehandel

Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Andreas-Marko Dymek

Dezentrale Blockchain Applikation für den OTC Energiehandel

Bachelorthesis eingereicht im Rahmen der Bachelorprüfung
im Studiengang Informations- und Elektrotechnik
am Department Informations- und Elektrotechnik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. -Ing. Wolfgang Renz
Zweitgutachter : Prof. Dr. Heike Neumann

Abgegeben am 22. Mai 2018

Andreas Dymek

Thema der Bachelorthesis

Dezentrale Blockchain Applikation für den OTC Energiehandel

Stichworte

Blockchain, Ethereum, Smart Contracts, P2P, SGAM, Kryptografie, Digitalisierung, Energy Plattform, Konsens

Kurzzusammenfassung

Das Ziel dieser Arbeit ist eine Technologie Studie zum Thema Distributed Ledger Technology (DLT). Mit der damit verknüpften Blockchain Technik wird ein Anwendungsfall für eine Peer-to-Peer Energiehandels Plattform entwickelt und getestet. Die Ethereum Blockchain hat sich als fortschrittlichste kristallisiert und ermöglichte die Entwicklung des Prototypen. Außerdem wird ein Ausblick auf kommende Neuerungen im Zusammenhang mit der Energiewirtschaft gegeben.

Andreas Dymek

Title of the paper

Decentral Blockchain application for the OTC Energytrade

Keywords

Blockchain, Ethereum, Smart Contracts, P2P, SGAM, Cryptography, digitization, Energy Platform, Consensus

Abstract

The aim of this thesis is a technology study on the Distributed Ledger Technology (DLT). The associated blockchain technology will be used to develop and test a use case for a peer-to-peer energy trading platform. The Ethereum Blockchain has the most advanced stage of development and enabled the design of the prototype. Furthermore a brief perspective is given on the possibility of upcoming innovations in the context of the energy industry.

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABI	Binärschnittstelle - Application binary interface
ASIC	application-specific integrated circuit - Anwendungsspezifische integrierte Schaltung
BFT	Byzantine fault tolerance
DER	Distributed Energy Resources
DLT	Distributed Ledger Technologie
DPoS	Delegated Proof-of-Stake
DSA	Digital Signature Algorithmus
ECDSA	Elliptic Curve Digital Signature Algorithm
HAW	Hochschule für Angewandte Wissenschaften Hamburg
HTTP	Hypertext Transfer Protocol
IED	Intelligent Electronic Device
IOT	Internet of Thinks
IPFS	InterPlanetary File System
IPC	Inter-process communication
JSON-RPC	JavaScript Object Notation Remote Procedure Call
MMLab	Multimedia Labor
NIST	National Institute of Standards and Technology
OTC	Over the Counter - Außerbörslicher Handel
P2P	Peer-to-Peer
PoA	Proof-of-Authority
PoC	Proof of Concept
PoET	Proof-of-Elapsed-Time
PoS	Proof-of-Stake
PoW	Proof-of-Work
Prosumer	Verbraucher und gleichzeitig Produzent

RIPEMD RACE Integrity Primitives Evaluation Message Digest

SCADA Supervisory Control and Data Acquisition

SHA-2 Secure hash algorithm

SHS Secure hash standart

TCP Transmission Control Protocol

tx Transaction - Transaktionen

1. Einleitung

Im Zuge der Energiewende werden zunehmend nachhaltige Energiesysteme angeschlossen, wodurch sich das Stromversorgungsnetz grundsätzlich verändert. Viele neue Energieerzeugersysteme sind regenerative Anlagen (Windenergie, Photovoltaik oder Biogasanlagen), dessen Energieeinspeisung aufgrund von Wetterverhältnissen stark variiert.
Der Anteil von erneuerbaren Energiesystemen an der Stromerzeugung 2017 ist von 29,0 % im Vorjahr auf 33,1 % gestiegen. Die Jahre vorher war die Steigung noch deutlicher BDEW [2017]. Aufgrund dieser neuen Dynamik wird die Steuerung der Versorgungsnetze für Netzbetreiber erschwert, da immer mehr Eingriffe ausgeführt werden müssen, um die Netzstabilität zu gewährleisten, die sogenannten Einspeisemanagementmaßnahme (Einsman) BNetzA [2017]. Auch für Bilanzkreisverantwortlichen (BKV) ist diese Entwicklung sehr schwerwiegend. Die dadurch entstehenden spontanen Abweichungen von gemeldeten Energiefahrplänen können zu hohen Kosten führen, da das Kompensieren durch Ausgleichsenergie erfolgt. Diese Kosten werden auf den Endkunden übertragen.
Die Herausforderung bei elektrischer Energie ist, dass diese heutzutage nicht direkt in großen Maßen gespeichert werden kann. Erzeugte Energie muss größtenteils sofort verbraucht werden, da es sonst zu Netz Instabilitäten führt. Um diese ständigen Eingriffe in Zukunft einfacher und sorgfältiger bzw. automatisierter zu ermöglichen, ist es notwendig, eine Digitalisierung der Energiewirtschaft zu realisieren. Schlussendlich gibt es nicht nur die technischen Herausforderungen der Einspeisemanagementmaßnahme, sondern auch die damit finanziell verbundenen Entschädigungen an den Anlagenbetreiber, welche sich für das Jahr 2016 auf 643 Millionen Euro beliefen BNetzA [2017].
Grundsätzlich eröffnet sich durch die Dezentralisierung des Stromnetzes eine Beteiligung und Teilhabe der Verbraucher am Energiemarkt und durch den Ausbau von privaten Anlagen werden diese zu sogenannten „Prosumer “. Dadurch könnte sich eine Unabhängigkeit von großen Energiekonzernen entwickeln und zu mehr Wettbewerb führen. Für private Prosumer wäre eine Stromhandelsplattform, die bilaterale Geschäfte sogenannte OTC-Geschäfte (Over The Counter) ermöglicht, eine gute Perspektive.
Eine Option zur Digitalisierung und für eine bilaterale Stromhandelsplattform bietet die Blockchain Technologie, welche die Datensouveränität und eine direkte Interaktion zwischen sich unbekannten Akteuren gewährleistet, ohne dass hierfür eine Vermittlung durch zentrale Instanzen notwendig ist F.Kirstein [2017]. Die Interaktionen beschränken sich nicht nur auf Transaktionen, sondern schließen auch Anwendungen und Prozessabläufe ein, welche durch die Blockchain-Architektur manipuliersicher, nachvollziehbar und effizient durchführbar sind.

1.1. Motivation und Relevanz

Wie in der Einleitung angedeutet benötigt, die Energiewirtschaft eine Entwicklung der bisherigen Automatisierung, eine größere Ausprägung der Digitalisierung und das damit verbundene einführen von neuen Technologien. Dies ist Notwendig um in Zukunft auf Herausforderungen zu reagieren, aber auch um eine Weiterentwicklung voran zuführen.
Eine mögliche Technologie um diese Entwicklung voran zu treiben könnte die Distributed Ledger Technologie (DLT 2) sein. Im Speziellen sei hier die Blockchain Technologie genannt, welche zur Zeit großes öffentliche und unternehmerisches Interesse erfährt.
Die Blockchain Technologie soll laut ihren Befürwortern nicht nur eine Weiterentwickelung der Digitalisierung und Automatisierung sein, sondern soll bestehende Produkte, Technologien oder Dienstleistung ersetzen können. Diese These gilt es in dieser Arbeit aufzuarbeiten.
Was jedoch über die Blockchain Technologie gesagt werden kann ist, dass es ein System ist, welches auf vielen einzelnen Lösungen basiert und diese geschickt miteinander verbindet, wodurch ein sehr stabiles dezentrales Netzwerk erstand.
Einer dieser Lösungen, der verteilten Konsensbildung, kann in vielen Abläufen die Rolle eines vertrauenswürdigen Dritten einnehmen. Somit könnten Geschäftsmodelle vieler Organisationen und Institutionen, welche heute die Rolle der Überwachung und Intermediäre besetzen, transparenter gestaltet werden oder sogar ersetzt werden.
Die mögliche Option mit der Blockchain Echte-Werte abzubilden, ist die Grundlage des Internet-of-Value, in der nicht nur Kryptowährungen sondern auch reale Waren und Werte wie Grundstücke, Lizenzen und Fahrzeuge abgebildet und gehandelt werden. Es ergibt sich eine neue Art des Ursprungs und Transparenz. Auch die Möglichkeit mit Hilfe von programmierten Verträgen (Smart Contracts) Prozesse auf der Blockchain zu automatisieren und dezentral auszuführen birgt ein großes Potenzial an Automatisierung, da der Aufwand extrem gering und die Transparenz für alle gleich ist.
Dabei besitzen öffentliche Blockchain Systeme nicht nur technologisch die Möglichkeit große Veränderungen herbeizubringen sondern auch Gesellschaftlich. Durch die Transparenz und die Irreversibilität aller Transaktionen sind diese für jeden sichtbar und nachvollziehbar. Mit einer transparenteren Methodik würde das Vertrauen der Gesellschaft in viele Bereiche Einzug erhalten.
Durch die vielfältigen Anwendungen dieser Technologie kommt es zu einer großen Aufmerksamkeit von unterschiedlichen Unternehmen und Start-Ups und erste Proof of Concept (PoC) werden entwickelt.

1.2. Aufgabenstellung und Aufbau der Bachelorthesis

Aus dem Titel der Arbeit „Dezentrale Blockchain Applikation für den OTC (Over the Counter) Energiehandel“ lässt sich die Aufgabenstellung und Struktur der Bachelorthesis ableiten.
Im ersten Schritt wird eine Einführung in die Distributed Ledger Technology (DLT) und die Blockchain Technik gegeben, um die Verständlichkeit der Arbeit zu verdeutlichen. Hierbei werden unter anderem kryptografische Grundlagen, die Netzarchitektur und verschiedene relevante Konsenses Algorithmen beschrieben.
Um die Theorie zu vertiefen wird in Kapitel 2.7 das Ethereum Blockchain-Protokolle erläutert und die Besonderheit des Protokolls verdeutlicht.
In Kapitel 2.12 wird die Aufgabenstellung aufgeworfen und ein Use Case im Kontext der Blockchain und programmierten Verträgen(Smart Contracts) in der Energiewirtschaft konzipiert. Für dieses wird eine dezentrale Anwendung mit Ethereum beschrieben, die Implementation dargestellt und getestet.
Zum Abschluss wird eine Zusammenfassung der Thesis dargelegt, ein Fazit der Arbeit und der verwendeten Technologie. Außerdem ein Ausblick auf die Möglichkeiten der Blockchain.

1.3. Zielsetzung und Forschungsfrage

Ziel dieser Arbeit ist es, einen funktionsfähigen Prototypen zu entwerfen und in eine Blockchain zu implementieren. Dieser Prototyp soll Grundlagen des theoretischen Konzepts der OS4ES [2017] aufnehmen, aber auch eigene vereinfachte Prinzipien aufstellen und detailliert aufzeigen. Zum anderen soll ein möglicher Nutzen für die Energiewirtschaft aufgezeigt werden.

Forschungsfragen :

- Kann die DLT eine weitere Automatisierung und Digitalisierung in der Energiewirtschaft bewerkstelligen?
- Kann eine OTC Energiehandelsplattform mit Hilfe der DLT bzw. einer Blockchain abgebildet werden? Welche Vor-und Nachteile, sowie Herausforderungen ergeben sich für den Einsatz in der Praxis?

2. Einführung in Distributed Ledger Technology

2.1. Motivation und Hintergrund

Kassenbücher (Ledger) sind seit der Antike ein extrem wichtiger Bestandteil des Handels. Durch diese werden das protokollieren von Wertgütern wie Geld und Grundstücken deutlich vereinfacht. Sie haben eine enorme Entwicklung vorgenommen, von Tontafeln zu Papyrus weiter zu Vellum bis hin zum Papier.
Jedoch war die Entwicklung meistens nur eine Übertragung von einem Medium zum anderen. Genauso verhält es sich in unserer heutigen Zeit. Papier wird von Bytes ersetzt. Doch neben des einfachen Austausch des Mediums haben sich neue Prinzipien entwickelt wie digitale verteilte Kassenbücher (Distributed Ledger Technology). Diese sollen deutlich mehr Potential und Gleichheit hervorbringen als traditionelle Kassenbücher.
Im Prinzip ist die DLT eine verteilte Datenbank, welches über ein globales Netzwerk für jeden Teilnehmer eine gemeinsame Schreib-, Lese- und Speicherberechtigung erlaubt. Durch diese gemeinsame Datenhaltung können komplexe Prozesse erleichtert werden. Grundsätzlich wären keine Intermediäre mehr nötig, um Transaktionen zu bewerkstelligen.
Grundlegende Technologie der DLT ist die „Blockchain“. Die erste Entwicklung ihrer Art war die Peer-to-Peer Internet Währung „Bitcoin“ welche im Jahr 2008 von Nakamoto [2008] veröffentlicht wurde. Die Blockchain dient hier als verteilte, öffentliche und dezentrale Datenbank. Bitcoin basiert auf mehrjähriger Forschung in der Kryptografie, verteilten Systemen und kryptografischen Währungen laut Swan [2016], jedoch wurde durch die Blockchain Lösung eine wahre Pionierleistung vollbracht welche zwei grundlegende Probleme löste :

1. Das „Double Spending“ Problem siehe Bonadonna [2016]
2. Das „Byzantine Generals“ Problem siehe Lamport u. a. [1982]

Double Spending beschreibt das doppelte Ausgeben eines Gutes an mehr als ein Ziel, d.h. eine Münze darf nicht die Möglichkeit besitzen gleichzeitig an zwei verschiedene Personen verteilt zu werden. In bisherigen System wird dieses über Intermediäre unterbunden bzw. verhindert. Ein Beispiel hierfür ist z.B. Paypal welches als zentrale Autorität dient, um Transaktionen auf Double Spending zu überprüfen.
Bei Bitcoin wird die Überprüfung darüber bewerkstelligt, dass alle Transaktionen in Blöcken in der Blockchain festgeschrieben werden, wodurch eine chronologische Verbindung von kryptografisch verifizierten Blöcken mit allen Transaktionen entsteht Nakamoto [2008].
Eine passende Zusammenfassung des Byzantine Generals Problems wird in Reischuk [1987] gegeben auf die an dieser Stelle zurückgegriffen wird:

„Eine byzantinische Armee, die aus mehreren räumlich getrennten Divisionen besteht, befindet sich in der Situation, einen gemeinsamen Schlachtplan entwerfen zu müssen. Jede Division wird von einem General geführt. Die Generäle können untereinander nur durch Austausch von Botschaften kommunizieren. Eine Zusammenkunft aller zur Beratung ist aus strategischen Gründen ausgeschlossen. Eine Entscheidung (etwa sofortiger Angriff oder nicht) muss daher von jedem General aus den Vorschlägen seiner Kollegen nach einem vorab festgelegten Verfahren vor Ort getroffen werden. Das Problem dabei ist, dass einige wenige, eventuell nicht loyale Generäle, die heimlich mit dem Feind paktieren, durch unterschiedliche Vorschläge an die anderen Generäle eine geschlossene Entscheidung der übrigen verhindern können. Dies wird das Problem der byzantischen Generäle oder das Problem der interaktiven Konsistenz genannt. “

Das Byzantine Generals Problem besteht aus dem Versuch, Akzeptanz über einen Handlungsverlauf oder den Status eines Systems über Austausch von Informationen über ein unzuverlässiges oder potentiell komprimiertes Netzwerk durchzuführen.

2.2. Kryptografische Grundlagen

2.2.1. Symmetrische und Asymmetrische Verschlüsselung

In Bitcoin und anderen Blockchain bzw. Kryptowährungen werden symmetrische Verschlüsselungen verwendet um mit Hilfe von privaten Schlüsseln (*Private Keys*) Transaktionen eindeutig zu signieren. Die Asymmetrische Verschlüsselung findet Verwendung zur Generierung von Bitcoin Adressen und innerhalb der Blockchain zum verifizieren und signieren von Transaktionen.
Antonopoulos [2014] vergleicht diese Verschlüsselungen mit dem Girokonto. Die Kontonummer steht für den öffentlichen Schlüssel (*Public Key*) und die PIN (Personal Identification Number) mit dazugehörigen EC Karte ergibt den privaten Schlüssel.
Die symmetrische Verschlüsselung ist eine historisch geprägte Verschlüsselung die schon von Julius Caesar (100 v. Chr.) im gallischen Krieg verwendet wurde.
Mehrere Parteien haben einen Schlüssel als gemeinsames Geheimnis vereinbart, um Nachrichten zu verschlüsseln und entschlüsseln. Es ist enorm wichtig den Schlüssel über einen sicheren Übertragungskanal zu verteilen, damit jene Sicherheit geboten werden kann, dass das System nicht komprimiert ist. Eine Nachweisbarkeit über den Versender und Empfänger gibt es bei dieser Art der Verschlüsselung nicht. Es werden zwei verschiedene Verschlüsselungen unterschieden Franco [2014]:

1. **Kanalverschlüsselung**, verschlüsselt jedes *Bit* eines Datenstrom einzeln
2. **Blockverschlüsselung**, verschlüsselt Blöcke vom Datenstrom, üblich 128 Bit lang

Das Bitcoin Core Wallet, die Referenz Wallet Implementierung von Bitcoin, benutzt eine AES-256 Blockverschlüsselung, um die privaten Schlüssel der asymmetrischen öffentlichen Schlüssel Verschlüsselung zu schützen. Eine Ausführliche Ergänzung zum AES-256 findet sich in Franco [2014].
Die Asymmetrische bzw. öffentliche Schlüssel Verschlüsselung wurde von Merkle und Hellman [1978] veröffentlicht. In dieser ist es nicht mehr nötig einen gemeinsamen geheimen Schlüssel zwischen den Parteien zu vereinbaren. Jeder erzeugt einen eigenen öffentlichen und geheimen privaten Schlüssel.
Beide Schlüssel sind über eine Einwegfunktion miteinander verbunden, d.h. der öffentliche Schlüssel kann aus dem privaten Schlüssel leicht berechnet werden aber nicht umgekehrt. Der öffentliche Schlüssel ermöglicht dem Nutzer, falls dieser im Besitz des privaten Schlüssels ist, Daten zu verschlüsseln, die digitale Signatur zu prüfen oder zu verifizieren. Der private Schlüssel wird dafür angewandt, Daten welche mit dem öffentlichen Schlüssel verschlüsselt wurden zu entschlüsseln, digitale Signaturen zu erzeugen und sich zu authentisieren.
Im Bitcoin Protokoll werden der Elliptische Kurven (*ECDSA* siehe Abb. 2.1) Algorithmus als Einwegfunktion zusammen mit dem Digital Signature Algorithmus (*DSA*) für das digitale signieren verwendet (siehe Abb. 2.2). Die genaue Definition der verwendeten elliptischen Kurve lautet *secp256k1* und wurde vom National Institute of Standards and Technology (NIST) veröffentlicht.
Die folgende Funktion definiert die Kurve:

$$y^2 = (x^3 + 7) \text{ über } \mathbb{F}_P \tag{2.1}$$

$\mathbb{F}_P$ ist ein endlicher Körper von Primzahl Ordnungen mit

$$p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1 \tag{2.2}$$

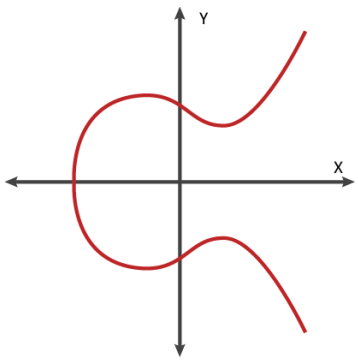

Abbildung 2.1.: ECDSA Algorithmus ähnlich zu secp256k1

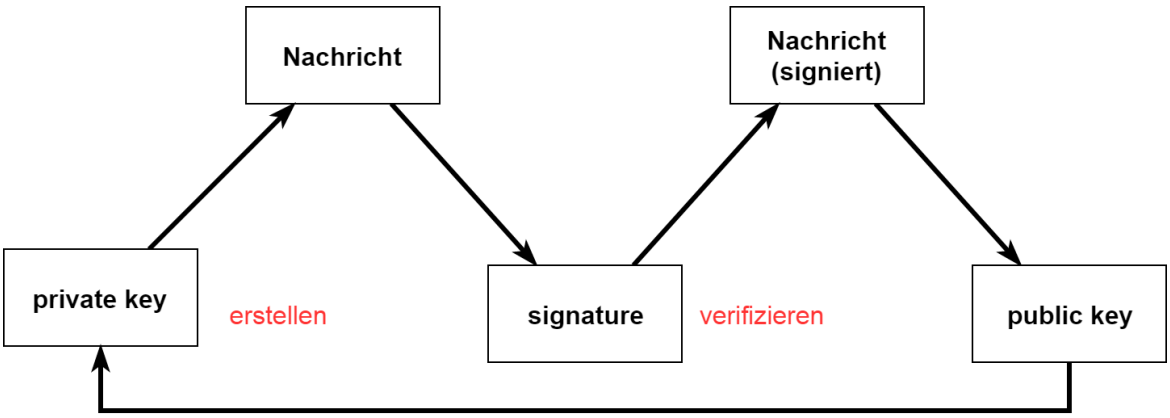

Abbildung 2.2.: Digitale Signierung

2.2.2. Hashfunktion

Eine kryptografische Hashfunktion ist in erster Linie eine mathematische Funktion, welche folgende Eigenschaften aufweist:

1. Der Eingabewert ist ein String mit beliebiger Größe.
2. Der Ausgabewert besitzt eine fest definierte Größe, welche von der verwendeten Funktion abhängt, Bspw.: Bitcoin - SHA256 - 256 bit.
3. Die Funktion ist deterministisch, ein gleicher Eingabewert führt zum gleichen Ausgabewert.
4. Es muss sich um eine Einwegfunktion handeln, damit aus einem Ausgabewert nicht ein Eingabewert gefunden werden kann.
5. Sie muss kollisionsresistent sein. Es soll praktisch unmöglich sein, zwei verschiedene Eingabewerte zu finden, welche den selben Hashwert ergeben.

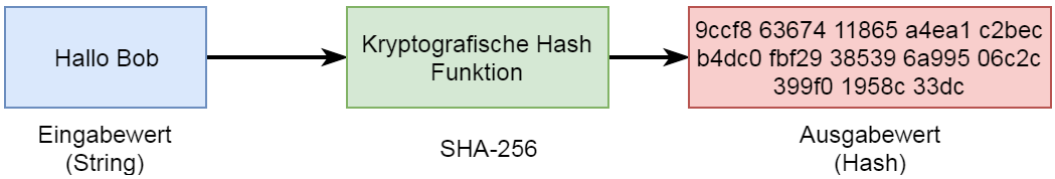

Abbildung 2.3.: Beispiel Hashfunktion

In allen Blockchain bzw. Kryptowährungen werden Hashfunktionen (vgl. 2.3) benutzt. In Bitcoin sind die Hashfunktionen SHA-256 und RIPEMD160 (RACE Integrity Primitives Evaluation Message Digest) implementiert. SHA-256 setzt sich aus einem Merkle Meta-Verfahren mit Davies-Meyer-Kompressionsfunktion zusammen und wurde unter dem Name SHA-2 im Secure Hash Standard (SHS) vom NIST 180-4 [2012] definiert. Das Prinzip verläuft vereinfacht dargestellt wie folgt:

1. Jeder String wird auf 512 Bit erweitert und dann in Blöcke zu 32 Bit Wörtern aufgeteilt.
2. Danach findet die Hashberechnung in 64 Durchläufen über sechs logische Funktionen statt.
3. Jeder Durchlauf ermittelt acht Hashwerte für die Berechnung der nächsten Iteration.
4. Zuletzt werden die Werte verkettet und bilden den Hashwert von SHA-256.

