Master's Thesis

Design and Implementation of a Distributed Energy Trading Platform for Local Electricity Markets Based on Hyperledger Fabric

Faculty III — Process Science

Energy Process Technology and Conversion Technologies
for Renewable Energies (EVUR)

Technische Universität Berlin

by

Raphaël Haupt

Industrial Engineering and Management:
Energy and Resource Management

Matriculation Number: 350355

raphael.haupt@posteo.de

Supervisor

Prof. Dr. Frank Behrendt

Specialist Olga Tcvetkova

Berlin, September 17, 2019

I hereby certify that the thesis I am submitting is entirely my own original work except where otherwise indicated. I am aware of the University's regulations concerning plagiarism, including those regulations concerning disciplinary actions that may result from plagiarism. Any use of the works of any other author, in any form, is properly acknowledged at their point of use.

Berlin, September 17, 2019

Abstract

This thesis proposes a Decentralised Application (DApp) design on top of the permissioned blockchain framework Hyperledger Fabric: an auction-based, Peer-to-Peer (P2P) marketplace which enables market participants of the same energy community to trade electricity with one another while keeping their respective buy and sell orders concealed. The design development is inspired by the *standards-based architecture modelling framework*[1] and consists of: (1) an *architecture model*, describing the technical and software components making up the decentralised marketplace—with a special focus on the blockchain network—as well as the information flow between these components; and (2) a *chaincode model*—the core component within the DApp architecture—defining the assets which reside on the blockchain and the transactional logic they are subject to. The design was evaluated in line with design science standards: all previously defined functional requirements could be validated through unit tests simulating multiple agents in interaction with the DApp. Sealed bidding was realised by means of private transactions and opens the door for system designers to implement strategy-proof market mechanisms despite relying on a blockchain-based system. This thesis contributes to the field of research investigating the Information and Communication Technology (ICT) layer of Local Electricity Markets (LEMs) by displaying a systematic approach to system design in a decentralised environment and by proposing a language-agnostic DApp model for community-based local electricity markets.

[1] Neureiter, C. ; Uslar, M. ; Engel, D. ; Lastro, G.: A Standards-Based Approach for Domain Specific Modelling of Smart Grid System Architectures. In: *2016 11th System of Systems Engineering Conference (SoSE)*, 2016, S. 1–6.

Contents

List of Figures

List of Tables

Acronyms

ABAC	Attribute-Based Access Control
AMI	Advanced Metering Infrastructure
BFT	Byzantine Fault Tolerance
CA	Certificate Authority
CFT	Crash Fault Tolerance
CTA	Consumer Trading Agent
DApp	Decentralised Application
DER	Distributed Energy Resources
DG	Distributed Generation
DLT	Distributed Ledger Technologies
DR	Demand Response
DSL	Domain Specific Language
DSO	Distribution System Operator
DSR	Design Science Research
EMTS	Energy Management Trading System
ESCO	Energy Service Company
ESP	Energy Service Provider
ESS	Energy Storage System
EV	Electrical Vehicle
EVM	Ethereum Virtual Machine
GHG	Green House Gas
HLUC	High-Level Use Case

ICT Information and Communication Technology

IO Information Object

IR Individual Rationality

LEM Local Electricity Market

LMO Local Market Operator

MAS Multi-Agent System

MCP Market Clearing Price

MGO Microgrid Operator

MSP Membership Service Provider

NEIC Nash Equilibrium Incentive Compatibility

P2P Peer-to-Peer

PE Pareto Efficiency

PKI Public Key Infrastructure

PoA Proof-of-Authority

PoS Proof-of-Stake

PoW Proof-of-Work

PTA Producer Trading Agent

PV Photovoltaics

SDK Software Development Kit

SGAM Smart Grid Architecture Model

TPS Transactions Per Second

UI User Interface

UML Unified Modelling Language

VCG Vickrey-Clarke-Groves

WBB Weak Balanced Budget

ZI Zero Intelligence

Chapter 1

Introduction

"Until you start focusing on what needs to be done rather than what is politically possible, there is no hope. We can't solve a crisis without treating it as a crisis. We need to keep the fossil fuels in the ground, and we need to focus on equity. And if solutions within the system are so impossible to find, maybe we should change the system itself."

Greta Thunberg, COP24 in Katowice, 2018 [1]

1.1 Contextual Introduction

Worldwide, schoolchildren take the streets under the banner of 'FridaysForFuture' to fight against what they call a climate crisis [2]; a defining aspect of the 2019 European elections was the 'rise of the Greens' [3, 4]; Germany will fail to meet the self-imposed Green House Gas (GHG) emission reduction targets for 2020 [5], and maybe also for 2030 [6]. It is safe to say that the pressure on policymakers to take climate action has never been higher.

According to a recent study published by the United Nations [7], the energy sector—responsible for at least 25% of annual greenhouse gas emissions [8]—presents the highest potential for reducing emissions within the next 10 years; but in order to leverage this mitigation potential, the share of renewable energy sources in the electricity mix has to grow significantly. However, the continuous integration of Distributed Energy Resources (DER)—an umbrella term encompassing both Distributed Generation (DG) units and Energy Storage Systems (ESSs)—entails additional stress on the power grid: fluctuating DG units like Photovoltaics (PV) systems or wind turbines increase the complexity of the energy system [9] and, some argue, may even jeopardise security of supply [10, 11].

In the last years, household consumers assume an increasingly important role in this 'energy transition' by investing in rooftop solar, often complemented by battery storage devices [12]. This new type of consumer—producing electricity in their domestic environment—is generally referred to as *prosumer* [13]. However, the integration of prosumers into the energy system is lagging behind [14]. Electricity which cannot be self-consumed or stored for later use is typically fed into the grid at fixed rates. In Germany, these rates are limited to a time horizon of twenty years, meaning that early adopters start falling out of the subsidy and have to find new ways for marketing their excess energy [15]. This strong dependency from public support schemes is being criticised both for its lack of long-term legal certainty and high costs for taxpayers [16]. Furthermore, the absence of price signals for both consumers and prosumers leads to existing flexibilities in demand and supply not being used, which results in unnecessary stress on the distribution grid [17].

To address the above challenges—the integration of fluctuating renewables into the grid and of prosumers into the markets—new forms of locally trading energy are being explored in first pilot projects [18] and discussed in scientific literature [19]: *Local Electricity Markets (LEMs)*, where consumers, producers, and prosumers can trade locally produced electricity. Studies [14, 20] have shown that Peer-to-Peer (P2P) trading—meaning that market participants can directly negotiate and trade energy with each other [21]—can reduce the electricity bill of market participants by up to 34% without and up to 60% with residential battery storage flexibilities being leveraged [14]. Furthermore, thanks to dynamic price signals, demand and supply are better aligned and power peaks are reduced which can lead to lower balancing costs for Distribution System Operators (DSOs) [22, 23] and alleviate the need for investments in additional grid infrastructure [24].

For LEMs to become a reality, secure and efficient *marketplaces* are needed [25]. Blockchain—essentially an immutable registry of transactions distributed across a P2P network [26]—is being discussed as key Information and Communication Technology (ICT) on which to build such marketplace applications [24, 25, 27–32]. Applications which run on a P2P network are generally referred to as Decentralised Applications (DApps) and consist of one or more interconnected *smart contracts*—self-executable code which defines the transaction logic [33]. Advantages include transparency, disintermediation, and immutability of transactions [34]. However, blockchain is still in its early days and still faces some challenges, namely, scalability [35], interoperability [36], and privacy [37].

1.2 Research Outline

Problem Definition & Research Question

In recent years, multiple projects using smart contacts for local energy trading have surfaced [18, 28] and appropriate DApps designs are being discussed in scientific literature [38–44]. However, research in this field is still at an early stage of development [45]. Most implementations are proofs of concept built on top of *Ethereum*—the first blockchain platform enabling the creation of DApps [33]. But since *Ethereum* is still very energy intensive and has difficulties to scale, most implemented energy trading marketplaces are not deployed on the public *Ethereum* network but on local instances in private environments. Market participants thus need permission to join the network—a design which is widely agreed to be a good fit for LEMs [46, 47]. Permissionned blockchain systems share the benefit that all actors are known which makes it possible to rely on less computationally complex consensus mechanisms—the protocol which ensures that all network participants have identical copies of the transaction log and agree on the state of the system [34]; this significantly increases the number of transactions the system can process and reduces the ecological footprint [48]. However, in the case of *Ethereum*, the content of transactions remains visible to all network participants which complicates the implementation of privacy-preserving features. Subsequently, the scientific community calls for more research in this field [25, 41–43] and encourages investigating alternative blockchain frameworks [46, 49–51].

Hyperledger Fabric—originally contributed by IBM and Digital Asset and now hosted by the Linux Foundation—is one of the most mature permissioned blockchain frameworks [52]. Since its release in 2017, more and more projects rely on *Hyperledger Fabric* [53–57] and leverage some of the framework's advantages: (1) its *modularity*, facilitating the integration with existing ICT infrastructure, for instance; (2) its good *scalability*, meaning it can process high levels of transactions in a short amount of time; and (3) its support for *private transactions*, making it possible to keep private data confidential between a subgroup of authorised parties. [58–60].

This makes *Hyperledger Fabric* a promising candidate for building a distributed energy trading platform. For instance, companies in the energy sector like TenneT, Vandebron and the sonnen Group have partnered with IBM on two blockchain projects—both using *Hyperledger Fabric* [53]. Nevertheless, no implementations have been found whose source code is publicly available and reviewed academically. This thesis intends to shed light on the gaps in the present state of research by

answering the following question: *how could a distributed energy trading platform for local electricity markets, based on Hyperledger Fabric, be designed?*

Contributions

This thesis contributes to current research in the following ways:

1. An overview of the current state of research in the field of LEMs is provided, with a particular focus on different market designs and blockchain as a suitable technology for the ICT layer (see Chapter 3).

2. Based on this overview, an application design for a decentralised energy trading platform on top of Hyperledger Fabric is proposed. This design comprises an *architecture model* and a *chaincode model*[1] (see Chapter 5).

3. Finally, an implementation of the application design discussed in (2) is presented (see Chapter 6), as well as an evaluation of the same (see Chapter 7). All code will be made available open-source.

1.3 Methodology & Structure

This thesis is to be classed into the field of constructive research. This approach *"is a research procedure for producing innovative constructions, intended to solve problems faced in the real world and, by that means, to make a contribution to the theory of the discipline in which it is applied"* [61]. The underlying methodology employed by this thesis is the Design Science Research (DSR) methodology. It is a common approach in the fields of engineering and ICT and focusses on the development or the functional improvement of artefacts. These artefacts range from human-computer interfaces, algorithms and methods, to process models. The DSR process proposed by [62] encompasses the following six steps:

1. **Problem identification and motivation:** Definition of the specific research problem and justification of the value of the solution.

[1] In Hyperledger Fabric, *smart contract* code which defines the transaction logic of the decentralised application is packed into *chaincode* and deployed to the blockchain network. The two terms are often used interchangeably (see Section 4.1).

2. **Objectives definition:** Inference of objectives and requirements from problem definition and literature. These can either be quantitative or qualitative and must be based on knowledge of the state of current solutions.

3. **Design and development:** Creation of the artefact which includes the activity of determining its desired functionality and architecture. Knowledge of theory that can be brought to bear in the solution is a necessary resource for successful completion of this step.

4. **Demonstration:** Demonstration of the artefacts capability to solve one or more instances of the problem described in (1). Examples include simulations, case studies, or proofs of concepts [63].

5. **Evaluation:** Assessment of the artefact's capacity to provide a solution to the problem. In this step, the objectives (2) are compared with observed results from the demonstration (4). Depending on the defined objectives, the comparison can focus on an artefact's functionalities, its performance, results from satisfaction surveys, or simulations. At the end of this step, an iteration back to (3) is possible in order to improve the artefact's effectiveness.

6. **Communication:** Dissemination to relevant audiences stating the problem's importance, the utility and novelty of the artefact, the accuracy of its design, and its effectiveness.

The focus of the thesis lies in the *design and development (3)* of a decentralised energy trading marketplace. The design process itself follows a methodical procedure inspired by the *standards-based architecture modelling framework* formulated by [64]. This *architecture framework* is based on the Smart Grid Architecture Model (SGAM) developed by [65]. The exact process is described in greater detail at the beginning of Chapter 5.

As suggested by [62], the outline of this thesis closely follows DSR approach. The exposé and this introduction *formulate the problem to be addressed and the motivation behind this research (1).* Chapter 2 gives background information and explains the concepts of relevance, and thus serves as a foundation for approaching the second step of the DSR process. Chapter 3 consists of a common literature review focussing on local electricity market designs as well as existing approaches to blockchain-based energy trading platforms in academia. A digression on the blockchain framework Hyperledger Fabric can be found in Chapter 4. Based on this technical analysis and the literature review in Chapter 3, the *key objectives for the proposed solution (2)*

are defined at the beginning of Chapter 5 (Section 5.1) before the proposed *solution design (3)* is presented—the key artefact in line with the DSR approach. Chapter 6 describes the implementation of the design which serves as a *demonstration (4)* of its functionalities. The artefact is *evaluated (5)* in Chapter 7 on the back of this implementation. For *communication (6)* purposes, this thesis and the source code of the implementation are made public on GitHub[2] and a publication in an academic journal is pursued. The DSR methodology and the structure of this work are illustrated in Figure 1.1; the colours indicate which part of the thesis discusses which step within the DSR process.

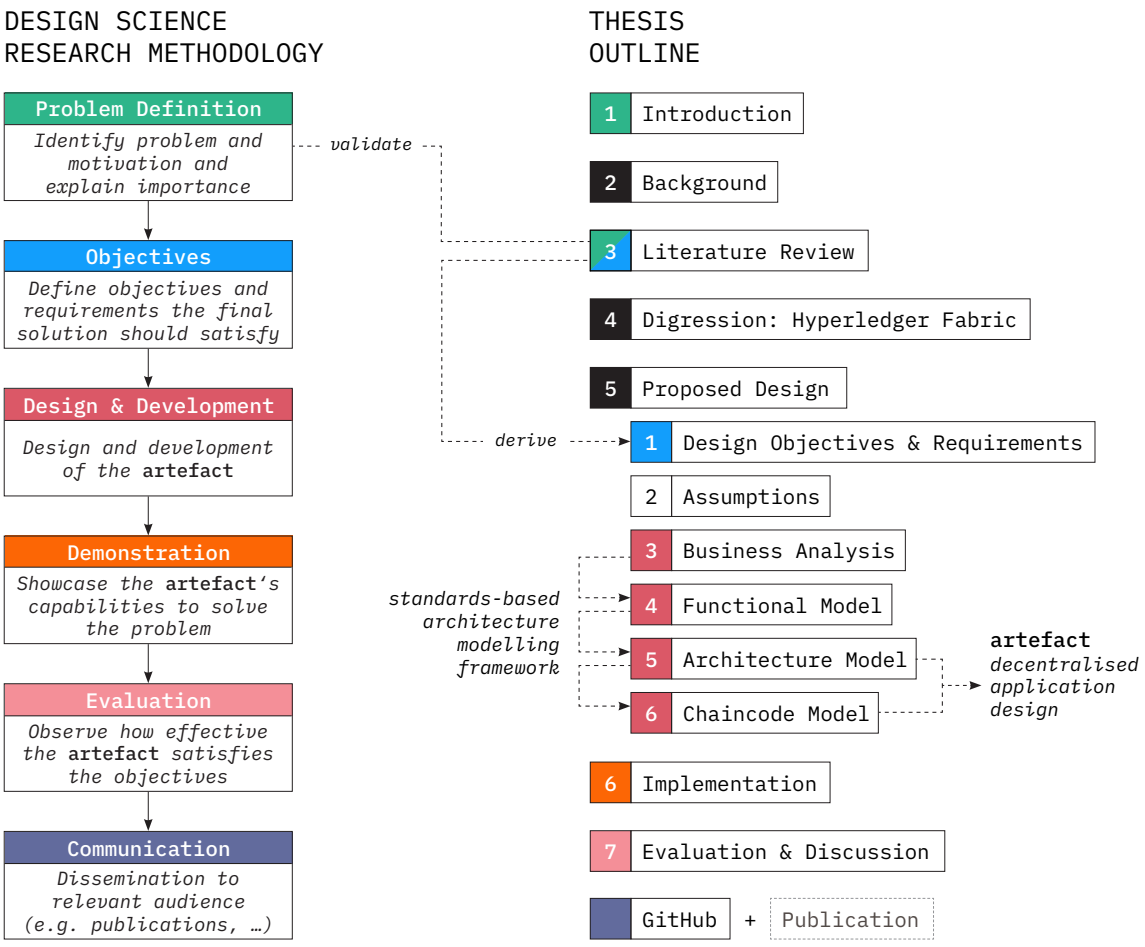


Figure 1.1: *Design science research methodology reflected in the thesis outline (own illustration based on [62])*

[2]See https://github.com/raphmc/thesis.

Chapter 2

Background

This chapter introduces concepts and terminologies which need to be defined before the literature review can be carried out (see Chapter 3). In Section 2.1, definitions of the most important terms are presented. Afterwards, an introduction into the blockchain technology is given in Section 2.2.

2.1 Definitions

This section is split into two parts: Subsection 2.1.1 introduces the *energy*-related jargon and Subsection 2.1.2 discusses *blockchain*-related definitions.

2.1.1 Decentralised Energy Jargon

Papers published in the broad field of 'decentralised energy' sometimes use the same terminologies in different contexts [19]. The following list consolidates the definitions used throughout this thesis.

Markets can be defined as *"a set of humanly devised rules that structure the interaction and exchange of information by self-interested participants in order to carry out exchange transactions at a relatively low cost"* [13].

Prosumer: describes *"an energy user who generates renewable energy in their domestic environment and either stores the surplus energy for future use or trades to interested energy customers"* [66].

Local electricity markets (LEMs): *"are geographically constrained market mechanisms with distinct pricing mechanisms between interconnected agents*

(i. e. producers, prosumers and consumers)" [42]. They *"provide households with access to a market platform for trading locally produced energy on a peer-to-peer [...] basis within their community"* [21], thereby reducing costs for market participants and helping the local community by keeping profits in its ranks [67].

Peer-to-peer (P2P): has its origin in computer science where it refers to a distributed network where *"the participants share a part of their own hardware resources [...]. These ressources are necessary to provide the Service and content offered by the network [...]. They are accessible by other peers directly, without passing intermediary entities. The participants of such a network are thus resource (Service and content) providers as well as resource (Service and content) requestors"* [68]. In the context of distributed energy, a Peer-to-Peer (P2P) market describes a market where producers, consumers, and prosumers (peers) can directly negotiate and trade energy with each other, without relying on a central authority [28].

Microgrid: is often described as *"a group of interconnected loads and distributed energy resources within clearly defined electrical boundaries that acts as a single controllable entity with respect to the grid. A microgrid can connect and disconnect from the grid to enable it to operate in both grid-connected or island mode"* [69]. Microgrids bring along important infrastructures and technologies that act as key enablers for Local Electricity Markets (LEMs)—be it in the field of monitoring, communication, or control [70]. However, a microgrid is not a requirement for LEMs which can also be a purely financial market structure operating on top of a regular distribution grid [46].

Smart meters: are part of the monitoring devices found in microgrids but are also being deployed outside of this local context. They are a prerequisite for efficient LEMs. Meters are considered *smart* when they have *"significant data processing and storage for various purposes such as: [...] data communication with the meter using secure and open standard protocols, [...] automatic reading of consumption measurements for billing and settlement, [...] real time consumption data to various actors (distributor, retailer, end user) and their automation of energy management systems"* or *"load response (load management and control)"*. [71]

2.1.2 Blockchain Definitions & Jargon

The blockchain technology is part of the broader family of Distributed Ledger Technologies (DLT). It stores transactions in a digital ledger and makes use of cryptography to secure the system [26]. This registry of transaction is distributed across a P2P network which can either be public or private [72].

No central authority is solely verifying the transactions and ensuring that all copies of the transaction log stay identical. This task is shared amongst network participants through a predefined protocol, the so-called consensus mechanism. Today, a multitude of different blockchain architectures exist, and almost as many consensus protocols [73]. [34]

As a consequence, the terminologies in the field are sometimes used ambiguously [74]. Before discussing more specific concepts of the blockchain technology in Section 2.2, the terms often used in conjunction with the word 'blockchain' are defined below on the basis of [34, 75, 76].

Blockchain technology encompasses the technology as a whole, i. e. its concepts, functionalities, and implementations.

Blockchain protocol describes the set of rules and algorithms all network participants adhere to. The consensus mechanism can be understood as part of this protocol.

Blockchain network refers to the network of physical devices—known as nodes—on which the ledger is stored and that run the protocol.

Blockchain framework is used to describe blockchains as foundational architectures on which to build applications. The terms *blockchain platform* or *blockchain architecture* are used interchangeably.

Blockchain system encompasses all blockchain participants (both transaction submitters and validators), the physical network layer, the consensus mechanism as well as all interactions between these three layers.

2.2 Blockchain Technology

Multiple publications [18,21,27,28,30] have pointed out that an adequately designed Information and Communication Technology (ICT) is a key success factor for the

realisation of LEMs and that blockchain seems to be a promising fit. This section gives an application-agnostic introduction into the workings of DLT and describes the benefits as well as the current limitations of the technology. A detailed analysis of its applicability in the context of decentralised energy trading can be found in Section 3.2.

2.2.1 Bitcoin: The First Blockchain

The concept of a blockchain was first introduced with the nascency of the digital currency *Bitcoin*. The whitepaper *Bitcoin: A Peer-to-Peer Electronic Cash System*—published under the pseudonym Satoshi Nakamoto—describes it as *"a system for electronic transactions without relying on trust"* [26]. This is achieved by cryptographically chaining transaction records together. For the sake of practicability, multiple records of transactions are put inside a block before the latter is validated and unalterably connected to the previous block of transaction records. This validation process is realised by so-called *miners*. Miners invest computational power to create consensus between all parties. Fittingly, this mechanism is called *Proof-of-Work (PoW)*. It guarantees the safety of the system in a trust-less environment. Essentially, the *miners* compete in solving a mathematical puzzle. The fastest in finding the solution proposes a new block, is rewarded for their effort with newly created *bitcoins*, and receives all transaction fees coming with the validated transactions [77]. Anyone with access to a computer and an acceptable internet connection can participate in this validation process and has full insight into the transaction history. The combination of these concepts results in an immutable and transparent database which allows P2P transmission of a digital asset without the need to rely on any intermediaries. This database is now commonly referred to as a *blockchain*. [78–80]

On top of this distributed ledger comes a second layer: the protocol. It represents the software system that transfers the asset over the underlying ledger [78]. The cryptocurrency itself is considered as the third layer. However, today there is a multitude of other digital currencies besides *Bitcoin*. They all either created a completely new system—i. e. a new blockchain, protocol and currency (e. g. Litecoin)—or built a new protocol and currency on top of the existing *Bitcoin blockchain* (e. g. Counterparty) [78]. They all strive to become a better performing digital currency, by improving aspects like the system's energy consumption, privacy, or scalability [81].

However, the applicability of distributed ledgers goes beyond the mere creation of digital money.

2.2.2 Ethereum: The First Smart Contract Platform

Since the launch of the public blockchain framework Ethereum in 2014, it is possible to create blockchain-based economic, market, and financial applications by means of so-called *smart contracts* [78]. Smart contracts are *"systems which automatically move digital assets according to arbitrary pre-specified rules"* [33] and can be seen as verified, decentralised state-changes. For example, a smart contract is automatically triggered as soon as a shipment has been delivered and a payment is sent to the supplier [80]. This transaction logic is executed on the distributed computing platform—often referred to as 'on-chain'. In the case of Ethereum, this Turing-complete[1] execution environment is called the Ethereum Virtual Machine (EVM). Once deployed, every node in the network can interact with these smart contracts. Thus, the miners in the Ethereum protocol not only validate the transactions but also execute the distributed code. [33, 83]

Ethereum was the first platform that allowed everybody to build and publish Decentralised Applications (DApps)—essentially complex smart contracts (or a set of inter-linked smart contracts which combined create a single application). Ethereum has an inherent cryptocurrency *Ether* and also created its own programming language called *Solidity*. Just like Bitcoin, Ethereum uses the PoW consensus algorithm to ensure the integrity of its ledger. While being a remarkably secure algorithm, it lacks in scalability [77]. In order to stay the go-to platform for the deployment of DApps, Ethereum has to increase the amount of transactions the network can process per second—known as throughput in the ICT jargon [84]. For this reason, Etherum is currently in the process of switching from a PoW consensus to Proof-of-Stake (PoS) [85]. PoS does not depend on computational power and solving cryptographic puzzles but rather on a validator's economic stake. Essentially, validators are chosen randomly for proposing the next block. However, the higher their deposit in *Ether*—their stake—the higher their chances of being selected. Besides the increased scalability, this consensus is also far more energy efficient and less prone to centralisation. [33, 76, 85]

[1] In computational theory, any system of data-manipulation rules is said to be *Turing-complete* if it is possible to use it to simulate any *Turing machine*, which means that the system is capable of recognising or deciding on other data-manipulation rule sets [82]. In the case of *Ethereum* this means that it can run any coin, script, or cryptocurrency project [78].

Since the introduction of Ethereum, many more blockchain platforms allowing for decentralised application design have emerged[2]. Of particular interest for the energy sector and this thesis, is the permissioned framework Hyperledger Fabric [49].

2.2.3 Public, Private & Consortium Blockchains

Before diving into Hyperledger Fabric, it is important to distinguish between the different types of blockchains—public, private, and consortium. A classification is presented in Table 2.1.

The two architectures introduced above—Bitcoin and Ethereum—are both **public blockchains**. That means that they are both completely open—anyone can join the network and submit transactions or participate in the consensus mechanism [87]. In order to become a miner, one only has to install a client running the blockchain protocol. Submitting transactions and interacting with the network can also be done though web services or a light client which eliminates the need to synchronise the full blockchain [88]. As a consequence, these blockchains are referred to as 'permissionless'.

A **private blockchain** system, on the other hand, is operated and controlled by a single organisation. This central authority has the power to attribute the rights to the other participants, i. e. decides who is allowed to execute read/write operations or takes part in the validation process. Thus, only a selected number of nodes takes part in the consensus. Often used by enterprises, private blockchains allow for increased privacy and scalability and are generally implemented for the purpose of transparency and auditability, typically within closed organisations [89]. Since the blockchain operators maintain the control over the network participants, private blockchains are also 'permissioned', i. e. the operator has to grant applicants the permission to join.

Lastly, there are **consortium blockchains**, also sometimes called **federated blockchains**. These terms apply when multiple parties operate a blockchain together and all parties are equally involved in decision-making and consensus. They can be seen as a hybrid between public and private blockchains [72]. Pre-selected nodes are responsible for achieving consensus. The ability to execute read operations might be completely open to anyone[3] but limited through a set of rules (e. g.

[2] e. g. Neo, Lisk, Cardano, Eos, Stellar [86].

[3] Ripple is a good example of a public, yet permissioned DLT. The consensus layer is restricted to a number of known nodes but anybody can join the network [90].

Table 2.1: *Possible configurations of blockchain systems*

Configuration	Consensus	Send transactions	Read operations	Level of decentralisation
Public, permissionless	yes	yes	yes	high
Consortium, (semi-)permissioned	no	yes / no	yes	medium
Private, permissioned	no	no	no	low

Sources: [76, 77] *Legend: yes = permissionless | no = permissioned*

not until the identity of an entity has been verified), or completely restricted to a group of predefined actors. Due to the different configurations possible with consortium blockchains, they are sometimes described as semi-permissioned whilst their consensus process is always permissioned. This type of blockchain makes sense when a group of actors share a common goal but the actors do not fully trust each other [60]. [48]

The governance structure of a blockchain system has a high impact on properties like performance, privacy, or level of centralisation. For instance, a public permissionless system has to protect itself from malicious agents by employing a complex, yet secure consensus mechanism like PoW. In a private permissioned system, on the other hand, all actors are known and a less computationally intensive consensus mechanism can be employed thanks to the increased level of trust between the network participants. Therefore, permissioned blockchains are more efficient[4] at the cost of centralisation. The possibility to keep data private or the ease of protocol updatability are other advantages permissioned blockchains enjoy over public ones [91]. As of today, Hyperledger Fabric is the most mature permissioned blockchain framework and therefore the system of choice for this thesis [60].

2.2.4 Hyperledger Fabric: A Privacy-Focussed Blockchain

Hyperledger is the umbrella project for all blockchain-related projects of the Linux Foundation. It was launched in 2015 and currently hosts 12 open-source projects. Half of these projects are blockchain frameworks whereas the other half are tools that are designed to work with these frameworks. Hyperledger Fabric was originally contributed by IBM and Digital Asset and was the first framework to be accepted under the umbrella project. It is still the most prominent and most mature of the frameworks [92]. Unlike Bitcoin or Ethereum, it has been developed to deploy and

[4]Propagating transactions and blocks in a public network is very time-consuming. This results in a low transaction throughput and high latency. The smaller number of validating nodes in a permissioned (consortium or private) blockchain translates into increased efficiency [77].

operate permissioned blockchains. It is very modular, supporting different consensus mechanisms or authentication systems. In contrast to Ethereum, the smart-contract logic—called chaincode in the context of Hyperledger Fabric—can be implemented with standard programming languages[5] and works without an inherent cryptocurrency. In terms of scalability, Hyperledger Fabric currently outperforms Ethereum and Bitcoin by two orders of magnitude[6]. Since the release of version 1.2 in July of 2018, the framework support so-called 'private data collections' which allow to keep certain data and transactions confidential among a subset of blockchain participants [95]. This paragraph is meant to give only a short introduction to Hyperledger Fabric. It will be discussed in greater depth in Chapter 4: *Digression: Hyperledger Fabric*. A glossary, listing the important terminologies, can be found in Appendix A.

2.2.5 Disruption Potential & Key Benefits

Distributed ledgers are seen to have the potential to not only disrupt products and services offered by incumbent businesses but to change the whole economy and society [96]. Energy and financial service industries have been identified as most sensitive to disruption in a survey conducted by [97]. Information technology and law are seen as close followers by the respondents. The disruptive potential of the technology for the energy sector has been pointed out by [27, 30, 98, 99].

Blockchains have multiple benefits that make them interesting for a broad set of applications. The following non-exhaustive list presents five key benefits of the technology and how these benefits are exploited by different actors:

1. **Efficiency:** By eliminating the intermediary, transactions can be settled between two parties without lag. Smart contracts even allow for programmed actions as soon as they get triggered which can reduce the need for human intervention and optimise processes while reducing cost and transaction time. Known actors within the energy sector bet on blockchain technology, e. g. to reduce costs related to energy trading, to facilitate peer-to-peer energy trading, or to streamline the billing process for autonomous vehicles. [100]

2. **Auditability:** Each transaction is chronologically and immutably stored within the blockchain, thus enabling an auditor to follow the history of each asset even if multiple parties are involved. The blockchain company Everledger

[5] As of today, chaincode can be written in Golang, node.js, and Java [93].

[6] Bitcoin can handle around 10 Transactions Per Second (TPS), Ethereum maxes out at 20 TPS, and Hyperledger Fabric has been shown to process over 3500 TPS [60, 94].

exploits this characteristic to guarantee the identity and legitimacy of objects—notably, diamonds. [101]

3. **Traceability:** Following goods in supply chain is one example of how an immutable and transparent transaction history can improve traceability. For instance, Maersk and IBM recently announced the co-creation of a *"blockchain-based platform for global trade"* to eliminate friction in global logistics [102]. Maersk has been experimenting with tracking containers since 2016. [78]

4. **Transparency:** Sharing the history of records with partners, customers, or public institutions can eliminate delays in commerce, improve a company's image, or facilitate regulation [103]. The creation of trust thanks to the immutability of a blockchain can be leveraged to reduce transaction costs and eliminate bureaucratic burdens [104]. The Energy Web Foundation offers an application called *EW Origin* based on a consortium blockchain that registers certificates of origin—certifying the generation of 1 MW of renewable energy—on-chain [105]. The start-up Provenance provides a blockchain application allowing for fully transparent food chains [106]. This way, the end consumer can trust the provenance of their food.

5. **Security:** Thanks to cryptography and the distributed nature of blockchains, authenticity of information can be guaranteed. Even though private blockchains are considered less safe compared to their public counterparts, they still represent a big step forward with regard to cybersecurity [79]. This aspect is of major interest within the Internet of Things environment whose expansion has been hindered by security issues for some time. [107–109]

2.2.6 Barriers & Limitations

Besides the above advantages about this new technology, there are also some barriers that limit its propagation. As with most emerging technologies, the immaturity of the technology leads to hesitant adoption since some security and governance questions are still unanswered [110]. It also means that some technical challenges are still not conclusively resolved, such as throughput, latency, the size of the database, lack of privacy, hidden centrality, and high costs as well as high energy consumption [78, 79, 111]. The uncertainty concerning to regulation is also a serious obstacle [27]. For example, the legal consequences of blockchain transactions are still unclear in many countries and cryptocurrencies remain a widely unregulated field [112].