Die Funktionalität von RIPEMD160 ist ähnlich dem SHA-256, nur dass es 80 Durchläufe mit fünf logischen Funktionen vollzieht.
Bitcoin nutzt SHA-256 und RIPEMD160 um Bitcoin Adressen zu generieren. Zunächst wird der SHA-256 angewandt und danach dass RIPEMD160 um die Adresse auf 160 Bit zu verkürzen. Außerdem wird der private Schlüssel mit dem SHA-512 erzeugt. Ethereum nutzt eine andere Hashfunktion als SHA-256 nämlich KECCAK-256, welches eine Weiterentwicklung von SHA-2 Hashfunktionen ist.

2.2.3. Hash Pointer

Der Hash Pointer (siehe Abb. 2.4) ist ein Zeiger welcher auf gespeicherte Daten zusammen mit dem Hash dieser Daten zeigt. Somit schafft ein Hash Pointer die Möglichkeit zu prüfen, ob Daten sich verändert haben (z.B. durch Manipulation). Sie sind ein Grundbaustein der Datenstruktur der Blockchain.

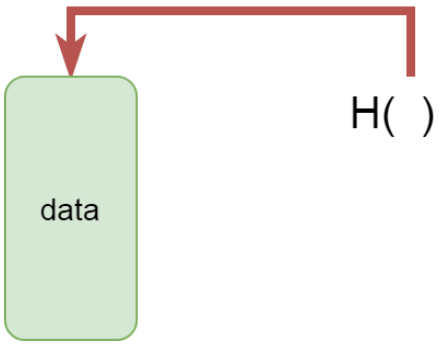

Abbildung 2.4.: Beispiel Hash Pointer

2.2.4. Hash-Bäume

Hash-Bäume (siehe Abb. 2.5) sind eine Art von Datenstruktur (Merkle tree) welche von Merkle [1979] erstmalig veröffentlicht und später patentiert wurden.
Im Grunde ist ein Merkle tree ein Baum von Hashwerten, die Blätter sind Datenblöcke welche jeweils einen Hashwert haben und in Paaren angeordnet sind. Die Blockpaare erzeugen einen Hashwert und werden in einem darüber liegenden Knoten zusammengefasst. Die neuen Blockpaare erzeugen wiederum einen Hashwert und werden zusammengeführt.
Diese Funktionalität verläuft soweit bis man am Ende des Baumes angelangt ist. Dort entsteht der Root-Hash. Durch dieses Verfahren ist es möglich große Datenmengen zu reduzieren bzw. umzuverteilen. Außerdem kann durch die Struktur schnell bewiesen werden, ob ein bestimmter Block ein gültiger Teil der Gesamtstruktur ist.
Um zu verifizieren ob ein Block gültig ist z.B. H_D, müssen lediglich die Blau markierten Hashwerte bekannt sein. Durch berechnen der Hashwerte kann sehr schnell und leicht verifiziert werden, ob ein Block gültig ist oder nicht, da bei veränderten Daten die Hashwerte nicht mehr übereinstimmen.

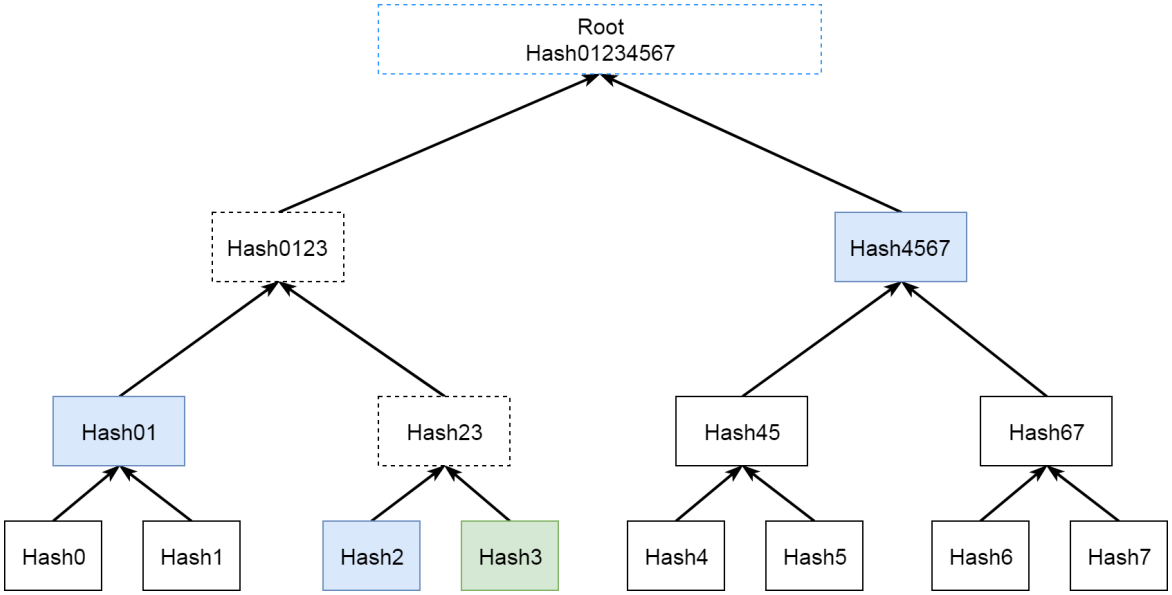

Abbildung 2.5.: Merkle Baum mit 8 Transaktionen

2.3. Aufbau der Blockchain

Die Blockchain Datenstruktur (siehe Abb. 2.6) ist eine geordnete, verlinkte Liste von Blöcken, in welcher Transaktionen und Zeitstempel protokolliert werden.

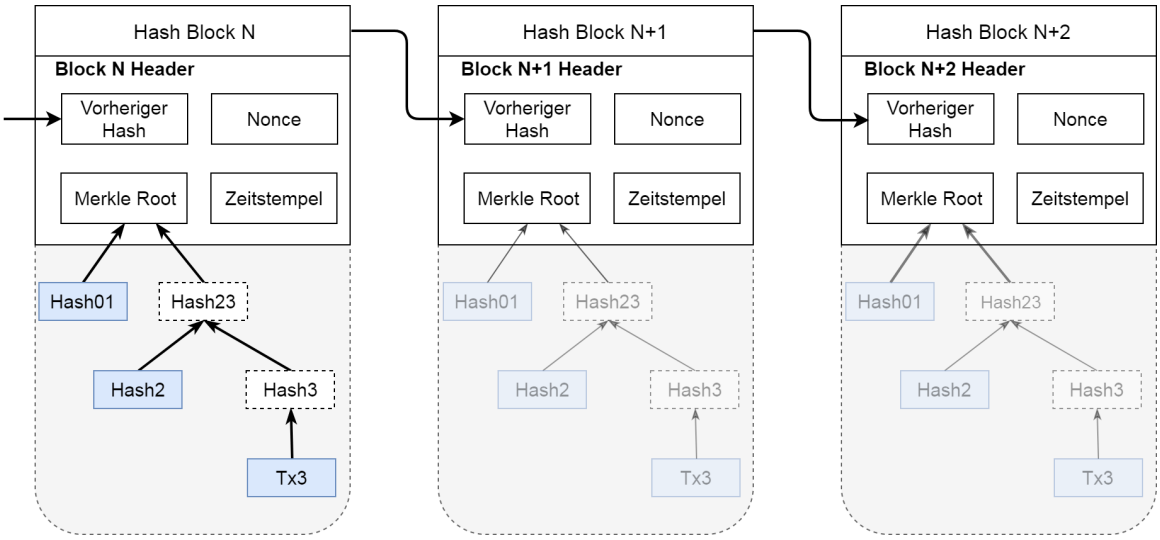

Abbildung 2.6.: Vereinfachter Header einer Blockchain mit jeweils 4 Transaktionen (tx) pro Block

Die Blöcke verweisen auf den Hashwert des vorherigen Blockes, wodurch die angesprochene Kette entsteht. Neben der Referenzierung auf den vorherigen Block wird in den Blöcken ein *Zeitstempel*, eine *Nonce*, welcher ein ansteigender Wert ist um zusätzlich den Hashwert des Blockes zu verändern und den Hashwert des Merkle-baumes eingetragen.
Dieser beschriebene Header wird speziell im Bitcoin Protokoll verwendet.
Im Ethereum Protokoll gibt es noch zusätzliche Werte wie das *Gas Limit*, den *State Root*, *Receipt Root*, *Uncles Hash*. Jeder neu erstellter Hashwert eines Blockes wird dementsprechend vom vorherigen Block Hashwert beeinflusst, wodurch die Veränderbarkeit der Blockchain entsteht.
Veränderliche Werte des Headers sind zu einem der Hashwert des *Merkle-Baum*, welcher ein Hashwert aus den unter ihm gesammelten Hashwerten ist, der Zeitstempel und die *Nonce*. Außerdem gibt es im *Proof-of-Work (PoW)* Konsenses Algorithmus noch die *Difficulty* welche den Schwierigkeitsgrad der Rechenoperation darstellt.

2.4. Peer-to-Peer Netzarchitektur

Blockchain Netzwerke basieren auf einer Peer-to-Peer (P2P) Netzarchitektur(siehe Abb. 2.7). Dies bedeutet, dass alle Teilnehmer des Netzwerkes untereinander direkt kommunizieren.
Bei einer P2P Architektur liegt eine Gleichberechtigung aller Nutzer vor, sodass alle Teilnehmer die selben Funktionen und Dienstleistungen nutzen oder anbieten können. Es ist als ein dezentrales zusammenhängendes Netz aufgebaut (siehe Abb. 2.7).

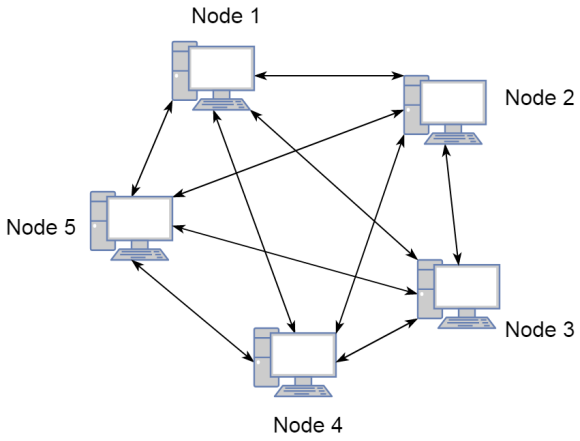

Abbildung 2.7.: P2P Architektur: Darstellung mit 5 Teilnehmern (Nodes)

Die Teilnehmer, auch als Netzknoten (Nodes) bezeichnet, verbinden sich mit dem gemaschten Netz, welches eine flache Topologie aufweist.
Im Blockchain Zusammenhang bedeutet dies, dass die jeweiligen P2P-Netze (bspw. Bitcoin-Netzwerk) für die Transaktionen und das Übertragungsmedium für die Konsensbildung verantwortlich sind.
Die Kommunikation innerhalb des Netzwerks findet über das TCP (Transmission Control Protocol) statt. Verschiedene Portnummern werden verwendet, um Zugriff auf unterschiedliche Netze zu erhalten (bspw. Bitcoin Hauptnetz - Port 8333; Ethereum Hauptnetz - Port 30303).

2.5. Transaktionen

Transaktionen sind einer der wichtigsten Aspekte einer Blockchain. In Bitcoin speziell ist alles darauf ausgelegt, Transaktionen zu erstellen, im Netzwerk zu verbreiten, zu validieren und schlussendlich zum Hinzufügen in dass globale Kassenbuch (Ledger).

Die Abb. 2.8 zeigt wie Transaktionen in Ethereum, einer Kryptowährung die voll-turing kompatible ist, in eine Blockchain geschrieben werden. Im Bitcoin Protokoll würden die Transaktionen ähnlich verarbeitet nur mit weniger Informationen. In Ethereum gibt es drei verschiedene Arten von Transaktionen:

1. Transfer von Ether (Ethereum eigene Währung).
2. Erstellen von programmierten Verträgen öfters auch als Smart Contracts benannt.
3. Ausführen von Verträgen (Smart Contracts) und die damit programmierten Funktionen.

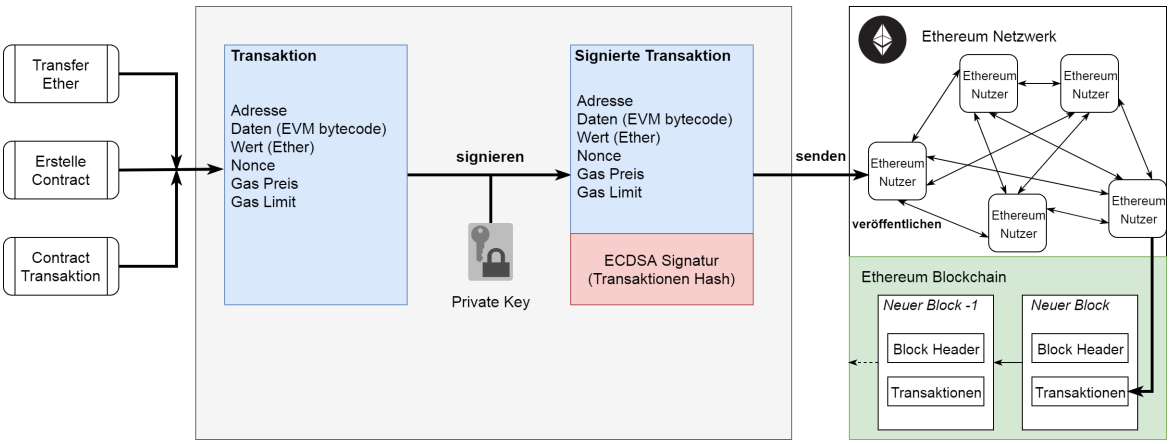

Abbildung 2.8.: Einfache Darstellung von Transaktionen in Ethereum

Eine der aufgelisteten Transaktionen wie beispielsweise „Erstelle Contract“ wird vom Nutzer ausgeführt. In der Transaktionen stehen die Ziel und Herkunfts-Adresse, die Opcode Daten für die Ethereum Virtuell Maschine (EVM), der Ether-Wert falls es zu transferieren von Werten kommt, der Gas Preis, welcher die Kosten des ausführbaren Vertrag darstellt, die Nonce und das Gaslimit, da nur begrenzte Funktionen ausgeführt werden können. Danach wird die Transaktion vom privaten Schlüssel des Nutzers signiert, um sie eindeutig zu verifizieren.
Nach dem signieren wird die Transaktion wie im Kapitel 2.2 über einen ECDSA Algorithmus zu einem Hash entwickelt. Diese Transaktion wird als nächstes für alle Nutzer des Ethereum Netzwerkes veröffentlicht. Je nach Konsens Algorithmus (siehe Abschnitt 2.6) wird die Transaktion in die Blockchain geschrieben.
Bei Proof-of-Work 2.6.1 agieren bestimmte P2P Knoten (Miner) darum, neue Blöcke zu erstellen, wodurch sie neben neuen Blöcken auch Transaktionen in die Blockchain schreiben können.

2.6. Konsens Algorithmen

Konsenses Algorithmen sind für jede Blockchain notwendig, da sich die Teilnehmer eines Netzwerkes auf eine Wahrheit einigen müssen. Schlussendlich besteht die Blockchain aus Blöcken die von allen Teilnehmern akzeptiert werden sollen. Man könnte diesen Mechanismus mit einen standardisierten Weg eines Gesetzesentwurf im Parlament vergleichen. Im Folgenden werden die populärsten Algorithmen vorgestellt.
Jedoch sei anzumerken, dass es nur ein kleiner Ausschnitt ist, es gibt gegenwärtig mehrere verschiedene Konsens Algorithmen die ihre Berechtigung haben und auf die verschiedenen Projekte zugeschnitten sind.

2.6.1. Proof-of-Work (PoW)

Proof-of-Work (PoW)(PoW) ist in der Informatik ein Mechanismus der gewährleistet, dass ein Nutzer Arbeit verrichtet, damit der Dienst oder das Netzwerk vor übermäßigen oder fälschlichen Anspruch geschützt wird. Diese Arbeit wird vom Rechner verrichtet, indem eine mathematischer Aufgabe (Differentialgleichungen, Kryptografische Primitiven) gelöst wird.
In Bitcoin und vielen anderen PoW basierten Blockchains basiert der Mechanismus auf dem Hashcash-Algorithmus. Dieser wurde ursprünglich dazu verwendet E-Mail-Spam zu vermeiden und wurde von Cynthia Dwork [1992] entwickelt.
Nutzer des PoW Algorithmus, welche auch Teilnehmer des Netzwerkes sind, werden als Miner Knoten (Miner Nodes) bezeichnet. Sie sind für das validieren von Transaktionen, dem akzeptieren von Blöcken und für die Erstellung von neuen Blöcken verantwortlich (siehe Abb. 2.9)

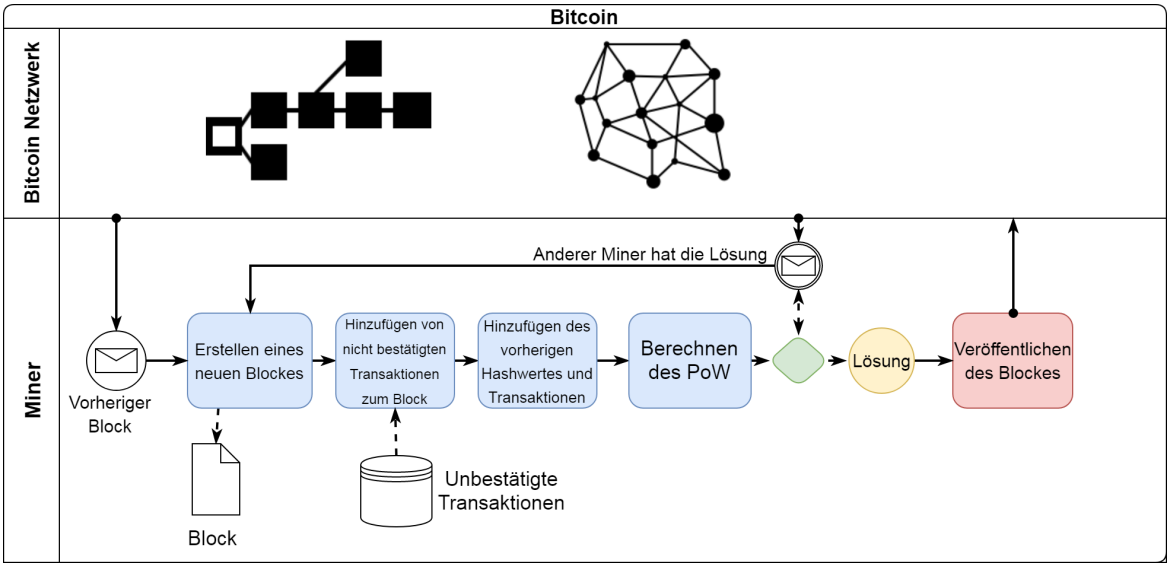

Abbildung 2.9.: Prozess vom Bitcoin Mining - erstellen von Blöcken, validieren von Transaktionen und bereitstellen des Block Beweises

Für jeden neu erzeugten Block wird eine Nonce (siehe Abb. 2.10) erstellt. Die Berechnung dieses Wertes würde enorm viel Rechenleistung benötigen. Um an das Ziel zu gelangen wird daher, ein passender SHA-256 Hashwert gesucht, welcher mit „000“ beginnt. Sobald dieses geschieht, wird die Aufgabe als gelöst angesehen. Als Anreiz wird der Miner, welcher diesen Wert als erstes errechnet und damit einen neuen Block generiert, mit einer vorher definierten Menge der jeweiligen Kryptowährungen (bspw. Bitcoin) belohnt. Durch diese Belohnung ist es lukrativer, die Regeln des Protokolls zu befolgen, als zu versuchen, sich durch betrügerisches Handeln zu bereichern.

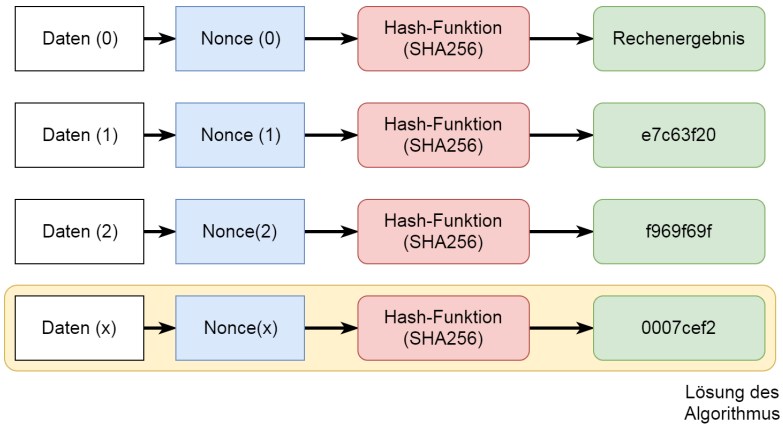

Abbildung 2.10.: Einfach Darstellung der PoW Berechnung

Der Nachteil von Proof-of-Work ist aber der mitunter sehr hohe Energiebedarf. Der rasante Anstieg des Bitcoin-Preises hat dazu geführt, dass sich professionelle „Mining Pools“ gebildet haben, die miteinander konkurrieren. Der Einsatz von sogenannten Anwendungsspezifischen integrierten Schaltungen (ASICs) ermöglicht die gezielte Berechnung des Hashcash-Algorithmus und führt zu einem immer weiter steigenden Rechen- und Energieaufwand. Die Schwierigkeit der mathematischen Aufgabe skaliert mit der Rechenleistung im Netzwerk, wodurch private Rechner diese Aufgaben nicht mehr in entsprechender Zeit lösen können. Dadurch ist das Bitcoin-Mining für Privatpersonen unattraktiv geworden und wird nun von wenigen großen Akteuren kontrolliert. Dies widerspricht der ursprünglichen Ideologie von Bitcoin als komplett dezentrales Zahlungsmittel. Außerdem ist die Skalierbarkeit und die Geschwindigkeit der Transaktionen stark beschränkt, da auf die Lösung des Rätsels gewartet werden muss.

2.6.2. Proof-of-Stake (PoS)

Proof-of-Stake ist ein weiterer Konsens Algorithmus für Blockchain Protokolle der von Buterin [2018] vorgestellt und die Funktionalität mathematisch bewiesen wurde. Es wurde entwickelt um die Blockgenerierung unabhängig von der Rechenleistung zu machen und die Blockerzeugungszyklen zu verkürzen. Während bei Proof-of-Work (PoW) die Berechnung einer kryptografischen Aufgabe für die Blockgenerierung nötig ist um neue Transaktionen einzufügen, so werden bei PoS Anteile einer Währung vorgehalten und für den weiteren Gebrauch gesperrt (Stake). Laut Buterin [2018] sieht ein PoS folgendermaßen aus, jeder der die Blockchain basierte Kryptowährung besitzt, kann durch hinterlegen einer bestimmten Menge ein Validator werden. Ein Validator wird beim erstellen und akzeptieren eines neuen Blockes eingebunden. In der *chain-basierten proof-of-stake* Variante wird über einen Pseudo-Zufallsgenerator ein Validator ausgesucht, um einen neuen Block zu erstellen.
Bei der BFT-proof-of-Stake Variante werden zufällige Validator ausgewählt, welche ein Recht erhalten einen Block vorzuschlagen. Dieser Block wird jedoch durch einen mehr Runden Prozess geschickt um von anderen Validatoren ausgewählt zu werden. Mit diesen Anteil der Währung validiert der Besitzer anteilig Transaktionen auf der Blockchain. Bei PoS werden jetzt Währungsanteile eingefroren und aufgrund dieser Anteile gibt es eine prozentuale Anteilnahme an der Lotterie, wer den nächsten Block erstellen darf. Im Gegensatz zu PoW wird hier keine elektrische Energie benutzt um ein Rätsel zu lösen.

2.6.3. Proof-of-Authority (PoA)

Proof-of-Authority (PoA) ist eine modifizierte Form von Proof-of-Stake (PoS). Anstatt des einfrieren "Staking" von Anteilen, welches zum Konsens im Netzwerk führt, wird lediglich nur ein Autoritätsnachweis benötigt.
Dieser Nachweis wird von anderen vertrauenswürdigen Autoritäten vergeben. Die Gründe für diesen Konsens Algorithmen liegen in der Tatsache, dass es Kommunikationsnetze geben muss, welche von mehreren Autoritäten kontrolliert und instand gehalten werden müssen.
Ein Beispiel hierfür ist das PoA Blockchain Projekt der EWF (Energy Web Foundation). Dieses Projekt hat sich der Aufgabe verschrieben, die Energiewirtschaft mit Hilfe der Blockchain Technologie weiter zu automatisieren und digitalisieren. Als Autoritäten dienen in der Blockchain namhafte Energieversorger. Gerade für Konsortium Blockchains ist dieser Konsenses Algorithmus interessant. Um Beeinflussungen zu verringern, darf eine Autorität nur in bestimmten Abständen agieren.
Für die Implementierung von PoA Algorithmen gibt es mehrere Lösungen. Eine davon ist Aura, entwickelt von Parity die folgendermaßen funktioniert. Neue Blöcke dürfen von einer Gruppe Autoritäten geschrieben werden, welche in einem Smart Contract eingetragen sind. Das Signieren und Erzeugen von Blöcken geschieht rundenbasiert. Jede Autorität erhält einen bestimmten Zeitraum, um einen Block zu erzeugen und zu signieren. Ist die Autorität offline oder reagiert nicht, wird diese übersprungen. Die Autorität welche den Block signiert hat, wird Primär ("primary") genannt. Um zu entscheiden wer zurzeit die aktuelle Primär ist, speichert jede Autorität den sogenannten Schritt ("step"). Der Schritt ist ein Wert welcher auf dem Unix Zeitstempel (abgkz. ut) basiert. Der aktuelle Schritt wird durch Teilen des Zeitstempels mit der geforderten Blockzeit (abgkz. bt) berechnet. Danach wird der Schritt durch eine Modulo Operation mit der Anzahl der Autoritäten (abgkz. a) berechnet. Eine Beispiel Rechnung sehe wie folgt aus:

$$bt = 3$$

$$ut = 1504857547$$

$$a = 6$$

$$step = \frac{ut}{bt} = \frac{1504857547}{3} = 501619182 \tag{2.3}$$

$$p \equiv step \pmod{a} \equiv 501619182 \pmod{6} \equiv 0 \tag{2.4}$$

Autorität 0 wurde bei dem Timestamp als “primary“ ausgewählt. Eine Blockzeit später würde die Autorität 1 als “primary“ ausgewählt werden.
Um jedoch sicherzustellen dass alle Autoritäten auf der selben Kette (Chain) sind, besitzt Aura eine Ketten Wertung (Chain scoring). Sollte eine Autorität offline sein bzw. wird ihre Verbindung unterbrochen, folgt sie nach einem wieder verbinden weiter ihrem Protokoll und versucht fälschlicherweise einen Block zu produzieren. Um dieses zu verhindern, muss sich die Autorität für die aktuelle und richtige Kette (Chain) entscheiden.
Dieses wird über eine Ketten Wertung realisiert. Die Rechnung sieht wie folgt aus :

$$score = U128_{max} \cdot h - s \tag{2.5}$$

h ist die Größe der Kette, s ist der Schritt des größten Blockes.
$U128_{max}$ ist dafür zuständig, dass h immer Vorrang vor s hat.
Das Prinzip ist, dass die unterbrochene Kette weniger Blöcke produziert hat als die Haupt-Kette (Main Chain), wodurch für die Autoritäten klar gestellt ist, welche die Haupt Kette ist.

2.7. Ethereum

Ethereum ist ein Projekt welches sich als Ziel gesetzt hat, eine Blockchain-basierte Plattform für Smart Contracts 2.8 und dezentralen Applikationen 2.9 zu schaffen. Die Einheit der Kryptowährung wird als Ether bezeichnet und fungiert in erster Linie als internes Zahlungsmittel. Das Ausführen von Operationen, das Versenden von Transaktionen, das Erstellen und Ausführen von Smart Contracts kostet in der Ethereum Umgebung Gas, einer zusätzlichen Währung die an Ether gekoppelt ist. Ethereum besitzt somit zum einen Ether, welches die Währung darstellt und zum anderen Gas, welches zum ausführen von Operationen benötigt wird. Gas und Ether sind jedoch untrennbar miteinander verbunden. Der Benutzer ist nur in der Lage Ether zu erwerben. Die Besonderheit von Ethereum ist, dass es auf einer vollturing fähigen Programmiersprache *Solidity* beruht welche auch für das programmieren von Verträgen (Smart Contract) genutzt wird.

2.8. Smart-Contracts

Smart Contracts wurden erstmals 1996 von Szabo [1996] in dem Artikel „Smart Contracts: Building Blocks for Digital Markets“beschrieben. Laut Definition nach Mitschele [2018] im Gabler Wirtschaftslexikon ist ein Smart Contract ein „Elektronischer Vertrag, der hinterlegte Regeln automatisch überwacht und definierte Aktionen bei Vorliegen eines Trigger-Events selbsttätig ausführen kann“.
Hiermit können komplexe Wenn-Dann-Beziehungen auf einer Blockchain dargestellt werden. Zwei Vertragsparteien treffen eine verbindliche Vereinbarung, sobald eine bestimmte Bedingung erfüllt ist, wird automatisch eine bestimmte Konsequenz ausgeführt.
Auch hier kann jede durchgeführte Aktion oder Transaktion über die Blockchain nachvollzogen werden. Es ist aber wichtig zu beachten, dass es sich bei Smart Contracts keineswegs um Verträge in einem zivilrechtlichen Sinne handelt. Smart Contracts können sehr unterschiedlich ausgestaltet werden und haben ein breites Spektrum von Einsatzmöglichkeiten. Es gibt dementsprechend verschiedene Ausprägungen von Smart Contracts, die nach ihrer Komplexität andere Aufgaben übernehmen können.
Die einfachste Variante ist die autonome Ausführung einer Zahlung, sobald eine bestimmte Bedingung erfüllt ist. Oftmals wird hier der Vergleich mit einem Getränkeautomaten herangezogen.
Weitaus komplexere Anwendungen sind dezentrale Applikationen (dApps) und dezentrale autonome Organisationen (DAOs), die in den folgenden Abschnitten separat erläutert werden. Besonders interessant ist die Kombination von Smart Contracts und IoT, da hierdurch neue Geschäftsmodelle wie „Micropayments“ oder „Pay-per-Use“ ermöglicht werden.
Zukünftig wäre es denkbar, dass das bloße Betreten eines Hotelzimmers die Erstellung eines Beherbergungsvertrags auslöst. Dies lässt sich über eine Chipkarte oder intelligente Türschlösser realisieren wie es das Startup Slock.it aus Mittweida versucht zu realisieren. Die Abwicklung der Buchung und Bezahlung soll vollkommen automatisiert geschehen.
Damit ein solcher Smart Contract ordnungsgemäß ausgeführt wird, ist eine externe Informationsquelle notwendig, die angibt, ob notwendige Vertragsbedingungen wirklich erfüllt sind oder nicht. In diesem Kontext wird eine solche Quelle als Oracle bezeichnet. Oracles können rein softwarebasiert sein oder auch auf Hardware zurückgreifen. Ein Software Oracle könnte etwa auf eine Datenbank mit aktuelle Veranstaltungen zugreifen und damit einen Smart Contract aktivieren, der den Zimmerpreis einer Wohnung automatisch anpasst.
Als Hardware Oracle könnte bspw. ein Smart Meter dienen, indem die Verbrauchsdaten in Echtzeit an einen Smart Contract übermittelt werden, der wiederum automatische Zahlungen vornehmen kann.

2.9. Dezentrale Applikationen

DApps (Dezentrale Applikationen) sind Applikationen, die nicht von einem einzelnen PC, sondern in einem P2P-Netzwerk ausgeführt werden. Es kann sich dabei um ganz normale Webseiten handeln, die für den Endnutzer nicht als solche erkennbar sind. Der wesentliche Unterschied ist, dass DApps keine klassische Datenbank nutzen, sondern mithilfe von Smart Contracts auf eine Blockchain zugreifen können. Der Smart Contract ist aber nur ein Teil einer DApp. Sie umfasst sowohl Frontend als auch Backend einer Anwendung.
Beispiele für DApps sind die Transaktionsplattform von Propy oder das Universal Sharing Network von slock.it. Große Daten sollten nicht in einer Blockchain gespeichert werden, da es das P2P Netzwerk negativ beeinflussen würde.
Beispielsweise könnte sich die Latenz des P2P-Netzwerk stark erhöhen. Die Größe der Blockchain würde sich extrem schnell erhöhen und nur wenige Nutzer wären in der Lage die gesamte Transaktionshistorie zu speichern.
Um diese Problematik zu umgehen gibt es zwei sinnvolle Entwicklungen, zu einem Inter-Planetary File System (IPFS) und zum anderen Swarm, beide sind im Grunde dezentrale Speichermöglichkeiten für große Daten.

2.10. Dezentrale autonome Organisationen (DAOs)

Eine Weiterentwicklung von Smart Contracts und DApps sind sogenannte dezentrale autonome Organisationen (DAOs). Diese unterscheiden sich grundsätzlich vom hierarchischen Aufbau von konventionellen Unternehmen.
Eine DAO verfügt nicht über Manager oder CEOs, sondern wird allein durch ihre Anteilseigner gesteuert. Entscheidungen werden entweder automatisiert oder per Abstimmung getroffen. Die Anteile einer DAO können in Form von Token erworben werden. Die Besitzer erhalten damit meist umfassende Mitspracherechte und werden am Gewinn der Organisation beteiligt. DAO-Token können als plattforminternes Zahlungsmittel dienen oder auch extern gehandelt werden.
DAOs ermöglichen eine Fülle an komplexen Geschäftsmodellen.Beispielhaft hierfür ist das Projekt Siemens Hutten-DDO, indem die Mitarbeiter von Siemens Anteile besitzen, um zu entscheiden wo Spendengelder des Unternehmens transferiert werden.

2.11. Ethereum-Infrastruktur

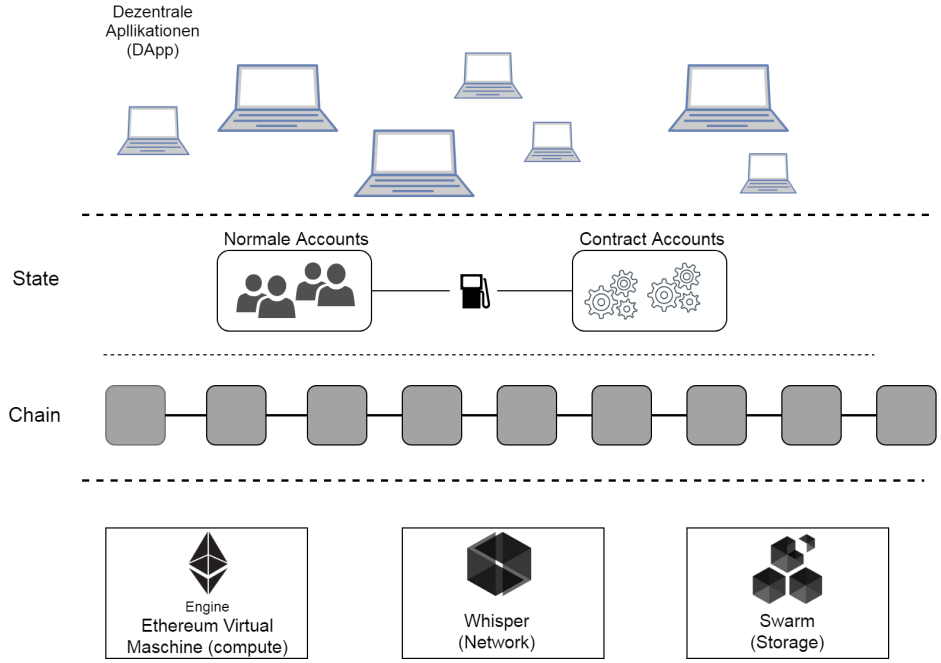

Abbildung 2.11.: Infrastruktur Ethereum

Die Infrastruktur welche zur Ausführung von Transaktionen jedoch besonders für Smart Contracts 2.8 benötigt wird, besteht aus der *Ethereum Virtual Maschine (EVM)*, welche als Laufzeitumgebung für Smart Contract dient, dem Kommunikationsnetzwerk *Whisper* für Smart Contracts und Accounts und dem dezentralen Speicher *Swarm* (siehe Abb. 2.11).
Accounts und Contract Accounts unterscheiden sich nicht in Ethereum. Sie besitzen verschiedene Attribute wie dem Kontostand (Balance), einem Zähler für Transaktionen, dem Contract Code und Zustände (States). Damit alle Teilnehmer des Netzwerkes die Zustände (World State) verfolgen können gibt es in Ethereum eine Weiterentwicklung des Merkle Hashbaumes, nämlich dem Patricia-Trie, welcher komprimierter die deutlich größeren Daten des Ethereum Protokoll aufnimmt und dadurch schneller agiert als das Bitcoin Protokoll. Übergänge von Zuständen werden durch Transaktionen bewerkstelligt.

2.12. Blockchain im Energiesektor

Wie in der Einleitung angesprochen gibt es einige Vorteile den Energiesektor weiter zu automatisieren und für kleinere Nutzer zu öffnen. Die dena GmbH [2016] hat in einem Umfragepapier mit Geschäftsführern großer deutscher Unternehmen in der Energiewirtschaft die Frage aufgeworfen, in welchen speziellen Bereichen die Blockchain einen Vorteil bringen könnte. Die Abb. 2.12 stellt in groben Zügen das Ergebnis dar. Es wird deutlich, dass gerade im P2P-Energiehandel, als Handelsplattform, im Netzmanagement, dem Zählwesen und dem Datentransfer ein Anwendungsfall im Energiesektor in Frage kommen könnte.

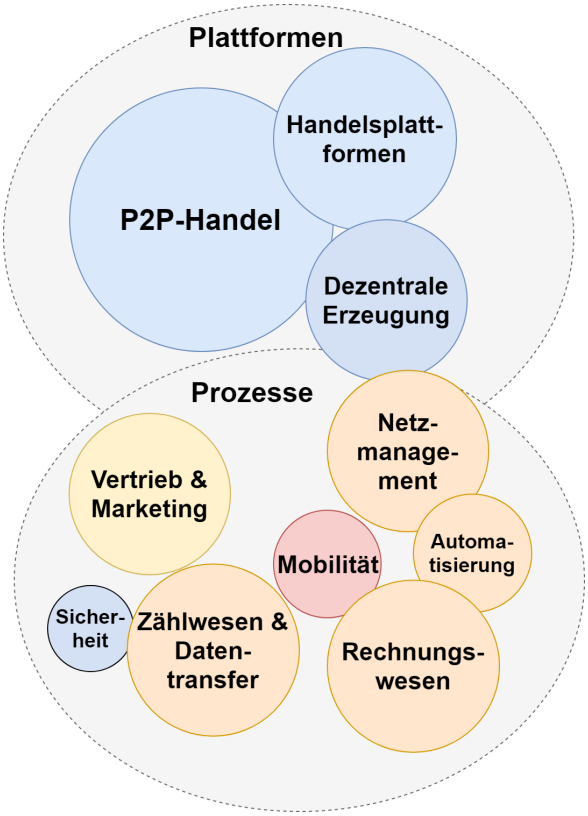

Abbildung 2.12.: Umfrage Ergebnisse potenzieller Anwendungsfällen der Blockchain im Energiesektor dena GmbH [2016]

2.13. Smart Grid Architecture Model (SGAM) Framework

In der Abbildung 2.13 wird das Grundmodell des SGAM-Framework dargestellt, welches erstmalig in CEN und ETSI [2012] definiert wurde und als Grundlage mit dieser Theorie dient. Es gibt fünf Interoperabilität Ebenen, die als Schichten (Layer) bezeichnet werden.

- Component Layer - Physikalische Verteilung der beteiligten Komponenten
- Communication Layer - Protokolle und Mechanismen für Informations-Austausch
- Information Layer - Übertragene Informations Objekte und Datenmodelle
- Function Layer - Funktionen und Services zwischen Komponenten
- Business Layer - Business View, z.B. ökonomische und regulatorische Aspekte

Die Ebenen interagieren miteinander und ergeben zusammenhängend eine kompakte und gute Darstellung des Anwendungsfalls. Somit können fehlende Komponente schnell und effektiv erkannt und behoben werden.
Die fünf Ebenen haben die selbe Struktur und sind auf der x-Achse in „Domains“ unterteilt. Die sich wie folgt im Themen-Bereich der Elektrische Energieversorgung beschreiben lassen:

- Generation - großvolumige Energieerzeugung
- Transmission - Übertragungsnetz
- Distribution - Verteilnetz
- Distributed Energy Ressources (DER) - verteilte Energieerzeuger
- Customer - Kunden bzw. Prosumer

Auf der y-Achse werden „Zones“ abgebildet, welche von der Automatisierungspyramide adaptiert wurden jedoch mit etlichen Veränderungen. Die Aufteilung sieht folgendermaßen aus:

- Process - physische Ausstattung der Energieversorgung
- Field - Schutz, Steuer und Überwachungselemente
- Station - Daten Aggregation der Field Zone z.B. lokales SCADA System
- Operation - übergeordnete Steuerung des Energiesystems, z.B. Verteilnetzsteuerung
- Enterprise - kommerzielle und organisatorische Prozesse
- Market - Markt Operation und Interaktion

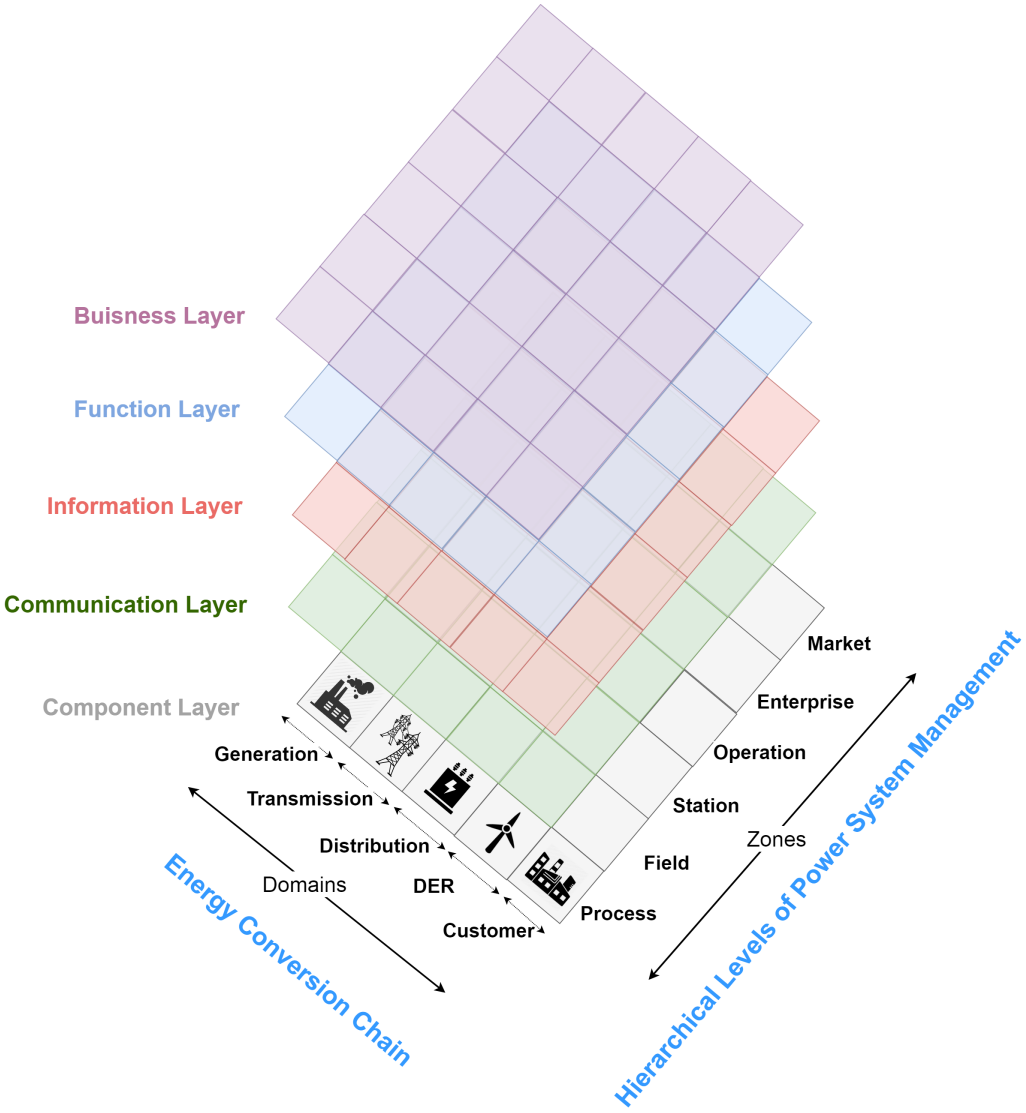

Abbildung 2.13.: SGAM Framework Allgemein CEN und ETSI [2012]

3. Anforderung und Analyse

3.1. Anwendungsfall OTC Energiehandel

In Deutschland gibt es mehrere Möglichkeiten des Stromhandels. Zu den bekanntesten zählen Strombörsen wie die europäische EEX oder anderen nationale und internationale Börsen. Der größte Teil des Handels, erfolgt außerhalb der Börsen über den OTC Energiehandel. Unter diesem versteht man einen bilateralen und direkten Handel zwischen zwei Partnern. Durch diese Art des Handel kann deutlich flexibler und günstiger Energie gehandelt werden. Diese Eigenschaften machen für unseren Anwendungsfall Sinn, da hier die regulatorischen Hürden nicht groß sind.
Aus diesem Grund wurde entschieden, einen Peer-to-Peer (P2P) Energiehandel, ein Zählwesen und den Datentransfer mit der Blockchain näher zu analysieren und einen passenden Anwendungsfall zu konzipieren. Die Blockchain dient als verteilte, dezentrale und kryptografische verkette Datenbank, die über Methodiken verfügt, dass unbekannte Nutzer vertrauenswürdig miteinander agieren können. Der Anwendungsfall ist dementsprechend eine dezentrale OTC (Over-the-counter) Energiehandels-Plattform. Der Schwerpunkt ist die Entwicklung der Smart Contracts, welche die gesamte Logik der Automatisierung bereitstellen.

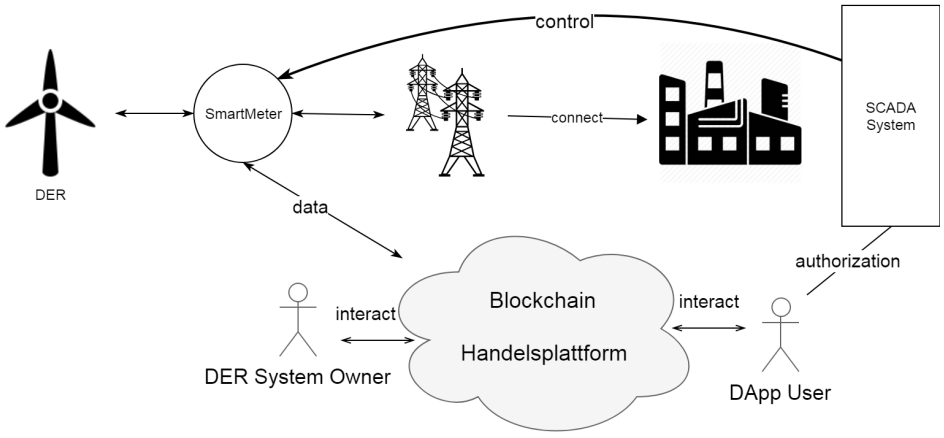

Abbildung 3.1.: Einfaches Diagramm des Anwendungsfall OTC Energiehandel

Im Anwendungsfall (vgl. Abb. 3.1) gibt es zwei verschiedene Akteure. Zu einem der DER (Distributed Energy Resources) System Owner, welcher einen kleinst Energieerzeuger darstellt. Zum anderen einen dezentralen Applikations-User (DApp-User) der Energie benötigt.