The wish for a clear regulatory framework has also been pronounced in the survey conducted by [97]. The interviewed professionals also identified insufficient knowledge as a significant barrier that is limiting adoption—it is a complex technology and not everyone is eager to invest time and effort to decipher its basic functioning [78]. Furthermore, in order to reach to a truly efficient economy on the back of immutable blockchains and smart contracts, interoperability and compatibility between different blockchains is of major importance [36, 113]. However, projects[7] working on solving this difficulty exist but most are still at an early stage of development [35].

Now that the basic terminology and concepts with relevance for this thesis are set out, the literature around market designs and ICTs systems for LEMs is reviewed in the next chapter.

[7]Two promising projects are *Polkadot* [114] and *Cøsmos* [115].

Chapter 3

Literature Review

This chapter summarises the existing literature around Local Electricity Markets (LEMs). As illustrated in Figure 3.1, LEMs can be broken down into three layers: (1) the *physical layer*, representing the physical appliances and the flow of electrons; (2) the *Information and Communication Technology (ICT) layer*, necessary for the exchange of information between market participants and technical components; and (3) the *market layer*, which describes the rules of the market such as the market matching mechanism and agent strategies. In some cases, the market layer is also referred to as the 'virtual' layer [28]. The *Advanced Metering Infrastructure (AMI)*[1] represents the connection between the two lower layers. The *blockchain-based energy trading marketplace* constitutes the link between the *market* and the *ICT layer* (see Chapter 5). [19]

In LEMs, one typically encounters some kind of smart grid infrastructure on the physical layer which is managed by smart grid protocols on the ICT layer [117]. This literature review, however, only focusses on the *market layer* (see Section 3.1) and the necessary *ICT* (see Section 3.2) to support it, since these are the two layers which are to be connected by means of a blockchain-based energy trading marketplace—the element of focus in this thesis (see Figure 3.1). Section 3.1 of this chapter analyses the former and focusses on: different market topologies, appropriate market matching and pricing mechanisms, bidding strategies, and the impact of flexibility on the market. Section 3.2 discusses how LEMs could be implemented with the use of blockchain as technology of choice for the ICT layer. At the end of each section, the deductions which can be drawn from the analysed literature and which are of relevance for the design of a decentralised energy trading platform are summarised.

[1] AMI is a collective term to describe the whole system of smart meters, the communication protocol, and all applications that enable the collection, storage, analysis, and transfer of energy consumption or production data, close to real-time [116].

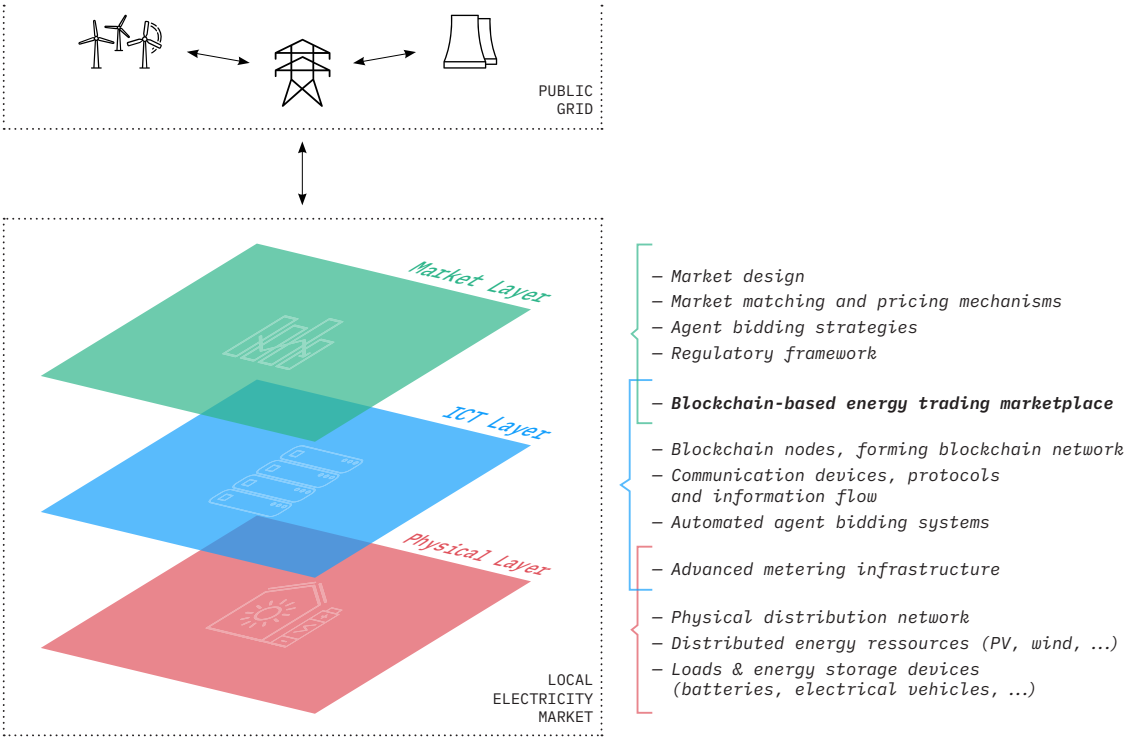


Figure 3.1: *The three layers of a local electricity market (own illustration based on [49] and [23])*

3.1 Market Layer of Local Electricity Markets

Despite increasingly decentralised production and management of energy, traditional electricity markets remain centralised, top-down systems [118]. LEMs on the other hand, have the potential to reorganise allocation of electricity, provision of flexibility, and even grid stabilisation in a bottom-up manner [28]. However, these markets can be organised in a number of ways. This section investigates the intricacies of market design in the context of LEMs in order to derive some key requirements an efficient decentralised energy marketplace should meet.

In Subsection 3.1.1 the different topologies of LEMs are discussed: (1) a *full Peer-to-Peer (P2P)* market design; (2) a *community-based* market design; and (3) a *hybrid* market design. Subsection 3.1.2 investigates market matching and pricing mechanisms and puts them in relation to publications from the field of LEM designs. In Subsection 3.1.3, the connection between market design and agent strategies is presented. Subsection 3.1.4 illustrates the importance of flexibility in the context of LEMs. Finally, Subsection 3.1.5 presents the deductions which can be drawn from this market-oriented literature review.

3.1.1 Local Electricity Market Topology

Different structures for locally trading electricity in a P2P manner exist. The three most common designs are full P2P markets, community-based markets, and an approach combining the first two—henceforth called hybrid markets. They differ from one another in their level of decentralisation and organisational management. [119]

Full P2P Market

Figure 3.2 illustrates a full P2P market design where market participants negotiate directly with each other without centralised supervision. Consequently, there is one price for each bilateral trade which can in theory be different for each and every trade. A major benefit of this design is the high level of privacy it brings along. Each market participant only shares information about the amount of energy and the price at which they are willing to trade, without exposing sensible data. Examples of such a design can be found in [21, 118, 120].

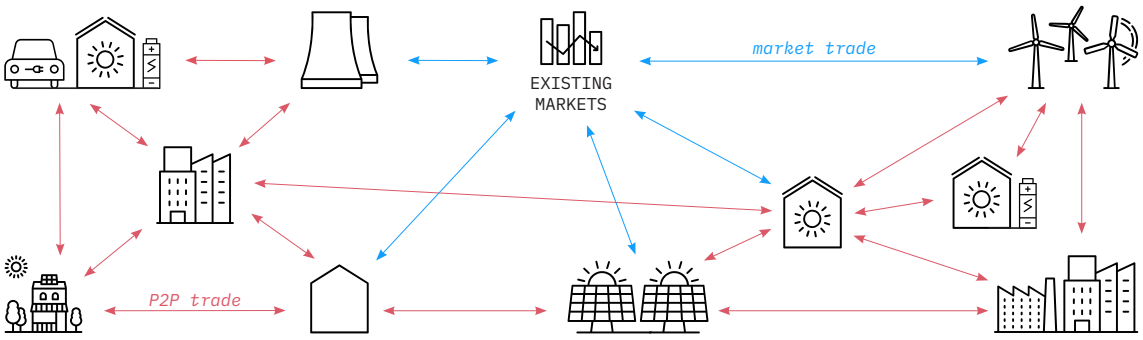


Figure 3.2: *Design of a full P2P market (own illustration based on [28])*

Community-based market

The design shown in Figure 3.3 is more centralised and relies on a Local Market Operator (LMO) or community manager. The role of this actor is to organise trading activities within the local community and act as intermediator between the community and the traditional energy system. While a full P2P design could theoretically be stretched out over a national (or even international) system, the community-based approach generally is linked to geographic proximity. Microgrids [121, 122] or a neighbourhood with a high density of prosumers [123, 124] are a perfect fit for this

market design. Because trading is being coordinated by the LMO, market participants don't know with whom they are actually trading electricity. The community is connected to the public grid which provides the local market with energy in times of high demand or acts a an energy sink for excess production. Every market participant is thus free to buy from or sell electricity to the public grid. The LMO generally aggregates all external trades, meaning that the local community operates as a single actor with the outside world. Some designs include the possibility for nesting of communities, here referred to as hyper-local communities. Refer to [125, 126] for more exemplary community-based markets.

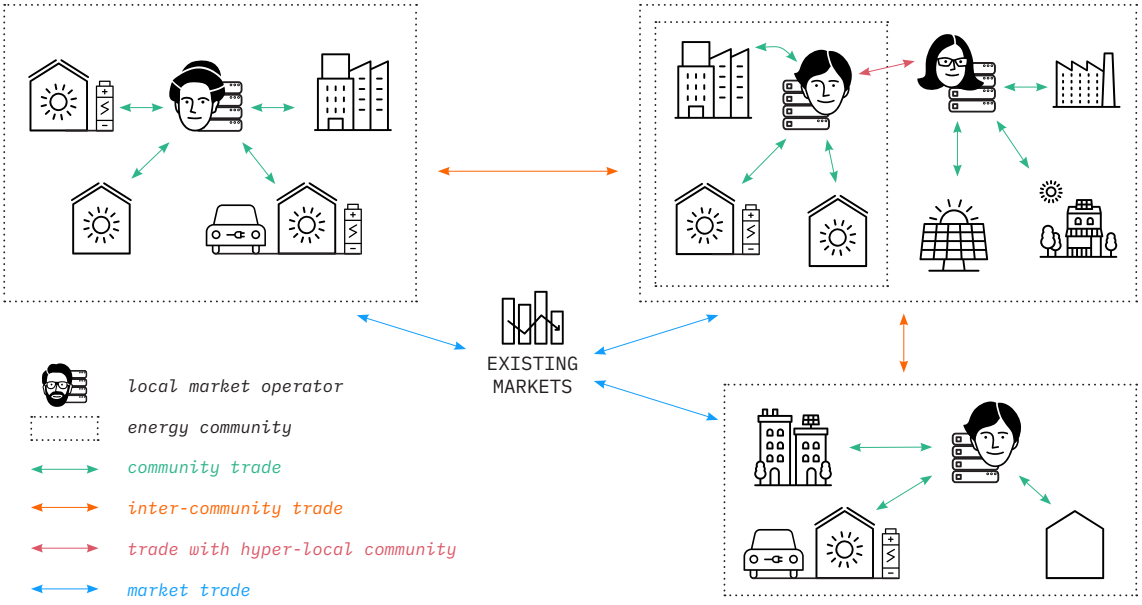


Figure 3.3: *Design of a community-based market (own illustration based on [28])*

Hybrid market

This design can be understood as a merger of the previous two designs. As shown in Figure 3.4, energy trading thus takes place on multiple layers. This means that on each layer, peers and local communities can interact directly with each other. The highest level is characterised by a mixtures of different actors (e. g. individual peers, communities or energy collectives) that trade in a full P2P fashion with each other but also engage in transactions with traditional markets. The lower levels are organised in a community-based manner, as described above. It is also possible for a local community (e. g. an agglomeration of houses in a neighbourhood) to be part of a bigger energy collective (e. g. a city district), as illustrated in Figure 3.4. Examples of such a hybrid design are [127, 128].

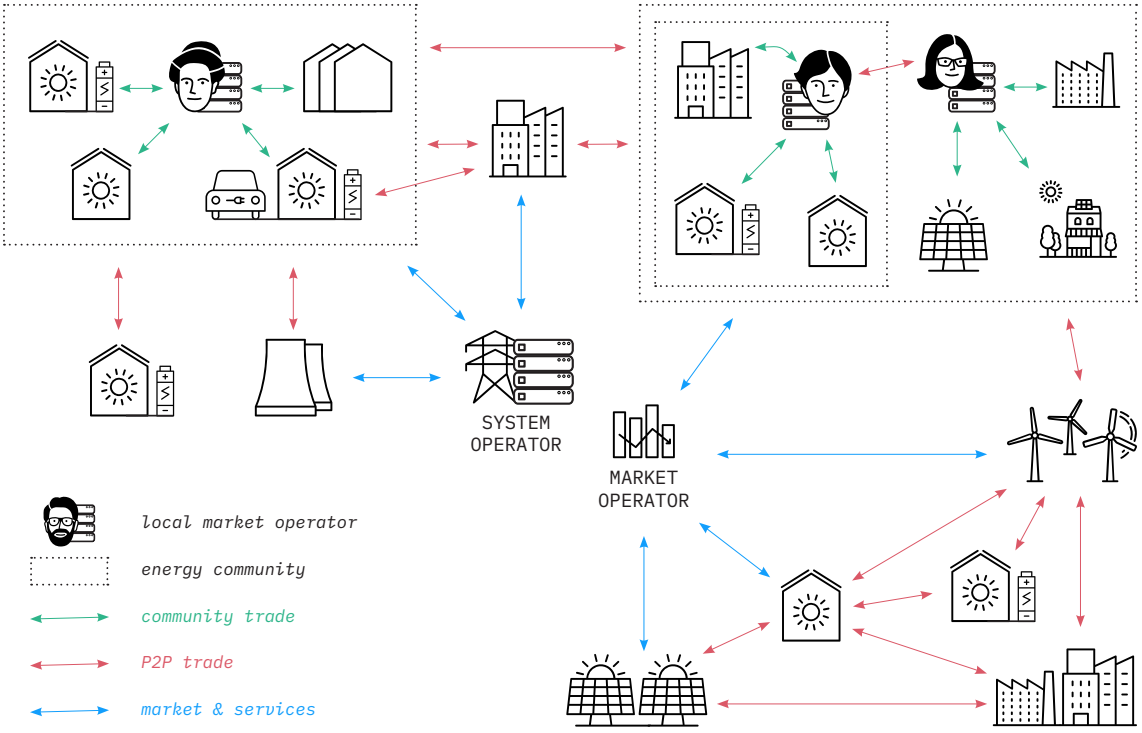


Figure 3.4: *Design of a hybrid market (own illustration based on [28])*

Comparison

All three presented market designs come with advantages and disadvantages, which are summarised in Table 3.1 based on [28] and [119].

A benefit of the full P2P approach is that energy consumption and consumer preferences are fully aligned. However, this design comes with a scalability problem with regard to the negotiation process since a high number of market participants results in a high level of computational complexity [133]. P2P market models have shown that this design can lead to sub-optimal energy prices for market participants while more centralised approaches can benefit from more efficient auction-based pricing mechanisms[2] [25, 28].

A community-based model, on the other hand, has the benefit of increasing the identification with the local community. Furthermore, the LMO can act as an aggregator and negotiate better deals with external electricity suppliers than market participants could do on their own. Grid operators may also benefit from this central actor who may provide them with services such as flexibility. This way, grid problems could be solved through cooperation of market participants rather than

[2]See Section 3.1 for an in-depth analysis on market matching and pricing mechanisms.

Table 3.1: *Comparison of full P2P, community-based, and hybrid market designs*

Design	Advantages	Challenges	References
Full P2P	Total freedom of choice and autonomy, empowering the active consumers	Investment and maintenance with ICT infrastructure in case of scalability to the whole energy system	[21, 118, 126, 129]
	Energy use aligned with each agent's preferences (e. g. cost, green, local, ...)	Potentially, slow convergence to obtain a consensus in the final delivery of energy	
	Complete 'democratisation' of energy use	Predicting system behaviour by grid operators, because of the lack of centralised control	
		Guarantee of safety and high-quality energy delivery	
Community	Enhancing the relationships and involvement of community members, because of sharing a common good (i. e. energy)	Reaching the preferences of energy use for all community members at all times	[121–126, 128, 130–132]
	Mobilising social cooperation and resilience in community members	For the LMO: aggregating all members' data and managing their expectations	
	Potential new services for grid operators provided by the community manager	Having a fair and unbiased energy sharing among community members	
	Good scalability thanks to limited scope of energy community		
Hybrid	ICT infrastructure and computation effort are scalable to the whole energy system	Coordinating internal trades in the communities with trades between high-level agents (e. g. community managers, utilities, ...)	[127, 128]
	Most compatible with the system in the next years. It can be seen as co-existent design of the two previous ones		

Source: based on [28] and [119]

reinforcing the grid, increasing the resilience and security of the power system. However, in a community market participants loose the freedom to choose their trading partners which may lead to misaligned interests between single community members and the community as a whole[3]. [28]

Finally, the hybrid design inherits some advantages and disadvantages from the other two design—namely, the ability for all agents to freely choose their level of involvement in the system, as well as the possibility to increase identification with

[3]For instance, in the *full P2P* approach market participants can chose exactly from whom they buy energy and thus have the freedom to only trade with their preferred actors, if they wish to do so. In the *community-based* approach, on the other hand, one cannot decide from whom in the local market the electricity "comes from" or to whom one is selling.

a local energy community. The computational burden of the full P2P design is significantly reduced due to smaller numbers of actors on each level. A model-based study by [28] concludes that a hybrid market is the most suitable design in terms of scalability. The authors also point out that a distributed, blockchain-based approach has the benefits of increasing the efficiency of the system while maximising the agents' privacy. [18] propose a similar design where trading takes place on three levels: (1) P2P trading within a microgrid; (2) P2P trading within cells of multiple microgrids; and (3) P2P trading among cells. These hierarchically organised P2P markets try to balance power and energy on each level, respectively. The authors argue that this design will *"increase the efficiency, flexibility and responsiveness of local resources"* and that blockchain is *"a very promising technique which can simplify the metering and billing system of the P2P energy trading market"* [18].

A major threat to all of these designs is the legal framework which, in most countries, limits the access to electricity markets for small actors [134, 135]. However, the increasing number of government-funded projects from the field is a sign that those barriers might significantly decrease, or eventually disappear completely, in the future [28]. For instance, France has paved the way for community-based LEMs thanks to the 2016 modified article L315-2 of the *code de l'énergie* [136] by allowing so-called *autoconsommation collective*—collective self-consumption. In the *Clean Energy For All Europeans* package, the European Commission further introduced regulatory changes to *"shift from centralised conventional generation to decentralised, smart and interconnected markets"* which *"will also make it easier for consumers to generate their own energy, store it, share it, consume it or sell it back to the market—directly or as energy cooperatives"* [137].

A structured literature review carried out by Mengelkamp et al. [19] found that recent work in the field of LEMs *"puts more emphasis on the community approach of electricity trading of prosumers and consumers"*. A possible reason for this development it that the implementation of a community-based market faces the least obstacles in most regulatory environments. For instance, the LMO could take the role of energy supplier for the community members and manage the compliance with legal requirements. Although not in the context of P2P trading, the *sonnenCommunity*[4] by the German battery storage manufacturer sonnen represents a real-world example where all community members virtually pool their battery storage and share self-produced electricity with each other [18, 138]. In this case, sonnen takes the role of the community manager, thus eliminating the need for a classical

[4]See https://sonnengroup.com/sonnencommunity.

energy provider. Furthermore, they currently run multiple pilot projects leveraging blockchain technology for transparency and accounting purposes [139]. *Piclo*[5], *Vandebron*[6], and LichtBlick's *SchwarmEnergie*[7] are other exemplary projects which follow similar approaches [18].

3.1.2 Market Matching & Pricing Mechanisms

The discipline of market design deals with *"the creation of a venue for buyers and sellers, and a format for transactions"* [140]. There is an abundance of market designs, each with their own set of strengths and weaknesses. This thesis focusses only on the mechanisms most relevant for this work which are presented hereafter.

In the academic literature on LEMs, auction-based markets are the predominant design; refer to [19] for a detailed overview. As a consequence, they are discussed in greater detail hereafter. For comparison, two non-auction-based market designs are touched at the end of this subsection.

Auction-Based Market Designs

Auction designs exit in any shade of colour. A subset of differentiating factors are single versus double auction, periodic versus continuous, or discriminatory versus uniform pricing. However, most auctions follow the same three steps: bidding, clearing, and pricing. The bidding phase may also include the possibility to place asks—offers that propose a good at a given price. Discrete (periodical) double auctions with a single Market Clearing Price (MCP) (uniform price) are most popular in the context of LEM, but other designs are also worth looking into.

Single auctions, also called *primary* auctions, are used for the allocation of a single good. They can be categorised into two camps: (1) *open-outcry*, where all bids are visible to all market participants; and (2) *sealed bidding*, where the bids remain hidden from other market participants. The most common designs from this first category are the *English* or the *Dutch auction*. In the case of an *English auction*, the auctioneer starts with a low price, bidders compete by placing ever higher bids and the highest bidder wins the auction. *Dutch auctions*, on the other hand, work the other way around, i. e. the auctioneer starts with a high price and lowers it until a bidder accepts it. [141]

[5] See https://piclo.energy.
[6] See https://vandebron.nl.
[7] See https://www.lichtblick.de/schwarmenergie/.

Typical designs that fall into the *sealed bidding* category are the *first-price sealed-bid auction* and the *Vickrey auction*. In the former—also known as *blind auction*—all bidders submit their bids simultaneously, thereby keeping the bids of the other market participants hidden. The highest bidder wins. The *Vickrey auction* functions similarly, except that the winning bidder only pays the second-highest price which leads to all bidders submitting their 'true' valuations. Figure 3.5 illustrates the described single auction designs. [141]

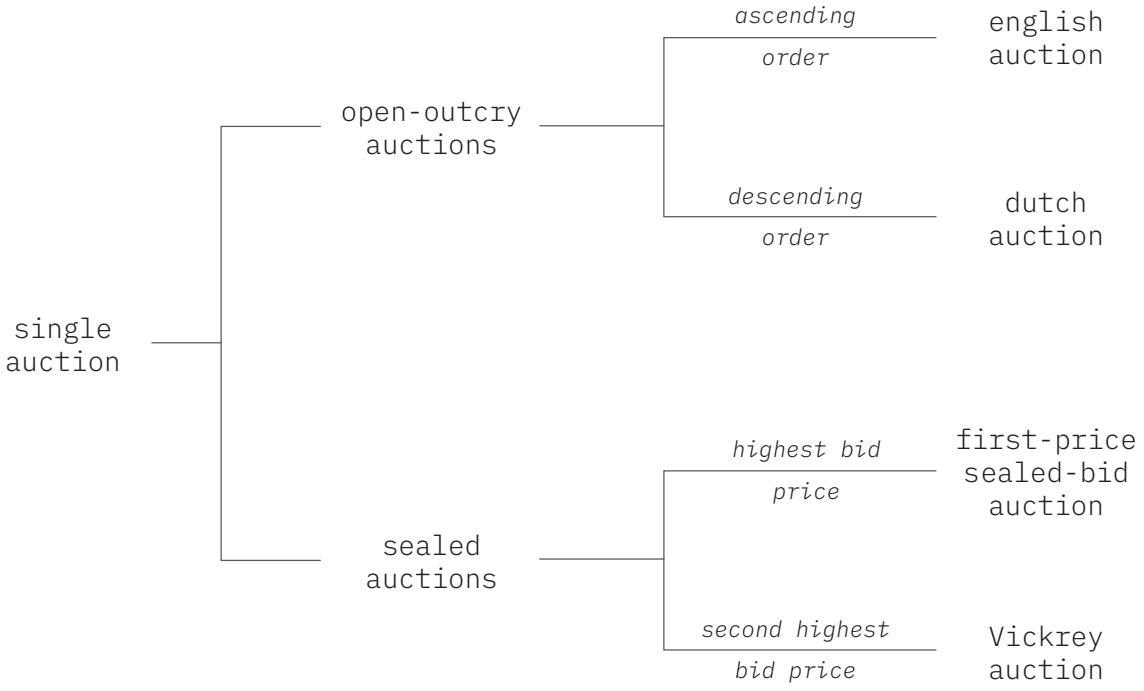


Figure 3.5: *Classification of single auction designs (own illustration based on [141] and [43])*

Double auctions are characterised by a process where potential buyers submit their bids and potential sellers submit their asks to an auctioneer. In the case of a discrete double auction with uniform pricing, the auctioneer then calculates the price p that clears the market for every trading period. Simply put, buyers whose bids are equal or higher than p and sellers whose asks are equal or lower than p are matched. However, *double auctions* come in as wide of a variety as *single auctions* which is why they are analysed in greater detail hereafter. [142]

A double auction can be *one-shot*, meaning it has only one trading period, or *repeated* when it has several periods and each agent receives a new allocation of commodities in every new period. *Periodic* or *discrete time* double auctions are cleared at a fixed time after the auction start. In contrary, *continuous* auctions do not rely on

a fixed time to match demand and supply. The auctioneer tries to clear the market each time a new order is placed. Figure 3.6 shows both demand and supply curves of a discrete time double auction. In this exemplary case, the MCP, also called *equilibrium price*, equals 25.

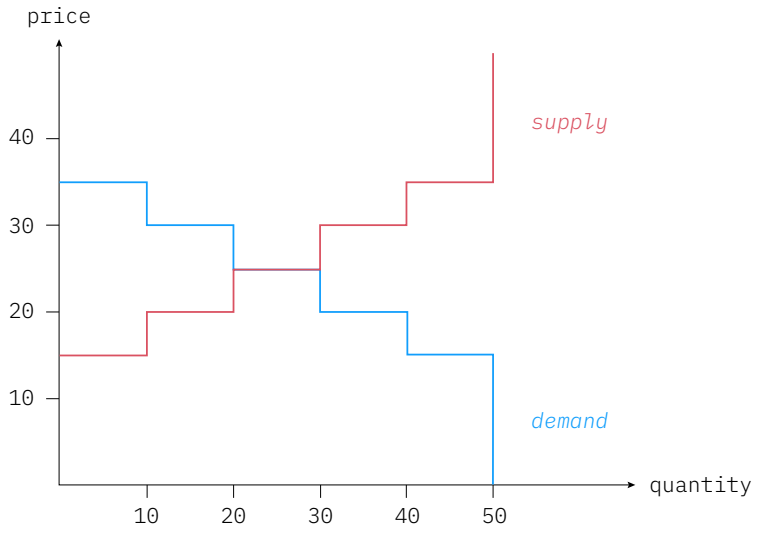


Figure 3.6: *Exemplary supply and demand curve of a double auction (own illustration based on [142])*

The Myerson-Satterthwaite theorem formulates four key principles an ideal auction mechanism should satisfy [143]. However, it is important to note that no auction mechanism can satisfy all four of these requirements—three out of four is the maximum [144]. These principles are:

1. **Individual Rationality (IR):** This requirement is satisfied as long as no agent finds themself in negative utilities after trading in the market.
2. **Weak Balanced Budget (WBB):** When a trade does not have to be financially subsidised by the market operator, it is called *budget-balanced*. If the market operator profits from the transactions between market participants the *weak budget balance* criterion is still satisfied.
3. **Nash Equilibrium Incentive Compatibility (NEIC):** This principle is met when all market participants have an incentive to report their true valuations—regardless of other agents' strategies. It is also referred to as *strategy-proofness* and *truthfulness*.
4. **Pareto Efficiency (PE):** In order to maximise social welfare and satisfy *Pareto efficiency*, the auction mechanism has to allocate the limited quantity of goods to those market participants who value them the most.

What follows is a selection of relevant auction designs and how they compare with regard to the four principles defined by the *Myerson-Satterthwaite theorm*:

Discriminatory k-Double Auction: Both buyers and sellers submit sealed bid prices B^P or ask prices S^P for one unit of a good. Bid prices are then sorted in descending order and ask prices in ascending order following the natural ordering rule. A trade occurs when $B^P p = kB^P + (1-k)S^P \geq S^P$ at the following price p:

$$p = kB^P + (1-k)S^P \tag{3.1}$$

The constant k is predetermined and either lies in the closed interval $[0, 1]$ or the open interval $(0, 1)$. In the case of $k = 1$, the trading price will be $p = B^P$, thus if $B^P > S^P$ the seller gains in utility since they receive more than asked. At $k = 0$, $B^P > S^P$, and a resulting trading price of $p = S^P$, the buyer gains in utility since they traded at a lower-than-expected price. At $k = 0.5$ neither side gains more utility than the other side. Since the trading price is determined between each winning buyer-seller pair, its pricing mechanism is called discriminatory. This design satisfies IR, WBB and PE but is not NEIC since market participants have an incentive to submit offers that differ from their true valuations in the interest of individual gain. [144]

Uniform k-Double Auction: This auction mechanism closely resembles the above design, except that all market participants trade at the same MCP. After following the natural ordering rule and sorting bids and asks as for the *discriminatory k-double auction*, the largest breakeven index γ is found where $B^P_\gamma \geq S^P_\gamma$. The price is calculated as:

$$p = kB^P_\gamma + (1-k)S^P_\gamma \tag{3.2}$$

B^P_γ and S^P_γ represent the bidding and asking prices where demand and supply match best, thus where the breakeven index is largest. Thus, the first γ sellers can sell to the first γ buyers. The constant k follows the same rules as for the *discriminatory k-double auction* as does its performance with regard to the *Myerson-Satterthwait theorm*. A new MCP is determined every auction period. [144]

The Average Mechanism: This mechanism is essentially a special case of the *uniform k-double auction* where $k = 0.5$. It follows the same steps and the MCP can be simplified as:

$$p = \frac{B_\gamma^P + S_\gamma^P}{2} \tag{3.3}$$

This mechanism is commonly found in literature about LEMs and satisfies the IR, WBB, and PE principles. Also here, sellers are incentivised to report higher valuations and buyers lower valuations, respectively, making it not *strategy-proof* and therefore fail at NEIC. [142]

Vickrey-Clarke-Groves (VCG) mechanism: This auction design belongs to the family of *Grove mechanisms* which are *strategy-proof* and *efficient* mechanisms for market participants with quasi-linear preferences [145]. This mechanism requires market participants to send sealed buy and sell orders (i. e. only visible to the auction system). The allocation is identical to the *uniform k-double auction*—the first γ sellers sell their goods to the first γ buyers. The particularity lies in the pricing mechanism: all buyers pay the lowest equilibrium price and all sellers receive the highest equilibrium price. It can be expressed as:

$$p_B = max(S_\gamma^P, B_{\gamma+1}^P) \tag{3.4}$$

and:

$$p_S = min(S_{\gamma+1}^P, B_\gamma^P) \tag{3.5}$$

where $B_{\gamma+1}^P$ and $S_{\gamma+1}^P$ represent the bid price and ask price following the largest breakeven index, respectively. If p_B is lower than p_S, the auctioneers needs to subsidise the trade. The satisfaction of the NEIC principle thus comes at the expense of the WBB principle. Just like the all previous designs, both IR and PE are met. [145]

Trade Reduction mechanism: This auction design works without the market operator's trade catalysis and still satisfies the principle of *strategy-proofness*. This is achieved by removing the last winning buyer and seller from the trade, thus limiting the set of successful trades to $\gamma - 1$. While satisfying the NEIC principle, this mechanism cannot be called *pareto-efficient* since it excludes the $\gamma^t h$ buyer and the $\gamma^t h$ seller from the auction. The pricing mechanism is as follows: buyers pay B_γ^P and sellers receive S_γ^P. Since B_γ^P can be equal or higher than S_γ^P, the auctioneer can financially gain from this transaction ($B_\gamma^P - S_\gamma^P$ per unit of good and transaction), thereby still satisfying the principle of *weak*

budget balance. Similarly to the VCG mechanism, sealed bidding is a requirement for satisfying NEIC. And just like all previous designs, it complies with IR. [146]

Table 3.2 summarises the satisfaction of the principles formulated by *Myerson-Satterthwaite* for the five auction mechanisms, respectively. The comparison shows that both *strategy-proof* auction designs, the VCG and trade reduction mechanisms, require sealed bidding.

Table 3.2: *Comparison of five different double auction designs with respect to their compliance with the four principles of an ideal auction presented by Myerson-Satterthwaite [143]*

Auction mechanism	**IR**	**WBB**	**NEIC**	**PE**
Discriminatory k-double auction	✓	✓	✗	✓
Uniform k-double auction	✓	✓	✗	✓
Average mechanism	✓	✓	✗	✓
Vickrey-Clarke-Grove mechanism	✓	✗	✓	✓
Trade reduction mechanism	✓	✓	✓	✗

Now that the basics of auction theory were discussed, two publications which rely on auction-based market matching in the context of LEMs are presented:

The Brooklyn Microgrid project is one of the most famous implementation of a LEM and was reviewed by Mengelkamp et al. [21]. The local market spreads over three distribution grid networks in Brooklyn, New York, and is run by a company called LO3 Energy. Market participants have the possibility to trade electricity among each other on a P2P basis. Energetic and financial transaction data is stored on a blockchain system. The microgrid is connected to the public grid, guaranteeing energy supply in times of low production or unsuccessful bidding in the local market. The implemented market design is a discrete time double auction with market periods of 15 minutes each. Trading is done by software agents but the participants have the option to define a preferences profile (e. g. favour locally produced electricity). However, this design is not set in stone and the operator has planned to experiment with other market and pricing mechanisms (e. g. discriminatory double auction).

Vytelingum et al. [147] chose a continuous double auction as their design of choice for a day-ahead smart grid market. The authors argue that this design has proven effective in similar systems. The order book is public, thus market participants have visibility of the unmatched offers. The authors also included a particularity in their design: only offers that improve themselves are accepted. This results in the fact

that buyers can only increase their bids while sellers can only decrease their asks. A similar policy is currently in place at the New York Stock Exchange [148].

The interested reader may further refer to Maity and Rao [67] for a comparison between a discriminatory and uniform pricing mechanism in a sealed-bid auction for a solar-powered microgrid.

Non-Auction-Based Market Designs

Bilateral P2P markets were first proposed by Blouin and Serrano [149]. Their design is based on random and anonymous pairwise meetings between buying and selling market participants. The benefit of this approach is that it does not rely on a central authority once implemented, thereby avoiding the aggregation of information at a central point. The mechanism functions as follows: buyers and sellers are matched randomly and then have the opportunity to come to an agreement regarding the price. If they fail to find common ground, they remain in the market and are matched with a new counterpart. If an agreement is found, the transaction is settled. Afterwards, they leave the market if the trade fully satisfied their demand or offer, or stay in the market to find a new trade partner. Subsequently, an individual price is found for every successful pair of buyer and seller.

A study conducted by Mengelkamp et al. [25] on market designs and bidding strategies on LEMs has shown that this P2P market design—in conjunction with intelligent bidding agents which are able to learn from past events—reached the highest amount of locally traded energy. According to the authors, this system finds all trades an auction-based market would find plus a number of matches facilitated by the P2P mechanism. However, the authors also state that auction-based markets have the potential to find the *"lowest feasible market prices"* [25]. Sousa et al. [28] further raise concerns regarding system scalability due to complex negotiation processes with respect to bilateral P2P markets.

Sorin et al. [118] propose a design which allows for multi-bilateral trading as well as product differentiation where market participants can express their preferences.

Price functions: a market design proposed by Mihaylov et al. [150] relies on price functions for market price determination. The prices for buying or selling

electricity depend on the total consumption and production in the local market. The objective of this approach to balance supply and demand by creating a financial incentive for market participants to act in a grid-friendly manner. In this example, the Distribution System Operator (DSO) controls these price functions. Therefore, it is not a completely decentralised system. The possible price range is not limited by the public grid, thus in times of high production the price for consuming energy can be as low as zero. The following formulae define the pricing mechanism:

For producers, the price is defined as follows:

$$g(x, t_p, t_c) = \frac{x \cdot q_{t_p=t_c}}{e^{\frac{(t_p - t_c)^2}{a}}} \tag{3.6}$$

And the price for consumers is determined by:

$$h(y, t_p, t_c) = \frac{y \cdot r_{t_c>>t_p} \cdot t_c}{t_c + t_p} \tag{3.7}$$

where:

x	=	the amount of electricity produced by a market participant
y	=	the amount of electricity consumed by a market participant
t_p	=	the total supply of electricity
t_c	=	the total demand of electricity
$q_{t_p=t_c}$	=	the maximum rate producers can get for their electricity
a	=	a scaling factor in the case of $t_p \neq t_c$
$r_{t_c>>t_p}$	=	the maximum cost for producers in times of low supply

Vytelingum et al. [147] and Seidl [49] show how these price functions can work in conjunction with other market mechanisms. Both use predefined functions to incorporate congestion management into their auction-based market designs. In his thesis, Seidl [49] also adopts this approach to account for voltage and frequency variations.

3.1.3 Agent Strategies

One has to keep in mind that the choice between different market and pricing mechanisms always affects the buying and selling strategy of market participants. In game theory and mechanism design, these market participants are called *agents* and represent individuals or users. The *strategy* of an agent defines *"its complete and*

contingent plan, which allows the agent to select different actions depending on every unique state of the game" [22].

On an abstract level, a LEM essentially represents a Multi-Agent System (MAS), i. e. a decentralised system composed of self-interested agents. In such systems, market-based approaches have proven to be most successful in achieving high levels of efficiency [151]. In the case of LEMs, market participants are represented by software agents which each try to buy electricity at low prices or sell it at high prices. The self-interest of these agents shifts consumption to times of low prices and production to times of high prices, thereby flattening the demand and supply curves. This effect has a positive impact on grid stability. [23, 152]

This thesis focusses on the design of a decentralised energy trading marketplace. The design of software agents is not the scope of this work but since agents are in direct interaction with this marketplace, a brief introduction into the field is given.

Literature on LEMs has shown some interest in the impact of agent strategies on ratios like market efficiency or locally traded energy. Therefore, agents are typically divided into *Zero Intelligence (ZI) agents* and *intelligent agents*. ZI agents place random offers (bids and asks) within a predefined price range, e. g. feed-in and buy prices of the public grid. Intelligent agents, in contrast, follow more complex patterns and often rely on complex algorithms fed with historical data or data representing the current state of the system. [147]

Lamparter et al. [153] conclude that LEMs can be understood as a MAS. The agent strategy proposed by the authors follows local policies representing the preferences of market participants or constraints of the agents' physical appliances. This ensures that strategies remain in a desired action space while at the same time leaving space for autonomous decisions. The bidding process essentially follows a three-layer approach: (1) information is gathered (information layer); (2) the information is crossed with the agent's appliance policies, returning a set of feasible actions (knowledge layer); and (3), these actions are ranked best-to-worst based on the agent's preference policies (behavioural layer).

Already mentioned above, Mengelkamp et al. [25] have compared two market designs and the impact of intelligent and ZI agent strategies on these markets in their paper *Trading on Local Energy Markets: A Comparison of Market Designs and Bidding Strategies*. The ZI agents placed random bids and asks in a spectrum of 12.32 to 28.69 Euro-cent per kWh. According to the authors, ZI agents are a good way of finding the lower market efficiency boundary. The intelligent agents learned from

past decisions by analysing historical data on *income* and/or *saved costs*. To avoid falling into a specific bidding pattern too quickly, a memory parameter $\lambda \in [0, 1]$ was introduced. Furthermore, a parameter traducing into the speed of learning was implemented with $\epsilon \in [0, 1]$. Strategic bidding—e. g. buying more than demand and try to sell the surplus at higher prices—has not been considered. The analysed parameters are *self-consumption inside the LEMs*, *total energy traded*, *market* and *overall weighted electricity prices*, and *revenue* and/or *costs* for all market participants. As expected, intelligent agents outperform their 'dumber' counterparts in all disciplines.

In 2018, the authors published another paper with the title *Intelligent Agent Strategies for Residential Customers in Local Electricity Markets* [154] where they analysed intelligent bidding strategies. The research context was a LEM with a double auction and uniform pricing. Erev-Roth learning—a form of reinforcement learning—is applied to maximise each agent's individual utility and further extended by *generation* and *storage states* as well as the concept of *time-dependend learning*. Results show that intelligent agent strategies can increase self-consumption within the local market from 43% to 54% with the right combination of the Erev-Roth algorithm and the extensions. The research also confirms that LEMs can keep profits within the local community and decrease the total cost of energy for market participants. Interestingly, consumers benefit more from the participation in the LEM than prosumers, since the latter already profit from direct self-consumption.

3.1.4 Impact of Flexibility

The aim of LEMs is to pave the way for local balances of energy generation and demand close to real time. In the long-term, community-based or hybrid LEMs could reduce the need for expensive transmission lines, lower curtailment of renewables, and cut down on redispatch costs. However, higher levels of self-sufficiency require increased flexibility. The steadily decreasing prices for lithium-ion batteries have sparked the deployment of storage technologies in prosumer households [155]. For instance, according to [12], today 50% of residential photovoltaic systems in Germany are sold along with battery storage devices. This fact opens the door to residential Demand Response (DR)[8] which can help reduce power peaks and eventually lower the cost of electricity supply. [157]

[8]Demand response is part of the broader family of demand side management practices which aim at improving energy systems from the consumption side. DR describes changes in consumption behaviour in reaction to dynamic price signals [156].

It is important to note that in a LEM which is connected to the public grid, without flexibility, bid and ask prices would mirror the boundary prices of the public grid [25]. For instance, in a system where one can buy electricity from the grid at an exemplary price of 25 Euro-cent per kWh and sell electricity back to the grid at an exemplary feed-in tariff of 5 Euro-cent per kWh, bid and ask prices would be 25 and 5 Euro-cent per kWh, respectively. This makes flexibility a key enabler for practical LEMs, because—only with flexibility—consumers and producers can react to price signals indicating scarcity or overproduction of energy.

In [157] the authors show that residential demand response can increase sufficiency in the local market by 16% while reducing the electricity price by up to 32%. The residual aggregated peak demand of market participants could even be reduced by 40%. As a consequence, residential demand response in combination with local energy trading can lead to more self-sufficient local communities.

Lüth et al. [20] compare two P2P market designs—one with decentralised and the other with centralised battery storage. They show that the decentralised approach can lead to cost savings for market participants of up to 17%. The centralised approach reduces the energy bill for end-users by 9%. In comparison to a reference scenario without P2P trade and battery storage, both designs are economically viable, reducing the overall costs for market participants by 31% and 24%, respectively.

In [14], Zepter et al. investigate how prosumer communities with residential storage can benefit from active participation in existing day-ahead and intraday markets (see Section 3.1.1 under *Hybrid market*). A Smart elecTricity Exchange Platform (STEP) serves as interface between wholesale markets and local communities. The authors conclude that the combination of P2P trading and residential battery storage can lead to savings of up to 60% for market participants and simultaneously increase the community's level of self-sufficiency. P2P trading and demand-side flexibility account for cost reductions of 34% and 20%, respectively.

In his dissertation [22], Dauer has developed an auction mechanism where DSOs can procure demand-side flexibility from consumers in a smart grid environment. Following a community-based approach, the flexibility is aggregated by a LMO. Simulations show that flexibility auctions have the potential to reduce the DSO's balancing cost by up to 43%.

Olivella-Rosell et al. [24] propose a similar framework for smart grid-dominated scenarios. The authors design a community-based local flexibility market selling flexibility services to DSOs and/or other balance responsible parties. The flexibility

transactions are supervised by a LMO. The publication shows that such a flexibility market could reduce energy prices as well as the need for investments in grid infrastructure.

3.1.5 Deductions

Summarising this first section investigating the market layer of LEMs, it can be said that it is difficult to extrapolate from the analysed studies to a perfect market design. The number of studies is still limited and the environments in which they have been conducted are very different. However, the following deductions can be drawn:

1. **Community-Based Market:** The community-based approach being the most 'realistic' approach with regard to the German regulatory environment of 2019 (see Section 3.1.1), the Decentralised Application (DApp) design should focus on community-based LEMs while allowing for a future evolution into a hybrid design.

2. **Double Auction:** Auction designs where both consumers and producers of energy can place orders are the predominant design [19] and have proven efficient in traditional energy markets [158]. No consensus exists whether these auctions should be designed as discrete time or continuous auctions.

3. **Sealed Bidding:** Strategy-proof auction mechanisms like the VCG auction rely on *sealed bidding* (see Section 3.1.2 under *Auction-Based Market Designs*). The ability to keep placed orders hidden from other market participants is of particular importance in consumer-centric markets [159], e. g. small market participants should not be outperformed by bigger market participants with access to more sophisticated analytic tools.

4. **Uniform Pricing:** The analysed literature is undecided whether discriminatory or uniform pricing is the better fit for LEMs. Therefore, it is important to consider established markets. Kahn et al. [160] and Cramton et al. [161] have analysed the Californian electricity market and come to the conclusion that uniform pricing results in a lower overall cost of electricity than discriminatory pricing. Similarly, the German day-ahead EPEX spot market is designed as a discrete time market with a uniform pricing mechanism [158]. This thesis does not intend to settle this debate and ideally, a DApp enabling energy trading should be modular with regard to the implemented pricing mechanism.

3.2 Blockchain-Based Information & Communication Layer of Local Electricity Markets

The previous section shed light on the market layer of LEMs which defines the rules that govern the interactions and the exchange of information between market participants. While there is no consensus on which market design is best suited to promote the development of LEMs [31], all agree that efficient and secure protocols are needed on the ICT level. Many publications— [24, 25, 27–29, 31], to name a few—propose the blockchain technology as automated settlement system to keep track of electricity transactions within the local market.

A blockchain system is characterised by its decentralised data storage and computation (see Section 2.2), making it a predestined technology for coordinating Distributed Energy Resourcess (DERs). In a LEM, as stated before, the interplay of these DERs with local consumers can be understood as a multi-agent system [23]. This means each agent follows their own optimisation strategy [151]. However, for the success of a LEM, all agents have to follow specific predefined rules in order to guarantee maximum welfare for the local community. The smart contract logic in a decentralised blockchain application can enforce these rules in a fully transparent manner and automate energy trading based on them. This way, trust between the market participants is created without relying on a centralised intermediary. [41]

Furthermore, a blockchain system allows not only for transfer of information but for transfer of value [78]. In the case of energy trading, the values transferred over the decentralised application are *energy* and *money* [50]. In the blockchain jargon, it is often referred to as *tokenisation* of assets [75]. A decentralised system further has no single point of failure which makes it more resilient to malicious attacks compared to a centralised system. However, decentralisation alone is not enough to ensure increased security. In a blockchain system, the consensus mechanism also guarantees that every change to the system state is valid. [48]

Most futuristic is the idea of connecting local blockchain systems to other blockchains, thus creating a token economy [78]. Many actors work on the interoperability of blockchains allowing for cross-chain transactions[9]. For instance, the *Energy Web Foundation* is currently developing a decentralised application called *EW Origin* [105] to issue and trade *certificates of origin*[10] (see Subsection 2.2.5).

[9] Refer to *Polkadot* [114] and *Cøsmos* [115] as mentioned in Subsection 2.2.6.

[10] These certificates act as proof that a certain amount of energy has been produced by renewable energy sources [162].

The connection of a blockchain-based energy community application with the *EW Origin* application could increase the attractiveness of local electricity trade and potentially open the door to future carbon markets. Other examples include the combination with blockchain-based smart home systems [163], combining Electrical Vehicle (EV) charging [164] with a local market platform, or providing flexibility to DSOs or other interested parties [165].

3.2.1 State of the Art

Research focussing on the ICT level of decentralised energy trading is summarised in Table 3.3. The most relevant publications are presented in greater detail below.

Mengelkamp et al. [42] propose a concept, market design, and simulation of a LEM. Their simulation scenario encompasses 100 residential households which trade locally produced electricity on a decentralised market platform. The underlying blockchain framework is a private Ethereum network. The LEM is connected to the distribution grid and market participants can buy electricity from or sell electricity to an energy provider if needed. Consequently, grid sell prices and grid buy prices constitute the lower and upper price limit, respectively[11]. The market design of choice is a discrete time double auction with uniform pricing, implemented via a closed order book. If a market participant does not fulfil their contract (e. g. due to a forecasting mistake), the order gets cleared over the grid at an unfavourable price. This way, market participants are incentivised to submit accurate orders and it is ensured that all agreed transactions are met. Smart meters measure the demand and production and software agents forecast future unsatisfied demand and excess production for each market participant. These values are stored as information on the blockchain together with each market participant's financial balance. Orders are then placed based on the estimated demand or production in the next period and every agent's utility function. Bids are sent together with the monetary amount in order to guarantee the settlement of every transaction. The blockchain thus acts as an escrow agent for the local market. The authors argue that the technological evaluation of the blockchain as main ICT still needs to be conducted in terms of: computational resources; energy usage; transaction costs; block speeds; and scalability. They further suggest to use a blockchain framework which does not rely on the consensus mechanism Proof-of-Work (PoW), as does Ethereum. Electricity

[11] In this case, the assumed boundary prices are a grid sell price of 12.31 Euro-cent per kWh and a grid buy price of 28.69 Euro-cent per kWh [42].

Table 3.3: *Overview of publications regarding the information and communication layer of blockchain-based local electricity markets (in chronological order)*

Publication	**Title**
Mihalov et al. [38]	NRGcoin: Virtual currency for trading of renewable energy in smart grids
Mihalov et al. [150]	NRG-X-Change - A Novel Mechanism for Trading of Renewable Energy in Smart Grids
Murkin et al. [39]	Enabling peer-to-peer electricity trading
Mustafa et al. [40]	A local electricity trading market: Security analysis
Wang et al. [166]	A Novel Electricity Transaction Mode of Microgrids Based on Blockchain and Continuous Double Auction
Hahn [41]	Smart contract-based campus demonstration of decentralized transactive energy auctions
Sikorski et al. [167]	Blockchain technology in the chemical industry: Machine-to-machine electricity market
Münsing et al. [32]	Blockchains for decentralized optimization of energy resources in microgrid networks
Pop et al. [165]	Blockchain Based Decentralized Management of Demand Response Programs in Smart Energy Grids
Schlund et al. [168]	ETHome: Open-source Blockchain Based Energy Community Controller
Yu [44]	Design, Implementation, and Evaluation of a Blockchain-enabled Multi-Energy Transaction System for District Energy Systems
Mengelkamp et al. [42]	A blockchain-based smart grid: towards sustainable local energy markets
Biggelaar [169]	Towards Decentralized Grids – EnergyBazaar: decentralized free-market energy-trade within an isolated community micro-grid
Li et al. [47]	Consortium Blockchain for Secure Energy Trading in Industrial Internet of Things
Aitzhan and Svetinovic [170]	Security and Privacy in Decentralized Energy Trading Through Multi-Signatures, Blockchain and Anonymous Messaging Streams
Ferreira and Martins [171]	Building a Community of Users for Open Market Energy
Reekers et al. [172]	Analyse und Optimierung eines Demonstrators für elektrisches Netzmanagement und Energiehandel auf Blockchains
Seidl [49]	Implementation of Blockchain-Based and Grid-Friendly Local Energy Markets
Myung and Lee [43]	Ethereum smart contract-based automated power trading algorithm in a microgrid environment
Thut [173]	Development and evaluation of a DLT-based marketplace for sector coupling in quarters
Kirpes et al. [46]	Design of a Microgrid Local Energy Market on a Blockchain-Based Information System

taxes and fees (e. g. grid fees and EEG-surcharge[12]) are assumed to be fully scalable—essentially a constant percentage of the paid price—which currently is not consistent with German regulation. However, one can hope that regulation will evolve in favour of decentralised trade in the short-to-medium-term, as proposed by the European Commission [137]. Unfortunately, the publication does not include the smart contract code or the agents' bidding strategies.

Seidl [49] introduces a blockchain-based platform for grid-friendly P2P electricity trading in LEMs. The local market is connected to the public grid which subsequently acts as price limit for market offers, as seen in the previous example [42]. Demand and supply are matched through a double auction with discrete market closing times, limited to one bid per market participant and auction period. The pricing mechanism is a mixture between uniform pricing and price functions (see Section 3.1.2 under *Non-Auction-Based Market Designs*). The price functions ought to incentivise market participants to act within the limits of the physical grid and incorporate information on: the congestion-level at the transformer between the local grid and the public grid; voltage-levels within a circuit; and the frequency at every node of the system. The decentralised application is written in Solidity and deployed on a private, permissioned Ethereum blockchain using Proof-of-Authority (PoA)[13] as consensus mechanism. It is assumed that all market participants are equipped with smart meters acting as blockchain nodes, and that the LEM is not affected by any regulation, as trading happens *"behind the meter"* [49]. A custom token is introduced which is directly linked to the Euro in a ratio of 1000 tokens to 1 Euro. In order to react as quickly as possible to signals from the physical grid, trading periods have been reduced to 30 seconds. The author showed that smaller time frames are difficult to implement in this setting. All the code was made available upon request.

Hahn et al. [41] introduce a smart contract implementing a decentralised energy marketplace on a private Ethereum blockchain. The market is designed as a Vickrey single auction, as described in Section 3.1.2 under *Auction-Based Market Designs*. The decentralised application is realised on the campus of the Washington State University interacting with a 72 kW Photovoltaics (PV) array. This PV system thus offers its produced electricity via the marketplace and the different buildings

[12]The EEG-surcharge is added to the price of electricity per kWh consumed in Germany based on the *Erneuerbare Energien Gesetz*. It was introduced to promote development of renewable energy sources [15].

[13]PoA is becoming more and more common in permissioned blockchain systems and can be characterised by the existence of trusted nodes acting as authorities. Generally, each round a leader is elected who then publishes the new block. The identities of the authority nodes are public, meaning that they vouch with their reputation. The *EWF* blockchain also runs on a PoA consensus [105]. [174].

on campus are simulated to bid based on their consumption. All market participants are assumed to be equipped with a smart meter which also operates as blockchain node. The key entities interacting on this marketplace are: the seller agent, initiating a new auction when energy is available; the bidder agent, who can procure energy from the public grid or bid on the local market; the meter agent, reporting the consumed energy during agreed time period which authorises the payment; and the smart contract, which accepts and stores auction data from the previous agents, and executes the auction and the payment function. However, the penalties which arise when actual consumption or production differs from the contract are not described in the paper. The authors also published their smart contract code which may be interesting to any reader familiar with Solidity.

Myung and Lee [43] propose a decentralised power trading algorithm for a microgrid environment. Just as the previous two designs, the decentralised application is deployed on a private ethereum blockchain with smart contract code written in Solidity. The market design could be described as continuous English auction. Producing entities create a new auction whenever they have excess energy. Consuming entities can then bid on it, with the highest bid winning the auction (see Section 3.1.2 under *Auction-Based Market Designs*). The roles of the network participants can thus be described as follows: selling prosumer or producer, initialising the auction contract, selecting the winning bid, and providing the electricity; buying prosumer or consumer, bidding on available auctions; and a platform provider, running full Ethereum nodes[14] and making the marketplace available. Platform users are equipped with smart meters that act as light Ethereum nodes. No token is being created, instead, Ethereum's native cryptocurrency *Ether* is used for value transfer. The smart contract code can be found in the appendix of the paper.

Biggelaar [169] presents a decentralised trading platform called *EnergyBazaar* for an isolated community microgrid. On top of facilitating P2P trading of electricity in a local context, the algorithm focusses on optimal energy distribution taking into account grid stability constraints. Game theory is used to solve the economic dispatch problem[15]. Each agent is equipped with an energy storage system, acting as a buffer between residential consumption and the microgrid. The author states that there is a trade-off between economic gains and security of supply (a microgrid in island-mode needs to store energy for times of scarcity). The platform is implemented on

[14]While *full nodes* keep all transaction data (blocks), participate in the consensus (mining), and run all decentralised applications, a *light node* only runs the applications and pays transaction fees [43].

[15]Individual agents seek to maximise their social welfare while the community's collective task is to stabilise the grid [169].

Ethermint, a combination of *Ethereum* and *Tendermint*—essentially a permissioned version of Ethereum with a less computationally intensive consensus mechanism and still in an early development stage [175]. Smart contract code is written in Solidity. A disadvantage of this solution, the author argues, is that all market participants' information is completely public. Unlike previous system designs, buyers and sellers do not bid or place offers. Rather, they share their respective optimisation context with the network and the smart contract logic finds an optimum for the community. All code is publicly available on the author's GitHub page.

Yu [44] proposes a close to real-time multi-energy dispatch and settling system for heat and electricity in a district context. The dispatching logic is implemented in the smart contract code written in Solidity. Energy transactions and financial settlements are recorded on a private Ethereum blockchain using custom tokens. The local community is connected to the superordinate grid for balancing purposes. Similar to [169], the system is designed so that community welfare is maximised while taking into account the operational limits. It is assumed that households favour locally produced renewable energy as long as the price is reasonable. Entities in this system design are: houses and buildings, both residential and commercial consumers; PV arrays, as the only electricity producers in the system; one battery, shared amongst all members of the community; the public grid, acting as backup in case of energy surplus or deficiency; water tanks, providing flexibility for heating; and a heat pump, a thermal generator working as interface between the electricity and the heating network. Unlike previous designs, in this system, the consumers submit their predicted consumption for the next trading period and the producing entities then propose a price at which they sell their energy. The lowest offers are then selected under consideration of the community's total consumption. All code is available on GitHub.

Thut [173] presents a decentralised marketplace for sector coupling in quarters, considering both heat and electricity. The market is implemented as a discrete time double auction with a sub-market for each energy type and follows the design discussed in [42]. Similarly to [49], the decentralised application runs on a private Ethereum blockchain with PoA as consensus mechanism of choice. The author concludes that the high number of transactions make a theoretical migration to public Ethereum technically and economically infeasible. A private system, however, is dependent on intermediaries if it wants to interact with the world outside the chain. Some parts of the code are published in the thesis.

Mustafa et al. [40] propose a community-based LEM model implementing a double auction with uniform pricing. Financial transactions are settled via each market participant's supplier. Simulations show significant financial gains for both consumers and prosumers participating in the market, as well as ecological benefits. The authors further carried out a security analysis of their proposed design. On this back, they formulated 7 *secuirty and privacy requirements* which are intended to be used as a guide for future protocol design. Being of major importance for this work, these requirements are explained in greater detail in Section 5.1.

Kirpes et al. [46] propose an architecture for blockchain-based LEMs with a special focus on interoperability. The architecture design is based on a system analysis employing the Smart Grid Architecture Model (SGAM), a standardised reference model for smart grid systems [65]. The authors formulate 20 requirements for a blockchain-based LEM, on an organisational, informational, technical, and a blockchain-specific level. The five requirements regarding the blockchain-system are most important for this thesis and will be elaborated upon in Section 5.1: *Design Objectives & Requirements*. They further call for standardised data structures to foster future development and ensure interoperability and recommend the use of a private, permissioned blockchain framework in order to enforce user access rights.

3.2.2 Deductions

The key takeaway of this literature review is that research investigating blockchain-based marketplaces for LEMs is still a young discipline (maximum 5 years). However, many publications present starting points for future work. The most important deductions from this analysis are listed below.

1. **Permissioned System:** All analysed implementations are deployed on a permissioned system in a private environment. This makes it possible to restrict access to the LEM platform and facilitates the definition of access rights—a key element of a secure decentralised energy trading marketplace.

2. **Consensus Mechanism:** Most implementations are built on top of *Ethereum* and written in Solidity. However, some designs [49, 169, 173] implement alternative consensus mechanisms(e. g. PoA) or argue that an alternative to PoW should be found [42, 46].

3. **Privacy:** Most analysed publications do not consider privacy in their proposed marketplace designs. Some authors argue more privacy-preserving protocols should be investigated [40, 46, 169].

4. **Requirements:** Both [40] and [46] formulate some sets of requirements with regard to the ICT layer of LEMs. These should be considered when designing a DApp for energy trading.

This chapter analysed the existing literature around LEMs from two perspectives: (1) from a *market* perspective (see Section 3.1); and (2) from an *ICT* perspective (Section 3.2). The findings from this literature review validate the problem definition presented in Section 1.2, i. e. that existing blockchain-based energy trading protocols are still immature and that alternative blockchain frameworks should be investigated. Following the Design Science Research (DSR) methodology discussed in Section 1.3, the next step is to derive from these findings specific objectives and requirements the final solution should satisfy; a list of 24 requirements is presented in Section 5.1 of Chapter 5. However, before the proposed DApp design can be discussed, a short introduction into the workings of Hyperledger Fabric is given in Chapter 4.

Chapter 4

Digression: Hyperledger Fabric

In order to comprehend the application design described in Chapter 5, it is crucial to understand the basic functioning of the underlying blockchain framework. This chapter aims at clarifying the key concepts, the architecture, and transaction flow of Hyperledger Fabric version 1.4. Except where otherwise indicated, this chapter is based on the official documentation [176]. This chapter is not exhaustive, rather it is intended to give a good overview over the permissioned blockchain framework. The interested reader may refer to [176] for more detailed information.

4.1 Key Functionalities

As introduced in Subsection 2.2.4, Hyperledger Fabric is a modular blockchain framework maintained and developed under the umbrella of the Linux Foundation. It is best described as a platform for developing and deploying applications which uses blockchain technology as means to create an immutable history of records between a closed number of actors—a so-called consortium. Its key functionalities are:

Identity management Hyperledger Fabric is a permissioned blockchain system (see Subsection 2.2.3 under *Public, Private & Consortium Blockchains*). Access to these closed networks is managed by a module called the Membership Service Provider (MSP), administering user IDs and authenticating all network participants. This concept allows for permission control, e. g. that certain transactions can only be invoked by a certain (type of) participant.

Privacy and confidentiality A permissioned blockchain network can consist of a consortium of players with competing business interests. This may require to

keep certain information private. This level of confidentiality can be achieved in two ways with Hyperledger Fabric: (1) by means of *private channels*, essentially a completely separated blockchain visible to only a subset of network participants; (2) by means of *private transactions*, the content of which is only shared among a subset of predefined actors while the hash is still stored as a proof on the main-chain and visible to all actors in the channel.

Efficiency Unlike Ethereum (and most other common blockchain frameworks), nodes in Hyperledger Fabric can have different roles. Essentially, the execution of transactions is separated from ordering (consensus) and commitment (to the ledger). This design provides concurrency and parallelism to the network, increasing process efficiency and scalability (see Subsection 4.2.1).

Chaincode functionality Business logic in a channel[1] is encoded into chaincode and invoked by transactions. Chaincode can, for example, define parameters for a change of asset ownership, making sure that all transactions are subject to the same rules. Furthermore, there is a chaincode on the system level which defines the rules for creating new channels or the requirements for validating transactions.

Modularity The modular architecture gives creative and functional leeway to network designers. Different ways of handling identity management, consensus, and encryption can be plugged into any Hyperledger Fabric network. It is indented to insure interoperability with legacy Information and Communication Technology (ICT) infrastructure and operability in different regulatory environments.

4.2 Concepts in Detail

This section discusses the precise nature of some important concepts within the Hyperledger Fabric ecosystem. The following questions are addressed: (1) what are the different entities within a permissioned blockchain network and how is consensus achieved between these entities? (2) how is data stored and how can data be kept private between members of a same consortium? and (3) how is identity and membership managed within a Hyperledger Fabric network? Finally, these concepts are put

[1] In Hyperledger Fabric, there may be multiple channels within a single blockchain network. Each channel represents a unique blockchain with one or more chaincodes installed on it. As explained above, channels can also exist only between a subset of network participants.

together and used to explain the components making up an exemplary blockchain network.

4.2.1 Roles & Consensus

The following roles exist in Hyperledger Fabric:

Clients A *client* is an application that proposes a transaction on the blockchain network on behalf of a person.

Peers Each *peer* holds a copy of the ledger, thereby maintaining the state of the network. Hyperledger Fabric comes with two types of *peers*:

— *Endorsing peers* have chaincode installed, simulate transactions, and endorse them if valid.
— *Committing peers* verify the endorsements and validate transaction results before committing a new block of transactions to the distributed ledger.

However, every *endorsing peer* is also a *committing peer* and commits new blocks to the blockchain.

Ordering service The *ordering service* receives endorsed transactions, orders them into blocks, and distributes the blocks to the *committing peers*.

These roles are also reflected in the way the network reaches **consensus**. In the context of a distributed ledger system, *consensus* describes the process of reaching agreement on the next set of transactions, and the order in which they are added to the ledger [177]. In Hyperledger Fabric, consensus is made up of these three phases:

1. **Transaction proposal and endorsement:** A *client application* sends a transaction proposal to a set of peers. These peers invoke the chaincode and simulate the transaction and either endorse the results or not. The results are not applied to the ledger yet; instead, the *endorsement* is sent back to the *client application*. The *endorsement policy* states what a 'valid' endorsement must look like, i. e. the rules and the number of consortium members which must approve a transaction before it is passed to the *ordering service*.

2. **Ordering and packaging transactions into blocks:** With the valid endorsement(s) from phase 1, the *client application* now sends the endorsed transaction to the *ordering service*. The *ordering service* nodes receive transactions

from a variety of *client applications* concurrently and work together to sort these transactions into a deterministic sequence and aggregate them into a block[2].