Die Akteure dieser Plattform können einerseits ihre Energieanlage zum Verleihen anbieten, anderseits besteht die Möglichkeit des Ausleihen von Anlagen. Die Leihmöglichkeit der Plattform ist dementsprechend indirekt an eine Energiedienstleistung gekoppelt, um den Teilnehmern große Flexibilität zu gewährleisten. Die Verleih Plattform wird über Token realisiert, die den Zugriff auf eine Anlage erlauben. Dadurch wird gewährleistet, dass eine Energiedienstleistung punktgenau durchgeführt wird. Der Zugriff auf Anlagen ist zeitlich begrenzt und nur auf die Tätigkeit des Schalten frei.
Konzipiert wird jedoch nicht der Zugriff auf eine Anlange sondern der Zugriffstoken. Dieser Token würde auf einer entsprechenden und eingeschränkten Supervisory Controll and Data Acquisition (SCADA) Oberfläche einer DER-Anlage verwendet werden.

Angebote und Gesuche werden über Smart Contracts realisiert und in eine Blockchain eingefügt. Dadurch ist es für jeden Nutzer des Netzwerkes möglich, auf diese Applikation zuzugreifen.

Auch soll angemerkt werden, dass dieser Anwendungsfall im Kontext des OS4ES-Projektes OS4ES [2017] entwickelt wurde und als Machbarkeitsstudie für eine Blockchain-basierte Lösung ausgelegt wird. Bei einem Teil des OS4ES Projekt, welches unter anderem am MMLAB der HAW mitkonzipiert wurde, handelt es sich um ein dynamisches Datenregister, welches als Infrastruktur zur Bereitstellung von Energieflexibilitäten dienen soll. Gerade die entwickelte Möglichkeit, dass ein Zugriff auf eine DER Anlange über ein Ausleihen konzipiert werden könnte, wurde hier adaptiert.

3.2. Entwicklungstools und Techniken

Der Anwendungsfall für den OTC-Energiehandel wurde im vorherigen Kapitel kurz beschrieben. Hier wird eine Auswahl der geeigneten Blockchains und Ethereum Clients getroffen, sowie Tools, Techniken und Frameworks beschrieben. Diese werden im Kapitel 4 und 6 benötigt, um die Implementierung und Verteilung, das ausführen und testen der DApp zu realisieren.

3.2.1. Blockchain Protokolle

Zur Auswahl stehen verschiedene Blockchain Protokolle, die für einen OTC-Energiehandel in Erwägung gezogen werden können. Die Auswahl an Protokollen ist nicht vollständig und dient hier eines kleinen Vergleichs. In der Tabelle 3.1 werden das Ethereum, Hyperledger Fabric und Tendermint Protokoll miteinander in Vergleich gestellt.

	Ethereum	Hyperledger Fabric	Tendermint
Konsens Algorithmus	PoW/ PoS/ PoA	PBFT [2016]	BFT
Währung	Ja	Nein	Ja
Smart Contract	Solidity	alle Sprachen	alle Sprachen
Beschränkungen	Öffentlich/ Privat/ Virtuell	Privat	Öffentlich/ Privat

Tabelle 3.1.: Eigenschaften von Blockchain Protokollen

Die Entscheidung fiel auf das Ethereum Protokoll, da es gerade zur Entwicklung von Smart Contracts geeignet ist. Die Programmiersprache von Ethereum ist Solidity und zielt mit den entwickelten Methoden und Funktionen auf die Programmierung von Smart Contracts, welches ideal für die Entwicklung der DApp ist. Ein anderer Punkt ist die Tatsache, dass es für die Migration und Kompilierung diverse Frameworks zur Auswahl gibt. Außerdem verfügt das Ethereum Protokoll über eine virtuelle Blockchain namens *Ganache*, die speziell für das Testen und Entwickeln von Anwendungsfällen konzipiert wurde. Darüber hinaus ist es mit einem relativen kleinen Umstand möglich, entwickelte Smart Contracts auf andere Ethereum Typen zu übertragen und auszuführen. Daraus ergibt sich für die späteren Tests in Kapitel 6 die Möglichkeit, zwei Ethereum Blockchains miteinander zu vergleichen.

3.2.2. Ethereum Clients

In diesem Kapitel werden in der Tabelle 3.2 die populärsten Ethereum Clients beschrieben, aufgrund dessen eine Auswahl nach der Auflistung erfolgt.

Client	Sprache	Beschreibung
go-ethereum	Go	geth ist ein vollständiger Ethereum Knoten für Mac OSX, Windows und Linux/Unix sowie Raspberry Pi (ARM). Nutzt den MIST Browser und MetaMask. Besitzt eine Dapp-API. Es wird Mining unterstützt.
Parity	Rust	Parity eth ist ein vollständiger Ethereum Knoten für Mac OSX, Windows und Linux/Unix mit Fokus auf Performance. Nutzt Parity Browser. Besitzt eine Dapp-API, Multi-Wallet und POA Algorithmus.
cpp-ethereum	C++	cpp-eth ist ein vollständiger Ethereum Knoten für Mac OSX, Windows und Linux/Unix. Es wird Mining unterstützt.
pyethapp	Python	pyethapp ist ein vollständiger Ethereum Knoten für Mac OSX und Linux/Unix. Sehr lesbare Entwicklung. Für Forschung und akademische Zwecke geeignet. Nicht performant.
Ganache	JavaScript	Ganache ist ein simulierter Ethereum Knoten für Mac OSX, Windows und Linus/Unix. Speziell für Entwickler konzipiert. Vereinfachte Blockchain Einstellmöglichkeiten. Verfügt über ein simuliertes Mining.

Tabelle 3.2.: Auflistung von Ethereum Clients (vgl.Foundation [2018])

Damit der Aufwand eines lauffähigen P2P Netzwerk möglichst klein gehalten wird, wurde aus der Auswahl die Ganache Ethereum Blockchain genutzt (siehe Tabelle 3.2). Ganache ist eine virtuelle Ethereum Blockchain, welche das schürfen (minen) von Blöcke und das Bestätigen von Transaktionen simuliert. Diese visuelle Simulation bietet die Möglichkeit, ein System schnell aufzusetzen, sowie erweiterte Mining und Gas Einstellungen vorzunehmen. Außerdem werden mehrere Accounts mit vorhandenen ether zum Testen bereitgestellt. Auch die Vereinfachung des Debuggen durch die integrierte Protokollausgabe und das erweiterte Einblicken in Blöcke und Transaktionen ist für Entwickler sehr geeignet.

Um im Kapitel 6 einen Vergleich zwischen unterschiedlichen Ethereum Clients aufzustellen, wird eine zusätzlicher Client genutzt (vgl. Tabelle 3.3). Ein privater geth Ethereum Client mit 2-5 Teilnehmer Knoten (Member Nodes), einem Boot Knoten (Boot Node) und bis zu 5 Schürf Knoten (Miner Nodes). Um den Umstand des schürfens (minen) möglichst gering

zu halten, wurde im Einstellungsblock (Genesis Block) eine geringe Schwierigkeit (difficulty) eingestellt.

Netzwerk	Port	Netzwerk ID	Gas Limit
geth Privat	8101	1024	9000000
Ganache	7545	5777	6721975

Tabelle 3.3.: Eigenschaften von Blockchain Clients

Das Gas Limit in der Tabelle 3.3 definiert wie groß die ausführbaren Programme sind. Außerdem wird darstellt, auf welchem Port die Netzwerk Clients agieren und ihre zugehörige ID.

3.2.3. Truffle Framework

Truffle ist eine Entwicklungs-und Testumgebung für Ethereum. In dieser Arbeit wird Truffle zu Entwicklungszwecken verwendet, jedoch besonders für das automatisierte Testen der Smart Contracts. Truffle hat folgende Eigenschaften Coulter [2015–2018]:

- Kompilieren, verbinden, verwalten und entwickeln von Smart Contract.
- Automatisches Testen von entwickelten Contracts.
- Verwaltung zum entwickeln auf verschiedene Ethereum Netzwerke.
- Interaktive Konsole zur direkten Kommunikation.

3.2.4. Web3.js

Der Ethereum-Knoten (Node) bietet eine RPC (Remote Procedure Call) Schnittstelle. Über diese Schnittstelle wird der Zugriff auf Interaktion mit DApps, der Blockchain und Funktionen der Blockchain gewährleistet. Die Verwendung der RPC-Schnittstelle ist sehr fehleranfällig, besonders beim arbeiten mit der Binärschnittstelle (ABI). Um dieses zu umgehen, wurde die Web3.js JavaScript-Bibliothek konzipiert. Sie bietet eine weitere Abstraktionsschicht und arbeitet auf dem Ethereum RPC. Des Weiteren ist benutzerfreundlich und weniger fehleranfällig. Web3 beinhaltet das eth Objekt - web3.eth (speziell für Ethereum Blockchain Interaktionen) und das shh Objekt - web3.shh (für Whisper Interaktion) w. team [2017–2018].

4. Design und Implementierung

, welcher den Namen „AssetToken“ trägt

Im Proof of Concept des Anwendungsfall (siehe Abb. 4.1) wird eine Plattform entwickelt, welche auf Smart Contracts basiert, die in einer simulierten Ethereum Blockchain betrieben wird.

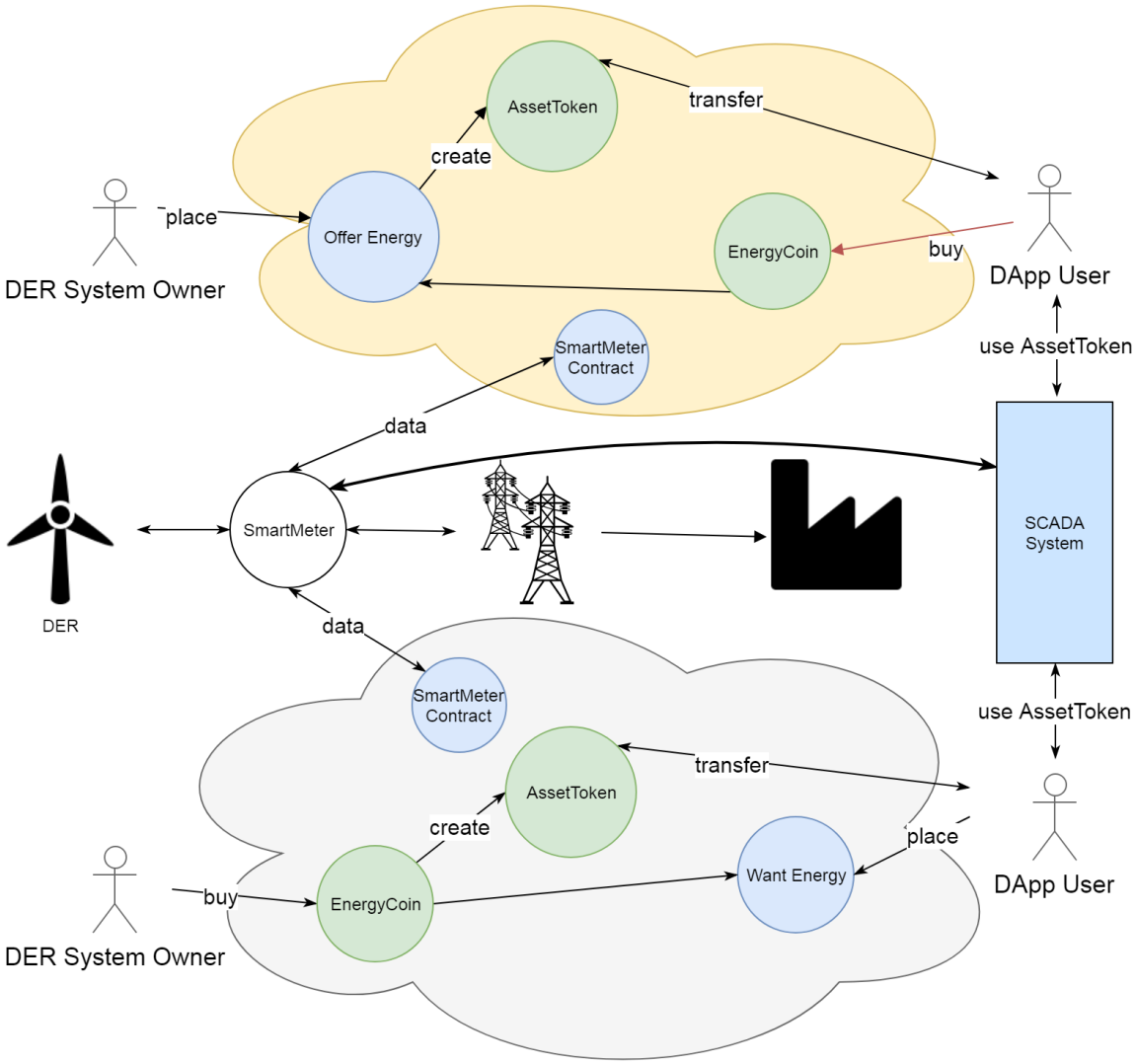

Abbildung 4.1.: Schematisches Diagramm des Anwendungsfall OTC Energiehandel

Im oberen Beispiel des Diagramm 4.1 (gelbe Wolke) platziert ein DER System Owner ein Angebot, dieses beinhaltet:

- Art des DER Systems (Windenergieanlage, Photovoltaikanlage etc.)
- Energiemenge in Kilowattstunde (kW/h) bzw. den Preis der Dienstleistung in Energy-Coins
- Zeitpunkt und Dauer der Dienstleistung

Die genannten Daten werden beim platzieren des Angebots in einem dezentralen Webspace gespeichert werden und der Link bzw. Hashcode an den Smart Contract „Offer Energy“ übergeben. Dadurch wird dieser in die Ethereum Blockchain geschrieben und für jeden Nutzer sichtbar. Wird der Smart Contract „Offer Energy“ von einem Akteur (hier *DApp User*) ausgewählt, prüft dieser an erster Stelle, ob der Akteur genügend „EnergyCoins“ hat, um die angebotene Menge zu zahlen. Der „EnergyCoin“ ist die interne Währung des Handelsmarktes und kann von jedem Benutzer gekauft bzw. bezogen werden. Sobald die Prüfung erfolgreich abgeschlossen wurde, aktiviert der Smart Contract „Offer Energy“ einen weiteren Smart Contract mit dem Namen „AssetToken“, welcher die Energie-Anlage und das Angebot digital darstellt. Der „AssetToken“ wird an den *DApp User* übertragen. Die tatsächlich geleisteten Verbrauchs- und Erzeugerdaten werden über die intelligenten Stromzähler, Smart Meter, in „SmartMeter “ Contracts geschrieben, um die Daten für die Teilnehmer einsehbar und transparent zu gestalten. Im unteren Beispiel (graue Wolke) platziert ein *DApp User* ein Gesuche über den Smart Contract „Want Energy“. In diesem kann der Nutzer verschiedene Kriterien einfügen wie:

- Art des gewünschten DER Systems (Windenergieanlage, Photovoltaikanlage etc.)
- gewünschte Energiemenge in Kilowattstunde (kW/h) bzw. den Preis der gewünschten Dienstleistung in EnergyCoins
- Zeitpunkt und Dauer der Dienstleistung

Ein geeigneter *DER System Owner* hat die Möglichkeit auf das Gesuche zu reagieren, indem er mit dem Smart Contract „WantRequest“ interagiert und dieser seinen „AssetToken“ generiert und an den *DApp User* transferiert. Das physikalische Smart Meter überträgt die Verbrauchs- und Erzeugerdaten wie im anderen Beispiel in ein „SmartMeter “ Contracts. Für beide Beispiele gilt zusätzlich, dass bei einem erfolgreichen Handel Reputation, genauer gesagt „RepCoins“, verteilt werden. Hierüber wird die vertrauenswürdig von Teilnehmern dargestellt.

4.1. Gesamtarchitektur

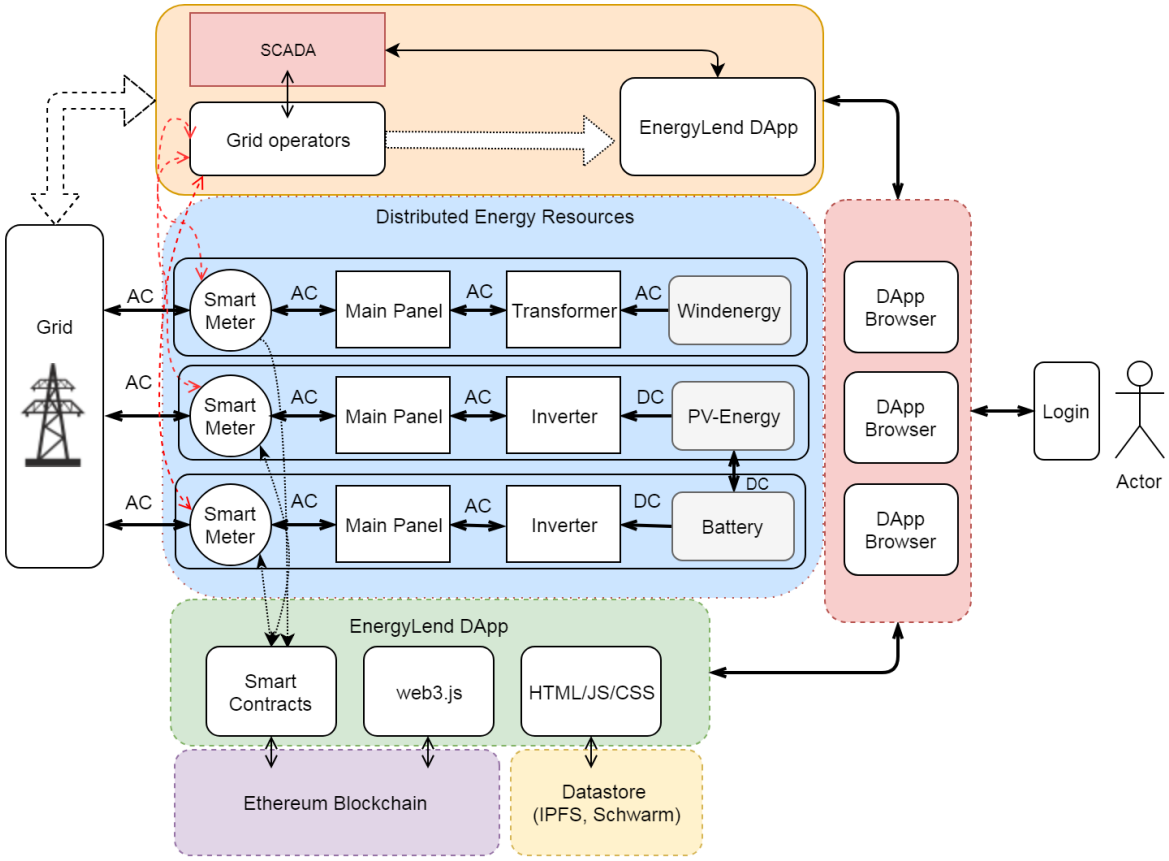

Abbildung 4.2.: Abbildung Gesamtarchitektur (einfache Darstellung)

Um die Architektur der Applikation zu verdeutlichen, wird in Abb. 4.2 die physikalische mit der software-technischen Modellierung auf einer gemeinsamen Ebene dargestellt und beschrieben. Die eindimensionale Darstellung dient des groben Verständnisses und wurde im Kapitel 5 differenziert dargestellt. Im Gegensatz zur Abb. 4.1 werden die einzelnen Aktionen der Teilnehmer nicht genau beschrieben, sondern ganz allgemein als Interaktion dargestellt. In Abb. 4.2 wurden einzelne Bereiche als voneinander getrennt agierende *Black Boxen* dargestellt, die über Schnittstellen verschiedenster Art miteinander kommunizieren. Zu diesen Black Boxen gehören:

- die Distributed Energy Ressources (DERs) in Blau
- die EnergyLend DApp in Grün
- die Ethereum Blockchain in Lila

- der DApp Browser in Rot
- den Netzbetreiber in Orange
- das Stromnetz in Weiß

Unter den Distributed Energy Ressources (DERs) werden beispielhaft eine Windenergieanlage (WEA), eine Photovoltaikanlage (PV) und ein Batteriespeicher aufgeführt.
Um eine WEA an das Stromversorgungsnetz anzuschließen, benötigt es jedoch zusätzliche Komponenten. Exemplarisch wäre ein Transformator, welcher zum wandeln der erzeugten Wechselspannung in die benötigte Netzübertragungsspannung dient. Bei der PV-Anlage wird ein Solarwechselrichter (Inverter) benötigt, der aus einer erzeugten Gleichspannung eine dreiphasige Wechselspannung moduliert. Soll in ein Mittelspannungsnetz eingespeist werden, so benötigt die PV-Anlage einen zusätzlichen Transformator. Häufig werden PV-Anlagen mit Batteriespeichern betrieben, um Energie für den eigenen Verbrauch zu speichern. In der Gesamtarchitektur kann das Batteriesystem jedoch auch als DER genutzt werden. Somit wäre eine Dienstleistung wie das Speichern von Energie gewährleistet. Jedes aufgeführte DER verfügt über physikalische Sicherheitsvorkehrungen zum ein- und ausschalten der Systeme im Fehlerfall. Ein wichtiges Bauteil stellt das Smart Meter dar. Es dient als Schnittstelle zur EnergyLend DApp. Dadurch können gemessene Daten zur Abrechnung einer Dienstleistung genutzt werden. Auch soll über diese Schnittstelle die Möglichkeit vorhanden sein, als Nutzer einer Dienstleistung über das SCADA System auf den Leistungsfluss der DER zugreifen zu können. Auch der Netzbetreiber kann direkten Einfluss auf die Anlagen haben, um bei aufgetretenen Problemen auf unterster Stufe einzugreifen. Die EnergyLend DApp wird mit ihren Schnittstellen, die miteinander interagieren, dargestellt. Die Smart Contracts haben Zugriff auf die Smart Meter Daten und sind in der Blockchain verankert und ohne Weiteres nicht widerrufbar. Web3.js ist die Programmierschnittstelle (API), die mit der Blockchain und den Smart Contracts als Benutzer zu kommunizieren. Unter HTML/JS/CSS werden Programmiersprachen aufgezeigt die für eine Frontend Lösung geeignet sind, um den Benutzer die Handhabung zu erleichtern. Damit Daten wie Typ, Zertifikate, Bilder oder Videoaufnahmen der Anlage dezentral gespeichert werden gibt es verschiedene Möglichkeiten Zu einem bietet Ethereum die eigene Speichermöglichkeit Schwarm an. Zu anderem das externe Speicherprotokoll InterPlanetary File System (IPFS) welches deutlich ausgereifter und populärer ist. Nutzer kommunizieren mit der EnergyLend DApp über eine Plattform, welche in der Abb. 4.2 als DApp Browser beschrieben wird. Es gibt verschiedene eigenständige Ethereum Browser, die als DApp Browser bezeichnet werden. Den von der Ethereum Foundation entwickelten *Mist-Browser* und der *Parity Browser* sind die populärsten. Jedoch gibt es die Möglichkeit jeden Internet-Browser über eine Erweiterung namens *MetaMask* an ein Ethereum Netzwerk einzubinden. DApp Browser ermöglichen die Kommunikation mit einer Ethereum Blockchain.

4.2. Ethereum EnergyLend DApp

Die prototypische Anwendung nennt sich „EnergyLend“, da es sich um ein Verleihsystem handelt. Die spätere Notation kann nicht mit der englischen Übersetzung aus dem Finanzumfeld gleichgesetzt werden.

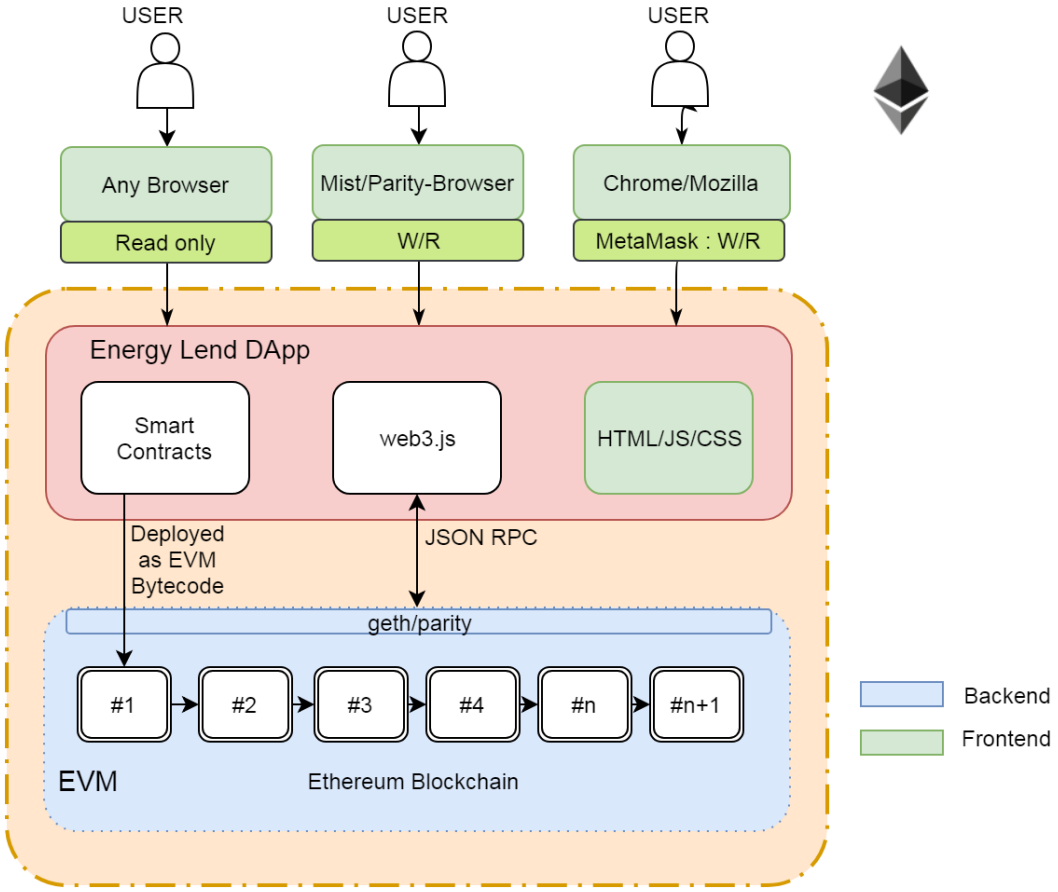

Abbildung 4.3.: Aufbau der EnergyLend DApp

Um als Nutzer mit einem Smart Contract und der Ethereum Blockchain zu kommunizieren, wird ein Web3.js (siehe 3.2.4) fähiger Browser wie Mist und Parity oder eine Browser Erweiterung namens MetaMask genutzt. Web3.js ist eine Javascript Bibliothek, die eine direkte HTTP oder IPC Verbindung zu einem Ethereum Node über einen Client herstellt, wodurch der Nutzer mit dem Netzwerk interagieren kann. Speziell wird über die JSON RPC Schnittstelle der Web3.js Bibliothek ein geth oder parity node angesprochen. Die Zuständigkeit hängt vom gewählten Web3 Browser ab. MetaMask und der Mist-Browser nutzen den geth Clients . Der Parity-Browser dagegen nutzt den parity Client (siehe 3.2.2). Ist der User mit

dem Netzwerk verbunden, kann dieser mit den Smart Contracts der EnergyLend DApp interagieren. Die Smart Contracts werden als Bytecode in der EVM ausgeführt und in die Blockchain geschrieben. Neben den Smart Contracts und der web3.js Bibliothek ist es auch möglich ein Front-End mit HTML, Javascript oder CSS für die DApp zu programmieren.

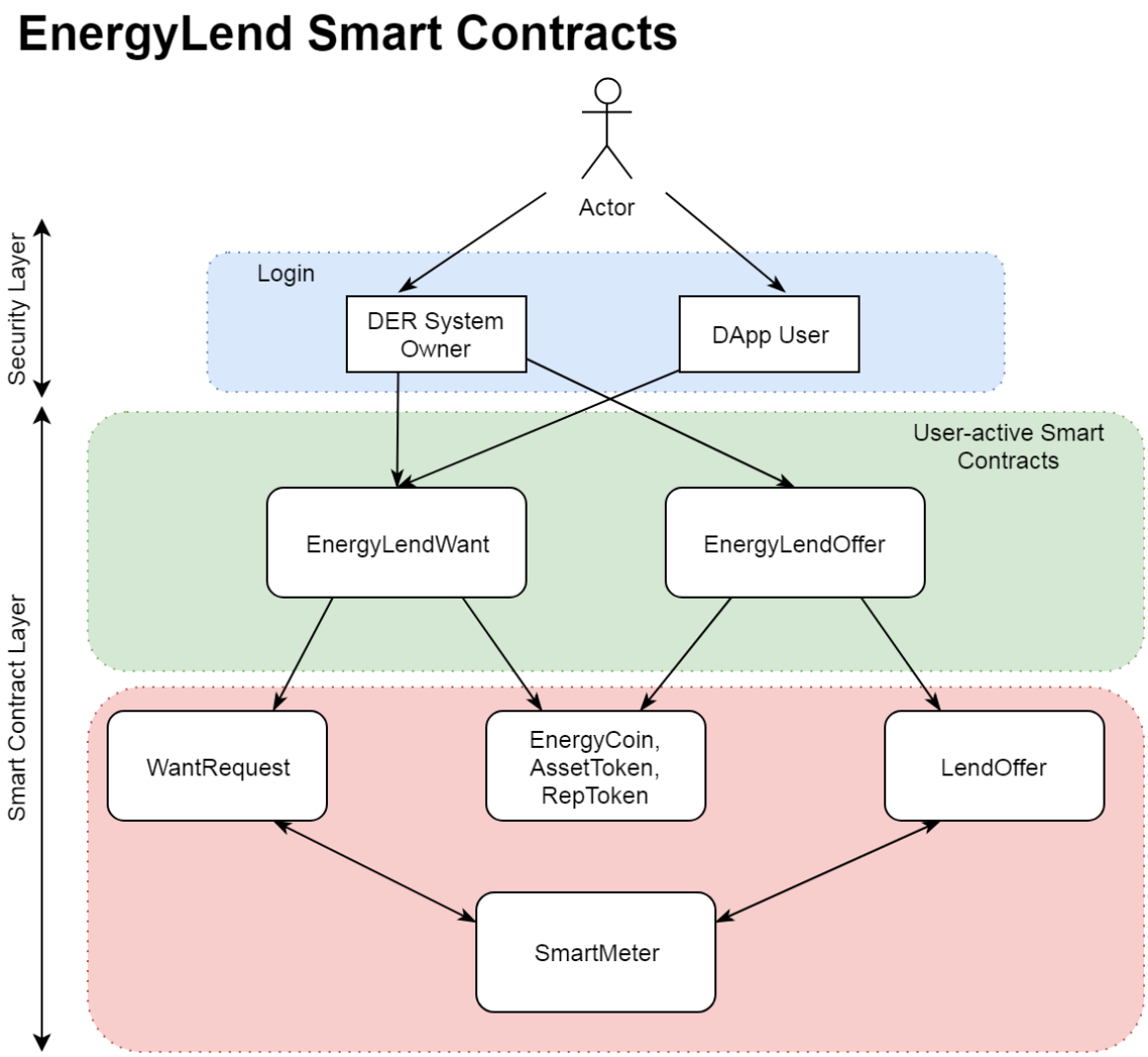

Abbildung 4.4.: Zusammenhang der Smart Contracts

In der Abbildung 4.4 wird dargestellt, wie die einzelnen Smart Contracts miteinander zusammenhängen. Um mit Smart Contracts zu interagieren, muss vorher aufgrund der Ethereum spezifischen Mechanik der Contract Code in ein Contract Erzeuger (creation) Block „geschürft“ werden. Die Blockgröße wird begrenzt über das Gaslimit im Ethereum Quell-

code. Aufgrund dessen wurden die Contracts in High- und Low-Level Contracts aufgeteilt. Dadurch wird die Skalierbarkeit der Applikation erhöht und die Erweiterbarkeit vereinfacht. Somit ergeben sich zwei Abläufe, die in den Abb. 4.13 und 4.14 näher dargestellt und erklärt werden. Die Smart Contracts „EnergyLendRequest“ und „EnergyLendOffer “ benötigen zum ausführen und ihren Prozessablauf zu folgen, Smart Contracts die sich hierarchisch unter ihnen befinden. Beispiele hierfür sind: „WantRequest“, „EnergyCoin“, „AssetToken“, „RepToken“ und „LendOffer“.
Im folgenden werden die grobe Funktionalität und das dazugehörige UML Diagramm der Smart Contracts dargestellt.

EnergyCoin und RepToken
Beide Smart Contracts sind Vererbungen des Smart Contracts „Token“ (Abb. 4.5), welcher das ERC20 Token Standard aufweist. Das ERC20 Token Standard wurde von der Ethereum Foundation konzipiert um einheitliche Token zu erzeugen. Dieser beinhalten sechs Funktionen und zwei Events um einheitlich über verschiedene Schnittstellen und Anwendungen angesprochen zu werden. Aus dezentraler Sicht spezifiziert der ERC20 Token folgende Punkte:

- Verteilung von Token von ihren Besitzer
- Verteilung von Token auf Wunsch vom Besitzer
- Zugriff auf Token Informationen
- Ereignisse des Token

Wird dem ERC20 Token Standard gefolgt, kann die Wirksamkeit und die Sicherheit von dezentralen Transaktionen erhöht werden. Ganz speziell hat der „EnergyCoin“ fortgeschrittene Operationen wie kaufen (*buy*) und verkaufen (*sell*) von „EnergyCoins“ im Austausch für *ether*. „RepToken“ dagegen ist kein übertragbarer Token, da es zur Überschreibung der Funktion *transfer* und *transferFrom* kommt.

Der RepToken steht für Reputationstoken, dieser Token dient dazu Nutzer und DER Betreiber zu bewerten und dadurch die Verantwortung und Zuverlässigkeit zu erhöhen. Nach jeder erfolgreichen Transaktion werden diese Token an die Akteure vergeben. Der EnergyCoin ist die Plattform eigene Währung.

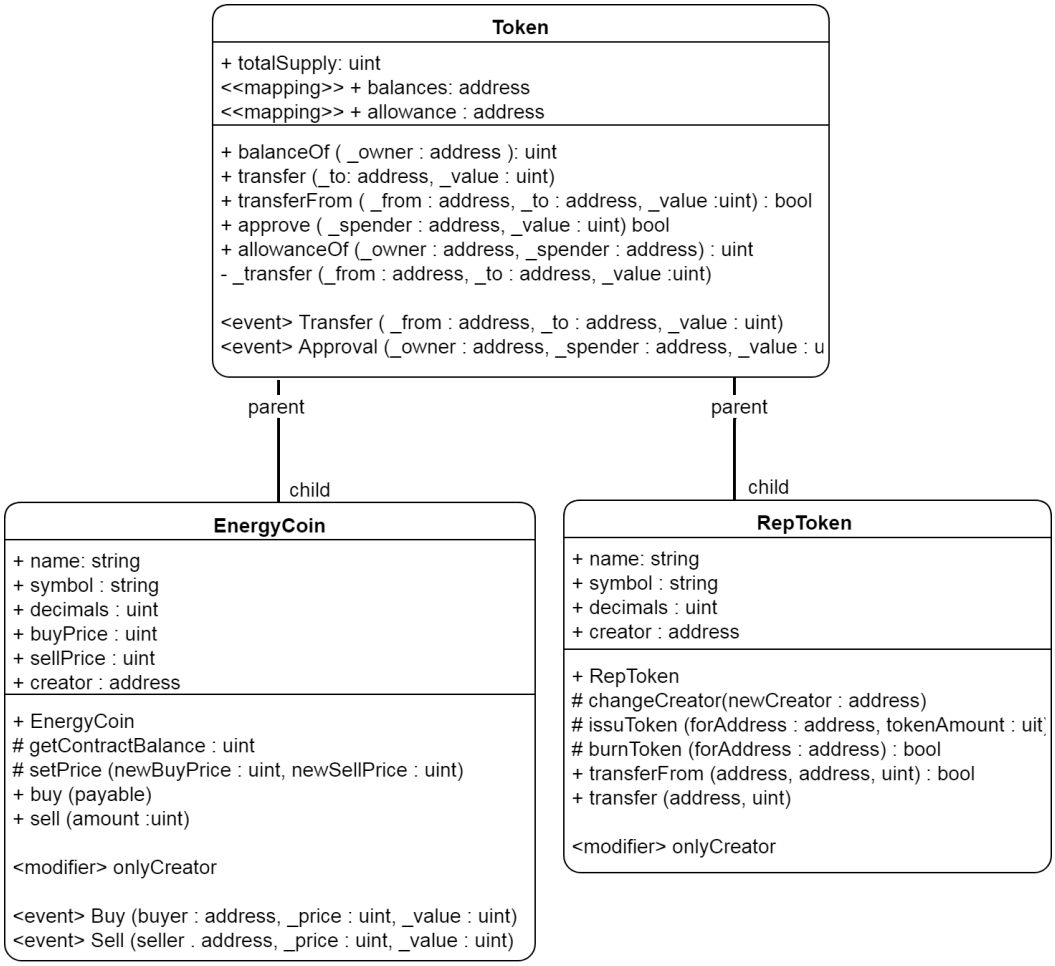

Abbildung 4.5.: UML Diagramm des EnergyCoin und RepToken

AssetToken

Anders als „EnergyCoin“ und „RepToken“ präsentiert dieser Token den Zugriff auf eine DER Anlage, also eines realen Objektes. Aufgrund dieser Eigenschaft bräuchte es einen Agenten, um die komplexe Beziehung zwischen dem virtuellen Token und der realen Anlage zu verwalten. Diesen sogenannten Internet of Thinks (IoT) Token zu entwickeln, wäre dementsprechend sehr komplex, aufgrund wessen es eine vereinfachte Lösung gibt. Um administrative Vorgänge wie Registrierung, Eigentumsrecht und Verteilung sicherzustellen, wird dem Erzeuger des Token eine zentrale Rolle im Smart Contract zugerechnet.

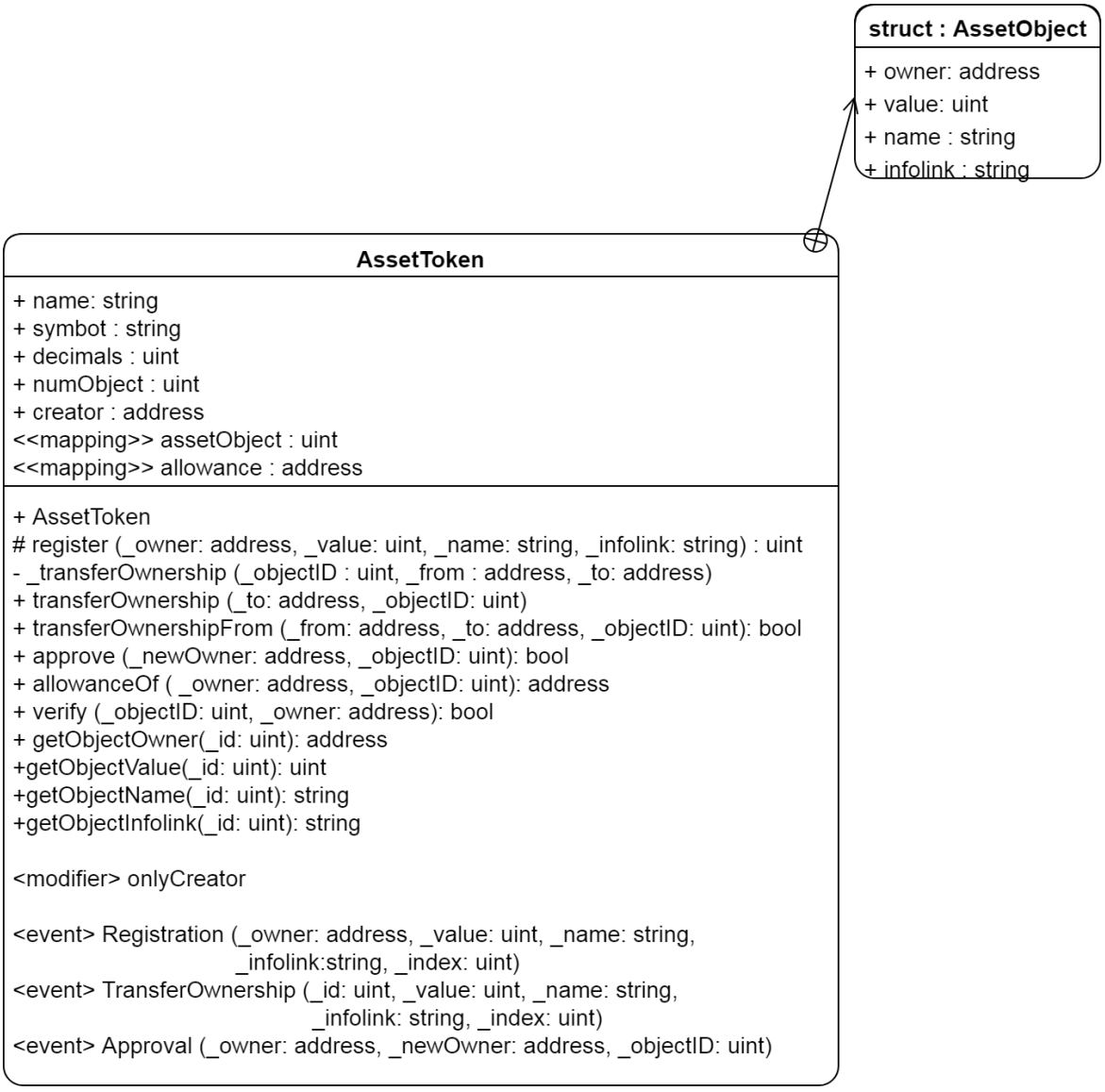

Abbildung 4.6.: UML Diagramm des AssetToken

EnergyLendRequest und WantRequest

Der Smart Contract „EnergyLendRequest“ stellt den Service für ein Energie Gesuche dar. In diesem Service werden alle Nutzer Adressen gespeichert, die mit dem Smart Contract „EnergyLendRequest“ in Aktion waren und sind. Außerdem werden alle Informationen zum DER System und zum Gesuche gespeichert. Für jeden Account wird einzeln der gesamte Verlauf im Solidity Datentyp *mapping* gespeichert und in die darunterliegende Blockchain geschrieben. Für jedes Gesuche wird ein eigenes „WantRequest“ Objekt erzeugt. Dies ermöglicht den weiteren Verlauf der Anzeige zu verfolgen und zum verwalten der Zustände des Objektes, welche in Abb. 4.8 dargestellt und weiter verdeutlicht werden.

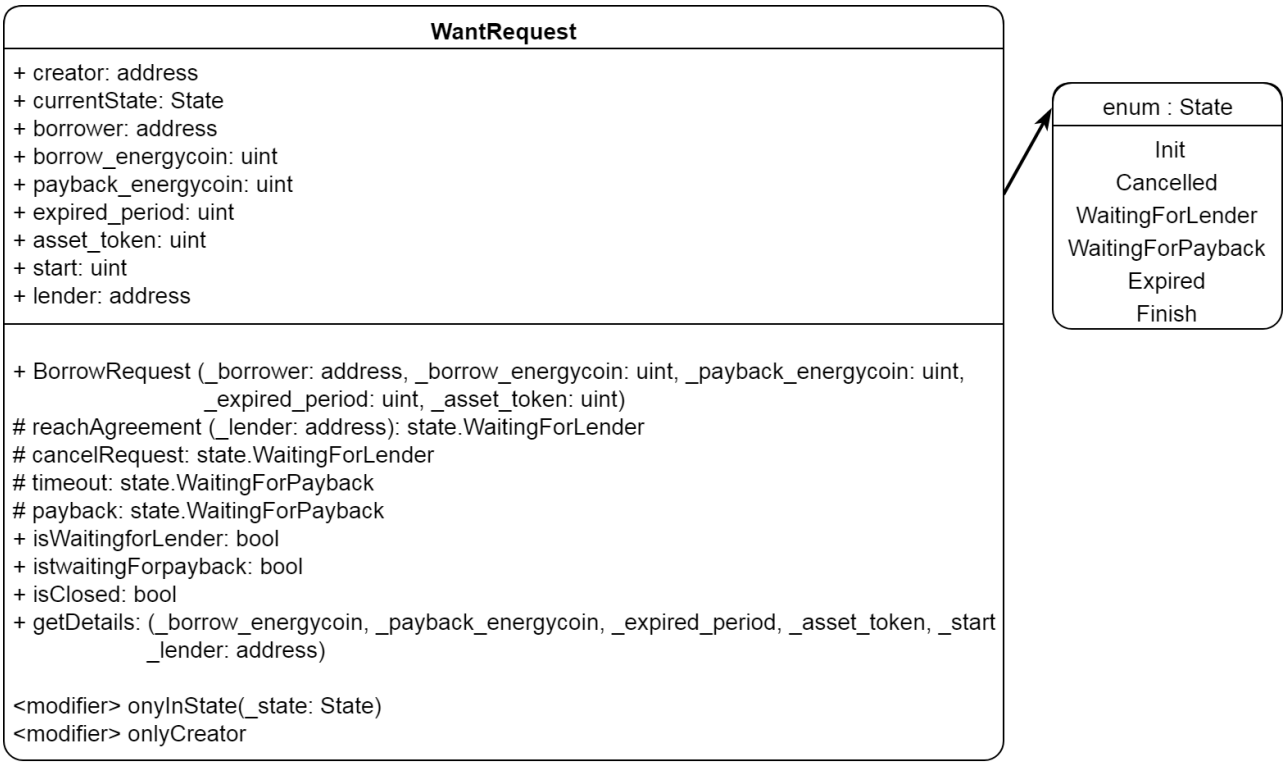

Abbildung 4.7.: UML Diagramm WantRequest

EnergyLendRequest

~ energybank: EnergyCoinInterface
~ energycredit: RepToken
~ energycollateral: AssetTokenInterface
- owner: address
+ waitingForLenderNum: uint
+ waitingForBorrowerNum: uint
+ waitingForPaybachNum: uint
+ closedNum: uint
+ totalAccountNum: uint
~ RepTokenRate: uint
<mapping> allAcountList: uint
<mapping> historyPerAccount: address

+ EnergyLendRequest
# setUpEnergyCoin (EnergyCoinAddress: address)
# setUpAssetToken (AssetTokenAddress: address)
+ postBorrowRequest (_borrow_energycoin, _payback_energycoin,
_expired_period, _asset_token: uint) : bool
+ acceptWantRequest (_request: address): bool
+ paybackLoan (_request: address): bool
+ defaultLoan (_request: address): bool
+ cancelWantRequest (_request: address): bool
+ getAll: address[]
+ getAllRequest (state: string): address[]
+ getUserRequest (account: address, state: string): address[]
+ _onlyInRequestState (_request: BorrowRequest, _state: string): bool
+ getRequestDetails (requestID: address): (borrower: address, borrow_energyCoin: uint,
payback_energyCoin: uint, expired_period: uint, asset_token: uint,
start: uint, lender: address)
+ getArraySize (_state: string): uint
+ getEnergyCoinAddr: address
+ getEnergyCollateralAddr: address

<<modifier>> onlyByOwner
<<modifier>> setUpReady

event Request (_from: address, _borrowAmount: uint, _paybackAmount: uint, _time: uint,
_token: uint)
event Create (_borrower: address, _borrow_energyCoin: uint, _payback_energyCoin: uint,
_expired_period: uint, _asset_token: uint)
event Accept (_request: address, _lender: address)
event Payback (_request: address)
event Timeout (_request: address)

Abbildung 4.8.: UML Diagramm EnergyLendRequest

EnergyLendOffer und LendOffer

Der Smart Contract „EnergyLendOffer“ stellt im Gegensatz zum „EnergyLendRequest“ einen Service für ein DER Angebot dar und agiert als paralleles System. Dadurch wird hier ebenso der Solidity Datentyp *mapping* verwendet, um alle Adressen von agierenden Nutzern und Informationen des DER System zu speichern. Der „LendOffer“ Contract ist zum verfolgen und zum verwalten der Zustände tätigt. Diese werden in Abb. 4.13 näher erläutert. Der wesentliche Unterschied in der Implementierung ist, dass sich mehrere Nutzer für ein Angebot interessieren können. Dadurch ergeben sich Auswahlverfahren für einen DER System Eigentümer. Aufgrund dieser Tatsache werden alle Daten von korrespondierenden Nutzern erfasst und der DER System Eigentümer entscheidet, wenn er seine Dienstleistung vergibt. Auch verwaltet der Contract die Zahlungen von „EnergyCoins“.