3. **Validation and commit:** First, the new block is sent from the *ordering service* to all *committing peers*[3]. Afterwards, every peer validates each transaction in the block, i. e. making sure the endorsement policy has been respected or checking that the transaction has not become invalid due to another recently added transaction[4]. Both valid and invalid transactions are added to the blockchain (or transaction log) and tagged accordingly; however, only valid transactions alter the peer's copy of the world state.

The cluster of *ordering nodes* forming the *ordering service* can currently be run on the back of three consensus protocols: (1) Solo, a centralised version for testing purposes only; (2) Kafka, a crash fault tolerant implementation available since version 1.0 and based on a 'leader and follower' model, where a leader node is elected and the follower nodes replicate its decision; and (3) Raft, similar to (2) but easier to manage in a setup with multiple organisations and only available since version 1.4.1.

It is important to note that the *ordering service* can currently only provide *Crash Fault Tolerance (CFT)* and no *Byzantine Fault Tolerance (BFT)*. These terms are best explained by looking at the threats a consensus protocol must be resilient against. For instance, machines and devices can fail, processes can crash, network connections can be interrupted. Consensus algorithms which can deal with these threats provide CFT. Another threat arises from the distribution of network components over different organisations with potentially conflicting goals: malicious agency. Consensus mechanisms which can deal with systems which may have malicious actors provide BFT. [178]

The developers behind Hyperledger Fabric are currently working on a BFT protocol for the *ordering service*, but no official release date has been announced so far [179]. However, it should be pointed out that this only concerns the *ordering service*.

[2]While in other blockchain frameworks—Ethereum, for instance—a transaction may be put into multiple blocks which then compete to form a chain, in Hyperledger Fabric, the ordering service guarantees transaction finality. This means that there can not be any ledger forks, i. e. once a transaction is validated it can never be reverted or dropped.

[3]It is worth noting that new blocks can also be cascaded to peers that are not directly connected to the orderer via a *gossip protocol* (see Subsection 4.2.2).

[4]This may happen when another transaction was still uncommitted—still with the ordering service, for instance—at the time the new transaction was endorsed. In this case, the new transaction is tagged as 'invalid' and will not change the world state of the ledger. However, it is still added to the blockchain to ensure transparency.

Peers already reject malicious attacks at *endorsement* level and further make sure all transactions are valid before *committing* them to their copy of the ledger.

4.2.2 Ledgers & Private Data

In Hyperledger Fabric, the ledger is made up of two distinct, though related, parts: (1) the *world state*, a traditional database which maintains a copy of the current values of a set of ledger states; and (2) a *blockchain*, essentially a log of transactions that records all changes that have resulted in the current world state. Data within the *world state database* is generally expressed as key/value pairs.

As mentioned before, Hyperledger Fabric also supports *private transactions* which enable a predefined subset of channel members to share confidential data among each other. Unlike data resulting from *public* transactions, this 'private data' is not stored in the *world state database*. A peer which is authorised to read and write private data maintains a separate *private state database*. As illustrated in Figure 4.1, data within this *private* database is stored in its unencrypted form. Both a hash of the 'key' and a hash of the 'value' are stored in the *world state database*. The set of authorised consortium members is defined in a so-called *private data collection*. There can be a multitude of these collections, each governing the access to private data between different subsets of network members. If a member is part of multiple authorised subsets, a separate *private state database* is maintained for each collection.

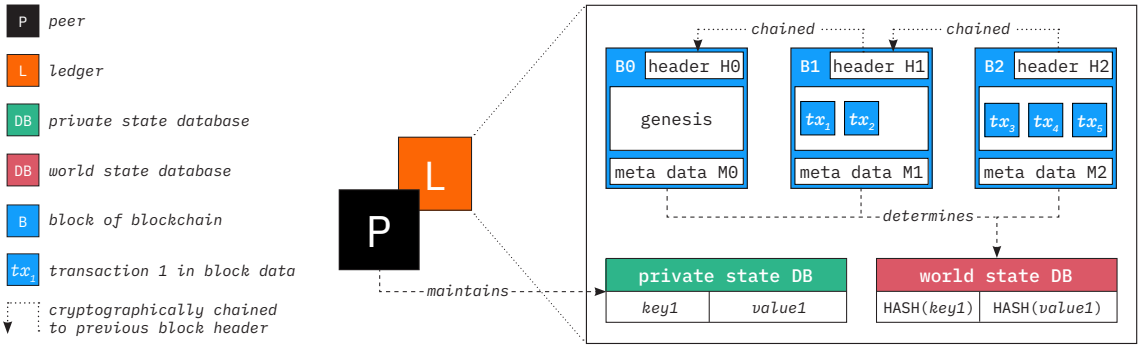


Figure 4.1: *Breaking down a Hyperledger Fabric ledger into its components: blockchain, world state database, and a exemplary private state database (own illustration based on [180])*

The flow of a transaction containing private data differs from the steps explained in Subsection 4.2.1 since this data cannot be distributed to all peers via the *ordering service* without compromising confidentiality. When a *client application* proposes a *private transaction*, it sends the private data along in a transient field of the proposal. The *endorsing peer* thus simulates the transaction and stores the results in a transient

data store—essentially a temporary storage only local to the peer. Peer-to-Peer (P2P) dissemination is carried out directly over the *gossip protocol* which is also responsible for broadcasting messages and data inside a same blockchain network[5]. Afterwards, the proposal response is sent back to the *client application* along with any public data and the hashes of both the private data 'keys' and 'values'. No private data is sent back to the *client application*. The endorsed transaction is then sent to the *ordering service*, packed into a block, and distributed to all peers in the channel. Thereafter, every authorised peer checks their local transient data store and validates the private data against the hashes in the new block. If valid, the private data is stored in the peer's corresponding *private state database*. Non-authorised peers simply commit the hashes of the private data to their copy of the *world state database*. The complete transaction flow for a private 'bid' transaction is illustrated in Figure 5.8 on page 74 as a sequence diagram.

4.2.3 Identity & Membership

All entities within a Hyperledger Fabric network (e. g. client applications, peers, orderers, network administrators) hold a digital *identity*. Each identity is encapsulated in a X.509 certificate, a digital certificate using the X.509 Public Key Infrastructure (PKI) standard in order to verify that a public key belongs to the entity contained within the certificate [181]. These certificates are issued by a Certificate Authority (CA), trusted authorities which are an integral part of internet security. Identities determine which entities inside the blockchain network have permissions over resources or access to information.

In Hyperledger Fabric, the MSPs are the *trusted* authorities which make an identity *verifiable*. These components define the rules which govern the validity of identities. Each organisation in the network has its own MSP that maps which identities belong to its organisation.

4.2.4 Blockchain Network

With this basic understanding of Hyperledger Fabric, it is worth taking a detailed look at the components which make up a blockchain network. An exemplary network

[5]This gossip-based dissemination protocol is further used for: (1) peer discovery, constantly identifying available peers and the peers which have gone offline; (2) distributing ledger data to all peers inside the network; and (3) bringing peers that were offline back into sync or new peers up to speed.

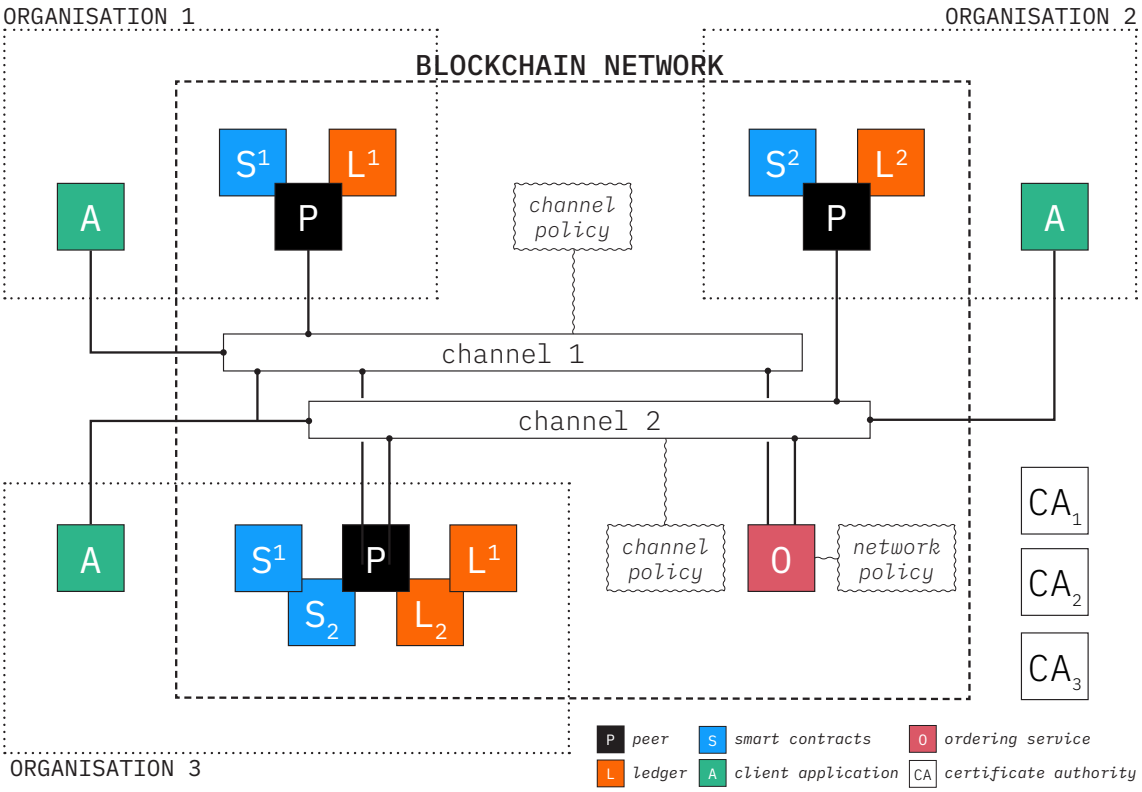


Figure 4.2: *An exemplary Hyperledger Fabric blockchain network with two channels and three organisations (own illustration based on [182])*

is illustrated in Figure 4.2. Three organisations are part of the blockchain network which comprises two different channels. ORG1 is part of the channel 1 consortium, ORG2 is part of the channel 2 consortium, and ORG3 is part of both consortia. The *ordering service [O]* is connected to both channels and is governed by the *network policy*, sometimes also referred to as *network configuration*. In this scenario it is made up of a single node belonging to an external organisation which does not intend to do business in the network and is subsequently not part of any consortium. In a 'real life' application it would probably be made up of a cluster of nodes, each belonging to a different organisation. The *ordering service* also has the power to add new organisations to the network. This power is restricted by the *network policy* which defines the rules for adding new organisations to the network or creating additional channels. Each channel is governed by the *channel policy*, sometimes also referred to as *channel configuration*. It defines the rules and the permissions different organisations have in the channel, i. e. the *endorsement policy* or which organisations have the right to add new members to the channel consortium. Even though the *ordering service* can create new channels and defines which organisations are part of the consortium in the first place, it can have no rights within the *channel policy*.

The peer of ORG1 has the *smart contract chaincode [S]* of channel 1 installed and maintains a copy of the *ledger [L]*, storing the blockchain and the world state database for this particular channel (see Figure 5.7). The *client application [A]* belonging to ORG1 is also connected to the channel—the means by which it can send transaction proposals to the peers of the consortium. The same applies to ORG2 which is only part of the second consortium and subsequently only holds the chaincode and a copy of the ledger of channel 2. ORG3, being part of both consortia, has two different *chaincodes* installed and maintains two separate ledgers, maintaining the world state of each channel, respectively. Consequently, the *client application* which is not part of the blockchain network is also connected to both channels and can communicate with all peers in the network.

In this scenario, each organisation has their own *certificate authority [CA]* which is also outside of the network. CA_1, for instance, is responsible for delivering trusted certificates to all entities of ORG1. All messages and transactions which are sent within the network are signed and encrypted with these certificates. This way, it is always possible to know the entity from which a message was sent and receiving entities can be sure that the message has not been tampered with while 'in transit' (see Subsection 4.2.3).

A last feature worth highlighting is the possibility for so-called *cross-channel transactions*. In this scenario, the chaincode of channel 1 could, for example, include functions which invoke *query* transactions of the channel 2 chaincode. It should be noted that *cross-channel transactions* could only be endorsed by the peer belonging to ORG3, as it is the only peer with both chaincodes installed. However, calling transactions of a different chaincode which *alter* the state of the other channel are not supported yet [183].

A non-exhaustive list of relevant concepts and terminologies within the Hyperledger Fabric ecosystem can be found in Appendix A.

Chapter 5

Proposed Design

As mentioned in Subsection 2.2.2, applications that run on a blockchain network are commonly referred to as Decentralised Applications (DApps). Unlike traditional applications where the backend code runs on centralised servers, the backend code of DApps runs on a P2P network and consists of one or a set of inter-linked smart contracts (see Section 2.2). In addition, a DApp can have a user interface and frontend code, just like a regular application, interacting with the backend. A decentralised energy trading platform for Local Electricity Markets (LEMs) is a DApp where market participants can buy and sell electricity under certain rules defined by the smart contact code. In the context of Hyperledger Fabric, this smart contact code is called *chaincode* and defines the business logic within the permissioned blockchain network.

As stated in Subsection 3.1.5, this thesis focusses on a community-based LEM since this market design has the lowest barriers in today's regulatory environment. At the end of this chapter, an *architectural model* (see Section 5.5) as well as a *chaincode model* (see Section 5.6) for a decentralised energy trading platform based on Hyperledger Fabric will be proposed. The sections before describe the procedure that was employed to design these models. The applied procedure is inspired by the *standards-based architecture modelling framework* formulated by [64]. This *architecture framework* is based on the Smart Grid Architecture Model (SGAM) developed by [65] and employs the same Domain Specific Language (DSL)[1]. It follows a three-stage approach, as illustrated in Figure 5.1. This chapter is organised, accordingly, as described in the next paragraph.

[1]DSL describes a (programming) language that *"offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain"* [184]. In this scenario, the SGAM proposes a DSL for the problem domain of *energy*. It is based on Unified Modelling Language (UML), a general-purpose modelling language which provides standards for visualising the design of a system [185]. An interactive version, called the

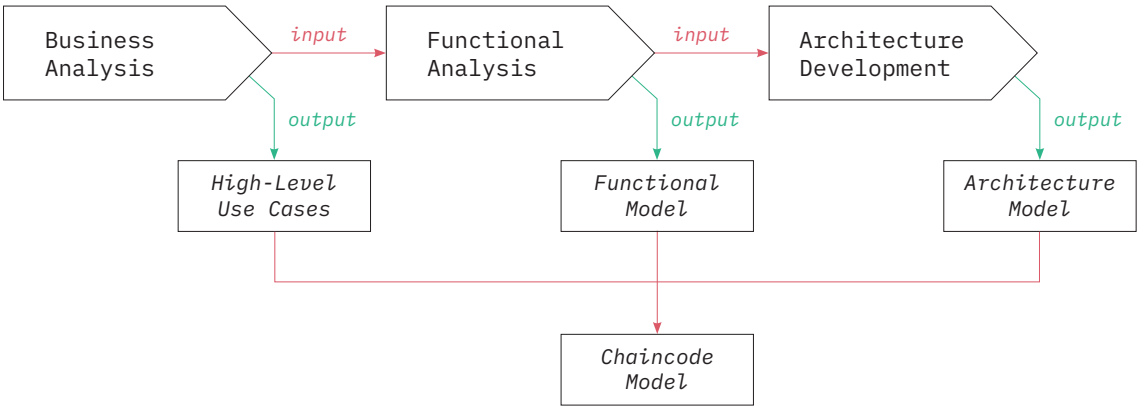


Figure 5.1: *Intended approach for using the standards-based architecture modelling framework, expanded to reflect a blockchain-based environment (own illustration inspired by [64])*

First and foremost, section 1 of this chapter formulates the requirements and design objectives the final solution should meet. The fulfilment of these requirements will be evaluated in Chapter 7. Section 2 formulates the assumptions that were made and on which the following sections and chapters are based. In accordance with the approach described in Figure 5.1, Section 5.3 consists of a *business analysis* describing the main *business case*, the *business actors* and their *business goals*, and derives some *High-Level Use Cases (HLUCs)* from it. In Section 5.4, the results from the previous section will be translated into a *functional model*. This model further decomposes the *business actors* into *logical actors* and identifies the *Information Objects (IOs)* that are shared amongst these actors. In the *architecture model* described in Section 5.5, the *logical actors* are mapped to *components*. It further describes how the IOs are exchanged between these *components* taking into account the topology of a Hyperledger Fabric blockchain network. Finally, as depicted in Figure 5.1, the three-stage approach proposed by [64] is extended in order to fit this blockchain-based use case; the outputs of the previous analyses are translated into a *chaincode model* which is described in Section 5.6. This language-agnostic model reflects the business logic of the energy trading use case. Together, the *architecture model* and the *chaincode model* form a *DApp design model* which constitutes the *Design Science Research (DSR) artefact* of this thesis.

SGAM-Toolbox 2.0, can be found here: https://sgam-toolbox.org/ (last accessed September 17, 2019).

5.1 Design Objectives & Requirements

The previous chapters elucidated the motivation behind this thesis and reviewed the relevant literature in order to answer the research question defined in Section 1.2. Following the DSR methodology presented in Section 1.3, the next step is to define requirements and objectives the proposed solution design should meet.

At first, some self-imposed *functional requirements* $\mathbf{R}_{func}$ and *design objectives* $\mathbf{O}$ are defined. The functional requirements *must* be satisfied and will be tested by means of the implementation described in Chapter 6. The design objectives are not part of the core use case of *community-based energy trading* but determine the direction in which the platform *should* be expandable. This section further includes two sets of requirements directly adopted from the literature analysed in Section 3.2: (1) *security and privacy requirements* for LEMs $\mathbf{R}_{sec}$ formulated by [40]; and (2) *blockchain-specific requirements* for LEMs $\mathbf{R}_{block}$ proposed by [46].

The following functional requirements and design objectives are set by the author of this thesis, based on the literature review presented in Chapter 3 and discussions with peers:

$\mathbf{R}_{func}$1: Multiple Order Placement Market participants must be able to place any number of buy/sell orders to reflect their individual buying and/or selling strategy.

$\mathbf{R}_{func}$2: Sealed Orders Market participants must be able to place sealed buy/sell orders in order to keep their strategy hidden from other market participants. Hence, only the Local Market Operator (LMO) and the market participant themselves must have visibility of the order details.

$\mathbf{R}_{func}$3: Smart Meter Readings Market participants must be able to report their actual consumption/production for each trading period.

$\mathbf{R}_{func}$4: Incentives for Accurate Bidding Market participants are incentivised to place accurate buy/sell orders and are penalised in case the data provided by their smart meters deviates from their initial order.

$\mathbf{R}_{func}$5: Internal and External Settlement Internal financial transactions (trades happening within the local market) and external financial transactions (trades with the public grid) are both settled via the DApp on the basis of a market participant's buy/sell orders and their actual consumption/production in a given trading period.

R_{func}6: Transparent Market Clearing If the demand and supply curves intersect, the market clearing transaction must be able to calculate a uniform Market Clearing Price (MCP) and inform all market participants if their buy/sell order was successful or not. The mechanism must be transparent in order to prevent manipulation.

R_{func}7: Adjustable Auction Times The duration of one auction period must be adjustable in order to give the LMO a degree of freedom with regard to local regulation or the ability to synchronise the local market with external market paces.

R_{func}8: Chronology of Events The DApp must enforce the chronology of events:
— **R_{func}8a: Bidding** Buy/sell orders can only be placed as long as the auction for a given trading period is still open.
— **R_{func}8b: Clearing** An auction can only be cleared if it is closed, i. e. once the auction time has elapsed.
— **R_{func}8c: Settlement** An auction can only be settled if all market participants have submitted their smart meter readings.

R_{func}9: Access Rights The DApp must be able to enforce access rights and prevent the following:
— **R_{func}9a: False Order** Market participants must not be able to place buy/sell orders on someone else's behalf.
— **R_{func}9b: False Reading** Market participants must not be able to send smart meter readings on someone else's behalf.
— **R_{func}9c: LMO Only** Market clearing and market settlement must be restricted to the LMO.

O1: Identity Management It must be possible to assign multiple identities to one actor in order to reflect more complex organisational structures of companies (e. g. the LMO has multiple employees with the right to perform a task in the network). Further, if a market participant looses their private key, or in case of a security breach, it should be possible to link a new identity to the given participant.

O2: Modular Market Design The platform should be modular with regard to the implemented market design. It should be possible to implement a con-

tinuous auction, switch to a discriminatory pricing mechanism, or change to open-outcry bidding[2].

O3: Platform Extensibility The platform should facilitate the integration of additional use cases and/or actors in order to adapt to future developments.

As described in Subsection 3.1.5, Mustafa et al. [40] analysed security problems and potential privacy threats for participants of a LEM in their paper *A Local Electricity Trading Market: Security Analysis* and derived a set of security and privacy requirements. These requirements are intended to be used as guidance for future LEM protocol designs and are listed below as direct citations:

R_{sec}1: Entity Authentication *"is important to ensure that entities can be assured of the identity of their communication partner. It is used to counter impersonation attacks. A liveness guarantee, i. e. the fact that the entity is active during authentication, is also an essential part of entity authentication."*

R_{sec}2: Message Authenticity *"guarantees an entity that the message it received has not been tampered with while in transit. It is used to detect message modifications. It can be achieved by means of a digital signature or a message authentication code. The advantage of digital signatures is that they are based on asymmetric keys, thus they also provide* ***non-repudiation****. Moreover, the integrity of the software running on SMs [smart meters] should be guarantied [sic] by using protected module architectures [...]."*

R_{sec}3: Authorisation *"is a process of determining if an entity has permissions to use/access resources, known as access control too. It can be coupled with entity authentication so the authorising party is aware of the identity of the entity requesting access. It is used to counter elevation of privilege attacks[3]."*

R_{sec}4: Confidentiality *"ensures that only the intended receiver(s) of a message can read the message. It is used to counter eaves-dropping attacks[4]. Confidentiality is achieved by encryption. Both symmetric and asymmetric encryption are possible, with symmetric encryption having the advantage of being*

[2] I. e. buy and sell orders are visible to all market participants (see Section 3.1.2 under *Auction-Based Market Designs*).

[3] Also known as *privilege escalation*, a *privilege attack* describes an network intrusion where an attacker gets elevated access to a system—and the data and applications on it thanks to programming errors or design flaws [186].

[4] Also known as *sniffing* or *snooping*, an *eavesdropping attack* describes an incursion where an unauthorised party tries to steal information which is transmitted by computers, smartphones, or other devices over a network [187].

less computationally heavy. Message authentication and confidentiality can be combined when using authenticated encryption."

R_{sec}5: User Privacy-preservation *"ensures that user privacy is protected as much as possible. To achieve this the 'principle of least privilege', i. e. only allow an entity to have access to data just sufficient for it to carry out its duties, should be applied. For example, the local market does not need to know the identities of users trading on the market, as long as it is assured that they are legitimate users. [...]"*

R_{sec}6: Non-repudiation *"is achieved when an entity cannot deny having sent a message when it did indeed send that message. This can only be achieved when messages are authenticated using a cryptographic key that only one entity has access to, i. e. using asymmetric cryptography."*

R_{sec}7: Availability *"is used to ensure that a system or a system resource is accessible upon demand by authorised entities. It is used to counter DoS [denial-of-service[5]] attacks. Availability can be achieved by using a combination of load and resource balancing, attack detection, message classification and filtering techniques."*

Kirpes et al. [46] formulate blockchain-specific requirements. They distinguish between *must*-meet and *should*-meet in order to indicate the importance of the requirements. They are listed below as direct citations:

R_{block}1: Smart contract capability *"Must implement a blockchain technology which is capable to deploy smart contracts and operate an efficient and secure market DApp."*

R_{block}2: Access rights *"Must implement a blockchain technology with suitable access rights (permissioned for most scenarios)."*

R_{block}3: Consensus mechanism *"Must implement a blockchain technology with a suitable consensus mechanism for this scenario."*

R_{block}4: Throughput *"Must implement a blockchain technology with transaction throughput suitable for the LEMs size (number of participants)."*

[5] A *denial-of-service attack* occurs when legitimate users of a service are prevented from accessing particular computer systems, services, devices, or other Information and Communication Technology (ICT) resources. Typically, these attacks are achieved by flooding servers, systems, or networks with a stream of requests [188].

R_{block}5: Token or Coin *"Should implement a blockchain-based token/coin which is used as a currency for trading energy."*

5.2 Assumptions

Due to the novelty of the use case, a number of assumptions are made with respect to the environment in which the proposed solution would be deployed and the interactions different actors would have with it:

A1 All consuming or producing parties are equipped with smart meters.

A2 All smart meters are connected to the blockchain network (either directly as blockchain node, or via a client application), hold their own private/public key pair to sign messages, and use protected module architectures to secure them from tampering (see Subsection 5.5.1).

A3 All market participants are connected to a power grid which itself has no transmission losses

A4 P2P-traded electricity is free of grid fees, taxes, and surcharges. It is assumed that trading happens 'behind the meter'.

A5 The local community is connected to the public grid which buys and sells electricity at fixed prices.

A6 A third party (e. g. a bank or the LMO) allows market participants to exchange money into locally used coins/tokens at a fixed rate and free of charge (see Section 5.5).

A7 All market participants have an internet connection and can connect to the blockchain via a user interface.

A8 All market participants are equipped with an automated trading agent, placing bids and ask in their regard, based on anticipated production or demand, the respective preferences, and risk appetite (see Section 5.5).

A9 After the dispatch of energy, all market participants automatically transmit their consumption or production data without delay.

5.3 Business Analysis of a Local Electricity Market

As illustrated in Figure 5.1, the *business analysis* is the first step of the modelling framework proposed by [64]. A *business case* is analysed with the objective to identify the *business actors*, the *business goals*, and the *high level uses cases* which describe a system's main functionality. As shown in Figure 5.2, the business case *energy trading in a local energy community* can be subdivided into four HLUCs: (1) trade within local community; (2) trade with the main grid; (3) trade with other communities; and (4) trade with a hyper-local community. These four HLUCs are represented by the differently coloured arrows in Figure 3.3 in Chapter 2.

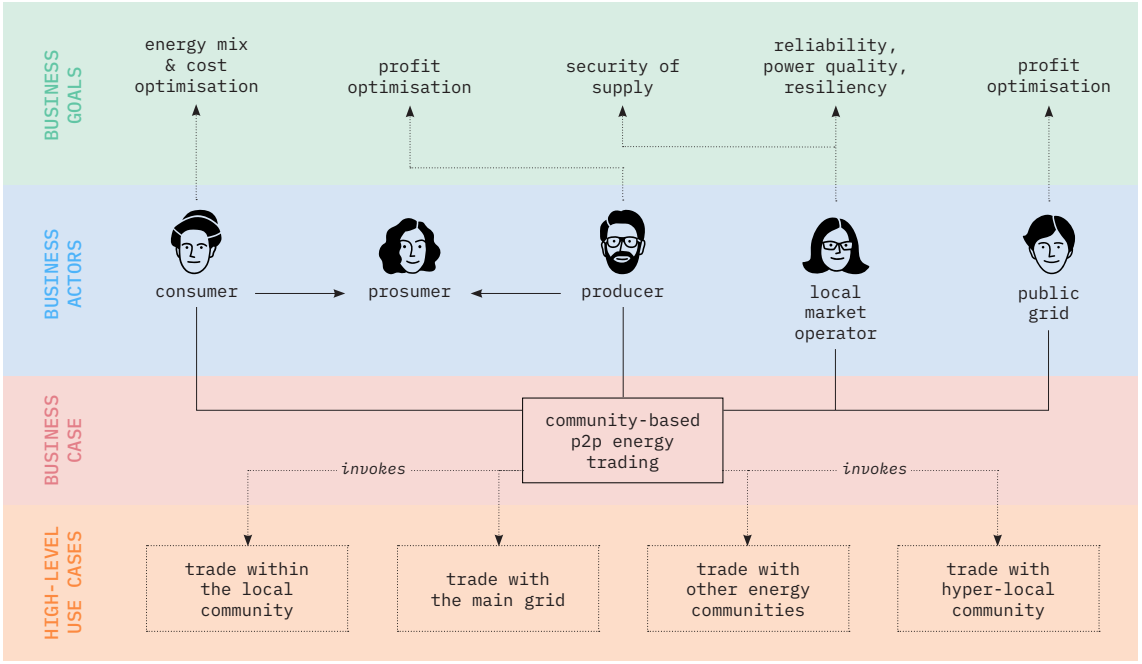


Figure 5.2: *Business use case model illustrating business actors, business goals and high-level use cases (own illustration based on [46])*

To keep a clear focus, this thesis concentrates on the HLUCs (1) and (2). Following the community-based approach, trade with the public grid is carried out by the LMO, sometimes referred to as community manager. Using the definitions from the *harmonised electricity market role model* [189], the stakeholders on the business layer are:

Consumer A party connected to the grid, consuming electricity. The *consumer*'s goal is to minimise the cost for their electricity while optimising the energy mix in line with their preferences (e. g. maximise the share of locally produced electricity).

Producer A party connected to the grid, producing electricity. The *producer*'s goal is to maximise the profits from selling locally produced electricity.

Prosumer A stakeholder combining the two previous roles.

Local Market Operator This party is no business actor to be found in [189]. The *LMO*'s goal is to maximise power quality, resiliency, and reliability of the local market while ensuring security of supply. P2P energy trading helps the LMO in achieving these goals. The LMO further bundles the energy demand or supply from the local market and acts as intermediator between the LEM and the rest of the power system. [46] argue that this role can be fulfilled by the *Microgrid Operator (MGO)* in case of a microgrid, or by any *Energy Service Company (ESCO)*. Alternatively, this role could be embodied by a local cooperative or energy collective.

Public grid A party that represents the connection to the 'traditional' energy system. In this use case this party is essentially a *trader*, buying electricity from, or selling electricity to the LMO. The *public grid*'s objective is to maximise profits from energy trading.

The community-based approach opens the door to use cases beyond energy trading, where the LMO sells services to other actors (e. g. to the *grid operator* or the *balance responsible party*). Whilst these use cases will not be considered in this solution design, it should allow for easy extension in future development.

5.4 Functional Model of a Local Electricity Market

The business actors described in Section 5.3 can have multiple functions and can be traced to (multiple) *logical* actors, as illustrated in Figure 5.3. The goal of the functional analysis is to further identify how these *logical actors* create *IOs* (see Figure 5.3) and how these objects flow between these actors (see the sequence diagram in Figure 5.4).

For instance, a **consumer** has as objectives to minimise the cost of electricity and has preferences regarding their energy mix. The consumer tries to minimise their cost by forecasting their future consumption (which might be partially flexible) and places a bid that reflects their *buying strategy*. However, forecasting and bidding is not carried out by the consumer in person, but by an automated *Consumer Trading*

Agent (CTA). In order to place correct *bids*, this trading agent needs to know the consumer's energy mix preferences and their risk appetite[6]. For this, the *"human" consumer* has to be able to communicate their preferences to the CTA (and be able to change them if needed), e. g. via a User Interface (UI)[7]. One can see that the business actor *consumer* can be split into two logical actors: (1) the *"human" consumer*; and (2) the *CTA*.

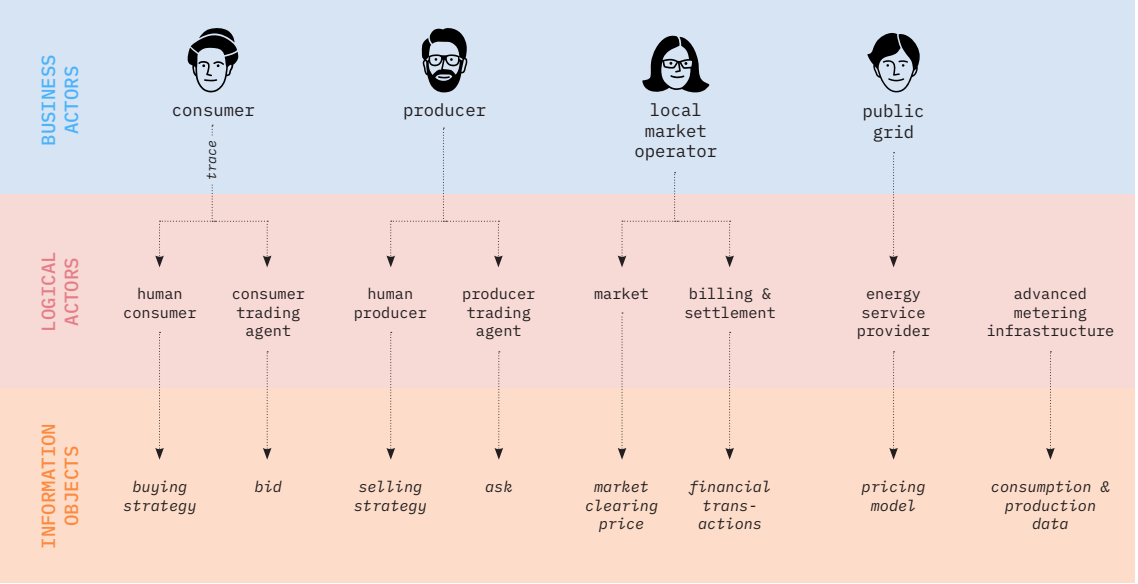


Figure 5.3: *Transformation of business actors into logical actors and derivative information objects (own illustration)*

The same applies to the business actor **producer** which can be divided into two logical actors: (1) the *"human" producer*; and (2) the *Producer Trading Agent (PTA)*. The PTA forecasts production and places *asks*—sellling orders—accordingly, potentially taking into account a producer's future consumption or the state of charge of an affiliated storage system, i. e. the *selling strategy*.

Additionally, every consumer and every producer is connected to a smart meter. Thus, a new logical actor, reading and reporting *consumption and production data* for all market participants, can be introduced: *Advanced Metering Infrastructure (AMI)*.

The business actor **LMO** can be traced to the two logical actors: (1) *market*, providing a platform for consumers and producers to match their demand and supply; and (2) *billing and settlement*, organising the financial transactions between all market

[6] A risk-averse consumer with high preferences for locally produced electricity is willing to pay a higher price for their electricity and subsequently the CTA should place higher bids.

[7] The exact connection between *logical actors* and *components* like a UI will be mapped in the next section (Section 5.5).

participants. Both functions can be fulfilled by the same DApp. The IO resulting from the market function is the *MCP* from which the successful and unsuccessful orders as well as the total amount of locally matched energy can be deducted. The billing and settlement function takes as input the consumption and production data from the AMI, calculates the amount of energy that has to be traded with the Energy Service Provider (ESP), and outputs a list of financial transactions, accordingly.

In a community-based LEM, the business actor **public grid** represents the connection to the rest of the power system and to traditional markets. This vague actor can fulfil a variety of functions. However, in the HLUC where the energy community can trade with the public grid, this actor essentially acts as an *ESP*. The DSL employed by [64] describes this logical actor as a party providing *"retail electricity, natural gas, and clean energy options, along with energy efficiency products and services"* [190]. The IO related to this actor is its *pricing model*, i. e. the prices at which electricity is bought and sold to the LEM[8].

Figure 5.4 depicts the interactions between these logical actors for a full cycle of energy trading within a community-based LEM. For illustration purposes, the logical actors *human consumer/producer* were left out of this sequence diagram, since they do not intervene in the operational trading process. However, they communicate their preferences and risk appetite to their respective CTA or PTA prior to the *bidding* phase. A full trading cycle can be divided into three periods:

1. **Bidding:** All market participants place their orders (bids & asks) for the next trading period, based on their respective buying/selling strategy. Bids are sent along with their monetary value which is locked by the DApp[9]. At the end of this bidding phase, the market is cleared and a uniform MCP is calculated. Successful bids/asks and unsuccessful bids/asks are tagged accordingly. In the case of unsuccessful bids, the locked funds are released.

2. **Consumption & Production:** All market participants consume or produce electricity. Market participants who where successful in the local market try to stay within their respective order. Market participants that were unsuccessful in the local market can still consume or produce and will be charged/remunerated at fixed rates set by the ESP.

[8]Depending on the contract between the LMO and the ESP this can, for example, be a flat tariff for buying and selling electricity or a more complex model with different price levels.

[9]In a blockchain-based system this can be done by representing a common fiat currency with a token in a fixed ratio.

3. **Billing & Settlement:** All smart meters in the local market report consumption/production data for the last period to the actor responsible for *billing and settlement*—in this case, the same DApp. With this data, this actor calculates the deviations between bids/asks and the actual consump-

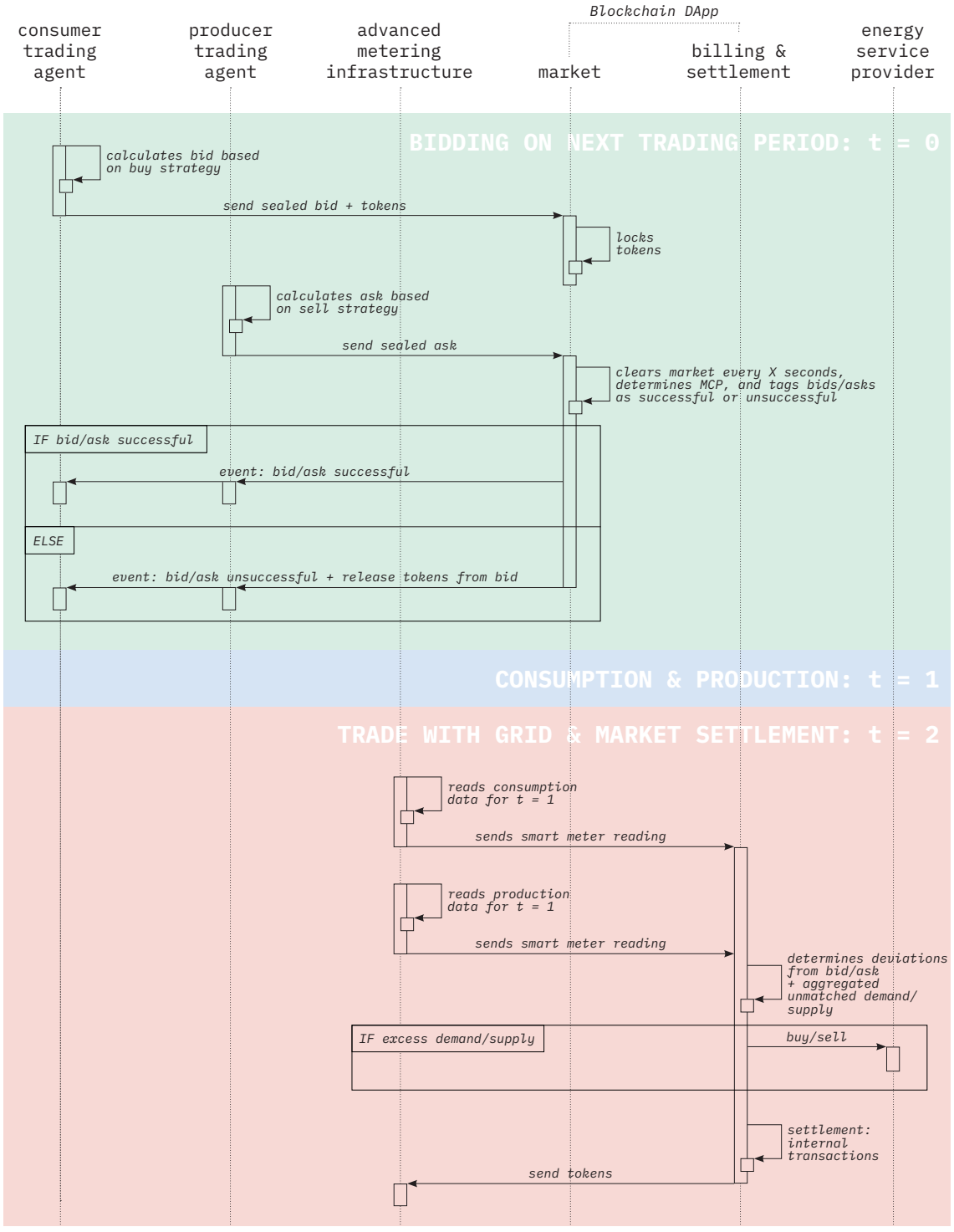


Figure 5.4: *Functional model depicted as sequence diagram showing the interactions between logical actors for a full trading cycle (own illustration)*

tion/production for every market participant and further calculates the total excess demand/supply that has to be traded with the ESP (if there is any). They then settle the financial transactions between the involved parties. All parties are obliged to honour their order, incentivising consumers/producers to place accurate bids/asks. This means that deviations from their original order is settled as if they were trading with the ESP (see Subsection 5.4.2 for more details).

Every logical actors performs a unique function which could be analysed in greater detail. For instance, one could investigate how the *consumer*'s socio-economic background affects their energy-mix preference or their risk appetite; or investigate the best production forecasting algorithm and subsequent optimal bidding strategy of the *PTA*. However, research on these topics is beyond the scope of this thesis which focusses on the design of the DApp performing the two functions *market* as well as *billing and settlement*.

5.4.1 Market: Market Matching & Bidding Format

The proposed market matching mechanism is a double auction with discrete market closing times. It is a common design in literature on LEMs [21, 42, 49, 173, 191] and has proven efficient in traditional energy markets [158]. As stated in $\mathbf{R}_{func}\mathbf{7}$, market clearing times should be variable for the LMOs to adapt to different local circumstances and varying regulation. The author of this thesis further suggests a closed order book, i. e. sealed bidding (see $\mathbf{R}_{func}\mathbf{2}$). This design opens the door to *strategy-proof* auction mechanisms, like the *Vickrey-Clarke-Groves (VCG) mechanism* (see Section 3.1.2 under *Auction-Based Market Designs*). How sealed bidding can be implemented on the basis of Hyperledger Fabric will be discussed in Section 5.5 and Section 5.6.

5.4.2 Billing and Settlement: Pricing & Settlement System

In conjunction with the discrete time double auction, uniform pricing is the proposed pricing mechanism. As described in Subsection 3.1.5, a uniform MCP has proven to result in lower overall cost of electricity in traditional energy markets [160, 161].

Further questions of interest, when taking a closer look at the settlement mechanism, are *at which point in the process transactions are settled* and *how it deals with deviations between bids/asks and the actual consumption/production*.

The first issue boils down to a decision between a *single-settlement* and a *multi-settlement system*. In a single-settlement system, transactions are settled ex-post, after the dispatch of electricity. In a multi-settlement system, transactions are settled as soon as the market is cleared, and later deviations from this first settlement are priced ex-post. These deviations—as stated in the second question—are often addressed with compliance penalties. [192]

This thesis proposes a single-settlement system which settles all transactions after the energy dispatch. However, in order to counter perverse incentives, no new MCP is calculated for the updated supply and demand curves but the original MCP is used to settle transactions. As stated above, all market participants have to honour their order and compensate deviations from their original bid/ask by trading with the ESP.

It must be said, that there is still no consensus on which market design is best suited for LEMs [31] and this thesis does not intend to provide the final answer (see Subsection 3.1.5). Consequently, as stated by **O2** in Section 5.1, the DApp design should be flexible enough to implement a different market design if needed.

5.5 Architecture Model for a Local Electricity Market

The *functional model* discussed in the previous section serves as input for developing the system's 'technical description', i. e. the *architecture model* (see Figure 5.1). The first step is to map the *logical actors* onto *technical components* (software and hardware); this trace-relation is described in Subsection 5.5.1. Afterwards, the *IO flow* between these technical components is analysed; the results of this analysis are discussed in Subsection 5.5.2.

5.5.1 Component Layer

The trace relation mapping *logical actors* to *technical components* is illustrated in Figure 5.5. Since this system design relies on a distributed blockchain network, some components are shared (*consensus*) or replicated (*blockchain ledger*) amongst actors. The blockchain network is described in greater detail in Section 5.5.1 under *Hyperledger Fabric Network for a Local Electricity Market*.

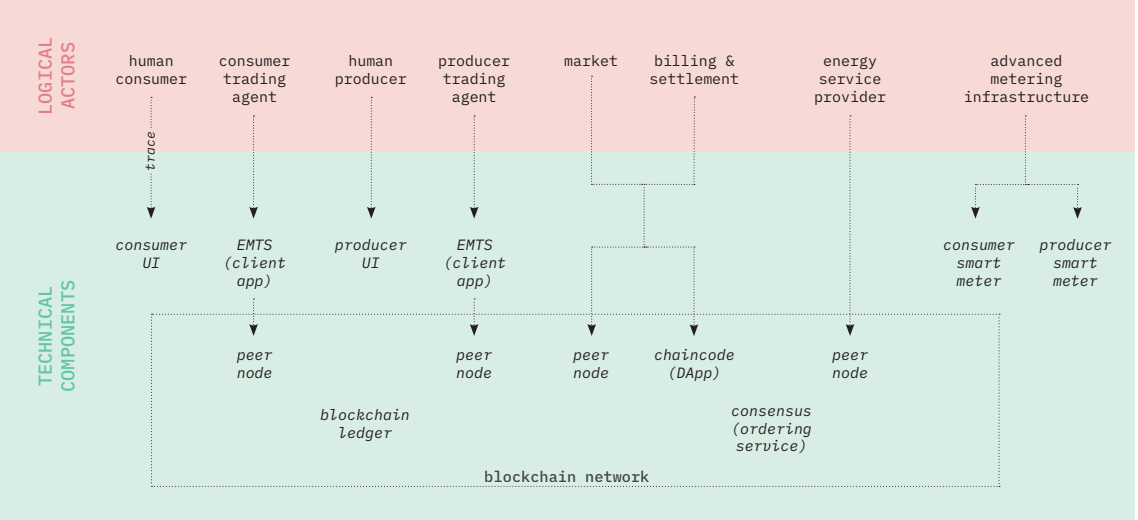


Figure 5.5: *Mapping of logical actors to technical components (own illustration inspired by [46])*

As mentioned in Section 5.4, the **human consumer** needs a *UI* through which they can communicate their energy-mix preferences and risk appetite. These inputs are then translated into a custom buying strategy.

The same applies to the **human producer**, who might, for instance, have preferences regarding the minimum state of charge of their battery or be very risk-averse. These informations must be communicated to the Energy Management Trading System (EMTS) through a *UI*.

The *EMTS* collects consumption or production data from the *smart meters*, inputs from the *UI* facing the end user, and potentially meteorological or historical data and calculates demand and supply forecasts. *EMTS* stands representatively for every device with enough computational intelligence[10] to compute bids and asks (and the corresponding schedule) based on this input data. Using Hyperledger Fabric terminology, the *EMTS* proposes transactions to the blockchain network through the *client application* which then sends this transaction proposal to a number of *peer nodes*[11]. This software code can easily be integrated into the energy management and trading software by means of Software Development Kits (SDKs)[12] for Hyperledger

[10] More research on this topic is needed, but, in theory, a device as small as a *Raspberry Pi* should be capable to carry out this task (e. g. for the Bitcoin blockchain: http://www.raspberrypifullnode.com (last accessed September 17, 2019)).

[11] As described in Chapter 4, transaction proposals are sent to a predetermined number *endorsing peers*, defined by the *endorsement policy*. A more detailed description of the transaction flow can be seen in Figure 5.8.

[12] A SDK generally describes *"a set of software development tools that allows the creation of applications for a certain software package, software framework, hardware platform, or computer system"* [193].

Fabric[13]. The **CTA** can thus be traced to a *consumer EMTS* as well as a *peer node*. Every node is part of the blockchain network and subsequently takes part in consensus-building, maintains a copy of the ledger, and has the chaincode installed[14]. The same applies to the **PTA** which is represented by a *producer EMTS* and a *peer node*.

The two logical actors **market** and **billing and settlement** can be traced to the *chaincode*, holding the business logic. The *chaincode* is the heart of the *DApp* and is installed on all *peer nodes* with the right to endorse transaction proposals—so-called *endorsing peers* in Hyperledger Fabric jargon. The *market* portion of the chaincode accepts buy and sell orders (bids and asks) in form of transactions and calculates the MCP at predefined market closing times. The *billing and settlement* portion of the chaincode accepts consumption and production data, signed by the *smart meters*, trades with the *ESP* to compensate for excess demand or supply, and then settles the financial transactions between all market participants. In order to comply with $\mathbf{R}_{func}\mathbf{2}$—privacy of buy and sell orders as described in Section 5.1—the *LMO* needs to maintain a *peer node* in order to fulfil the *market* and *billing and settlement* functions. More information in this regard can be found in Section 5.5.1 under *Hyperledger Fabric Network for a Local Electricity Market* and Subsection 5.5.2.

Since trades with the **ESP** are instantly settled and represented by token transactions inside the *chaincode* logic, the *ESP* also needs a *peer node* to ensure the procedural correctness of these transactions. However, a design where transactions with the *ESP* are not settled through the *DApp* but by traditional contracts between the *LMO* and the *ESP* is also conceivable. In this case, the *ESP* could be left out of the blockchain network completely. Whether this design would be favourable will depend on the regulatory framework and business reality. The *chaincode* design proposed by this thesis allows for both scenarios, however, the ”more complex” one *with* the *ESP* will be discussed henceforth, as it is easier to remove an actor than add one in retrospect.

As stated before, the **AMI** is made up of a multitude of *smart meters* on the physical level. Every consumer and producer is equipped with such a device which reads and reports consumption and production data for every trading period. A *smart meter* holds its own public/private key pair and can thus sign all the data before sending it to the *EMTS* which then sends a transaction proposal into the blockchain network.

[13]As of today, the officially supported client-SDKs for Hyperledger Fabric exist for node.js and Java. Unofficial releases, still in development, exist for Python, Go, and REST [194].

[14]If it is an *endorsing peer* the chaincode—the smart contract logic for the channel—is installed and transaction proposals can be endorsed. A *committing peer* only maintains a copy of the *ledger*.

Since it is assumed that there are no transmission losses (see A3 in Section 5.2), the *ESP* does not need to be equipped with a *smart meter*. Electricity which is not traded in the local market is automatically balanced as if it was traded with the *ESP*.

The core of this thesis is to design a decentralised energy trading platform for community-based LEMs on top of the blockchain framework Hyperledger Fabric. Therefore, it is worth taking a closer look at the *blockchain network* part illustrated in Figure 5.5.

Hyperledger Fabric Network for a Local Electricity Market

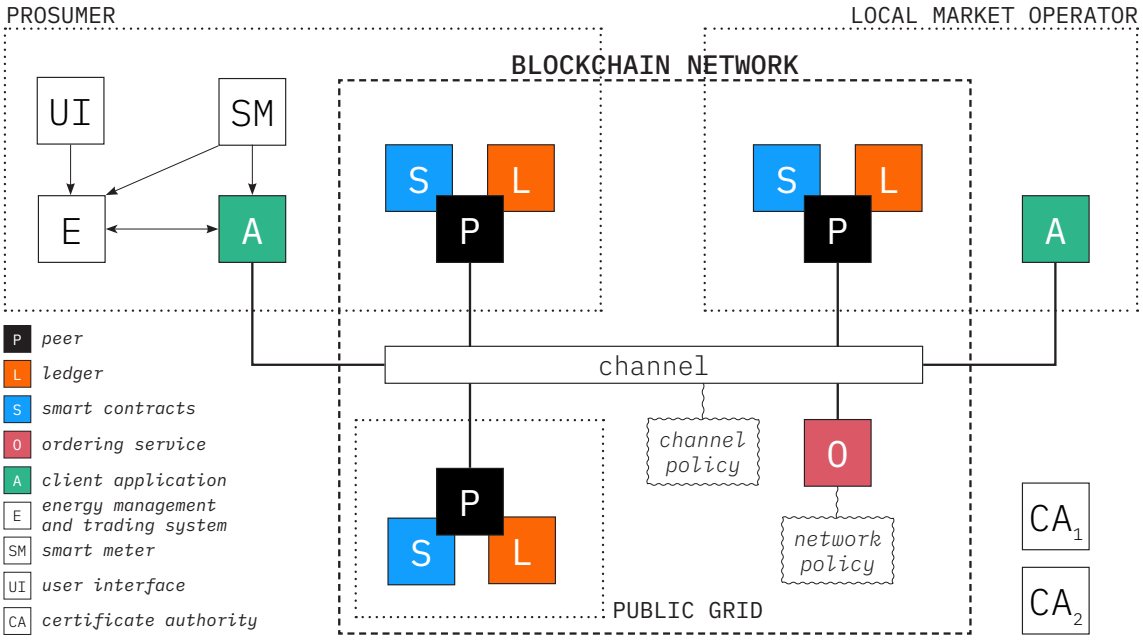


Figure 5.6: *Topology of a Hyperledger Fabric network for a community-based local electricity market (own illustration)*

The blockchain network presented in Figure 5.5 can be broken down into even more granular components. These are illustrated in Figure 5.6. The two business actors **consumer** and **producer** can be represented by the same set-up and are aggregated under the actor **prosumer**. It has further to be said that this set-up represents the smallest possible configuration of blockchain components for every actor. In a more 'realistic' environment, for instance, it would be advisable for the **LMO** to have more peers in order to increase the availability and resilience of the system.

All actors taking part in the *community-based electricity trading use case* form a *consortium*. The members of this LEM *consortium* are all connected to the same *channel* which represents the main communications mechanism by which they can

communicate with each other. The *channel* can be configured in such a way that only the LMO can add new members to it. However, a less centralised *channel configuration* where a predefined share of all members have to agree in order to add new market participants is also conceivable. The same applies to the *network configuration* which describes the rules for new members joining the *consortium*.

In the lower right corner of Figure 5.6—outside of the blockchain network—are two *Certificate authorities [CA]*. These institutions issue certificates to consortium members and network nodes. These certificates are used to map identities to organisations—as the upper-right peer node belonging to the LMO, for instance. Since the LMO is responsible for introducing new *consortium* members, it is reasonable to assume that the same *Certificate Authority (CA)* that issues certificates to the LMO also provides all market participants—consumers, producers, and prosumers—with certificates. The external public grid actor could, however, already rely on another *CA* for other services and use the same certificates to issue identities in this blockchain network[15].

In the upper left corner, the **prosumer** actor holds one *endorsing peer [P]* inside the blockchain network. It is an *endorsing peer* because it holds a copy of the *ledger [L]* and has the *smart contract chaincode [S]* installed in order to process transaction proposals and simulate their outcome. This *peer* is connected to the *channel*. Also connected to the *channel* is the *client application [A]*. It is this (software) component which acts like a door to the blockchain network and which sends transaction proposals in the prosumer's regard. It receives consumption and/or production data from the *smart meter [SM]* as well as bids/asks from the *EMTS*. The *EMTS* can also query the state of the system via this *client* or receive event notifications from inside the network[16].

The **LMO** also is represented by at least one *endorsing peer* inside the blockchain network which is connected to the same *channel* as the other *peers*. Further, a *client application* is connected to the *channel* which acts as a 'scheduler' for the LEM. At predefined time intervals, this *client application* submits transaction proposals, e. g. to clear the market or to settle an energy auction. These transactions underly the logic defined in the *chaincode*. It is thus not possible for the LMO to clear an auction before it is closed. A more detailed description of the chaincode logic is given in Section 5.6.

[15]Scenarios with more CAs or only one CA are also possible. The 'best' set-up will depend on the local circumstances.

[16]If a bid/ask was successful or not, for instance.

The **public grid** actor only holds an *endorsing peer* inside the consortium network. This represents the minimal set-up and is possible because the external actor does not at all interfere in the operations of the LEM[17]. If money flows within the local market are represented by token transactions, the public grid needs to act as an auditor, supervising the token flow. Because transaction *endorsement* is part of the consensus building in Hyperledger Fabric (see Subsection 4.2.1), the peer node must be an *endorsing peer*, enabling it to reject invalid transaction proposals.

As explained in Subsection 4.2.1, the *ordering service* is responsible for ordering and packaging endorsed transactions into blocks—providing delivery guarantees. In a production environment, this service would be provided by a cluster of nodes operating under a distributed protocol (see Subsection 4.2.1). In this scenario, both the LMO and the public grid actor could contribute ordering nodes. It is further advisable that a subgroup of market participants also contribute an ordering node in order to distribute consensus building over all interest parties. The *ordering service* further has the sovereignty to add new actors to the blockchain network. The rules and permissions for doing so are defined in the *network policy*.

As stated before, in order to transact, *client applications* submit transaction proposals to *endorsing peers* and need to receive a predefined number of positive responses—*endorsements*—before they can send the transaction to the *ordering service*. This number is fixed in the *endorsement policy* and embedded in the *channel policy* which is defined at the moment of *chaincode* instantiation on the *channel*. In order to maintain an efficient and scalable system, a policy, which demands that all member organisation of the *consortium* endorse every transaction proposal, should be avoided. A reasonable policy could require the signatures from: (1) a member peer associated with the LMO; and (2) two or three peers, each belonging to a different market participant[18]. This channel-wide policy should further be complemented by a more specific policy, applying to all transactions that target the public grid actor's token balance. These transaction proposals should also be signed by a peer related to the public grid actor, in addition to the parties described in (1) and (2)[19].

[17] In a slightly broader use case—e. g. where the ESP dynamically adjusts electricity prices, or where the local market sells flexibility to the public grid—a *client application* would allow this actor to actively take part in the operations within the local market.

[18] The exact number should be chosen in accordance with the network size and requirements regarding system efficiency. A market design with short trading periods may require a smaller amount of endorsements in order to increase transaction throughput.

[19] This so-called 'key-level endorsement policy' allows for a granular modelling of the validation process. Refer to [195] for a detailed explanation of the subject.

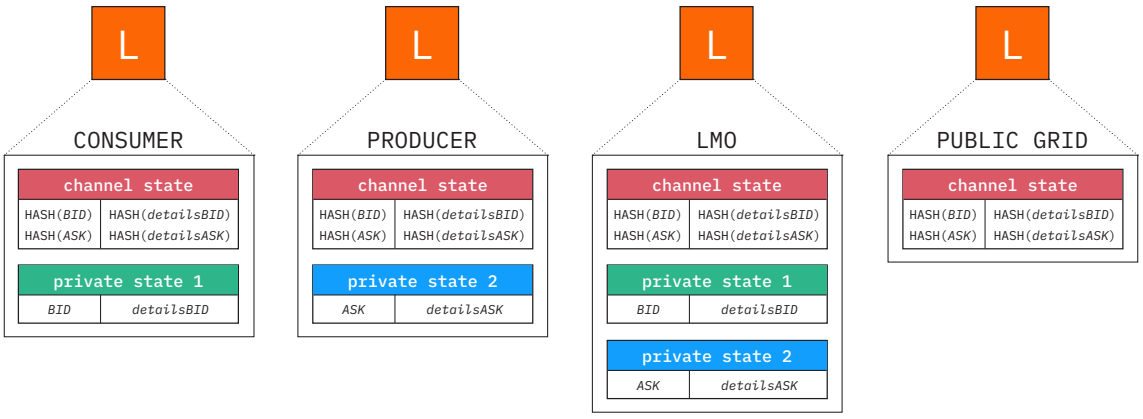


Figure 5.7: *Comparison of ledger (L) compositions for a network of four actors: consumer, producer, local market operator (LMO), and public grid (own illustration)*

Lastly, it is worth taking a closer look at the *ledgers* maintained by all peers. As described in Chapter 4, a *ledger* stores all transactions from a particular *channel* in a *blockchain* and determines the *world state* based on this interlinked transaction record. This *world state* is stored in a classical database for increased speed and easy queries. However, in the chosen market design, bids and asks submitted by the market participants are to remain sealed from the eyes of other market participants and only be shared with the LMO (see Subsection 5.4.1). In Hyperledger Fabric, this is made possible by means of *private transactions* (see Subsection 4.2.2). So, essentially, a bid/ask is split into two transactions: (1) a *private transaction*, holding information on price and amount; and (2) a *public transaction*, visible for all market participants, containing information regarding the auction period, the sender, and other unproblematic data. The private details of a bid/ask are stored in a *private state database*, shared between the invoking party and the LMO. A hash of the transaction is stored on the public blockchain for auditing purposes. The *public transaction* follows the classical transaction flow, which is described in greater detail in Figure 5.8. This means, in addition to the *blockchain* and the database storing the *world state*, every ledger in a market participant's peer further stores the private details of their bid/ask in a *private state database*. Every LMO's peer ledger thus maintains n such *private state databases*, where n equals the number of market participants in the Hyperledger Fabric network. Figure 5.7 illustrates the different ledgers compositions, representatively for the four actors: LMO, public grid, consumer, and producer. BID and ASK are both keys while detailsBID and detailsASK represent the private values stored under the respective key.

5.5.2 Information Flow Between Components

The functional model described in Section 5.4 identified the IOs that are created by each *logical actor*. This subsection now further analyses how these objects are shared between network components described in Subsection 5.5.1. The same set-up as illustrated by Figure 5.6 is assumed.

Table 5.1 shows which components have visibility on which IOs. It thereby differentiates between *processing (p)* or *storing (s)* the IO, or having the right to *query (q)* for it. The high-level IO *bid order* is further broken down into more granular components.

Table 5.1: *Access to information objects for every component inside and outside the blockchain network. Differentiation between processing (p), storing (s), and having the right to query (q) for the object. Exemplary breakdown of a bid transaction (tx) into more granular information objects*

	Prosumer							**LMO**				**Other**				
Information object	*UI*	*SM*	*EMTS*	*A*	*P*	*BC*	P_{DB}	*A*	*P*	*BC*	P_{DB}	P_{PG}	BC_{PG}	P_n	BC_n	*O*
Buying/selling strategy	p		s													
Consumption/production data	q	p/s	p/q	p	p	s		q	p	s		p	s	p	s	p
Market clearing price	q		q	q	p	s		p	p	s		p	s	p	s	p
Financial transactions	q		q	q	p	s		p	p	s		p	s	p	s	p
Public grid pricing model	q		s	q	q	s		q	p	s		p	s	p	s	p
Bid order	q		p/s	p	p	s	s	q	p	s	s	p	s	p	s	p
Public tx proposal				p	p				p					p		
Public tx proposal response				p	p				p					p		
Endorsed public tx				p												p
Block containing public bid tx					p	s			p	s		p	s	p	s	p
Private tx proposal				p	p											
Private tx response				p	p											
Hash of endorsed private tx					p											p
Block containing private bid tx hash					p	s			p	s		p	s	p	s	p
Private bid details				p	p		s		p		s					

Legend: *BC=blockchain, P_{DB}=private state database of prosumer, P_{PG}=peer of public grid, P_n=peer of n other market participants, others: see Figure 5.6*

The prosumer's **buying/selling strategy**, for instance, can be input via the *UI*—processed—and then is stored by the *EMTS*. It can further be assumed that the prosumer has the possibility to query for all high-level IOs via the *UI*.

The **consumption/production data** is recorded by the *smart meter [SM]* and shared with the blockchain network for market settlement and billing. Since financial flows are generated on the back of these readings, it is important for auditability reasons that this information is visible to all actors. It is worth noting that even though the *ordering service [O]* processes all transactions, it can be set-up in a way that it does not have visibility on the transaction content.

The **market clearing price** is generated when the LMO invokes the transaction to clear the market. This transaction proposal is generated by the LMO's *client application [A]*. This transaction then changes the world state and finds its entry into all *blockchains [BC]*. The same applies for the **financial transactions** which

result from the invocation of the settleAuction transaction. The exact flow for the clearAuction transaction is illustrated in Figure 5.9.

The **public grid pricing model** is assumed to be exogenously given at chaincode instantiation. It should be stored by all *EMTS* and all ledgers and can be queried for via all market participants' *UIs*.

The **bid order** stands exemplarily for a more complex IO which involves a private and a public part, as explained in Section 5.5.1 under *Hyperledger Fabric Network for a Local Electricity Market*. The private bid details of the prosumer are stored in a *private state database [P_{DB}]* which only they and the LMO have access to. A step-by-step transaction flow is shown in Figure 5.8 for an exemplary consortium made up of four actors—consumer, producer, LMO, and public grid.

The sequence diagram shows the invocation of two transactions—a public and a private transaction. One can see how the private details of the bid are only shared with the authorised peers. The hash of this confidential data still follows the same path as the public endorsed transaction and is packaged into a block. This design makes it possible to resolve potential conflicts over the 'true' state of the data stored in *private state databases*. The diagram also illustrates the three phases of consensus building in Hyperledger Fabric: (1) transaction proposal and endorsement; (2) transaction ordering and packaging into blocks; and (3) validation and commit (see Subsection 4.2.1).

However, one can see that there is no 'real' consensus over the state of the private details of a bid because only two actors within the consortium—the consumer and the LMO—have reading access on it. In a LEM with more market participants, this means that the LMO is the only actor with visibility of all private data objects. Hence, when the LMO clears the market and computes the MCP, the other members of the consortium would need to trust that the LMO has not altered the private details of the orders at some point. This centralisation of trust would eliminate most benefits of using a distributed system and question the use of a blockchain-based framework in the first place.