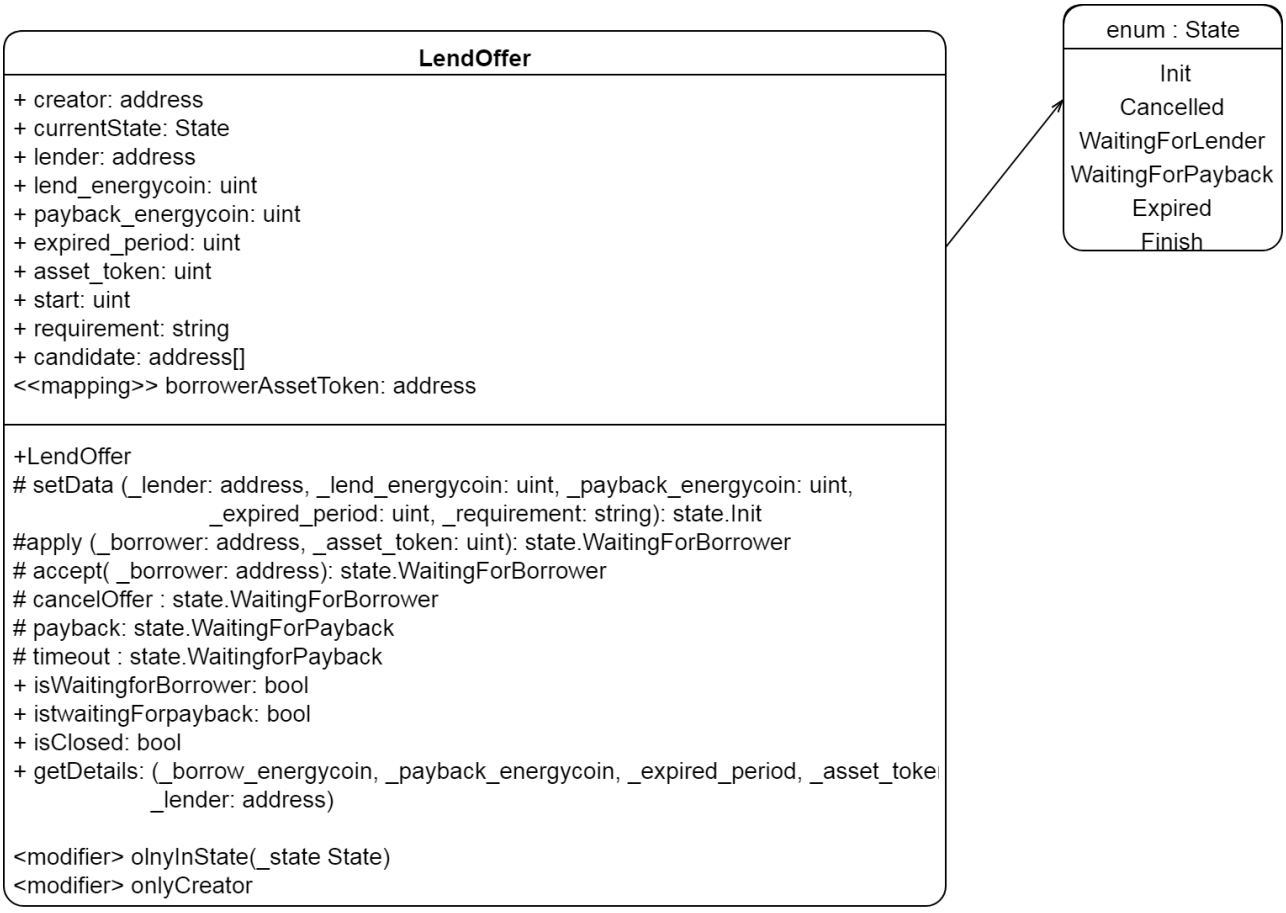

Abbildung 4.9.: UML Diagramm LendOffer

EnergyLendOffer

~ energybank: EnergyCoinInterface
~ energycredit: RepToken
~ energycollateral: AssetTokenInterface
- owner: address
+ waitingForLenderNum: uint
+ waitingForBorrowerNum: uint
+ waitingForPaybachNum: uint
+ closedNum: uint
+ totalAccountNum: uint
~ RepTokenRate: uint
<mapping> allAcountList: uint
<mapping> historyPerAccount: address

+ EnergyLendRequest
# setUpEnergyCoin (EnergyCoinAddress: address)
# setUpAssetToken (AssetTokenAddress: address)
+ postBorrowRequest (_borrow_energycoin, _payback_energycoin,
_expired_period, _asset_token: uint) : bool
+ acceptBorrowRequest (_request: address): bool
+ paybackLoan (_request: address): bool
+ defaultLoan (_request: address): bool
+ cancelBorrowRequest (_request: address): bool
+ getAll: address[]
+ getAllRequest (state: string): address[]
+ getUserRequest (account: address, state: string): address[]
+ _onlyInRequestState (_request: BorrowRequest, _state: string): bool
+ getRequestDetails (requestID: address): (borrower: address, borrow_energyCoin: uint,
payback_energyCoin: uint, expired_period: uint, asset_token: uint,
start: uint, lender: address)
+ getArraySize (_state: string): uint
+ getEnergyCoinAddr: address
+ getEnergyCollateralAddr: address

<<modifier>> onlyByOwner
<<modifier>> setUpReady

event Request (_from: address, _borrowAmount: uint, _paybackAmount: uint, _time: uint,
_token: uint)
event Create (_borrower: address, _borrow_energyCoin: uint, _payback_energyCoin: uint,
_expired_period: uint, _asset_token: uint)
event Accept (_request: address, _lender: address)
event Payback (_request: address)
event Timeout (_request: address)

Abbildung 4.10.: UML Diagramm EnergyLendWant

SmartMeter

Dieser Contract dient zum interagieren mit einem Smart Meter in der physikalischen Ebene.

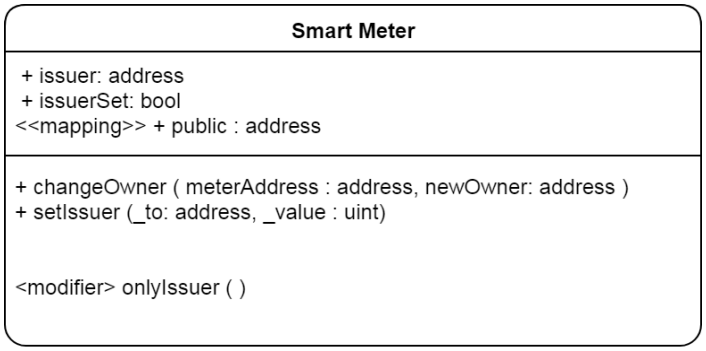

Abbildung 4.11.: UML Diagramm SmartMeter

Gesamt UML Diagramm

In Abbildung 4.12 wird das Gesamt UML Diagram dargestellt. Es wird nochmals deutlich gezeigt wie die einzelnen Smart Contracts miteinander zusammenhängen. Eine noch ausführlichere Darstellung befindet sich im DVD-Anhang A.1.3.

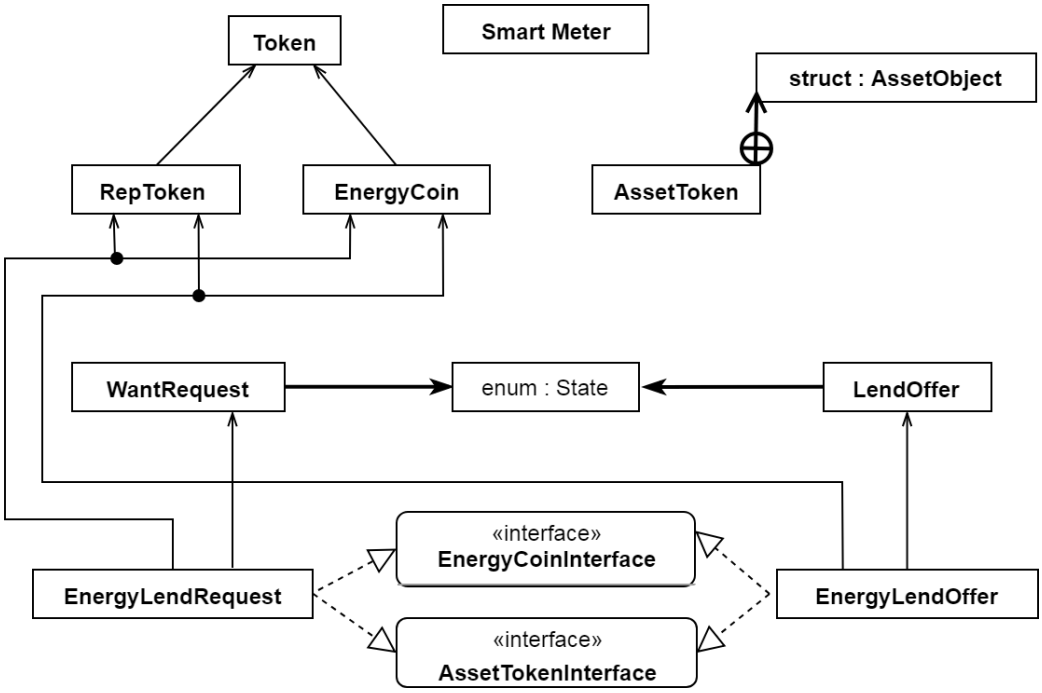

Abbildung 4.12.: Gesamt UML Diagramm

Ablaufdiagramm Smart Contracts

In den Ablaufdiagrammen wird dargestellt, welche Aktionen durchlaufen werden, um eine erfolgreiche bzw. nicht erfolgreiche Transaktion zu bewerkstelligen.

Im Ablauf Diagramm des DER Angebots 4.13 bzw. in den UML Diagrammen der involvierten Smart Contracts 4.9 und 4.10 wird der gesamte Angebots Prozess zusammengefasst dargestellt. Im Ablaufdiagramm werden auch die in Abb. 4.9 dargestellten Zustände und dessen Wechsel abgebildet. Speziell lauten die Zustände :

- Inititalization: Init: kein Angebot (lendOffer)
- Cancel: Angebot wurde vom DER owner abgebrochen, es wurden noch keine Einzahlungen vorgenommen.

- WaitingForLender: Das Angebot wurde aufgegeben mit den relevanten Informationen Preis der Dienstleistung und die Art des DER System. Außerdem wird in diesem Zustand auf Anfragen gewartet.
- waitingForPayback: DER System owner akzeptiert eine Anfrage, dadurch wird der AssetToken versendet und ein SmartMeter Contract erstellt um die physikalischen real Daten aufzunehmen. Außerdem stellt der DApp User eine Sicherheitsanzahlung in Form von 1000 EnergyCoins in die EnergyLend. Dieser Zustand dient dazu auf den AssetToken
- Default: AssetToken und Angebot werden ungültig außerdem wird die Sicherheitsanzahlung vernichtet.
- Finish: AssetToken wurde pünktlich zurückgebracht, es kommt zur Abschlussabrechnung und der *SmartMeter* Contract schreibt die wahren Verbrauchsdaten in die Blockchain außerdem wird der RepToken vergeben. Auch wird die Sicherheitszahlung an den DApp User überwiesen

Die Zustände des Contracts werden automatisch aktualisiert sobald eine bestimmte Transaktion durchgeführt wird. Der gesamte Prozess ist in der Ethereum Blockchain vollständig dezentralisiert und ohne eine dritte vertrauenswürdige Partei realisiert. Wichtige Ereignisse wie Erstellen eines zusätzlichen Contract, das Einzahlen von EnergyCoins, das verteilen des AssetToken und RepToken werden vollständig in der Blockchain protokolliert.

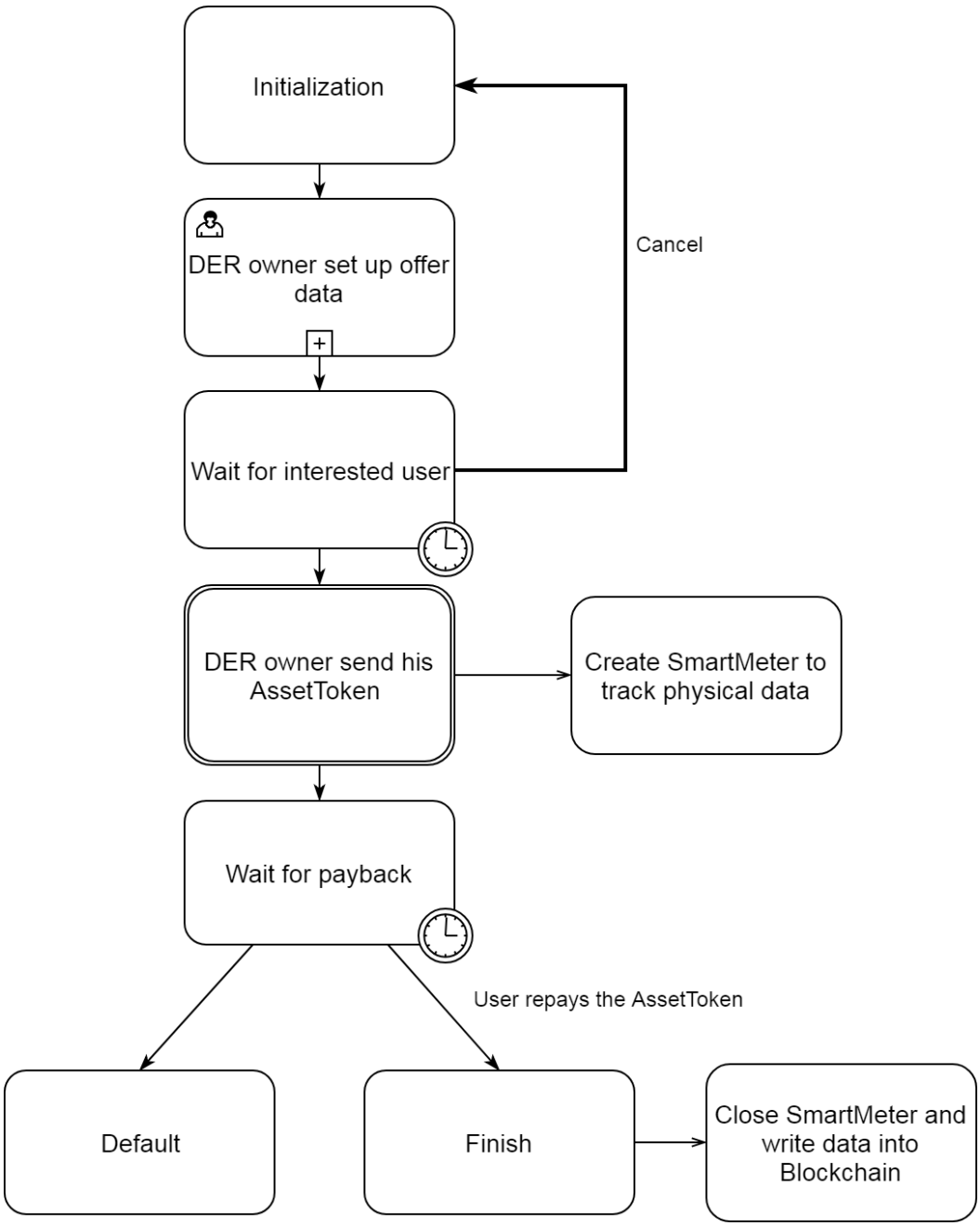

Abbildung 4.13.: Ablaufdiagramm eines DER Angebots

Die Zustände der Contracts *WantRequest* 4.7 und *EnergyLendWant* 4.8 sind die selben wie 4.2 aufgezeigt. Der Ablauf unterscheidet sich dementsprechend nicht groß vom DER Angebot 4.13. Der Unterschied beim DER Gesuche (WantEnergy) 4.14 ist, dass ein Nutzer

ein gewünschtes Angebot mit seinen spezifischen Wünschen anlegt und sich ein geeigneter DER system owner meldet. Der weitere Ablauf ist identisch.

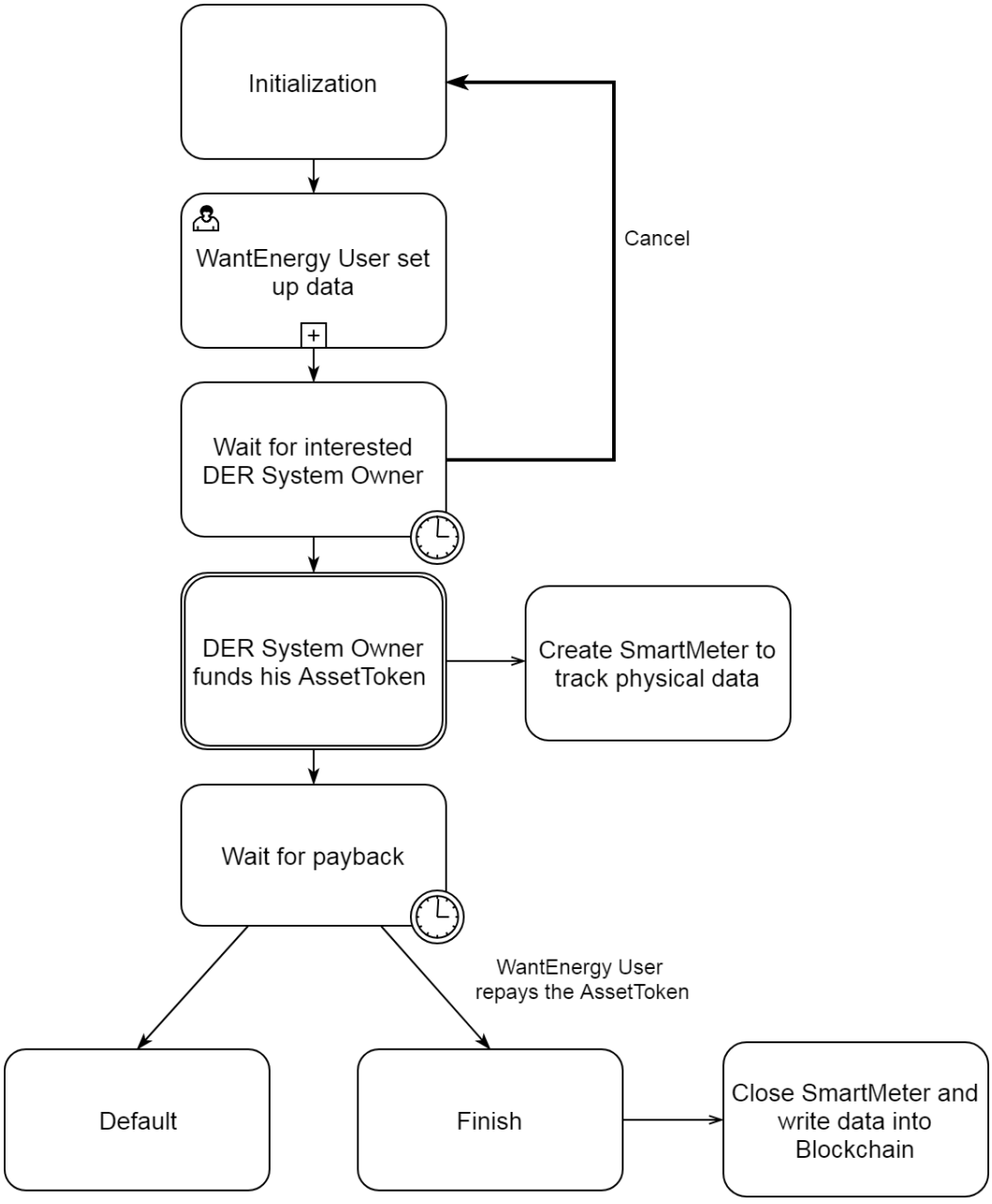

Abbildung 4.14.: Ablaufdiagramm eines DER Gesuche

5. Abbildung des Anwendungsfall in SGAM

Dieses Kapitel gilt als zusätzliche Einordnung des realisierten Anwendungsfall. Die bezogene Architektur dient lediglich des Gesamtbildes. Durch dieses SGAM soll die tiefe eines sich weiterzuentwickelnden Prototypen dargestellt werden.

Um den Anwendungsfall nach einen aktuellen Standard zu realisieren wurde die EnergyLend DApp und eine dazugehörige Blockchain nach dem allgemeinen Smart Grid Architektur Model (SGAM) (siehe Kapitel 2.13) entwickelt.