How this problem is solved can be best explained with the aid of Figure 5.9. This sequence diagram illustrates the interactions between different components of the blockchain network when invoking the transaction that clears an energy auction. The LMO's *client application* is the clock of the system and the impulse generator for the clearAuction() transaction. The beginning and the end time of every auction is hard coded and visible to all parties. The chaincode makes sure the clearAuc-

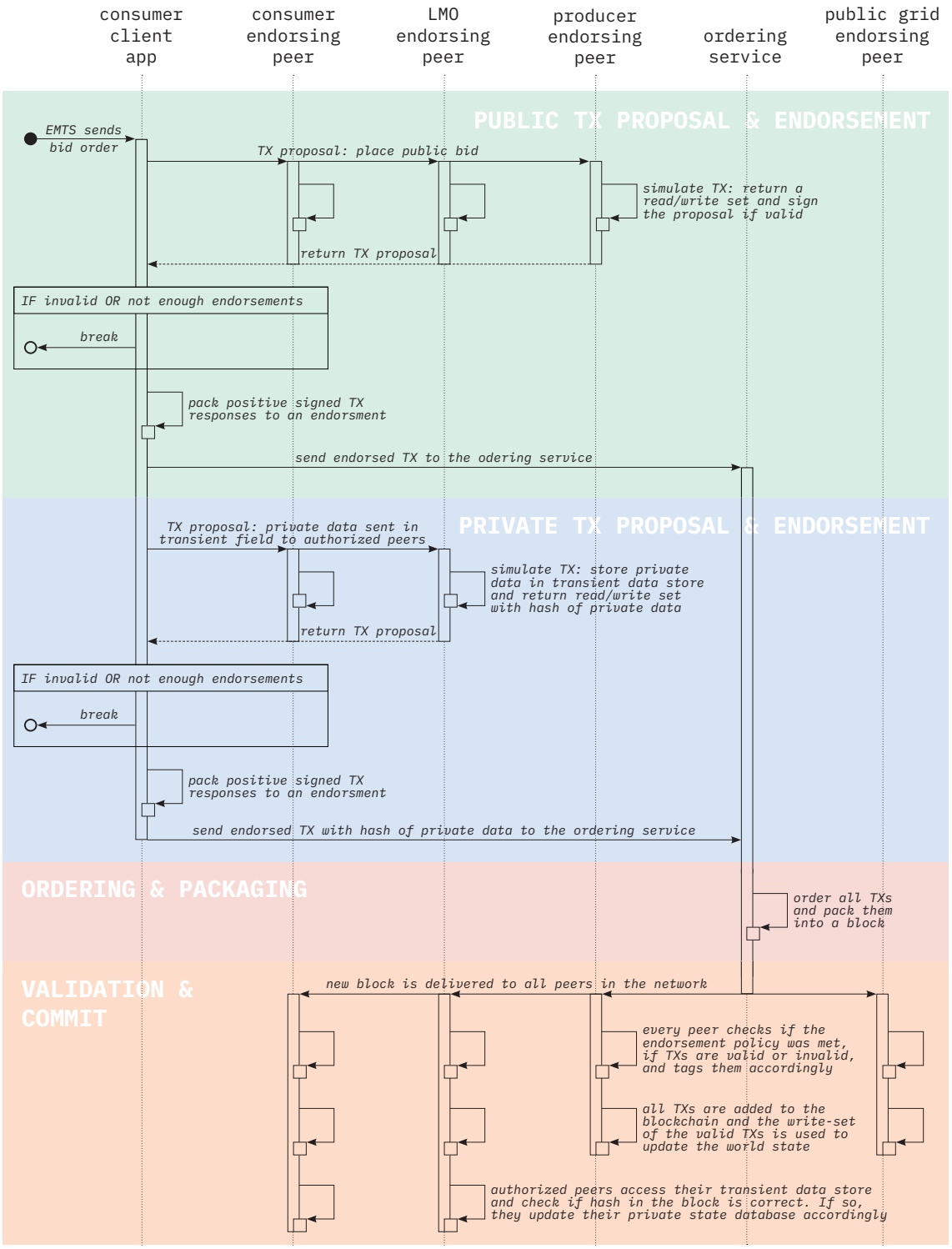


Figure 5.8: *Transaction (TX) flow for a bid order sent by a consumer's EMTS in a Hyperledger Fabric consortium, made up of a consumer, a producer, the LMO, and the public grid actor (own illustration)*

tion() transaction only is endorsed if the auction time has elapsed—meaning the auction status is 'closed'. Since the LMO is the only actor with access to all private state databases containing the private bid and ask details, the *client application* aggregates all these details and sends them along with the transaction in the tran-

sient input field. Data in this transient field is only shared with authorised peers. These peers then simulate the transaction and endorse it if valid. The results of this transaction are: (1) the MCP, which further is used by the chaincode logic to automatically tag bids and asks as 'successful' or 'unsuccessful', (2) a hash of the full transaction proposal, which is added to the blockchain for audit purposed; and (3) a hash of each input element (the private details of each order), which can then be verified by each market participant to be sure the inputs have not been manipulated before computation. As for the bid transaction described in Figure 5.8, the endorsed transaction is then sent to the *ordering service* where it is packed into a new block and then delivered to all peers in the network. Every peer verifies the validity of each transaction in the block before committing it to the ledger and only updates the world state for the valid transactions.

5.6 Chaincode Model

From a system developer's perspective, a blockchain is merely a data layer protected by a logic layer defining what the outside world can do with the inner data. Hence, this logic layer—called chaincode in Hyperledger Fabric—is at the heart of the system. It defines which assets are represented on the blockchain, which transactions can be executed and how exactly these transactions alter the state of these assets, and formulates which actors are allowed to invoke these transactions and under which circumstances. This subsection proposes a language-agnostic chaincode model on the back of a UML class diagram, illustrated in Figure 5.10.

Before looking at the diagram in greater detail, it is useful to break down a Hyperledger Fabric application into three components: (1) *participants*, describing the actors that are part of the business network; (2) *assets*, which can represent everything from tangible[20] to intangible[21] objects; and (3) *transactions*, defining the instructions for modifying these assets. This design follows a *model-controller pattern*. *Models* describe the shape of the data which is stored on the blockchain and *controllers* define the rules and the actions that apply to these models. Every model instance is stored on the world state database as *key/value-pair*. Every *key* is composed of the model's type and its unique identifier. For this reason, all models have the two attributes `type` and `id`. Table 5.2 shows exemplary *key/value-pairs* for the proposed chaincode model. It is further worth pointing out that chaincode always

[20] e. g. real estate, hardware, ...
[21] e. g. contracts, intellectual property, ...

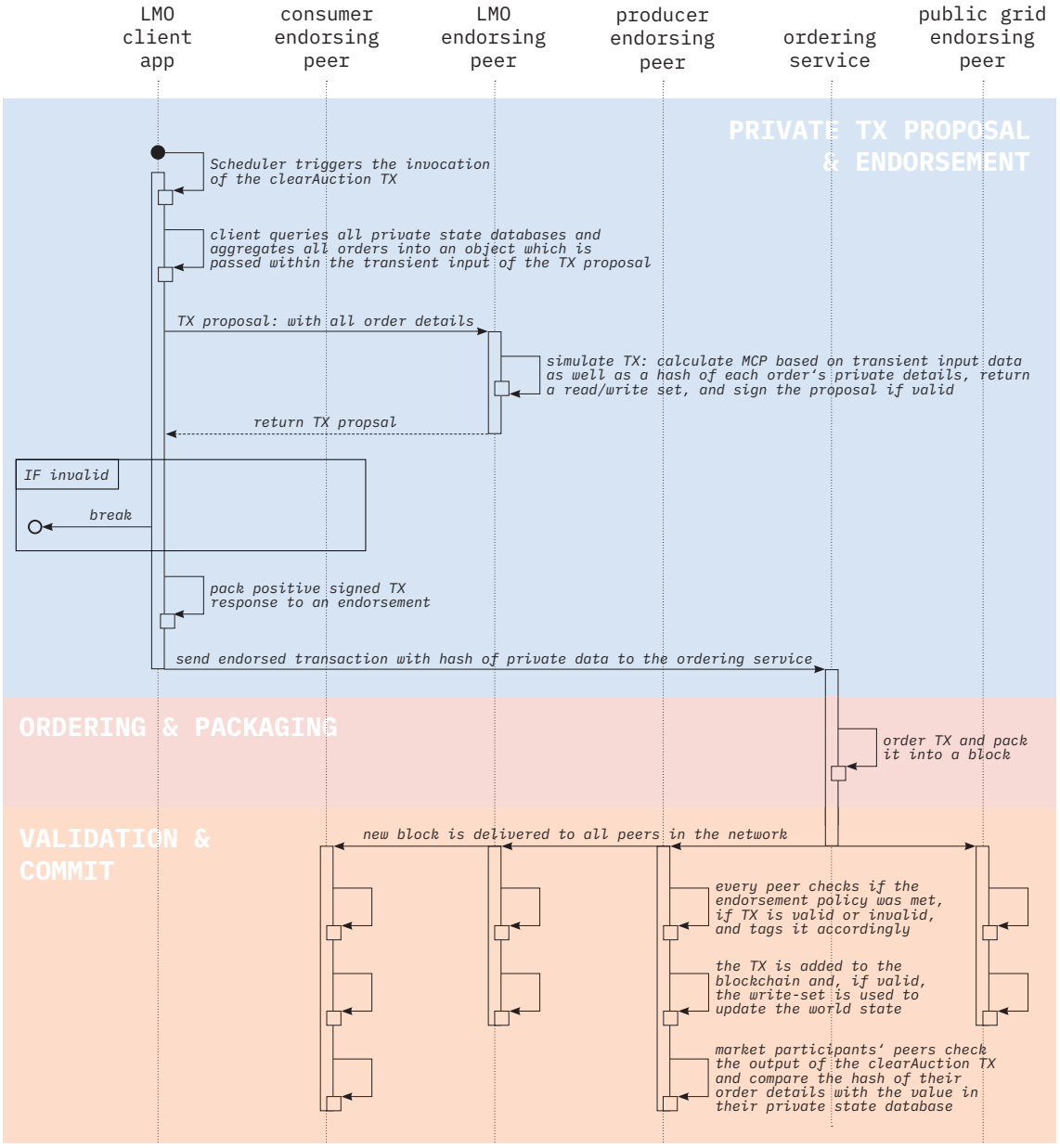


Figure 5.9: *Sequence diagram of the market clearing transaction (TX) sent by the local market operator in a Hyperledger Fabric consortium, made up of a consumer, a producer, the LMO, and the public grid actor (own illustration)*

has to be deterministic [196], i. e. when two *endorsing peers* simulate a transaction, the outcomes of this transaction always have to be identical[22].

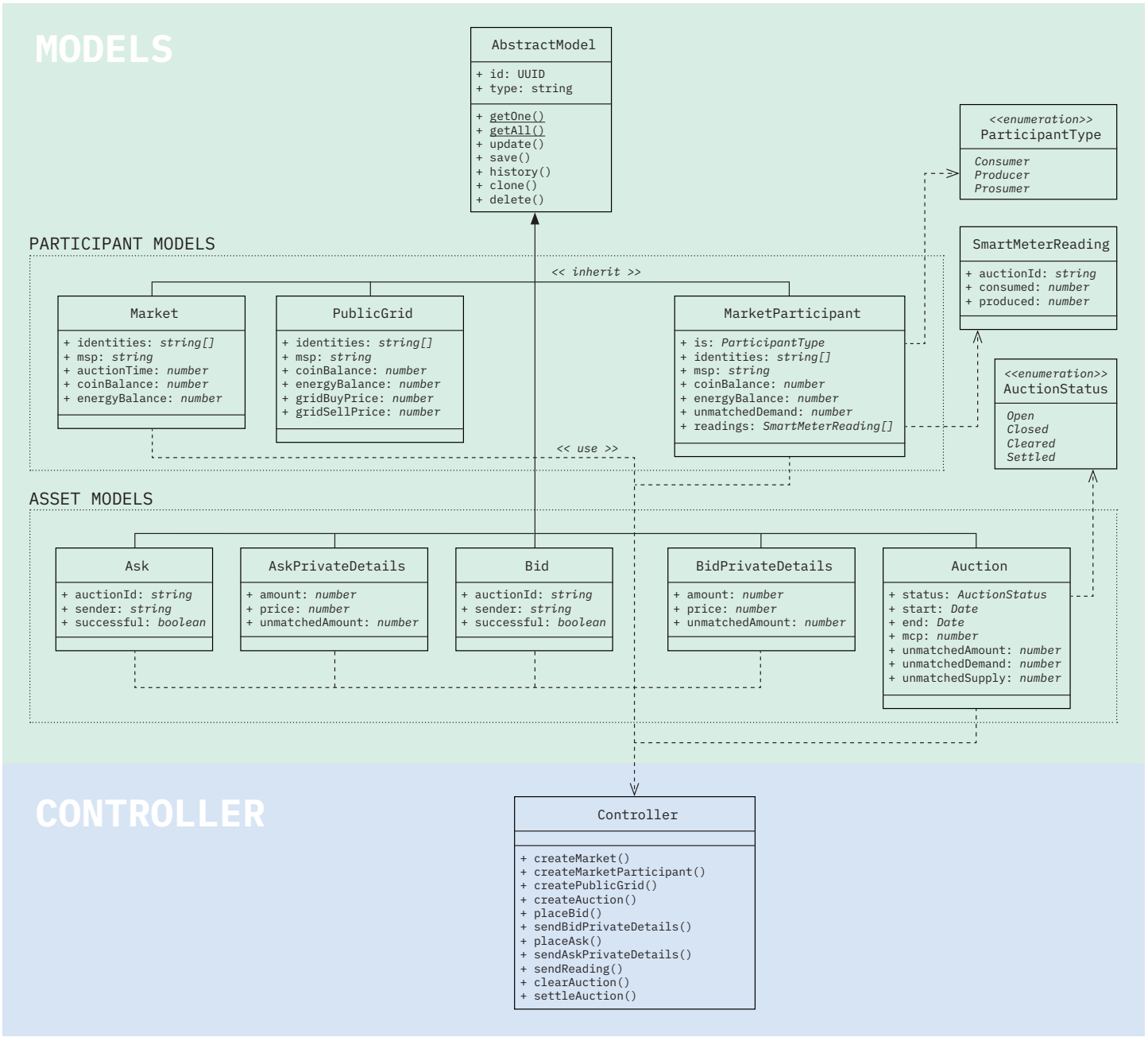
Participants

As described in Subsection 4.2.3, the CAs are responsible for identity management and the issuance of certificates. The Membership Service Provider (MSP) then maps

[22]For instance, chaincode containing methods which create random numbers is not deterministic since its invocation does not yield consistent results.

these identities to an organisation in the blockchain consortium. This way, when a party invokes a transaction, the transaction is signed and all peers know who the originator of the transaction is. These access control rules define the permissions over resources and the access to information and can be hard-coded into the chaincode based upon an identity's attributes. This design pattern is called *Attribute-Based Access Control (ABAC)* [197]. Essentially, the different types of consortium members are represented as *models* in the chaincode, each linked to their respective CA identity or identities. When a transaction triggers a chaincode execution, the *controller* checks if that identity has the right to perform this task or not. This way, it

Figure 5.10: *Chaincode model illustrated as unified modelling language class diagram (own illustration)*



is possible to define granular access rights, such as identityA from organisationB can approve a transaction.

In a community-based LEM there a three types of actors: (1) market participants, which can be either consumer, producer, or prosumer; (2) the LMO, which can, for instance, have multiple employees with different access rights; and (3) the public grid actor, which, just like the LMO, will have multiple identities linked to its organisation. As seen in Figure 5.10, these actors are each represented by one class called MarketParticipant (1), Market (2), and PublicGrid (3). They all have the attributes msp and an array of identities. The identities array can be read as a list of all certificates with the right to sign a transaction in the actor's regard.

All three models further have the attributes coinBalance and energyBalance. These attributes represent an actor's token accounts, with one token representing monetary value and the other representing energetic value. These token values should be stable, for instance, a monetary token could represent 1 Euro-cent while an energetic token equals 1 kWh. However, the exchange rate must be fixed in advance but the ratio can be chosen depending on the local circumstances. As specified in **A6** (see Section 5.2), it is assumed that a third party allows market participants to exchange fiat currency into locally used tokens. This service could, for example, be provided by the LMO or by a bank. In the latter case, the bank would have to become a member organisation of the blockchain network. However, it is possible to move these exchange operations to another channel. This way, the bank would not have any visibility on the transactions happening in the 'primary' channel. Thanks to the support of *cross-channel transactions* in Hyperledger Fabric, a consortium member of the 'primary' channel could interact with consortium members of the 'secondary' channel, as long as they are part of both consortia. This use case, however, is not the primary focus of this thesis which is why enough token liquidity is assumed within the local market. The energyBalance attribute allows the tracking of energy flows and can take positive (consumption) as well as negative (production) values. The energyBalance value of the Market model should always be zero, as the market does not participate in energetic transactions. However, since it acts as an intermediary between the public grid actor and the market participants the value can change 'in the course' of a transaction and is used to verify the correct calculation of energetic transactions.

In addition, the MarketParticipant model has an attribute called is which can take the values consumer, producer, or prosumer. This way consumers can, for instance, be forbidden the right to place sell orders. Another attribute which is only found

in the MarketParticipant model, is an array of readings. These readings are a model themselves called SmartMeterReadings which, however, are never stored on the ledger as their own key/value-pair. They are encapsulated in the readings attribute of the MarketParticipant model and are therefore called a 'flat' model.

The Market model further requires the attribute auctionTime, specifying the duration of one auction period, as well as the two attributes gridBuyPrice and gridSellPrice, indicating the rate at which a market participant can buy electricity from or sell electricity to the public grid, respectively (see **A5** in Section 5.2). In the eyes of the author, a future version of this decentralised energy marketplace should allow to dynamically set these exogenous energy prices. Additionally, the local Distribution System Operator (DSO) should be added to the actors of the energy-community consortium and make sure the energy transactions found by the market can actually be carried out on the physical level (see Section 7.3: *Further Work & Outlook*).

While there can be multiple instances of the MarketParticipant model, in the proposed design, there should only be one instance of each the Market model and the PublicGrid model. However, the chaincode model could also be used in a use case with more than one external public grid actor, for instance.

Assets

There are three types of assets the lifecycles of which are to be recorded on the blockchain. They are represented by the three models Bid, Ask, and Auction. As for the *participant models* described above, all these *asset models* have the attributes type and id which—taken together—compose they *key* under which all *values* for this asset instance are stored on the ledger (see Table 5.2).

The Auction model describes how an energy auction is represented on the blockchain. For every trading period, a new Auction instance is stored on the ledger. Their creation can—but does not have to—be restricted to the identities which can be linked to the Market instance, and thus can be identified as belonging to the LMO's organisation. An Auction instance is described by the following attributes:

— start, describing the time at which the trading period begins;
— end, the time at which no more orders can be placed;
— auctionStatus, which can take the values *open*, *closed*, *cleared*, or *settled*;
— MCP, the equilibrium price for this auction period;
— unmatchedAmount, the energy that could not be matched in the LEM;

Table 5.2: *Exemplary key/value-pairs for the proposed chaincode model in the form they are stored on the ledger*

Key	**Value**
[string in type#id format]	[string in JSON format]
World state database	
"MarketParticipant#PAR1"	{"is":"consumer","coinBalance":2000,"energyBalance":100,"readings":[...], "msp":"Org1MSP","identities":["F4:F5:9C:61:22:06:19:B5:22:D6:59:38:C1:3A:9C:76:B9:2B:7B:3E"]}
"MarketParticipant#PAR2"	{"is":"prosumer","coinBalance":1000,"energyBalance":-100, "readings":[...],"msp":"Org2MSP","identities":["F3:34:06:92:46:7B:9D:07:90:E4:65:3E:CE:6A:79:EB:3C:3F:01:CD"]}
"Market#MKT"	{"auctionTime":900000,"coinBalance":0,"energyBalance":0, "msp":"Org3MSP","identities":["4B:B9:69:E9:BB:5E:8D:41:53:90:E1:B9:0D:71:39:0D:64:A1:9A:1E"]}
"PublicGrid#GRID"	{"coinBalance":12500,"energyBalance":-500,"gridBuyPrice":25, "gridSellPrice":5,"msp":"Org4MSP","identities":["2A:25:5C:5E:63:82:41:55:FA:D2:C6:B3:E7:7E:C4:E4:8B:4D:DB:85"]}
"Auction#AUC1"	{"start":1559670042190,"end":1559679042190,"status":"cleared","mcp":10, "unmatchedAmound":10,"unmatchedDemand":0,"unmatchedSupply":10}
"Bid#AUC1_PAR1_1"	{"auctionId":"AUC1","sender":"PAR1","successful":"true"}
"Ask#AUC1_PAR1_1"	{"auctionId":"AUC1","sender":"PAR2","successful":"true"}
Private state database of PAR1	
"Bid#AUC1_PAR1_1"	{"amount":20,"price":15,"unmatchedAmount":0}
Private state database of PAR2	
"Ask#AUC1_PAR2_1"	{"amount":30,"price":10,"unmatchedAmount":10}

— unmatchedSupply, the offered energy that could not be sold within the LEM;
— unmatchedDemand, the amount of unsatisfied energy demand within the LEM.

It can be noted that the last three attributes do not necessarily carry crucial information which needs to be stored on the blockchain which means that they could be stored off-chain in a traditional database. By storing the information on-chain, however, the settleAuction transaction does not have to recalculate those values which increases the speed. Depending on whether data storage or transaction speed is the limiting factor in a real-life implementation, the system designers can choose to prioritise the one over the other and implement the transaction logic accordingly.

The Bid model essentially consists of two classes: Bid, and BidPrivateDetails. Instances of the Bid class are stored on the public ledger and it contains the follow-

ing specific attributes: auctionId, indicating the auction period for which the bid is placed; sender, the id market participant placing the bid; and successful, which is *false* by default but can be set to *true* by the clearAuction function (see Section 5.6: *Transactions*). Instances of the BidPrivateDetails, on the other hand, are not stored on the public ledger but are stored in private state databases instead. Each BidPrivateDetails instance shares the same type and id values as its 'public' Bid counterpart in order to link the two instances to one another. Additionally, each instance has the three 'private' attributes: the amount, indicating the amount of energy in *energy token* units which the market participant is bidding for; the price, which the market participant is willing to pay per *energy token* unit; and an optional attribute unmatchedAmount, which can only be set by the clearAuction transaction in case the bid can only be partially matched[23].

The Ask model consists of the two classes Ask and AskPrivateDetails. Both share exactly the same attributes as their Bid counterparts and will not be explained in greater detail.

Transactions

Transactions in Hyperledger Fabric can be of three different types: (1) *instantiate*, which installs chaincode on a channel; (2) *invoke*, which calls a particular function defined in the chaincode; and (3) *query*, which only reads the data on the ledger without going through the process of ordering, validation and commit. The interesting transactions for the chaincode model are the *invoke* transactions because they define how instances of the previously described data models—participants and assets—can be created, updated, or deleted[24]. Every model has some internal methods which are called from within the chaincode. These are:

— save(), which saves the instance to the ledger;
— update(attr), which updates one specific attribute of the instance;
— delete(), which deletes the instance;
— clone(), which creates a copy of the instance;

[23]The implemented market design allows for partial matching of orders. There are arguments against this design—a washing machine either needs all the energy to wash or none, for instance—and arguments in favour—market participants might place a multitude of small bids to increase their chance of getting the highest amount of energy which can 'pollute' the system, for instance. This chaincode model, however, leaves all the liberty to system designers to implement the market clearing mechanism they wish within the clearAuction transaction.

[24]Of course, if data is deleted from the world state database, the transaction which triggered the deletion is still recorded on the blockchain.

— history(), which returns the lifecycle for the given instance;
— getAll(), which returns all instances of this model type;
— getOne(id), which returns one specific instance of this model type.

The external functions that can be invoked by transactions are defined in the *controller*. These are illustrated in Figure 5.10 and are explained in greater detail below. It has to be pointed out that the focus lies on the 'main' functions which provide the basic functionalities of the decentralised marketplace. In a real application, it could certainly be useful to have a number of helper functions. However, these will not be elaborated upon, as they are of little interest from a system design perspective.

The createMarketParticipant() function registers a new MarketParticipant instance. It can be configured in a way that only the LMO can invoke this function or that only an identity related to the organisation of the to-be-created MarketParticipant can invoke it. Of course, all functions creating a new model instance should make sure that no model with the same id already exists. The function createPublicGrid() works in a similar way. If the use case only allows for one public grid actor, the function should be restricted to only allow for the creation of one PublicGrid instance. This certainly holds true for the createMarket() function which creates a Market instance of which there can only be one. If the LMO is the only party allowed to register new participants, the three functions above could be placed in a separate *controller* entirely. However, for simplicity, in this chaincode model, all invokable functions will be placed in a single *controller* class.

The createAuction() function creates a new Auction instance. The status is *open* by default and only identities linked to the LMO should be allowed to invoke this transaction. The auction duration—substracting the start attribute from the end attribute—should be slightly shorter than the auctionTime attribute in the previously created Market instance. This way the MCP can be computed before the start of the next auction period. This is crucial, because market participants need to know if their bids or asks were successful or not. The lifecycle of an Auction can thus be summarised as: (1) *open*, accepting buy or sell orders; (2) *closed*, when the end time was passed; (3) *cleared*, after the MCP has been computed; and (4) *settled*, after the settleAuction() transaction was invoked.

The placeBid() function and the placeAsk() function are almost identical and are the 'public' part of the order placement. Both can only be invoked by identities related to a MarketParticipant instance. The first can further be restricted to *consumers*

and *prosumers*, while the latter can be restricted to *producers* and *prosumers*. Bid and Ask instances created this way are only valid if the Auction which is specified in the auctionId attribute is still *open*. Furthermore, the sender attribute has to match the party invoking the transaction to prevent placing bids or asks in another market participant's regard. When the placeBid() or the placeAsk() transactions are invoked when the auction status is still *open* but the end value is smaller than the current time, then the Bid or Ask are not saved to the ledger but the auction status is changed to *closed*.

The sendBidPrivateDetails() and sendAskPrivateDetails() are the 'private' counterpart to the previous two 'public' transactions and are subject to the same rules. The exact transaction flow showing the difference between a 'public' and a 'private' transaction can be seen in Figure 5.8.

The clearAuction() function is passed an auctionId as well as all private bid and ask details for the given auction period in a transient field, and clears the given Auction instance. The chosen market design essentially manifests itself in this function. This chaincode model proposes a design for a discrete time double auction with uniform pricing, as described in Subsection 5.4.1. However, the model could be easily adapted to a continuous double auction with a discriminatory pricing mechanism. In this design, however, the clearAuction() transaction can only be invoked by an identity linked to the LMO and only if the auction is *closed* or if the transaction timestamp is higher than the auction end attribute. This design guarantees that the LMO acts according to the rules. The full transaction flow for this transaction is illustrated in Figure 5.9.

The sendReading() function saves a passed SmartMeterReading object in the readings array of a MarketParticipant instance. The invoking party's identity has to match a value within the identities array of this MarketParticipant to make sure the data comes from the right smart meter[25].

The settleAuction() function is subject to similar rules as the clearAuction() function. It can only be invoked at the end of auction period t+2, if t is the period in which buy and sell orders are placed. Its invocation is further restricted to members of the LMO's organisation. As pointed out in Subsection 5.4.2, this design follows an approach where market participants are penalised if their actual energy usage or production differs from their buy or sell order. The settleAuction() automatically

[25]If the smart meter happens to be directly connected to the blockchain network, its public key could be amongst the trusted identities specified in the identities attribute of the given MarketParticipant.

compensates deviation from the order by trading with the public grid actor. However, all these trades are first aggregated for all market participants and then settled as a one-time transaction between the Market and the PublicGrid instances. This means, if two deviations cancel each other out[26], the market participants are still charged as if they were trading with the public grid, but the LMO gets to keep the penalties. This way, market participants are incentivised to place accurate bids/asks and the collected penalties can be part of the LMO's revenue stream. However, further research needs to analyse this penalty design from a game theoretic point of view to make sure no paradoxical incentives are created. If so, for example, classic price functions could be implemented in the transaction logic to describe the penalty design (see Section 3.1.2 under *Non-Auction-Based Market Designs*).

As explained at the beginning of this subsection, *query* transaction can also be defined in the chaincode, together with sensible access rights. An example could be a *query* function called getBidById() which returns both the private and the public part of a bid if the invoking identity has the right to access the private state database. However, since these transactions are not part of the core use case of *energy trading*, they will not be explained in greater detail.

This chapter proposed an *architecture model* as well as a *chaincode model* for a *community-based energy trading marketplace* based on Hyperledger Fabric. The *architecture model* was developed on the basis of a business and a functional analysis. Building up on these results, the *chaincode model* translates the previously identified business actors, their different functions, and their respective range of authority into smart contract logic. It follows a *model-controller* pattern, the benefit of which lies, amongst others, in the transportability of the model classes. Because the programming languages that can be used for chaincode development—currently, node.js, Java, and Golang—are also common in other layers of application development, the same models can be used and passed from one layer to another. So, the Auction object created in the web browser using JavaScript, for instance, could easily be passed to the backend *client application* written in node.js, which then uses this same Auction object as input for the createAuction() transaction. An example of this 'model chain' can be found in the code accompanying this thesis. The framework conditions and more details regarding the *implementation* of the proposed artefact are described in Chapter 6.

[26] E. g. a consumer underestimated its consumption by 10 units and a producer underestimated its production by the same amount.

Chapter 6

Implementation

The implementation of the proposed Decentralised Application (DApp) model is a fundamental part of the Design Science Research (DSR) process described in Section 1.3. It serves the purpose of demonstrating the designed artefact's usability and evaluating how well it satisfies the previously defined design objectives and functional requirements (see Section 5.1). This chapter is thus split into two parts: the first part describes the implementation of the DApp model and the second part illustrates the testing environment which was developed in order to evaluate the artefact, i. e. the DApp model.

6.1 Implementation of the Proposed Design

The DApp design is made up of an *architecture model* and a *chaincode model*. The former describes the different technical and software components which make up the proposed distributed energy trading platform and illustrates the information flow between these components. The *chaincode* is a core component of the DApp *architecture* and describes the smart contract logic which defines the instructions for reading and updating data shared between consortium members. The *chaincode model* presented in Section 5.6 is a language-agnostic design of this transactional logic and could be implemented with any of the programming languages available for chaincode development—currently node.js, Golang, and Java. In this case, the *chaincode model* was implemented in node.js, a JavaScript runtime environment. This language was chosen because JavaScript is currently one of the most widespread general purpose programming languages (and because the author is quite familiar with it). The *chaincode* was written on top of Hyperledger Fabric version 1.4—the

latest stable version as of writing this thesis. To help with development, some supporting tools were used which are described below.

6.1.1 Development Environment

The *Convector Suite*, a set of open source tools, was used to help with application development. It comprises the following three tools:

1. Convector Smart Contracts, a framework streamlining smart contract development. It is based on TypeScript, a superset of JavaScript which compiles into plain JavaScript [198]. It is part of *Hyperledger Labs*, a pre-stage to becoming a full-fledged Hyperledger project, since March 2019 [199].
2. Hurley, a command line tool which is used to deploy a, and interact with a Hyperledger Fabric blockchain network.
3. CLI Convector, a command line tool which helps starting and building a *Convector* project.

The project was built with the help of the CLI Convector tool and the *chaincode model* was implemented using Convector Smart Contracts which has the benefit of being a higher-level framework, abstracting complexities and thus reducing the risk of errors in the code. As it compiles into native JavaScript it can be deployed on any Hyperledger Fabric network. Hurley was used for testing during development and evaluation (see Section 6.2). The implementation was developed on macOS 10.14.5 installed on a MacBook Pro of 2016.

Prerequisites for Hyperledger Fabric

```
Node.js: 8.15.0
Docker: 18.09.5
Docker-compose: 1.23.1
```

Convector Suite Tool Versions

```
Convector Smart Contracts: 1.3.3
```

```
Hurley: 1.0.5
Convector-CLI : 1.1.3
```

6.1.2 Smart Contract Code

All code can be found on GitHub under https://github.com/raphmc/thesis. The *chaincode* was implemented in accordance to the Unified Modelling Language (UML) class diagram illustrated in Figure 5.10. It should be seen as a proof of concept which serves the evaluation of the DApp model (see Chapter 7). Important choices which were made during the development process are listed below.

Coin A coin with a fixed value of 1 Euro-cent per coin was implemented as means to follow the cash flow within the local market. However, with the release of Hyperledger Fabric 2.0, native support for a FabToken was announced [200]. This coin will be better suited for a production-ready implementation than any custom-designed coin could be. Consequently, no safety features which would be necessary in a 'real-life' environment were included. For instance, a MarketParticipant can have a negative coinBalance. It has to be pointed out that a token has no real value in closed blockchain systems. It merely serves the purpose of facilitating accountability. Ideally, in the future, a Local Electricity Market (LEM) blockchain is connected to a public blockchain and uses an external token which has some value outside of only the local market.

Energy Flow The flow of energy is represented by each participant's energyBalance. It can thus be seen as an 'energy token'. An exchange rate of 1 token : 1 kWh was defined and implemented.

Auction Every Auction has a start and an end attribute. These are implemented as UNIX Epoche time which is equivalent to the number of seconds that have elapsed since 00:00:00 of 1 January 1970 [201]. An auction's duration has to match the auctionTime attribute from the Market instance. When the placeBid, sendBidPrivateDetails, placeAsk, or sendAskPrivateDetails transaction is invoked, the *chaincode* logic makes sure the passed auction is still *open* by comparing the transaction's timestamp with the end attribute of the given auction. In case the auction is still 'open' but the transaction timestamp is greater than end, the auction's status is changed to *closed* but the order is rejected.

Market Clearing The market is cleared when the clearAuction transaction is invoked. The algorithm goes through all bids and asks for the given auction period and creates a demand and supply curve. The demand curve orders the bids in descending order while the supply curve orders the asks in ascending order. These curves are stored as arrays with the position of the array representing the price. For simplicity, all bid and ask prices are assumed to be integer values. The value of the array at each price point represents the amount of energy that would be bought or sold at given price. For finding the Market Clearing Price (MCP), the algorithm goes through both arrays and identifies where supply is higher than demand. It then compares demand at that price point with supply one price point lower. If the latter is smaller than the former, the lower price point is identified as the MCP. In the opposite case the higher price point is the new MCP. This way, it is guaranteed that efficient supply is available for the corresponding demand. The algorithm then iterates through the totality of bids and updates the successful attribute to *true* if the price is equal or higher than the MCP. The same applies to all asks if the price is equal or lower than the MCP. If demand and supply are not equal at the MCP, there will be one bid or ask with an unmatched amount. Partial matching is allowed in order to maximise the allocation of locally produced energy. In case no intersection of the two curves is found, the MCP is set to -1 to indicate that no solution could be found. Finally, the auction is set to *cleared*, the unmatchedAmound, unmatchedDemand, and unmatchedSupply updated, and the MCP is returned.

Market Settlement The market is settled when the settleAuction transaction is invoked. The settlement algorithm goes through all bids and asks for the given auction period and selects the successful ones. It then iterates through all market participants and compares the actual smart meter data with the successful orders. In case of deviations, the difference is settled at grid prices via the Market instance. This means both the MarketParticipant's and the Market's energyBalance and coinBalance are updated. At the end, the accumulated energyBalance of the Market is calculated. If it is negative, energy is bought from the Grid; and if it is positive, energy is sold to the Grid. This means that the Market's energyBalance is always *0* but the coinBalance can be positive. This way, participants are incentivised to place accurate orders.

Query Transactions In addition to the invokable functions that are part of the *controller* described in Figure 5.10, some invokable 'query' functions are im-

plemented. For example, a 'query' called getAuctionById() which returns a specific Auction instance when passed a valid auctionId was added. These are necessary for the unit tests explained below.

6.2 Testing Environment & Scripts

There are currently two types of clients which allow the interaction with a Hyperledger Fabric network: a command line tool and the Software Development Kits (SDKs). While the former is typically used for testing and administration purposes, the SDKs are designed for developing *client applications* which must interact with a deployed blockchain network—primarily, to invoke chaincode functions (see Section 5.5). SDKs are currently available for Java and node.js. Support for Golang and Python is announced, but still in development.

For testing the above *chaincode*, some unit tests were written that make use of the node.js SDK. A unit test is essentially a script which tests some parts of the code (e. g. a function, a module, or a group of functions). It does that by calling the code under test and comparing the results with expected values or events. In this case, the open source test framework mocha[1] is used to test the *chaincode*. For simplicity, a single test script called test.sh was written. When this script is executed, it deploys a new *blockchain network* and installs the *chaincode* on all peers. Once the network is functional, it runs the mocha test script. The following sections explain those steps in greater detail.

6.2.1 Blockchain Network

Before the *chaincode* can be tested, it has to be installed and instantiated on a network. The test network is deployed using Hurley. It is made up of one channel ch1 to which are connected a total of 4 organisations: org1, the Local Market Operator (LMO); org2, the public grid actor; and org3 and org4, two market participants. Each organisation has 2 members, each with their own public/private key pair. The blockchain network is thus made up of 17 docker containers: one container running the *ordering service*, operating with the solo consensus mechanism (see Subsection 4.2.1), plus 4 containers per organisation, namely:

— CA container: the Certificate Authority (CA) for the given organisation.

[1] See https://www.npmjs.com/package/mocha (last accessed September 17, 2019).

— peer container: one *endorsing peer* belonging to the given organisation.
— couchDB container: the *world state database* maintained by the above peer.
— chaincode container: the *chaincode* which is installed on the above peer.

Two *private data collections* are defined—one for each market participants. These collections define which organisations have access to the *private state database*. For market participant 1 (org3), for example, the policy is defined as follows: "OR('org1MSP.member', 'org3MSP.member')". It means that only members of org1 (LMO) and org3 can read or write data to the *private state database*.

6.2.2 Exemplary Unit Test

All unit tests are executed in chronological order within the same mocha script. The mocha script is structured in the following way:

import Before the tests can start, all the necessary imports have to be defined. The following are worth mentioning:

— expect from the *assertion library* chai[2] which makes it possible to compare 'expected' results with 'actual' results. Its chainable language makes it easy to construct assertions with good readability. An assertion could look like this, for instance: expect(placedBids).to.be.an('array').lengthOf(2).
— FabricControllerAdapter, a utility class which allows to communicate with a Hyperledger Fabric implementation through the network. It involves identity of the request, network topology resolution, and cryptographic administration. It is part of the *convector* framework.
— ClientFactory, allows to reuse the logic and structure in the chaincode in the *client application* (backend server). It is part of the *convector* framework.
— EnergymarketController, the *controller* as described in Figure 5.10.
— Models, all *participant* and *asset models* as described in Figure 5.10.

describe The function in which all tests are defined—also called a *test suite*. The two parameters are: (1) a meaningful name describing the overall test; and (2) the function which contains one or more tests—also called *test cases*. Inside this describe block, some global variables are defined which each *test case* needs to have access to.

[2]See https://www.chaijs.com (last accessed September 17, 2019).

before Runs before all tests in this describe block. In this case, the ClientFactory is used to create multiple *client applications*—each using a different FabricControllerAdapter and different cryptographic material to connect to the network. This way, multiple actors are created which can invoke transactions via their respective *client application*. For instance, invoking the createAuction transaction as 'user1' of org1 can be done by calling energymarketCtrl.org1.createAuction() and passing a valid Auction instance.

it A function which describes a specific *test case*. Again, it takes two parameters: (1) the name of the test case; and (2) the function which holds the body of the test. In this case, a total of 13 different unit tests are formulated within the defined block—each within an it block. A simple test for illustration purposes could look like this:

```
it('should create an energy auction', async () => {
  auction = new Auction({
    id: 'AUC1',
    start: Date.now(),
    end: ( Date.now() + market.auctionTime )
  });
  await energymarketCtrl.org1.createAuction(auction);
  let savedAuction = await energymarketCtrl.org1.getAuctionById(auction.id);
  expect(savedAuction.id).to.eql(auction.id);
});
```

The name of the test describes the 'expected' behaviour: *should create an energy auction*. In the test, a new object of type Auction is created which is then passed as parameter to the createAuction() function. One can see that energymarketCtrl.org1 is making the call which in this scenario is equivalent to the *client application* of the LMO. This test also makes use of the auxiliary 'query' function described above in order to retrieve the newly created Auction instance from the *world state database*. Finally, the expect() method is used to compare the 'initial' auction object with the response from the blockchain network.

With all this in place, it is now possible to simulate different scenarios with a substantial amount of flexibility. From within the same script, it is now possible to: (1) create objects using the same *participant* and *asset models* defined in Figure 5.10; (2) invoke all transactions defined in the *controller* of the UML class diagram, plus some additional 'query' transactions which return specific data stored in the CouchDB

databases; (3) invoke these transactions via different *client applications*—each representing a different member belonging to one of the four exemplary organisations; (4) define multiple unit tests which each test a certain behaviour of the *chaincode* implementation. This behaviour is evaluated by comparing the test results with some predefined values or events via the expect() method.

After running the mocha script, a summary of all the unit tests is returned, indicating which tests 'passed' and which 'failed'. For readability, all tests were designed so that they are expected to 'pass'. If a transaction call is expected to fail, for instance, the expect() call anticipates the transaction rejection. An exemplary assertion for this scenario could look like:

```
await expect(energymarketCtrl.org4.placeBid()).to.be.rejectedWith(Error);
```

This chapter discussed the implementation of the DApp design (Section 6.1) as well as the testing environment (Section 6.2) which was built to evaluate the proposed artefact. The testing scenario and expected behaviour for each of the 13 unit tests are explained in the next chapter in Section 7.1.

Chapter 7

Evaluation and Discussion

This last chapter is split into three sections. Section 7.1 evaluates the proposed Decentralised Application (DApp) model presented in Chapter 5 on the basis of the implementation discussed in Chapter 6 and data from literature. Following the Design Science Research (DSR) methodology described in Section 1.3, this evaluation assesses if the proposed solution satisfies the design objectives and requirements formulated in Section 5.1. Section 7.2 discusses the results of the evaluation; and finally, Section 7.3 presents some promising starting points for future research.

7.1 Evaluation of the Proposed Application Design

This section goes through all 24 requirements formulated in Section 5.1. The *functional requirements* ($\mathbf{R}_{func}$) are evaluated on the basis of unit tests which simulate a specific scenario in order to validate a certain functionality. The exact testing environment and approach is explained in Section 6.2. As the DApp could not be deployed in a 'real-life' environment, the fulfilment of some *security and privacy requirements* ($\mathbf{R}_{sec}$) and *blockchain-specific requirements* ($\mathbf{R}_{block}$) can not be answered conclusively or only by means of secondary data from literature. All unit tests can be found in the code accompanying this thesis or on GitHub[1]

$R_{func}1$: Multiple Order Placement

In the unit test 1, a market participant PAR1 of type prosumer places multiple buy and sell orders for the same auction period AUC1 by invoking the transactions placeBid, sendBidPrivateDetails, placeAsk, and sendAskPrivateDetails. Querying the ledger

[1] See https://github.com/raphmc/thesis.

shows that all orders are successfully stored in the world state database. $\mathbf{R}_{func}$**1: positive**.

$R_{func}2$: Sealed Orders

In the unit test 2, PAR1 and PAR2 both try to access the private details of the buy and sell orders, placed in unit test 1, by invoking the transactions getBidPrivateDetails and getAskPrivateDetails for the auction period AUC1. PAR2's transaction is rejected. This scenario can be underlined when comparing the ledgers of the three actors PAR1, PAR2, and MKT (representing the Local Market Operator (LMO)): the CouchDB databases of PAR1 and MKT both contain the private order details, which is not the case for PAR2. $\mathbf{R}_{func}$**2: positive**.

$R_{func}3$: Smart Meter Readings

In the unit test 3, PAR1 submits its 'actual' energy usage by invoking the transaction sendReading. Querying the ledger shows that the SmartMeterReading was successfully added to the array of readings of PAR1. $\mathbf{R}_{func}$**3: positive**.

$R_{func}4$: Incentives for Accurate Bidding

In the unit test 4, 4 market participants each place 2 orders for auction period AUC2. The market is cleared, a uniform Market Clearing Price (MCP) is calculated, and all orders are tagged as *successful* or *unsuccessful*. All market participants then submit their actual energy consumption and production (which differs up to 10% from their total predicted consumption or production). The market is settled and the coinBalance and energyBalance of all market participants is updated. The test shows that the deviations from the initial orders are penalised relative to the gridSellPrice or gridBuyPrice, creating an incentive for market participants to place accurate orders. $\mathbf{R}_{func}$**5: positive**.

$R_{func}5$: Internal and External Settlement

The same unit test 4 as for $\mathbf{R}_{func}\mathbf{4}$ shows that both internal market trades and external trades (with the public grid) are settled in the same transaction. $\mathbf{R}_{func}$**5: positive**.

$R_{func}6$: Transparent Market Clearing

The unit test 5 follows the same steps as unit test 4 for a third auction period AU3. The results of the clearAuction transaction are stored in a variable res. The test shows that the hashes of the orders' private details are identical with the hashes stored in res. $\mathbf{R}_{func}$**6: positive**.

$R_{func}7$: Adjustable Auction Times

The unit test 6 creates two sets of auctions. In the first set all Auction instances have a duration of 900 000 milliseconds (15 minutes) and in the second set the duration is set to 60 000 milliseconds (1 minute). Querying the ledger shows that all instances are successfully stored in the world state database. $\mathbf{R}_{func}$**7: positive**.

$R_{func}8$: Chronology of Events

In the unit test 7, a market participant tries to place an order for AUC1 which is already closed. The transaction is rejected. $\mathbf{R}_{func}$**8a: positive**.

In the unit test 8, a new Auction instance AUC6 is created and then immediately the clearAuction transaction is invoked. The transaction is rejected because AUC4 is still *open*. $\mathbf{R}_{func}$**8b: positive**.

The unit test 9 follows the same steps as unit test 4 but for the previously created auction period AUC6. However, only 3 out of the 4 market participants submit their smart meter readings. As expected, when the settleAuction transaction is invoked, it is rejected. $\mathbf{R}_{func}$**8c: positive**.

$R_{func}9$: Access Rights

In the unit test 10, a market participant PAR1 tries to submit a buy order in the name of PAR2. As expected, the placeBid transaction is rejected because the invoking party's identity does not match an identity of PAR2. $\mathbf{R}_{func}$**9a: positive**.

In the unit test 11, a market participant PAR1 tries to send a SmartMeterReading in the name of PAR2. However, the smart contract code does not allow a participant to store a SmartMeterReading as another participant and always stores it in the invoking party's instance, i. e. PAR1. $\mathbf{R}_{func}$**9b: positive**.

In the unit test 12, a market participant PAR1 tries to invoke the clearAuction transaction. As expected, the transaction is rejected because the invoking party's identity does not match an identity linked to the LMO. $\mathbf{R}_{func}$**9c: positive**.

O1: Identity Management

The current *participant models* all include the msp and and array of identities linked to the respective participant. This design allows for granular access rights definitions as shown by $\mathbf{R}_{func}$**9** but also makes it possible to link multiple identities to a single *business actor*. In case of a security breach, the compromised identity could be removed from the array of 'trusted' identities, the market participant could be provided a new certificate from the Certificate Authority (CA), and this new identity could be added to the above array. Of course, this procedure must be supervised by the LMO and protected by chaincode logic. This level of administration was not implemented as it is not part of the core use case of *community-based energy trading* and can thus not be tested through a unit test. However, as the *chaincode model* shows, complex *identity management* can be **guaranteed**.

O2: Modular Market Design

The solution presented in Chapter 6 has implemented a sealed bidding, discrete time double auction with uniform pricing. This market mechanism manifests itself mainly in the two transactions clearAuction and settleAuction. *Discriminatory pricing* could be achieved by using the bidding prices of the orders in the settleAuction transaction instead of a MCP. A *continuous auction* could be implemented by reducing the number of Auction instances to 1 and by triggering an adapted clearAuction transaction whenever a new buy or sell order has been placed. If desired, these two transactions could be altered in order to implement other market matching and/or pricing mechanisms. *Open-outcry bidding* could be achieved by removing the private transactions sendBidPrivateDetails and sendAskPrivateDetails and including the private details in the placeBid and placeAsk transactions. It can thus be said that the *architecture model* is fully portable and that the *chaincode model* would need only minor changes to support a new market design. In the eyes of the author, this means that *market design modularity* can be **guaranteed**.

O3: Platform Extensibility

The whole *community-based electricity trading* use case takes place in one Hyperledger Fabric channel. As mentioned before, Hyperledger Fabric allows for the creation of multiple channels (see Section 4.1) and one member can simultaneously be part of multiple channels without the need to add additional blockchain infrastructure. Cross-channel transactions (see Section 5.6 under under *Participants*) make it possible to connect use cases with each other by making transactions in one channel depend on the ledger state of another channel—or channels, for instance. An actor who is only part of one channel has no visibility of the transactions happening in the other channel(s). Further, the *chaincode model* is designed in a way which facilitates extensibility, separating participants, assets, and transactions into *participant models*, *asset models*, and a *controller*. This design makes it uncomplicated to add new assets to a given use case[2] or introduce a new actor[3]. It can thus be concluded that *platform extensibility* can be **guaranteed**.

$R_{sec}1$: Entity Authentication

As mentioned in Section 4.1, *identity management* is one of the key functionalities of Hyperledger Fabric. The CAs provide verifiable identities to all network participants. The Membership Service Provider (MSP) manages which identities are part of the energy consortium and maps these identities to organisations. All end-users, client applications, and peers have their own identity in order to sign transactions or messages, this way *entity authentication* can be **guaranteed**.

$R_{sec}2$: Message Authenticity

Each identity in Hyperledger Fabric is provided with a private and a public key—so-called *assymetric cryptography*. All transactions or messages sent in the network are signed with the private key of the sending party. All receiving entities can then verify whether the message was tampered with while 'in transit'. Hence, *message authenticity* can be **guaranteed** for all messages sent between entities of the blockchain network as well as the client application connected to a channel (see Figure 5.6 for a detailed overview). As specified in **A2** of Section 5.2, all smart meters are assumed to hold their own private/public key pair and be secured from tampering by

[2] E. g. trading heat in the same community or creating certificates of origin on the basis of the produced electricity.

[3] E. g. include the local grid operator or an external energy service company.

protected module architecture. More research is need to investigate the integration of the Public Key Infrastructure (PKI) already in place for smart meters with a decentralised energy trading marketplace running on Hyperledger Fabric. However, this goes beyond the scope of this thesis.

$R_{sec}3$: Authorisation

As explained in the *chaincode model* described in Section 5.6, all actors of the Local Electricity Market (LEM) are represented as *participant models* in the chaincode. All participant models hold a list of identities that are linked to this particular participant. This means that, in addition to low-level security features like *entity authentication* and *message authenticity* provided by core Hyperledger Fabric, granular access rights can be hard-coded in the chaincode logic. This functionality was evaluated **positive** in $\mathbf{R}_{func}\mathbf{8}$ by means of three unit tests 10, 11, and 12. Subsequently, it can be said that suitable access rights that ensure *authorisation* can be **guaranteed** by the chaincode logic.

$R_{sec}4$: Confidentiality

Being a permissioned blockchain system, Hyperledger Fabric provides substantial *confidentiality* features. The *network policy* controls who has access to the blockchain network and the *channel policy* provides the same control on an even more granular channel level. As described in Section 4.1, inside the same blockchain network, transactions and data can be kept hidden from unwanted eyes through the creation of multiple *channels* or by means of *private transactions*. The DApp design presented in Chapter 5, makes use of the latter to keep private bid and private ask data hidden from other market participants. This feature was evaluated **positive** in $\mathbf{R}_{func}\mathbf{2}$ on the basis of unit test 1. Subsequently, by relying on a permissioned blockchain system and through the use of private transaction, *confidentiality* can be **guaranteed** for this DApp design.

$R_{sec}5$: User Privacy-preservation

In the *chaincode model* presented in Chapter 5, market participants are only identified by their id which can be seen as a *pseudonym*. However, for real *privacy-preservation*, a new *pseudonym* would have to be used for every trading period in combination with *anonymous signatures* in order to ensure *non-linkability* [40]. This

level of privacy is currently not supported in the proposed DApp model. While hiding the 'real' identities of market participants from other market participants can be achieved, the LMO will always have full disclosure of information. As a consequence, the current solution **cannot guarantee** full *user privacy-preservation*.

$R_{sec}6$: Non-repudiation

The same *assymetric cryptography* employed by Hyperledger Fabric which also guarantees *message authenticity*, ensures that *non-repudiation* of transactions and messages can be **guaranteed**. Every message is signed with a private key which makes it impossible for a sending party to start a dispute over its origin.

$R_{sec}7$: Availability

The *availability* of the DApp will depend on the particular configuration of the blockchain network which itself is highly dependent on the local circumstances. As described in Chapter 4, Hyperledger Fabric is being designed by well-known actors in the field of Information and Communication Technology (ICT) and tailored towards the enterprise world. An organisation which is part of a blockchain network can own a multitude of peers in order to increase its availability. Studies carried out by [60, 94] have further shown that high transaction speeds can be achieved with the correct set-up. It can thus be assumed that a highly available system can be deployed without any problems. However, additional research would have to analyse the performance and the resiliency of this DApp in 'real' conditions. Hence, **no final answer** can be given with regard to the requirement of *availability*.

$R_{block}1$: Smart contract capability

As explained in Chapter 4 and demonstrated in the implementation presented in Chapter 6, Hyperledger Fabric supports the deployment of smart contract code in the form of *chaincode* which is installed on all endorsing peers in a channel. As shown with the *chaincode model* presented in Section 5.6, this logic layer can be used to build complex and secure DApps. The fulfilment of this requirement can thus be **guaranteed**.

$R_{block}2$: Access rights

This requirement proposed by [46] is equivalent to the requirement of *authorisation* formulated by [40]. As explained in $\mathbf{R}_{sec}\mathbf{3}$, the enforcement of *access rights* can be **guaranteed**.

$R_{block}3$: Consensus mechanism

As discussed in Section 2.2, a consensus mechanism like Proof-of-Work (PoW), currently used by frameworks like Bitcoin or Ethereum, would not be sensible in the context of a LEM, due to its high power consumption. Hyperledger Fabric provides 'pluggable' consensus mechanisms—currently, two mechanisms providing Crash Fault Tolerance (CFT), namely Raft and Kafka, as well as a protocol called solo, for development purposes only. As discussed in Subsection 4.2.1, Hyperledger Fabric developers are currently working on releasing a Byzantine Fault Tolerance (BFT) consensus mechanism which would increase the resiliency of the *ordering service* against malicious attacks. However, the fulfilment of requirements $\mathbf{R}_{sec}\mathbf{1}$, $\mathbf{R}_{sec}\mathbf{2}$, and $\mathbf{R}_{sec}\mathbf{3}$, and the fact that identities are all known by the LMO, means that the risk for such attacks is already minimised. Another point in favour of Hyperledger Fabric is that consensus is achieved without relying on any form of cryptocurrency mining which would not be sensible in a local energy community. Even though it can be assumed that at least one of the consensus protocols provided by the Hyperledger Fabric framework is suitable for this community-based energy market use case, **no final answer** can be given without testing its performance in 'real-life' conditions.

$R_{block}4$: Throughput

Due to the lack of first-hand performance data, the findings presented by [58] are used as orientation. [58] analysed the performance of Hyperledger Fabric version 1.4 under different conditions and produced the following results: in a set-up of 128 peers and 325 channels, the researchers measured transaction throughputs over 10 000 Transactions Per Second (TPS) (maximum of 13 000 TPS). Interestingly, the throughput went up with the number of peers and channels in the blockchain network. For comparison, a set-up of 1 channel and 2 peers only yields throughputs in the range of 1000 TPS.

Let's assume the proposed decentralised marketplace is deployed on a network capable of operating at a conservative level of 1000 TPS. If trading periods are set to 15 minutes (= 900 seconds), the system could process a total number of 900 000 transactions per trading period. Let's further assume that all market participants $n_{\text{participants}}$ send a total of x_{orders} orders per trading period. To this number, the smart meter readings for the last trading period have to be added. Given a buy or sell order consists of two transactions—one public and private—the maximum number of transactions for one trading period equals:

$$max_{\text{transactions}} = 2(n_{\text{participants}} \cdot x_{\text{orders}}) + n_{\text{participants}} \tag{7.1}$$

If the maximum number of processable transactions $max_{\text{transactions}}$ equals 900 000, as assumed on the basis of [58], the equation can be rearranged to:

$$n_{\text{participants}} = \frac{max_{\text{transactions}}}{2x_{\text{orders}} + 1} = \frac{90\,0000}{2x_{\text{orders}} + 1} \tag{7.2}$$

This means, even if each participant sends a total of $x_{\text{orders}} = 10$ orders per trading period of 15 minutes, the system could handle over 40 000 participants. Even if the duration of a trading period is reduced to 1 minute, the system could sustain over 2 500 market participants. This means that the maximum number of peers that can participate in the network—also called horizontal scalability—seems to be the limiting metric worth analysing in future research. Unfortunately, no numbers could be found in this regard for Hyperledger Fabric version 1.4.

Conclusively, first-hand performance data is needed in order to fully validate this requirement. But, based on [58], it **can be assumed** that a DApp running on Hyperledger Fabric provides a high enough transaction *throughput* to sustain a community-based LEM.

$R_{block}5$: Token or Coin

Unlike the previous four requirements defined by [46], this requirement is formulated as *should*-have and not *must*-have. The *chaincode model* proposed in Section 5.6, integrates a LEM-internal *coin* which is sent along with bid orders, refunded if the bid is not successful, and finally settled by the settleAuction transaction. This way, the DApp takes the role of an escrow agent and the solvency of all market participants can be assured. As explained in **A6**, it is assumed that a third party (e. g. a bank or the LMO) allows all consortium members to exchange 'real' money

into locally used coins. As elaborated in Section 5.6 under *Participants*, thanks to the possibility of cross-channel transactions in Hyperledger Fabric, these exchanges could take place in a different channel entirely. However, the implementation presented in Chapter 6 simplifies the coin transactions and enables market participants to have negative coinBalance values. Hence, a coin is introduced but without all the safety features which would be necessary in a 'real-life' environment. The reasoning behind this decision is that, with the introduction of a *FabToken* which has been announced with the release of Hyperledger Fabric version 2.0 [200], coin transactions will be supported out-of-the-box. The author thus decided not to implement a design which will be obsolete in a few months. However, the unit test 4 demonstrates that the settleAuction transaction successfully resolves all coin movements on the back of smart meter readings and the results from the clearAuction transaction. It can thus be **guaranteed** that financial transactions within an energy community can be mapped through the use of a local *coin* on the basis of the proposed platform design.

7.2 Discussion of Results

This section discusses the proposed DApp design. The results of the evaluation carried out in Section 7.1 are summarised in Table 7.1. Critical aspects like *privacy* and *interoperability* are discussed in greater detail.

Functional Requirements

As Table 7.1 shows, all *functional requirements* defined in Section 5.1 are satisfied. Consequently, the *chaincode model* proposed in Section 5.6 can be fully validated. However, $\mathbf{R}_{func}\mathbf{1}$ should be reevaluated as to limit the number of orders a market participant can place per trading period; this way, spamming can be prevented and the ICT infrastructure necessary to guarantee system *availability* ($\mathbf{R}_{sec}\mathbf{7}$) can be better predicted.

The fulfilment of requirement $\mathbf{R}_{func}\mathbf{2}$ (*sealed bidding*) is of particular importance, because it means that all types of auction mechanisms can be implemented, despite relying on a blockchain-based system. For instance, strategy-proof mechanisms like the *Vickrey-Clarke-Groves (VCG) auction* could reduce the risk of market manipulation in LEMs (see Section 3.1.2 under *Auction-Based Market Designs* and *Deductions*). This is of particular importance when both bigger and smaller players

Table 7.1: *Overview of evaluation results for all 24 requirements and design objectives*

Requirements & objectives	Positive or guaranteed	Can be assumed	No final answer	Cannot be guaranteed
$R_{func}1$	•			
$R_{func}2$	•			
$R_{func}3$	•			
$R_{func}4$	•			
$R_{func}5$	•			
$R_{func}6$	•			
$R_{func}7$	•			
$R_{func}8$	•			
$R_{func}9$	•			
O1	•			
O2	•			
O3	•			
$R_{sec}1$	•			
$R_{sec}2$	•			
$R_{sec}3$	•			
$R_{sec}4$	•			
$R_{sec}5$				•
$R_{sec}6$	•			
$R_{sec}7$			•	
$R_{block}1$	•			
$R_{block}2$	•			
$R_{block}3$			•	
$R_{block}4$		•		
$R_{block}5$	•			

are present on the same market. The implemented DApp has further proven to successfully enforce the *chronology of events* ($\mathbf{R}_{func}\mathbf{8}$) as well as granular *access rights* ($\mathbf{R}_{func}\mathbf{9}$). Both create a well-defined possibility space for all business actors within a local community. This 'hard-coded trust' reduces agency risks and allows for automated settlement of financial transactions on the basis of smart contracts.

In its current form, the DApp design proposes a solely virtual marketplace without considering the physical constraints. Integrating grid operators into the platform is an important next step. **O3** has shown that the platform is extendable in multiple

ways. Hyperledger Fabric's *channels* in combination with *private* and *cross-channel transactions* make it possible to develop complex system designs. And most importantly, further use cases can be connected to the platform without current consortium members needing to install additional blockchain infrastructure.

It is necessary to stress that the current version of the proposed design is geared towards smaller-scale market participants. Large industry consumers, for instance, would probably not profit from participating in a fully auction-based LEM with only one external Energy Service Provider (ESP). Their disproportionate demand would likely lead to a convergence of ask prices and grid buy prices. However, as shown by **O2**, the *architecture model* as well as the *participant models* of the *chaincode model* provide a solid foundation for the implementation of new market designs. The author of this thesis is convinced that there is no encompassing solution that fits all LEM scenarios and that the 'best' market design for a given local community has to be chosen and developed on a case-by-case basis, taking into account the particular stakeholders and other project-specific factors. Hybrid LEM designs (see Section 3.1.1 under *Hybrid market*) could provide big actors with the level of flexibility they need to fully profit from both community-internal and external Peer-to-Peer (P2P) trade and should be investigated in further research.

A design feature in line with bigger actors is the possibility for flexible identity management shown in **O1**. The representation of business actors in the chaincode and the option to link multiple *identities* to one business actor makes it possible to define granular access rights, even for members within a single organisation. And it further enables the LMO to react to problems like security breaches or the loss of a market participant's private key. The author of this thesis is convinced that the ease-of-use of blockchain-based system must be on par with existent centralised solutions for wide-spread adoption to take place. In this regard, a certain level of flexibility—supervised by chaincode logic—is indispensable.

Security & Privacy

Many *security and privacy-related requirements* formulated by [40] are addressed by core functionalities of a permissioned blockchain framework like Hyperledger Fabric (e. g. asymmetric message encryption or consortium member administration). System *availability* ($\mathbf{R}_{sec}\mathbf{7}$) would need to be tested in 'real' conditions in order to collect data of the DApp performance first-hand. Due to limitations with regard to time and access to infrastructure, the *demonstration* of the proposed DApp could not be

realised in a representative environment. Field tests are thus the logical next step and should be part of any further research.

A requirement which the proposed solution does not fully satisfy is *user privacy-preservation* ($\mathbf{R}_{sec}\mathbf{5}$). In the current design, for instance, smart meter data is visible to all energy consortium members. One might wonder why the sendReading transaction is not designed as a private transaction based on the sealed order transactions model. Even though this would be perfectly feasible, all the benefit would be lost after invoking the settleAuction transaction, since energy consumption and production data could be fully derived from the flow of coins. It is, however, necessary to record unencrypted financial flows on the blockchain in order to guarantee that all members of the consortium have equal information on the state of the system. Otherwise, malicious agents could use the information asymmetry to manipulate the system in their favour.

This problem is best described as a trade-off between privacy and decentralisation. More privacy means less actors have access to certain data which goes against the principle of full decentralisation employed by public blockchains like Bitcoin or Ethereum. However, if blockchain-based marketplaces for LEMs are to experience adoption in the real world, they must come with a clear value proposition for the end user. The author of this thesis is convinced that privacy is of high value and that blockchain-based systems can only be successful if they respect it. Using private transactions to enable *sealed bidding* is a first step in the right direction, but is in no way sufficient. Future work should investigate mechanisms to also keep other data confidential.

Promising approaches which could provide more privacy for participants of LEMs are, for example, the use of *zero-knowledge proofs*[4], *homomorphic encryption*[5], or *ring signatures*[6]. In a permissioned blockchain network where the identities of the consortium members have to be verified, zero-knowledge proofs [202] and ring signatures [204] can be used to preserve an actor's *anonymity* while guaranteeing the trustworthiness of the respective actor. Zero-knowledge proofs could also be used to verify the authenticity of energy usage data [205–207]. Homomorphic encryption

[4] *Zero-knowledge proof* is a method that allows one party (prover) to prove to another party (verifier) that they know a certain value without having to disclose the given value [202]. This protocol can, for example, be used to verify an identity without the verifying party knowing who the identity belongs to.

[5] *Homomorphic encryption* transforms unencrypted data (plaintext) into encrypted data (cyphertext) that can still be worked with as if it was in its unencrypted form. This way, complex operations can be carried out on encrypted data without compromising the encryption [203].

[6] *Ring signatures* are used in cryptography to digitally sign messages as a group of people (which all have keys) without disclosing the identity of the signee [204].

is discussed as means to perform computations—market clearing or settlement, for instance—on smart meter data without compromising their privacy [203, 208, 209].