5.1. Component Layer

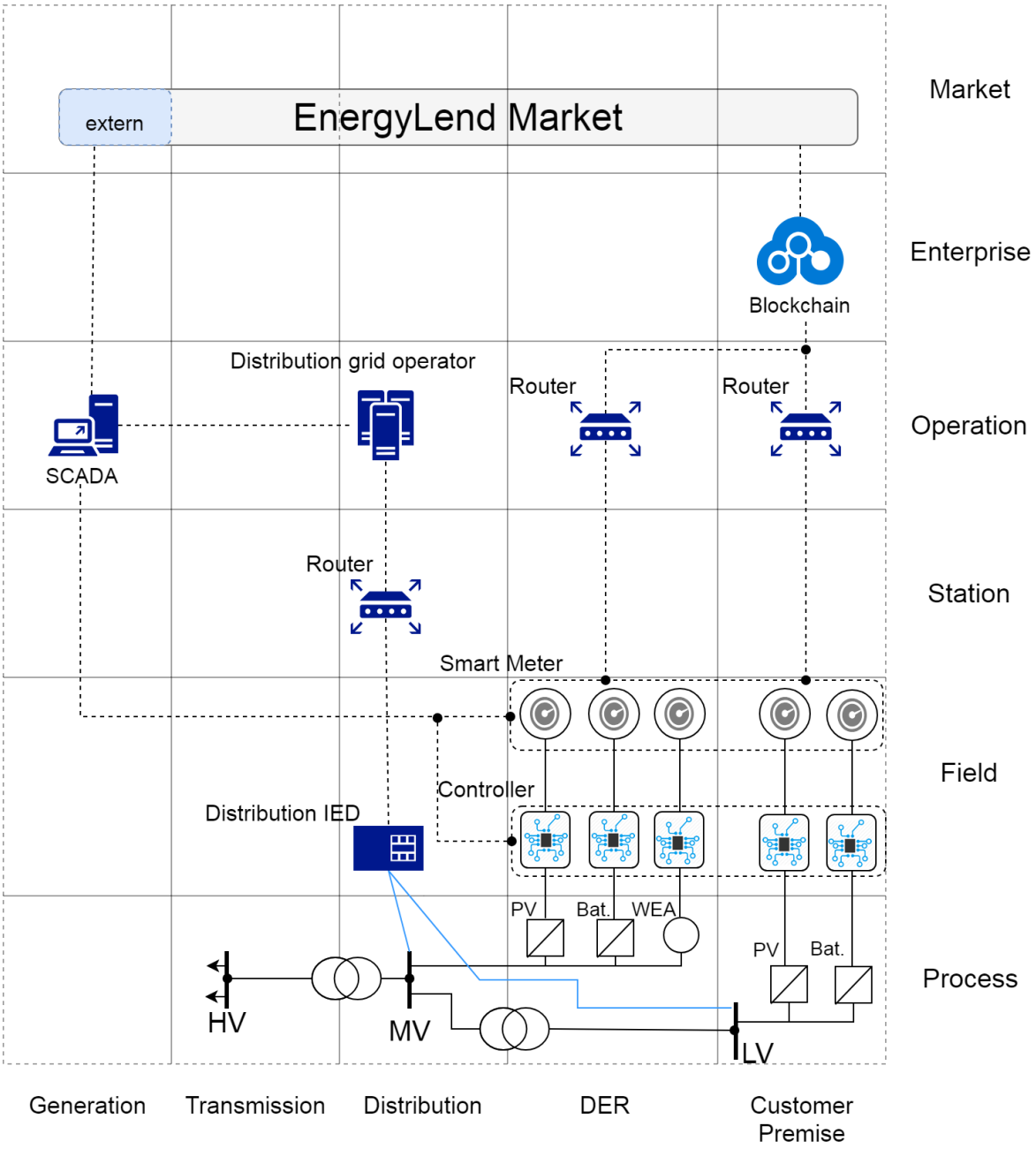

Abbildung 5.1.: Component Layer : EnergyLend DApp

Die Komponenten Schicht (Component Layer) zeigt, die Verteilung und Verbindung der physikalischen Komponenten. Es werden die Hoch- und Höchstspannungsebene (HV), die Mit-

telspannungsebene (MV) und die Niederspannungsebene (LV) in ihren aktiven „Domains“ dargestellt. Die Hoch- und Höchstspannungsebene dient zum Übertragen von elektrischer Energie, zur Versorgung Klein- und Großraumbereichen, zum Anschluss an Kraftwerke und Industrieanlagen und zum überregionalen Energieaustausch. Die Transformation von einer Spannungsebene in eine andere erfolgt in Umspannwerken über Transformatoren. Die Mittelspannungsebene dient zur Verteilung der elektrischen Energie auf Strecken von wenigen Kilometern bis zu 100 km. Üblicherweise werden Distributed Energy Resources (DERs) ,in meinem Beispiel Photovoltaikanlagen (PV), Batteriesysteme (Bat.) und Windenergieanlagen, an die Mittelspannungsebene angeschlossen, da diese des öfteren auf Feldern und Gegenden ohne große Abnehmer aufgebaut werden. Die Niederspannungsebene (LV) dient überwiegend dem Anschluss an die Endverbrauchern, jedoch entwickelt sich in den letzten Jahre die Prämisse, dass elektrische Energie, besonders durch PV Anlagen, auch beim Verbraucher erzeugt werden können. Dadurch agiert dieser als Prosumer d.h. Erzeuger und Verbraucher gleichzeitig. In meinem Beispiel agiert eine PV Anlage und ein Batterie System als Prosumer. Die jeweiligen Energie Erzeuger sind an einen Controller angeschlossen, welches die Systeme überwacht und Eingriffe in die Steuerung erlaubt. Neben dem Controller verfügt jedes DER über ein Smart Meter, um Erzeuger und Verbrauchsdaten aufzunehmen. Die Kommunikation der Smart Meter wird über Router realisiert, welche mit dem Internet über einen sicheren Kanal verbunden sind. Dadurch kann gewährleistet werden, dass die Smart Contracts der Blockchain die nötigen Informationen erhalten um die Automatisierung fortzuführen. Um auf DER Systeme Einfluss zu nehmen, könnte es einen externen Bereich in der EnergyLend Market DApp geben. Dadurch wäre es möglich den Controller der Anlage zu steuern und zu überwachen. Über ein Distribution IED in Verbindung mit einem Router kann ein Distribution Grid operator die Leistungsdaten und andere Energie-relevanten Eigenschaften überwachen und notfalls über das SCADA System eingreifen.

5.2. Communication Layer

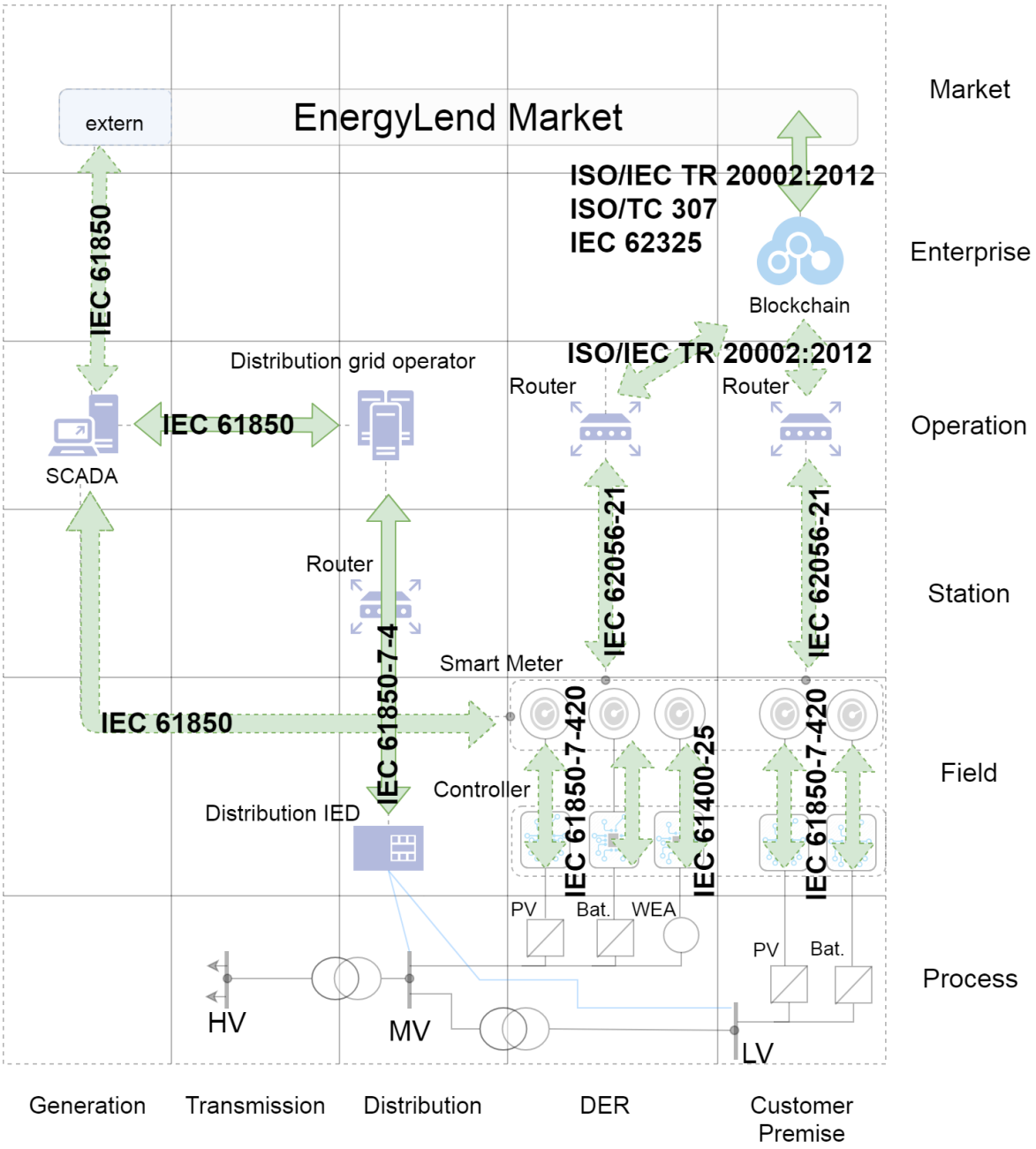

Abbildung 5.2.: Communication Layer: EnergyLend DApp

In der Kommunikationsschicht (Communication Layer) werden die verwendeten Protokolle, Normen und Mechanismen für den allgemeinen Informations- und Datenaustausch der

jeweiligen Layer im Anwendungsfall dargestellt . Die benötigen Informationen über diese wurden aus Dr. T. Koch [2015] und IEC [2018] gewonnen. Die Norm IEC61850 definiert die wichtigsten Informationen für Funktionen und Geräte, den Informationsaustausch für Schutz, Überwachung, Steuerung und Messungen, aber auch die digitale Schnittstelle für Primärdaten und eine Konfigurationssprache. Sie ist für eine Digitalisierung und erweiterte Automatisierung unumgänglich. Als Übertragungsprotokoll wird TCP/IP definiert. Außerdem sind zwei Peer-to-Peer (P2P) Dienste zur echtzeitfähigen Kommunikation in der Norm beschrieben, die auf dem Ethernet-Protokoll aufsetzen. Für die Verbindung zwischen der externen Umgebung und dem SCADA System, dem SCADA System und den DER Controller bzw. Smart Meter kommt die gesamte IEC61850 in Frage. Die IEC 61850-7-420 gilt für DER Systeme. IEC 61850-7-4 definiert grundlegende Kommunikationsstrukturen für Stations- und Feldbezogene Geräte und kompatible Logikknoten- und Datenklassen wie die Verbindung zwischen der Distribution IED und dem Distribution Grid Operator. Für Windkraftanlagen (WEA) gilt die IEC61400-25, welche die Kommunikation zum Überwachen der Anlange aber auch die Kontrollmechanismen für WEAs definiert. Die IEC 62056-21 bestimmt das Protokoll für den Datenaustausch mit einem Smart Meter. In der ISO/IEC TR 20002:2012 wird die Kommunikation, die Voraussetzungen und die Probleme eines P2P Netzwerkes beschrieben. Anzumerken sei, dass die Blockchain ein P2P Netzwerk darstellt, sich jedoch in einigen Punkten eines herkömmlichen P2P Netzwerk unterscheidet. Da es der Zeit keine passendere Norm gibt, wird auf diese zurückgegriffen. An einer Blockchain eigenen Norm, der ISO/TC 307, wird zurzeit gearbeitet und würde wahrscheinlich in meiner Darstellung die ISO/IEC TR 20002:2012 ersetzen ISO2017. Die IEC 62325 beschreibt Standarts für eine Kommunikation eines deregulierten Energie Marktes.

5.3. Information Layer

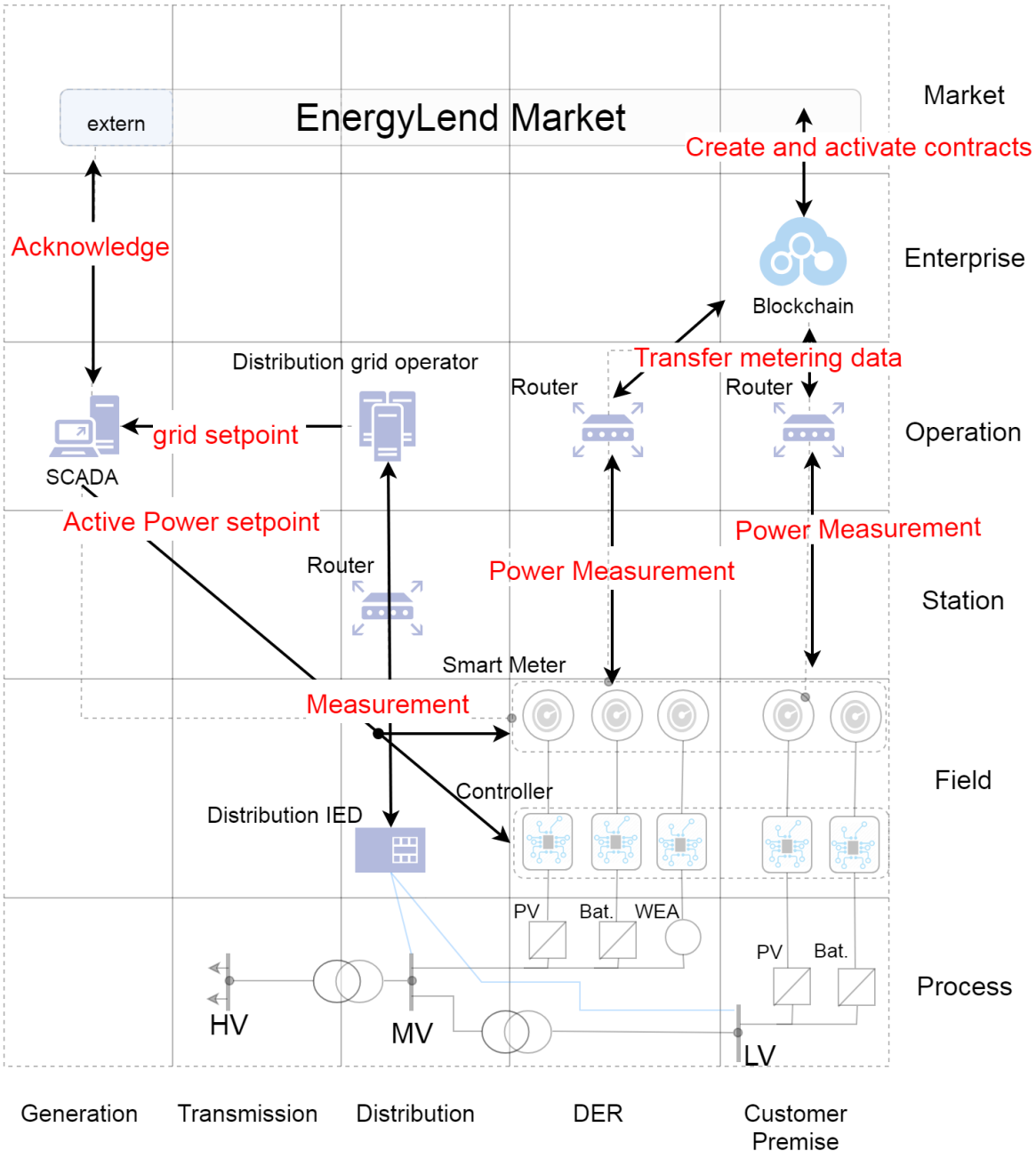

Abbildung 5.3.: Information Layer: EnergyLend DApp

Die Informationsschicht (Information Layer) beschreibt in kurzer Weise wie Informationen zwischen Komponenten, Akteuren und Funktionen erfolgen. Die Informationsobjekte werden

aus dem Anwendungsfall abgeleitet. Zwischen der externen Anwendung des EnergLend Marktes und dem SCADA System wird eine Legitimation des Zugriffs verlangt. Das SCADA System setzt Sollwerte an die DER Controller und Smart Meter. Die Distributed IED sendet über einen Router Messdaten an den Distribution grid operator, dieser kann auf das SCADA System zugreifen und die Energienetz Parameter zum Fahren der Anlage verändern. Die Smart Meter übertragen die aufgenommenen Messdaten über einen Router an die Smart Contracts der Blockchain. Die EnergyLend Market DApp erstellt Smart Contracts und interagiert mit diesen.

5.4. Function Layer

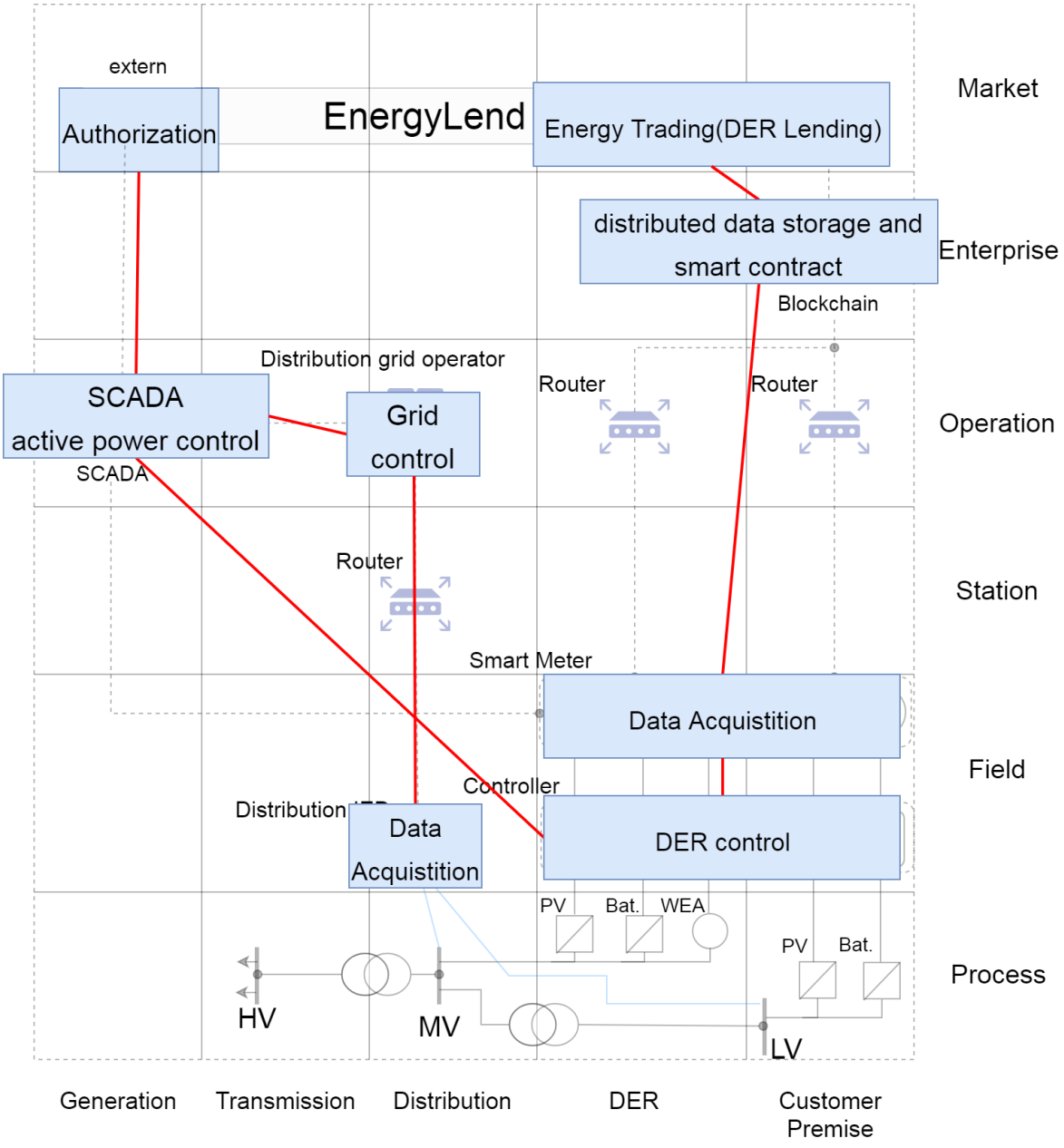

Abbildung 5.4.: Function Layer: EnergyLend DApp

In der Funktionsschicht (Function Layer) werden Funktionen und deren Wechselbeziehungen zu „Domänen“ und „Zonen“ dargestellt. Die Funktionen wurden aus dem Anwendungsfall

abgeleitet. Über eine externe Schnittstelle des EnergyLend Market kann ein Nutzer seinen AssetToken zum Autorisieren auf eine bestimmte Anlage anwenden. Über das verbundene SCADA System ist es dem Nutzer möglich, auf den elektrischen Energiefluss der Anlage zu zugreifen und die gewünschte Energiedienstleistung an den Controller zu übertragen. Die genutzte Energiedienstleistung ist lediglich auf Active Power (siehe A.1.3) in unserem Beispiel beschränkt. Als Active Power gilt das Fahren einer Anlage anhand der Wirkleistung. Das Distributed IED dient als Mess- und Kontrolleinheiten zur Beobachtung des Zustandes des Energienetzes. Das Smart Meter ist als Mess- und Kontrolleinheit zur Beobachtung des Zustandes der Anlage notwendig aber auch für die externe Abrechnung. Die Blockchain im Zusammenhang mit der zu entwickelten EnergyLend DApp ist für die gesamten Marktdarstellung zuständig und somit für das Speichern von Dienstleistungsdaten und dem Ausführen von Prozessen (Smart Contracts) maßgebend.

5.5. Business Layer

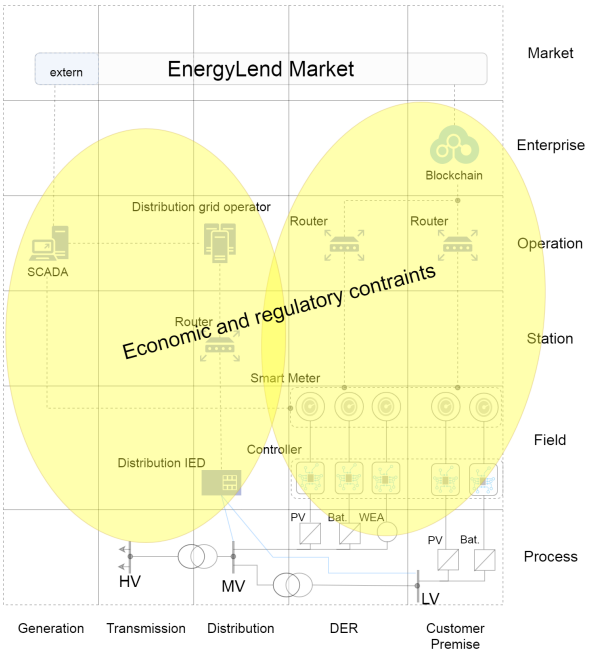

Abbildung 5.5.: Business Layer: EnergyLend DApp

Die Geschäftsschicht (Business Layer) zeigt die Zonen in denen sich der Anwendungsfall aufhält und sich die ökonomischen und regulatorischen Einschränkungen befinden.

6. Test des Systems und Ergebnisse

In diesem Kapitel soll der in Kapitel 3.1 entwickelte Anwendungsfall der EnergyLend DApp anhand von Funktions- und Vergleichstest validiert werden.

6.1. Smart Contract Testing

Um die Smart Contracts der DApp zu testen, wurden spezielle Testcases im Truffle Framework in JavaScript entwickelt. Die entwickelten Szenarios testen vollkommen automatisch die Funktionalität der DApp. Für das EnergyWant DER Angebot (Abb. 4.13) werden zwei Szenarios ausgeführt:

- Szenario 1 : DER System Owner stellt eine Anlage in den Markt. Die Bedingung der Dienstleistung sind folgende : 100 kW für 10 Sekunden zu einem Preis von 100 EnergyCoins. Ein DER User reagiert auf das offerierte Angebot und zahlt eine Sicherheitsanzahlung an den DER System Owner. Dieser erhält diese nach der erfolgreichen Dienstleistung zurück. Der „AssetToken“ wird an den DApp User vergeben. Nach 10 Sekunden der geleisteten Energie wird der „AssetToken“ zurückgegeben und zahlt 100 EnergyCoins für die Dienstleistung.
- Szenario 2 : Der Ablauf ist identisch, lediglich erfolgt hier keine Rückgabe des „AssetToken“ wodurch das Angebot als ungültig deklariert wird. Die Sicherheitsanzahlung wird zerstört und der „AssetToken“ an den DER System Owner zurückgeben.

Für das EnergyWant DER Gesuche (Abb. 4.14) werden ebenfalls zwei Szenarios ausgeführt:

- Szenario 3 : DApp User stellt ein Gesuche in den Markt. Die Bedingungen des Gesuche sind folgende : 100 kW für 10 Sekunden zu einem Preis von 100 EnergyCoins. Zur Auswahl stehen zwei Anlagen, „WindEnergy1 DER“ und „WindEnergy2 DER“. Beide Anlagen können die geforderte Menge liefern. Es wird sich für „WindEnergy2 DER“ entschieden. Der DApp User zahlt dem „WindEnergy2 DER Owner“ eine Sicherheitsanzahlung damit dieser die Anlage nutzen kann. Diese erhält er nach erfolgreichen Handel zurück. Nach der Sicherheitsanzahlung wird der „AssetToken“ der Anlage an den DApp User vergeben. Dieser gibt nach 10 Sekunden denn „AssetToken“ zurück und zahlt 100 EnergyCoins. Der DER System Owner erhält „RepToken“ als Zeichen des erfolgreichen Angebots.
- Szenario 4 : Der Ablauf ist identisch, lediglich erfolgt hier keine Rückgabe des „AssetToken“ wodurch das Angebot als ungültig deklariert wird. Die Sicherheitsanzahlung wird zerstört und der „AssetToken“ an den DER System Owner zurückgeben.

Die Test Szenarios dient nicht nur dem prüfen der richtigen Ausführung der Smart Contracts, sondern auch dem messen der benötigten Laufzeiten in den jeweiligen Blockchain Netzwerken 6.1. Die Tests können auf allen Ethereum Netzwerken ausgeführt werden.

Netzwerk	Szenario 1	Szenario 2	Szenario 3	Szenario 4
geth Privat	1304.142 s	1026.043 s	2397.161 s	2042.344 s
Ganache	44.722 s	39.463 s	59.929 s	56.434 s

Tabelle 6.1.: Laufzeit der ausgeführten Smart Contracts

Die Ergebnisse in Tabelle 6.1 zeigen, dass das Ganache Netzwerk die bessere Leistung besitzt. Dieses Ergebnis ist natürlich klar da es sich bei Ganache um ein simuliertes Netzwerk handelt. Die Laufzeit ist fast proportional zur Anzahl der Transaktionsoperationen die verschieden bei den Szenarios sind. Szenario 3 benötigt am längsten da es die meisten Transaktionen erstellt. Da es zwei verschiedene DER System Owner bietet, einen Entscheidungsprozess zur Auswahl und dem Transferieren des „AssetToken“. Szenario 2 dagegen hat die kürzeste Laufzeit. Die Ergebnis Protokolle finden sich im DVD Anhang A.1.2.

6.2. Netzwerk Test

Mithilfe des Truffle Frameworks wurde ein Leistungstest für die verschiedenen Blockchain Netzwerken programmiert. In diesem Leistungstest wird 20 Mal das Szenario 1 ausgeführt und der Datendurchsatz gemessen.

Netzwerk	Datendurchsatz/s
geth Privat	45.74
Ganache	169.81

Tabelle 6.2.: Datendurchsatz Test der Ethereum-Netzwerke

Wie in der Tabelle 6.2 ersichtlich ist besitzt das Ganache Netzwerk den deutlich höheren Durchsatz als die private Blockchain. Was nicht überrascht da in dieser tatsächlich geschürft (minen) werden muss. Auch das erhöhen von Schürf-Knoten (mining node) hat keine deutliche Verbesserung im Datendurchsatz gebracht. Die Ergebnis Protokolle finden sich im DVD Anhang A.1.2.

6.3. Beurteilung der Arbeit

Das entwickelte System der „EnergyLend“ DApp ermöglicht es, einen „AssetToken“ ohne Intermediär zwischen zwei unbekannten Teilnehmern zu verleihen. Der „AssetToken“ dient wie in den vorherigen Kapiteln schon erwähnt, als Zugriffsberechtigung für DER Systeme. Wodurch ein Handel von Energy und Kryptowährung virtuell prototypisch realisiert wurde. Die in Abbildung 4.7 und 4.14 dargestellten Ablaufdiagramme wurden zum großen Teil realisiert. Jedoch konnte das erzeugen eines „SmartMeter Contract“ nicht weiter implementiert werden, da es zu Problemen beim ausführen des weiteren Code kam. Die Implementation wurde dementsprechend zurück genommen. Dadurch ergibt sich eine Einschränkung im Zählwesen des Anwendungsfalls. Es soll außerdem angemerkt werden, dass die Blockchain mit ihren bisherigen Tools und Frameworks, noch sehr rudimentär ist. Das Debuggen ist im Truffle Framework nicht immer klar ersichtlich und durch die Beschränktheit der Konsole deutlich erschwert. In der Zeit in der diese Arbeit entwickelt wurde, wurden 8 neue Versionen von Solidity und Truffle veröffentlicht. Bei den Veröffentlichungen wurden grundsätzliche System-Punkte verändert wie vorhandene Funktionen, Zugriffsbeschränkungen und Namensänderungen von Methoden. Auch an der EVM wurden große Veränderungen vorgenommen wodurch sich einige Smart Contracts gänzlich anders verhielten und zu Fehlerfällen führten.

7. Schlussbetrachtung

In den vorherigen Kapiteln wurde eine Einführung in das Themengebiet Distributed Ledger Technology, insbesondere in die Blockchain Technologie gegeben. Darüber hinaus wurde mit Hilfe der Ethereum Plattform ein Prototyp einer dezentralen Anwendung (DApp) umgesetzt und evaluiert. Dieses Proof of Concept (PoC) dient als Entwicklung für Smart Contracts in einer Blockchain. Die DApp wurde dabei im Rahmen eines Use Case im Kontext eines Peer-to-Peer OTC Energiehandel entwickelt und evaluiert. Dieses Kapitel fasst die Erkenntnisse und Ergebnisse dieser Bachelorarbeit zusammen. Anschließend werden weitere Entwicklungen im Umfeld von Blockchain und Ethereum in einem Ausblick betrachtet und ein Fazit geschlossen.

7.1. Zusammenfassung und Fazit

Das Ziel dieser Arbeit ist eine Technologie Studie zum Thema Distributed Ledger Technology (DLT) anzulegen und mit der verknüpften Blockchain Technik einen Anwendungsfall für eine Peer-to-Peer Energiehandels Plattform zu prototypisieren. Die aufgeworfenen Forschungsfragen beziehen sich zum einen darauf, ob die DLT einen Anteil dazu beitragen kann, die Energiewirtschaft weiter zu automatisieren und perspektivisch zu digitalisieren. Zum anderen ganz speziell ob eine OTC (Over the Counter) Energieplattform mit einer Blockchain unter Zuhilfenahme von programmierten Verträgen, sogenannten Smart Contracts, realisiert werden kann, und welche Vor aber auch Nachteile entstehen würden.
Um die Forschungsfragen zu beantworten, wurden anfänglich die Grundlagen der DLT und der Blockchain thematisiert. Somit wurden die Herausforderungen von bisherigen Peer-to-Peer Netzen mit sich unbekannten Nutzern angesprochen und welche Lösungen die Blockchain über ihre Konsens Verfahren hier bereitstellt. Auch gibt es einen Einblick in die Kryptografischen Prinzipien um die Sicherheit jeder Transaktion, jedes Nutzers und des gesamten System zu gewährleisten. Daraufhin ging es speziell um Ethereum, welche die Blockchain Plattform für den Anwendungsfall darstellt. Hier wurden die Eigenschaften dieser Plattform erläutert. Um den Anwendungsfall für die dezentrale Applikation (DApp) zu begründen und zu verdeutlichen wurde eine Meinungsstudie der Gema unter Energieunternehmen herangezogen. Daraus ergibt sich die Anforderung, dass eine Peer-to-Peer Energiehandels-Plattform auf Basis der Blockchain ideal als Proof of Concept (PoC) ist. Im Nachfolgenden wurde ebendieses Design der DApp und die dazugehörige physikalische Zusammenhänge dargestellt. Mit Hilfe des Smart Grid Architecture Model (SGAM) Framework wird der Anwendungsfall detailliert analysiert und Schichtweise herausgearbeitet um das Gesamtkonzept darzulegen. Die Implementierung zeigt die in der Programmiersprache *Solidity* entwickelten Smart Contracts und ihr agieren miteinander, um die OTC Energieplattform zu realisieren.

Die Entwicklung des Prototypen demonstriert, dass sich mit verhältnismäßig allgemeinen Programmieraufwand eine effektive und funktionsfähige Prozessanwendung erstellen lässt. Die erstellten Smart Contracs können von jedem Benutzer des Netzwerkes aktiviert werden, welcher die Rahmenbedingungen zum interagieren bereitstellt. Die Prozesse des Energiehandels wurden über ein virtuelles Verleihsystem realisiert indem ein Token (AssetToken) den Zugriff auf eine DER Anlage darstellt und welcher an die Teilnehmer vergeben wird. Durch dieses autarke System ist die Automatisierung deutlich erhöht und sogenannte Intermediäre sind für die Ausführung von Transaktionen und für den Unterhalt des Netzwerkes und der Plattform nicht mehr notwendig.
Die Arbeit zeigt, dass die Distributed Ledger Technology hier die Blockchain die Möglichkeit aufweist, reale Prozesse einfach und effektiv zu realisieren. Gerade im Bezug auf eine zunehmend dezentrale Energiewirtschaft harmoniert die verteilte Systemarchitektur der Blockchain hervorragend. Durch die allgegenwärtige Transparenz, die potenzielle Kostenreduktion durch Dezimierung von Intermediären und der automatischen Buchhaltung würden in erster Linie Kosten für Unternehmen im Energiesektor reduziert werden. Dieser Aspekt könnte auch Einfluss in eine Senkung des Strompreises haben.
Wie in der Einleitung angesprochen entstehen für Bilanzkreisverantwortliche (BKV) bei Abweichungen von gemeldeten Energiefahrplänen hohe Kosten die auf den Endkunden übertragen werden. Um diese Prognosefehler zu senken, könnte die Blockchain bzw. der entwickelte Prototyp, zum kurzfristigen Nachkaufen oder Verkaufen von Energie genutzt werden. Es darf jedoch nicht vergessen werden, dass die Blockchain Technologie noch in den Anfängen und bestehende Projekte noch weit entfernt von einer Marktdurchdringung sind. Sie sollte als Software Werkzeug wahrgenommen werden, um Prozesse in erster Linie zu automatisieren und Transparenter zu gestalten.
Auch gibt es große regulatorische Bedingungen die geklärt werden, unter anderem des Datenschutzes und dem Haftungsrecht. Gerade im Bezug auf Smart Contracts gibt es rechtliche Risiken, da es noch unklar ist ob eine Entscheidung die durch den Programmcode getroffen wurde, auch von Gerichten als verbindlich anerkannt wird. Auch stellt sich die Frage, inwiefern der Programmcode für die Marktteilnehmer verständlich ist. In der Vergangenheit gab es etliche Beispiele für die voraus liegenden Herausforderungen dieser Technologie. Von fehlerhaften Smart Contracts und Wallet Programmen, die große Verluste für die Nutzer darstellten, der zu geringen Skalierbarkeit der Transaktionen und des zu hohen Energieaufwand des Konsens Algorithmus PoW (siehe 2.6.1).

7.2. Ausblick

Die Entwicklungen der Distributed Ledger Technology gehen weiter. So entstehen neben der Blockchain Weiterentwicklungen wie der Hedera Hashgraph oder Iota. Hashgraph ist eine Datenstruktur, die einem Baum nachempfunden wurde und ohne dazugehörige Kryptowährung agiert. Es besteht bisher lediglich das technische Grundgerüst welcher darauf ausgelegt ist, Infrastrukturen für Unternehmen und ähnliches zu ersetzen. Es sind mehrere parallele Rechenprozesse und Transaktionen möglich, wodurch die Transaktionsgeschwindigkeit (250.000+ tx/s) enorm ist. Als Konsens wird ein Gossip-over-Gossip-Verfahren (Tratsch über Tratsch) verwendet, bei dem Informationen über Hashs verteilt und im Abstimmungsverfahren validiert werden. Dadurch wird kein Energieaufwand für den Konsens benötigt. IOTA dagegen verwendet einen direktionalen azyklischen Graphen (DAG), umgangssprachlich als *Tangle* bezeichnet, als Struktur. Jeder Datenknoten kann mehr als einen Partner haben, wodurch sich die Struktur in verschiedene Richtungen entwickeln kann und dadurch die gleichzeitige Verarbeitung von Transaktionen bewältigt. Ebenso kann ein Teil des Netzwerkes von Rest losgelöst agieren und später mit dem Hauptnetz wieder vereinen. Um eine Transaktion vorzunehmen, müssen mindestens zwei andere Transaktionen verifiziert werden. Diese Verifikation geschieht über eine einfache Berechnung wie bei PoW 2.6.1, jedoch deutlich geringer. IOTA hat sich auf eine sichere Kommunikation und Zahlung zwischen Maschinen im Rahmen des Internet of Things (IoT, Internet der Dinge) ausgerichtet. Auch im Bezug auf Ethereum gibt es etliche Weiterentwicklungen und interessante Projekte, jedoch besonders hervorzuheben ist die Energy Web Foundation (EWF). Diese hat sich der Aufgabe verschrieben eine Blockchain für den Energiesektor zu entwickeln. Diese schon bestehende Blockchain beruht auf einem PoA 2.6.3 Konsens Algorithmus, um Energiekosten zu sparen und Transaktionen schnell zu validieren. Interessant ist, dass es mehrere Tools und Eigenschaften beinhaltet um Anwendungsfälle des Energiesektors einzufügen. Beispielsweise wäre da die Möglichkeit eine Art Führung und Überwachung der Blockchain und Transaktionen einzuführen indem verschiedene Rollen definiert werden können. Die gegebene Möglichkeit private Transaktionen zu versenden würde Datenschutzbedenken verändern. Um die Skalierungsproblematik zu beheben wurden zwei Lösungen gegeben. Zu einem Raiden, welches als Off-Chain Bezahlkanal dient und dadurch extrem hohe und günstige Transaktionen möglich sind. Für jede mögliche Dienstleistung wird ein privater Bezahlkanal eröffnet und Transaktionen ohne Gebühr ausgeführt. Beim schließen des Kanals werden Gebühren nötig, da die Abschlussrechnung als Transaktion in die Blockchain eingefügt wird. Damit sind sogenannte Mikrotransaktionen möglich und Abrechnungen können in Echtzeit durchgeführt werden, ein Beispiel wäre das Aufladen eines Elektroautos. Die andere Implementierung ist Polkadot, welche als Multichain- und Übersetzungsarchitektur agiert und individuell angepasste Sidechains miteinander verbindet. Dadurch besteht die Möglichkeit regionale Sidechains aufzubauen, welche von unterschiedlichen Akteuren implementiert wurden. Bei überregionalen Transaktionen und Verträgen verbindet Polkadot die Teilnehmer miteinander

und bewerkstelligt die Kommunikation. Auf diesem Wissen könnte man den bestehenden Anwendungsfall neu auslegen und modifizieren. Die Smart Meter würden als direkte Partner (Light Node) eines Bezahlkanals agieren können. Auch die Eigenschaft, dass es Beschränkungen innerhalb der Blockchain gibt, würde die Selektion in DER System Betreiber und normalen Nutzer deutlich vereinfachen. Gerade im Punkt Anwendungsfreundlichkeit gibt es viele Möglichkeiten für den Anwendungsfall EnergyLend. Ein erster Schritt wäre die Entwicklung eines Frontend um die Ausführungsschritte zu visualisieren. Es könnte auch an der Weiterentwicklung der Smart Contracts gearbeitet werden, eine Möglichkeit wäre ein Gruppen Service. Bisher ist es in den Smart Contracts *LendOffer* und *WantRequest* nur möglich mit einer Partei zu agieren. In einem Gruppen Service könnten mehrere Teilnehmer an einer Dienstleistung teilnehmen wodurch die Flexibilität deutlich erhöht würde.

Tabellenverzeichnis

Abbildungsverzeichnis

Literaturverzeichnis

[ISO2017] : *ISO/TC 307 Blockchain and distributed ledger technologies*

[180-4 2012] 180-4, Federal Inf. Process. Stds. (NIST F.: Secure Hash Standard. 2012. – Forschungsbericht

[Antonopoulos 2014] ANTONOPOULOS, Andreas M.: *Mastering Bitcoin: Unlocking Digital Cryptocurrencies*. O'Reilly Media, 2014. – URL https://www.amazon.com/Mastering-Bitcoin-Unlocking-Digital-Cryptocurrencies/dp/1449374042?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1449374042. – ISBN 978-1449374044

[BDEW 2017] BDEW: Stromerzeugung und -verbrauch / BDEW Bundesverband der Energie und Wasserwirtschaft e.V. URL https://www.bdew.de/media/documents/20171220_PI_Anlage_Zahlen-Fakten.pdf, 2017. – Forschungsbericht

[BFT] BFT: Byzantine Consensus Algorithm. In: *tendermint*. – URL https://github.com/tendermint/tendermint/wiki/Byzantine-Consensus-Algorithm

[BNetzA 2017] BNETZA: Monitoringbericht 2017. In: *Bundesnetzagentur* (2017), S. 492. – URL https://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Allgemeines/Bundesnetzagentur/Publikationen/Berichte/2017/Monitoringbericht_2017.pdf?__blob=publicationFile&v=4

[Bonadonna 2016] BONADONNA, E.: Bitcoin and the Double-Spending Problem. In: *Cornell University* (2016). – URL https://blogs.cornell.edu/info4220/2013/03/29/bitcoin-and-the-double-spending-problem/

[Buterin 2018] BUTERIN, Vitalik: Proof of Stake. In: *Ethereum* (2018). – URL https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ

[CEN und ETSI 2012] CEN, CENELEC ; ETSI: Smart Grid Reference Architecture / CEN, CENELEC and ETSI. 2012. – Forschungsbericht

[Coulter 2015–2018] COULTER, T.: Truffle / Truffle. URL http://truffleframework.com/docs/, 2015-2018. – Forschungsbericht

[Cynthia Dwork 1992] CYNTHIA DWORK, Moni N.: hashcash / hashchash.org. URL http://www.hashcash.org/, 1992. – Forschungsbericht

[Dr. T. Koch 2015] DR. T. KOCH, Dr. J. H.: Die IEC 61850 als Standard fÃ¼r die Digitalisierung des Energiesystems. In: *etz-elektrotechnik & automation* (2015). – URL http://www.etz.de/5725-0-Die+IEC+61850+als+Standard+fuer+die+Digitalisierung+des+Energiesystems.html

[F.Kirstein 2017] F.KIRSTEIN, Dr. Klaus-Peter E.: DIGITALE IDENTITAETEN IN DER BLOCKCHAIN. In: *Frauenhofer Fokus* (2017). – URL https://cdn0.scrvt.com/fokus/1b5163cdc614d2a3/3101727645ae/Fh_FOKUS_BlockIdent.pdf

[Foundation 2018] FOUNDATION, Ethereum: Ethereum clients. In: *Ethereum Docs* (2018). – URL http://ethdocs.org/en/latest/ethereum-clients/choosing-a-client.html

[Franco 2014] FRANCO, Pedro: *Understanding Bitcoin*. John Wiley & Sons, Ltd, oct 2014

[dena GmbH 2016] GMBH dena: Blockchain in der Energiewirtschaft - Eine Umfrage unter Fuehrungskraeften der deutschen Energiewirtschaft. In: *dena GmbH* (2016)

[IEC 2018] IEC: Smart Grid Standards Map. IEC, 2018. – URL http://smartgridstandardsmap.com/

[Lamport u. a. 1982] LAMPORT, Leslie ; SHOSTAK, Robert ; PEASE, Marshall: The Byzantine Generals Problem. In: *ACM Transactions on Programming Languages and Systems* 4/3 (1982), July, S. 382–401. – URL https://www.microsoft.com/en-us/research/publication/byzantine-generals-problem/

[Merkle und Hellman 1978] MERKLE, R. ; HELLMAN, M.: Hiding information and signatures in trapdoor knapsacks. In: *IEEE Transactions on Information Theory* 24 (1978), sep, Nr. 5, S. 525–530

[Merkle 1979] MERKLE, Ralph C.: Secrecy, Authentication, and Public Key Systems / Stanford University. URL http://www.merkle.com/papers/Thesis1979.pdf, 1979. – Forschungsbericht

[Mitschele 2018] MITSCHELE, Prof. Dr. A.: Smart Contract. In: *Gabler Wirtschaftslexikon* (2018). – URL https://wirtschaftslexikon.gabler.de/definition/smart-contract-54213

[Nakamoto 2008] NAKAMOTO, S.: Bitcoin: A Peer-to-Peer Electronic Cash System / -. URL https://bitcoin.org/bitcoin.pdf, 2008. – Forschungsbericht

[OS4ES 2017] OS4ES: Open System for Energy Services. In: *ew-Magazin fÃ¼r Energiewirtschaft, Heft 10/2017* (2017)

[PBFT 2016] PBFT: 2016. In: *hyperledger fabric* (2016). – URL https://github.com/hyperledger-archives/fabric/blob/master/docs/protocol-spec.md

[Reischuk 1987] REISCHUK, R.: Konsistenz und Fehlertoleranz in Veneilten Systemen - Das Problem der Byzantinischen Generale. (1987). – URL http://www.springer.com/de/book/9783540184782?utm_medium=referral&utm_source=google_books&utm_campaign=3_pier05_buy_print&utm_content=de_08082017#otherversion=9783662011102

[Swan 2016] SWAN, Melanie: *Blockchain*. O'Reilly UK Ltd., 2016. – URL http://www.ebook.de/de/product/23526564/melanie_swan_blockchain.html. – ISBN 1491920491

[Szabo 1996] SZABO, Nick: Smart Contracts: Building Blocks for Digital Markets. (1996). – URL http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html

[w. team 2017–2018] TEAM w.: web3.js - Ethereum JavaScript API / web3.js. URL https://github.com/ethereum/wiki/wiki/JavaScript-API, 2017-2018. – Forschungsbericht

BNetzA [2017]

A. Anhang

A.1. Auf der CD beiliegende Daten

A.1.1. Ordner „EnergyLend“ mit der entwickelten Software

- Ordner „contracts“ - Beinhaltet Smart Contracts
- Ordner „execTest“ - Beinhaltet die Test Scripte
- Ordner „migration“ - Beinhaltet die Migrationsdateien
- Ordner „testingPrivate“ - Beinhaltet Scripte und Shell zum Testen des Privaten Netzwerkes
- Ordner „privateNetwork“ - Beinhaltet Dateien und Shells zum Erzeugen eines Privaten Netzwerk
- Ordner „pwd“ - Beinhaltet Passwörter und Adressen für das Private Netzwerk
- README.txt - Beinhaltet die notwendigen Befehle zum Testen der Software

A.1.2. Ordner „testResults“ mit den Ergebnissen der Test

- Ordner „functionTesting“ - Beinhaltet die Ergebnisse der Funktionstests
- Ordner „systemTesting“ - Beinhaltet die Ergebnisse der Systemtests

A.1.3. Ordner „Abbildungen“

- GesamtUMLStruktur
- DER SystemModell OS4ES
- ES-ActivePower OS4ES

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §16(5) APSO-TI-BM ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen habe ich unter Angabe der Quellen kenntlich gemacht.

Hamburg, 22. Mai 2018	
Ort, Datum	Unterschrift