Another approach which could increase privacy in community-based LEMs, is the aggregation of market participants into small *prosumer pools*. This way, end-user bidding and smart meter data could be anonymised and the omniscience of the LMO would be constrained. The proposed DApp design could be extended in the following way to support this change: multiple *client applications*—each belonging to another end-user—could be connected to one *peer node* which is operated by the aggregator; both end-user buy/sell orders and smart meter data are sent as private transactions and only stored in the aggregator's ledger; the aggregator then participates in the market in the same way a prosumer would in the current design—except that, after market clearing and settlement, an additional step of internal settlement would be necessary. While this approach looks promising and would be easily compatible with the current design, further research is necessary in order to correctly assess the potential risks and rewards associated with the solution.

From a qualitative point of view, Hyperledger Fabric looks like it could satisfy all *blockchain-specific requirements*. However, as for $\mathbf{R}_{sec}\mathbf{7}$, more field research is necessary to fully prove the fitness of the framework with regard to performance-related requirements like $\mathbf{R}_{block}\mathbf{3}$ and $\mathbf{R}_{block}\mathbf{4}$. A few features which make Hyperledger Fabric stand out in comparison to Ethereum are: (1) the option to write smart contract code in *general purpose programming languages*, which makes it possible to use already existing, security-proven libraries and rely on a wide developer base; (2) the notion of *channels* as well as *private* and *cross-channel transactions*, which provide tools for privacy-preserving system designs; (3) strong *identity management*, which guarantees interoperability with existent ICT infrastructure and facilitates the formulation of granular access rights.

Coin & Blockchain Interoperability

A challenge during the DApp design process was the representation of a token. With Ethereum, ERC token standards provide system designers with a number of pre-defined token contracts. A token is implemented as one contract holding all token holders' balances—one instance maintaining the token state. Applying this same approach to Hyperledger Fabric would, however, increase the risk of invalid transactions since the state of the token—stored as a key/value pair on the ledger—would be updated with every token-moving transaction. If two of these transactions are

packed into the same block, the second one will be tagged 'invalid' since the state of the token asset will already have been updated by the first transaction (see Subsection 4.2.1). To counter this issue, token balances are stored directly in the *participant models*. This design makes it possible to have two parallel token-moving transactions as long as no party involved in the first transaction is also part of the second transaction[7]. However, token transactions will be simplified with the release of Hyperledger Fabric version 2.0 which will have native token support in the form of a FabToken [200].

A feature which is available since version 1.3 is the support for *Ethereum Virtual Machine (EVM) bytecode smart contracts* and the corresponding *web3 provider* [210]. This essentially means that Ethereum-based DApps can be deployed on permissioned Hyperledger Fabric infrastructure. Hence, if Ethereum-based smart contracts prove to be a better fit for P2P energy trading use cases, they could still be deployed on a Hyperledger Fabric blockchain network following the proposed *architecture model*.

As pointed out in Subsection 2.2.6, the interoperability of blockchains is agreed to be of major importance in order to foster the adoption of the technology [36, 113]. Otherwise, isolated token-based economies are created with coins that hold value only within the closed system. $\mathbf{R}_{block}\mathbf{5}$ formulated by [46] states that a LEM trading platform should implement a token or coin *"which is used as a currency for trading energy"*. If the blockchain platform is isolated, the implemented coin mainly serves two purposes: (1) tracing the *flow of money* within the local market, facilitating accountability; and (2), if a positive coin balance is needed in order to trade, guaranteeing the *solvency* of market participants which increases security and creates trust. However, if, as pointed out in Section 5.6, a bank or the LMO is responsible for the exchange of 'real' money into locally used coins, the system is not all that decentralised. One could argue that it resembles a pre-paid system where market participants have to 'fill up' their balances before they can transact in the market. The author of this thesis fears that such a pre-paid token system would increase the complexity for end users with comparably little benefit, i. e. only guaranteeing (2): solvency. In this case, it might be easier to rely on a traditional 'invoice-based' settlement between market participants and the LMO. A similar approach is proposed by [40] and should be considered if this approach is further investigated.

[7] Since in this *chaincode model* only the settleAuction transaction actually updates the market participants' coinBalances, this problem would not arise due to the intervals between the auction periods. However, with respect to *platform extensibility* formulated in **O3**, the more scalable solution was chosen.

A better solution would be to connect the LEM trading platform to a *"disparate ecosystem of blockchains"* [211]. This way, a real 'token economy' is created [78]. As mentioned in Subsection 2.2.6, *Polkadot* [114] and *Cøsmos* [115] are two protocols which are currently working on creating a 'network of blockchains'. Blockchains worth considering are:

1. **Bitcoin** [26], the first blockchain with the highest market capitalisation (as of today [86]). It could serve as value exchange for participants of a LEM;
2. **Ethereum** [33], the platform with the highest number of DApps as of today [212]. The connection to this ecosystem could provide market participant with access to other services and serve as means of exchanging value between the local market and the 'outside world';
3. **Energy Web Chain** [213], a consortium blockchain for the energy sector, launched in June 2019. The connection to other energy-centred DApps could propel the connection with other use cases (e.g. Electrical Vehicle (EV) charging [214] or issuance of certificates of origin [105]);
4. **Libra** [215], a recently announced consortium blockchain promising a 'global currency' with stable value backed by a basket of fiat currencies and government securities. It could serve as 'stable' means of exchange of value between the local market and the 'outside world';
5. **ZCash** [216], an anonymous payment system. It could help conceal the flow of money between different systems and thus increase privacy for participants of a LEM.

However, most of the blockchains mentioned above are still in an early phase of development—and inter-blockchain transactions even more so [217]. The scientific community should thus focus on creating standards in order to facilitate communication between the different systems [36].

Blockchain Trilemma

In the eyes of the author, Hyperledger Fabric is well positioned to establish itself as go-to framework for private permissioned blockchain systems—in great part thanks to the high level of flexibility it provides to system developers (see Subsection 2.2.3). As readers of this thesis will have noticed, this high level of flexibility

entails an equally high level of complexity. A recurring challenge when designing a blockchain-based system for a given use case is to balance the three cornerstones of the *blockchain trilemma*—efficiency, security, and decentralisation—which can never all be satisfied at the same time [218]. In order to appropriately weight these characteristics, it is crucial to understand the subtleties of both the use case and the underlying blockchain framework. This thesis deconstructed the community-based energy trading use case into its individual parts in order to understand the business goals (see Section 5.3) and functions (see Section 5.4) of each actor and developed an *architecture* (see Section 5.5) and a *chaincode model* (see Section 5.6) which fit the discovered findings. For instance, a bit of decentralisation was given up in order to conceal market participants' individual bidding strategy through the use of private transactions. This decision could only be made on the basis of a detailed analysis of the transaction flow for private transactions (see Figure 5.8), the access to Information Objects (IOs) for different system components (see Table 5.1), and given the special role of the LMO in this use case.

As pointed out in Section 3.1.1, the community-based market design is the most 'realistic' under current regulation; and as a consequence, this thesis chose to focus on this design. However, as it gravitates around the LMO, it is by nature a more centralised approach when compared to the full P2P design, for instance: even though the LMO's scope of action is restricted by the chaincode logic, they maintain the marketplace, aggregate market participants' data, and represent the connection point to the public grid. Consequently, the blockchain system reflects this level of centralisation and trades *decentralisation* for *security* and *efficiency*. In the eyes of the author, this makes sense in the current regulatory environment. The deployment of permissioned blockchains in private environments gives 'traditional' players in the energy sector the chance to gain first-hand experience with the new technology with little associated risk. This is important in an industry which provides fundamental services to society. However, as argued above when discussing *blockchain interoperability*, closed systems are limited in their reach. More *decentralised* approaches—in combination with a hybrid market design, for instance—should thus be part of further research. In this context, it could be worth looking into energy-centred consortium blockchains like the *Energy Web Chain* [213] mentioned above or *Exergy* [219] which is currently being developed by LO3 Energy, the same company operating the Brooklyn Microgrid.

Conclusively, it can be said that more research investigating the ICT layer of LEMs is needed, before secure, efficient, and privacy-preserving energy communities can

become a reality. The DApp model presented in this thesis does neither claim to be universal nor conclusive. But it contributes to the above research by demonstrating a systematic approach to system design in a decentralised environment and represents a good starting point for further research exploring the intricacies of privacy and performance in the ICT layer of LEMs. It thereby successfully answers the research question formulated in Section 1.2.

7.3 Further Work & Outlook

In the course of this chapter, a couple of points emerged which could not be addressed by this thesis. This last section shortly summarises the potential research openings deriving from this work:

Demonstration and field-testing of the proposed implementation in order to extract first-hand data. The **performance** of the DApp could be analysed by looking at the following metrics:

- *transaction throughput and latency*, in order to derive the minimum duration of one trading period and the maximum number of market participants.
- *vertical scalability*, to investigate if there is an upper limit with regard to the number of nodes Hyperledger Fabric can efficiently operate.
- *block speeds and sizes* have an impact on the previous two metrics and should be analysed in order to better estimate the maximum size of a local energy community.
- *computational complexity*, in order to derive the necessary resource to provide a seamless service.
- *energy usage*, to weight the energy consumption associated with operating the platform against its potential to create more sustainable local communities.
- *blockchain size*, to correctly estimate the need for storage capacity on peer nodes.
- *chaincode programming language*, to find out if DApp's performance is affected by the language choice.

A field demonstration is further needed to provide insight with respect to the **resiliency** of LEM operated on the basis of a decentralised platform. Some questions worth taking a closer look at are:

- *What happens if the whole system breaks down and how could that be prevented?*

— *What happens if isolated devices or network components break down?*
— *Can such a platform put in danger the security of supply?*

Finally, the **acceptance** amongst members of the public should be inquired. The barriers to participation for the average citizen have to be better understood. In this context, it is important to define a clear value proposition, backed by accurate data.

Privacy-preservation and security for participants of LEMs should be further explored. Generated data must be kept private in order to limit the profiling capabilities of data collectors, i. e. derive consumption patterns, presence profiles, or other behavioural patterns. In this context, it should be investigated how the PKI for smart meters, which is already in place, can be seamlessly integrated in the DApp so that their data cannot be fully trusted. Other promising approaches have been presented in Section 7.2.

Consider physical constraints and transmission losses in the decentralised marketplace and add the local grid operator(s) to the list of platform participants. The introduction of dynamic grid tariffs or other pricing system could be worth investigating, especially in combination with auction-based allocation of demand and supply. Price functions which take into account the state of network (such as congestion, voltage, or frequency) in close to real-time could also be an interesting path.

Include other use cases and energy carriers into the marketplace in the interest of integrated energy. Potential use cases which hold promise are:

— *Heating and cooling* which could be traded over the same marketplace [220] (potentially on a different channel).
— *Flexibility*, provided by stationary storage devices or electrical vehicles, could be incorporated into the platform. It should be investigated how mobile devices can be part of the blockchain architecture and what market designs are suitable for trading flexibility in a community-based LEM.
— *Auxiliary grid services* could be provided by the energy community to the 'traditional' energy system. The community could act as *virtual power plant* or participate in the market of *control energy* and provide ancillary services, for instance.

Energy management and trading systems for market participants of LEMs (see Section 5.5) should be investigated and automated agent strategies should

be developed. Machine-learning algorithms, as described in Subsection 3.1.3, could provide small-scale market participants with intelligent prediction tools which are necessary for efficient trading systems.

The business case and regulatory environment need to be examined carefully for all actors in the community-based energy trading use case. Interesting research directions include:

— *The role of the LMO*, a role which could, for example, be taken by public utilities which are searching for new business models in the face of digitalisation [221].
— *The connection to the public grid* with a special focus on the contractual relationship between the energy community and the external ESP.
— *The regulatory framework* remains a major barrier for the implementation of consumer-centric markets. Research should be directed both-ways: (1) *how can these markets see the light in the current regulatory environment*; and (2) *how does this environment need to evolve in order to promote new market structures*. For this, it could be interesting to compare different countries with each other.
— *Compare different P2P market topologies* as described in Subsection 3.1.1. A hybrid design could provide more freedom to market participants with respect to their individual energy preferences and facilitate the integration of big players into local energy communities. In this context, energy-centred consortium blockchains, like the *Energy Web Chain* or *Exergy* mentioned above, should be considered as decentralised infrastructure on which to deploy energy trading marketplaces.

Although the blockchain technology has received a lot of attention lately, it is important to look beyond the promises of this new technology and make sure it solves a real problem in each given scenario—and ideally, it should solve this problem in a more effective way than any centralised solution. This thesis proposes a DApp model for an energy trading marketplace in community-based LEMs; but the actual implementation of such marketplaces is a task for start-ups and industry. Only they can decide whether or not this technology provides tangible benefits to their businesses. If this work helps answer this question and contributes to the discussion around decentralisation and privacy in consumer-centric energy communities, it can be called successful.

Conclusion

This thesis proposed a Decentralised Application (DApp) design for a community-centred energy marketplace, based on the permissioned blockchain framework Hyperledger Fabric. The design was developed upon a procedural approach inspired by the *standards-based architecture modelling framework* [64] employing Domain Specific Language (DSL) language in line with the Smart Grid Architecture Model (SGAM) [65]: a *business analysis* was carried out, identifying the use case-specific business actors, their respective business goals, and the High-Level Use Cases (HLUCs); and a *functional analysis* was realised, mapping business actors to logical actors and identifying the Information Objects (IOs) associated with each of the latter. Two models were derived from these analyses: (1) an *architecture model*, describing the technical and software components for each business actor—with a special focus on the necessary blockchain network to support a community-based Local Electricity Market (LEM)—as well as the information flow between these components; and (2) a *chaincode model*, the core component within the DApp architecture defining the assets which reside on the blockchain and the transactional logic they are subject to.

In line with Design Science Research (DSR), the proposed solution was first implemented and then evaluated with respect to previously formulated objectives and requirements of three different natures—functional, security and privacy-oriented, and blockchain-specific (see Section 5.1). All functional requirements could be validated on the basis of unit tests simulating multiple agents in interaction with the DApp. An aspect worth highlighting is the possibility for market participants to place sealed buy and sell orders. This functionality opens the door to strategy-proof market designs, like the Vickrey-Clarke-Groves (VCG) auction, and thus has the potential to reduce the risk of bigger players in an energy community abusing their market power.

A security and privacy requirement which could not be fully satisfied is the preservation of user-privacy. In the presented design, energy and cash flows are visible to all

members of the blockchain consortium. This is in great part due to the consensus-based nature of Distributed Ledger Technologiess (DLTs) which complicates the implementation of both privacy and decentralisation. Ring signatures, zero-knowledge proofs, and homomorphic encryption were identified as promising starting points for increasing privacy in consumer-centric energy communities and should be further investigated in follow-up research.

Within the scope of this work, the DApp could not be deployed in a realistic environment. Subsequently, some blockchain-specific requirements around performance could not be answered conclusively. From a qualitative point of view, however, Hyperledger Fabric looks to be a promising framework for building energy-related DApps. Some key features that stand out in particular and differentiate it from a framework like Ethereum, for instance, are: (1) its *permissioned nature*, which makes it possible to restrict access to a service and which provides advanced identity management; (2) its support for multiple *channels* and *cross-channel transactions*, which facilitates the separation of concerns in a scenario with multiple interconnected use cases; and (3) *private transactions*, which enable members within a same channel to share confidential information with only a subset of selected peers. Further research should focus on gathering first-hand data from field testing and should focus on including physical constraints into the auction-based marketplace.

As the literature review confirmed, more research is needed in the field of blockchain-enabled, consumer-centric energy markets. This thesis contributes to this field of research by displaying a systematic approach to system design in a decentralised environment and by proposing a language-agnostic DApp model of a Peer-to-Peer (P2P) energy marketplace for community-based LEMs. The successful implementation of a sealed-bidding mechanism demonstrates that maintaining privacy in distributed systems is a feasible, yet complex endeavour. One can hope that, in doing so, this thesis represents a piece of the puzzle in the quest for fair, secure, and privacy-preserving local energy communities.

Appendix A

Hyperledger Fabric Glossary

What follows is a non-exhaustive list of important terms in the Hyperledger Fabric ecosystem. These are based on the official documentation [176] and [222].

Anchor Peer *Anchor peers* are responsible for communication between organisations making *peers* in different organisations aware of each other.

Blocks A *block* is made up of a *header*, *block data* where the transactions are stored, and *block metadata* which contain information about the various nodes involved in the process of block creation.

Certificate Authorities In order for anybody to interact with the network, an identity is needed. The *Certificate Authority (CA)* provides a verifiable digital identity for all actors in the blockchain network (can be external). A *CA* is built into Hyperledger Fabric if needed.

Chaincode The *chaincode* holds the transaction logic, otherwise referred to as *smart contract*. Note that not every *peer* in a *channel* needs to have the *chaincode* installed.

Channel A *channel* is essentially an independent *ledger* of transactions shared among a group of network participants. Transactions in a *channel* are only visible to *channel* members. A Hyperledger Fabric blockchain network can have multiple *channels*.

Channel Policy A *channel* is governed by its members who are subjected to the *channel policy* describing the rules that govern the *channel*. It is not identical with the *network configuration*.

Consortium Literally means 'a group with a shared destiny'. In Hyperledger Fabric it represents a group of organisations that share a need to transact.

Committing Peer Every *peer* in the network is a *committing peer*. *Committing peers* all hold a copy of the *ledger* and the *world state*.

Endorsing Peer Has *chaincode* installed and can thus simulate and endorse transactions. Is also a *committing peer*.

Endorsement Policy Defines the rules and the number of organisations which must approve a transaction before it is ordered into a *block*. Each *chaincode* has its own *endorsement policy*.

Hash Function A *hash function* takes data of arbitrary length (numbers and letters) and maps it to an encrypted output of fixed length. The same input always yields the same output.

Hash A *hash* describes the result of data which was altered by a *hash function*. The *hash function* takes data of arbitrary size and turns it into data of fixed size.

Leader Peer Organisations can have multiple *peers* in one *channel*. *Blocks* need only be distributed to one *peer* per organisation. The *leader peer* disseminates the new *block* from the *ordering service* to all other *peers* of its organisation.

Ledger A *ledger* is an append-only file consisting of two parts: (1) a blockchain, holding all past transactions inside *blocks*; and (2) the *world state*, a database created based on the validated and committed transactions in the *blockchain* containing the latest state of all business objects.

Membership Service Provider The *Membership Service Provider (MSP)* is a trusted authority in the blockchain network. It is responsible for identifying which roles different actors inside an organisation can play in the blockchain network. New *nodes* can only join the network through the *MSP*. Therefore, it verifies the identities issued by the *CA*[1] are trusted by the network.

Network Policy Determine which organisations (or actors inside organisations) have control over the *network configuration*.

Ordering Nodes Is similar to a network administration point. The *nodes* in the *ordering service* act as support for the *application channels* and order the

[1]There can also be a chain of trust from a *Root CA* over one or more *Intermediate CAs*

endorsed transactions into *blocks*. Thereby, they ensure that all *committing peers* have an identical copy of the *ledger(s)*.

Peer Nodes Every *peer* in Hyperledger Fabric is a *node* in the network and holds a copy of the *ledger* for every *channel* it is a member of. There are two types of peers: *committing peers* and *endorsing peers*.

Private Data Collection These collections are used to keep (part of) the data in a transaction confidential between a subset of organisations. This *private data* is stored separately from the *channel ledger* in its own private state database.

Public Key Infrastructure The *Public Key Infrastructure (PKI)* provides the means for secure communication in a blockchain network. The digital certificates issued by the CAs are used to sign and decrypt messages. The *PKI* issues a list of identities which are then mapped to the different organisations by the *MSP*.

System Chaincode Is installed on every *peer* and defines the operating parameters for the entire *channel*.

Transaction Proposal A request to invoke a *chaincode* function sent by a *client application* to a number of *endorsing peers*.

World State Represents a snapshot of the current *state* of all *assets* or objects in the network. It is stored in a separate (usually a graph) database.

Bibliography

[1] FridaysForFuture: *Greta Thunberg Addressed the COP24 Plenary Session December 12 2018*. https://www.fridaysforfuture.org/greta-speeches, 2019

[2] Waldholz, Rachel: Students Demand Climate Action with "Fridays for Future" School Strikes. In: *Clean Energy Wire* (2019), Februar

[3] Comment, Full: *Andrew Coyne: The Rise of the Greens, and the Danger It Poses to the Liberals | National Post*. Juni 2019

[4] Abidor, Mitchell: What's Left of the Left? The European Elections and the Rise of the Greens. (2019), Juni. – ISSN 0015–7120

[5] Bundesministerium für Umwelt, Naturschutz und nukleare Sicherheit: Klimaschutzbericht 2018 zum Aktionsprogramm Klimaschutz 2020 der Bundesregierung. (2018), S. 174

[6] Oe, Pao-Yu ; Göke, Leonard ; Kemfert, Claudia ; Kendziorski, Mario ; von Hirschhausen, Christian: *Erneuerbare Energien Als Schlüssel Für Das Erreichen Der Klimaschutzziele Im Stromsektor*. Deutsches Institut für Wirtschaftsforschung, 2019. – ISBN 978–3–946417–25–5

[7] United Nations Environment Programme: *Emissions Gap Report 2018*. S.l. : UNEP, 2019. – ISBN 978–92–807–3726–4. – OCLC: 1082969108

[8] IPCC (Hrsg.) ; Pachauri, R. K. (Hrsg.) ; Mayer, Leo (Hrsg.): *Climate Change 2014: Synthesis Report*. Geneva, Switzerland : Intergovernmental Panel on Climate Change, 2015. – ISBN 978–92–9169–143–2. – OCLC: 914851124

[9] Karimi, M. ; Mokhlis, H. ; Naidu, K. ; Uddin, S. ; Bakar, A. H. A.: Photovoltaic Penetration Issues and Impacts in Distribution Network – A Review. In: *Renewable and Sustainable Energy Reviews* 53 (2016), Januar, S. 594–605. http://dx.doi.org/10.1016/j.rser.2015.08.042. – DOI 10.1016/j.rser.2015.08.042. – ISSN 1364–0321

[10] BDEW: Kraftwerks-Kapazitäten in der Europäischen Union schmelzen dahin. (2018)

[11] Wetzel, Daniel: Stromausfälle: Auch in Deutschland Wird Der Mega-Blackout Wahrscheinlicher. (2019), Juni

[12] Hockenos, Paul: *In Germany, Consumers Embrace a Shift to Home Batteries*. https://e360.yale.edu/features/in-germany-consumers-embrace-a-shift-to-home-batteries, März 2019

[13] Weinhardt, Christof ; Gimpel, Henner: Market Engineering: An Interdisciplinary Research Challenge. In: Jennings, Nick (Hrsg.) ; Kersten, Gregory (Hrsg.) ; Ockenfels, Axel (Hrsg.) ; Weinhardt, Christof (Hrsg.): *Negotiation and Market Engineering*. Dagstuhl, Germany : Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007 (Dagstuhl Seminar Proceedings)

[14] Zepter, Jan M. ; Lüth, Alexandra ; Crespo del Granado, Pedro ; Egging, Ruud: Prosumer Integration in Wholesale Electricity Markets: Synergies of Peer-to-Peer Trade and Residential Storage. In: *Energy and Buildings* 184 (2019), Februar, S. 163–176. http://dx.doi.org/10.1016/j.enbuild.2018.12.003. – DOI 10.1016/j.enbuild.2018.12.003. – ISSN 0378–7788

[15] Tews, Kerstin: Mapping the Regulatory Features Underpinning Prosumer Activities in Germany. (2016)

[16] Obermüller, Frank: Missstand EEG: Besser fördern, was wenig kostet / IW-Kurzbericht. 2019 (14/2019). – Research Report

[17] Davison, M. J. ; Summers, T. J. ; Townsend, C. D.: A Review of the Distributed Generation Landscape, Key Limitations of Traditional Microgrid Concept Amp; Possible Solution Using an Enhanced Microgrid Architecture. In: *2017 IEEE Southern Power Electronics Conference (SPEC)*, 2017, S. 1–6

[18] Zhang, Chenghua ; Wu, Jianzhong ; Long, Chao ; Cheng, Meng: Review of Existing Peer-to-Peer Energy Trading Projects. In: *Energy Procedia* 105 (2017), Mai, S. 2563–2568. http://dx.doi.org/10.1016/j.egypro.2017.03.737. – DOI 10.1016/j.egypro.2017.03.737. – ISSN 1876–6102

[19] Mengelkamp, Esther ; Diesing, Julius ; Weinhardt, Christof: Tracing Local Energy Markets: A Literature Review. (2019). http://dx.doi.org/10.13140/rg.2.2.17644.21128. – DOI 10.13140/rg.2.2.17644.21128

[20] Lüth, Alexandra ; Zepter, Jan M. ; Crespo del Granado, Pedro ; Egging, Ruud: Local Electricity Market Designs for Peer-to-Peer Trading: The Role of Battery Flexibility. In: *Applied Energy* 229 (2018), November, S. 1233–1243. http://dx.doi.org/10.1016/j.apenergy.2018.08.004. – DOI 10.1016/j.apenergy.2018.08.004. – ISSN 0306–2619

[21] Mengelkamp, Esther ; Gärttner, Johannes ; Rock, Kerstin ; Kessler, Scott ; Orsini, Lawrence ; Weinhardt, Christof: Designing Microgrid Energy Markets: A Case Study: The Brooklyn Microgrid. In: *Applied Energy* 210 (2018), Januar, S. 870–880. http://dx.doi.org/10.1016/j.apenergy.2017.06.054. – DOI 10.1016/j.apenergy.2017.06.054. – ISSN 0306–2619

[22] Dauer, David: *Market-Based Allocation of Local Flexibility in Smart Grids: A Mechanism Design Approach*. Karsruhe, Germany, KIT, Dissertation, 2016

[23] Zhang, Chenghua ; Wu, Jianzhong ; Zhou, Yue ; Cheng, Meng ; Long, Chao: Peer-to-Peer Energy Trading in a Microgrid. In: *Applied Energy* 220 (2018), Juni, S. 1–12. http://dx.doi.org/10.1016/j.apenergy.2018.03.010. – DOI 10.1016/j.apenergy.2018.03.010. – ISSN 0306–2619

[24] Olivella-Rosell, Pol ; Lloret-Gallego, Pau ; Munné-Collado, Íngrid ; Villafafila-Robles, Roberto ; Sumper, Andreas ; Ottessen, Stig ; Rajasekharan, Jayaprakash ; Bremdal, Bernt: Local Flexibility Market Design for Aggregators Providing Multiple Flexibility Services at Distribution Network Level. In: *Energies* 11 (2018), April, Nr. 4, S. 822. http://dx.doi.org/10.3390/en11040822. – DOI 10.3390/en11040822. – ISSN 1996–1073

[25] Mengelkamp, E. ; Staudt, P. ; Garttner, J. ; Weinhardt, C.: Trading on Local Energy Markets: A Comparison of Market Designs and Bidding Strategies. In: *2017 14th International Conference on the European Energy Market (EEM)*, 2017, S. 1–6

[26] Nakamoto, Satoshi: Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. – Technical report

[27] Bogensperger, Alexander ; Zeiselmair, Andreas: *Die Blockchain-Technologie: Chance zur Transformation der Energiewirtschaft? Berichtsteil Anwendungsfälle*. München : Forschungsstelle für Energiewirtschaft e.V. Forschungsgesellschaft für Energiewirtschat mbH, 2018. – ISBN 987–3–941802–42–1

[28] Sousa, Tiago ; Soares, Tiago ; Pinson, Pierre ; Moret, Fabio ; Baroche, Thomas ; Sorin, Etienne: Peer-to-Peer and Community-Based Markets: A Comprehensive Review. In: *Renewable and Sustainable Energy Reviews* 104 (2019), April, S. 367–378. http://dx.doi.org/10.1016/j.rser.2019.01.036. – DOI 10.1016/j.rser.2019.01.036. – ISSN 13640321

[29] Espe, Eunice ; Potdar, Vidyasagar ; Chang, Elizabeth: Prosumer Communities and Relationships in Smart Grids: A Literature Review, Evolution and Future Directions. In: *Energies* 11 (2018), September, Nr. 10, S. 2528. http://dx.doi.org/10.3390/en11102528. – DOI 10.3390/en11102528. – ISSN 1996–1073

[30] Deutsche Energie-Agentur ; Richard, Philipp ; Mamel, Sara ; Vogel, Lukas: Blockchain in the Integrated Energy Transition / German Energy Agency. Berlin, Februar 2019. – Dena Multi-Stakeholder Study

[31] Lüth, Alexandra R. ; Zepter, Jan Martin W.: *Rethinking the Role of Prosumers*, Norwegian University of Science and Technology, Diss., 2018

[32] Münsing, E. ; Mather, J. ; Moura, S.: Blockchains for Decentralized Optimization of Energy Resources in Microgrid Networks. In: *2017 IEEE Conference on Control Technology and Applications (CCTA)*, 2017, S. 2164–2171

[33] Buterin, Vitalik: Ethereum White Paper: A Next Generation Smart Contract & Decentralized Application Platform. 2014. – Technical report

[34] Bashir, Imran: *Mastering Blockchain: Distributed Ledger Technology, Decentralization, and Smart Contracts Explained, 2nd Edition*. Packt Publishing Ltd, 2018. – ISBN 978–1–78883–867–2

[35] Kim, S. ; Kwon, Y. ; Cho, S.: A Survey of Scalability Solutions on Blockchain. In: *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, S. 1204–1207

[36] European Union Blockchain Observatory and Forum: Scalability, Interoperability and Sustainability of Blockchains. 2019. – Technical report

[37] Lin, Iuon-Chang ; Liao, Tzu-Chun: A Survey of Blockchain Security Issues and Challenges. In: *International Journal of Network Security* 19 (2017), September, Nr. 5, S. 653–659. http://dx.doi.org/10.6633/IJNS.201709.19(5).01. – DOI 10.6633/IJNS.201709.19(5).01. – ISSN 1816–353X

[38] Mihaylov, M. ; Jurado, S. ; Avellana, N. ; Moffaert, K. V. ; Abril, I. M. ; Nowé, A.: NRGcoin: Virtual Currency for Trading of Renewable Energy in Smart Grids. In: *11th International Conference on the European Energy Market (EEM14)*, 2014, S. 1–6

[39] Murkin, Jordan ; Chitchyan, Ruzanna ; Byrne, Alastair: Enabling Peer-to-Peer Electricity Trading. In: *Proceedings of ICT for Sustainability 2016*. Amsterdam, the Netherlands : Atlantis Press, 2016. – ISBN 978–94–6252–224–4

[40] Mustafa, Mustafa A. ; Cleemput, Sara ; Abidin, Aysajan: A Local Electricity Trading Market: Security Analysis. In: *2016 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. Ljubljana, Slovenia : IEEE, Oktober 2016. – ISBN 978–1–5090–3358–4, S. 1–6

[41] Hahn, A. ; Singh, R. ; Liu, C. ; Chen, S.: Smart Contract-Based Campus Demonstration of Decentralized Transactive Energy Auctions. In: *2017 IEEE*

Power Energy Society Innovative Smart Grid Technologies Conference (ISGT), 2017, S. 1–5

[42] Mengelkamp, Esther ; Notheisen, Benedikt ; Beer, Carolin ; Dauer, David ; Weinhardt, Christof: A Blockchain-Based Smart Grid: Towards Sustainable Local Energy Markets. In: *Computer Science - Research and Development* 33 (2018), Februar, Nr. 1, S. 207–214. http://dx.doi.org/10.1007/s00450-017-0360-9. – DOI 10.1007/s00450–017–0360–9. – ISSN 1865–2042

[43] Myung, Sein ; Lee, Jong-Hyouk: Ethereum Smart Contract-Based Automated Power Trading Algorithm in a Microgrid Environment. In: *The Journal of Supercomputing* (2018), November, S. 1–11. http://dx.doi.org/10.1007/s11227-018-2697-7. – DOI 10.1007/s11227–018–2697–7. – ISSN 0920–8542, 1573–0484

[44] Yu, Qianchen: Design, Implementation, and Evaluation of a Blockchain-Enabled Multi-Energy Transaction System for District Energy Systems. (2018). http://dx.doi.org/10.3929/ethz-b-000257364. – DOI 10.3929/ethz–b–000257364

[45] Abdella, Juhar ; Shuaib, Khaled: Peer to Peer Distributed Energy Trading in Smart Grids: A Survey. In: *Energies* 11 (2018), Juni, Nr. 6, S. 1560. http://dx.doi.org/10.3390/en11061560. – DOI 10.3390/en11061560. – ISSN 1996–1073

[46] Kirpes, Benedikt ; Mengelkamp, Esther ; Becker, Christian ; Weinhardt, Christof ; Schaal, Georg: Design of a Microgrid Local Energy Market on a Blockchain-Based Information System. In: *Unpublished* (2019). http://dx.doi.org/10.13140/rg.2.2.18627.45607. – DOI 10.13140/rg.2.2.18627.45607

[47] Li, Z. ; Kang, J. ; Yu, R. ; Ye, D. ; Deng, Q. ; Zhang, Y.: Consortium Blockchain for Secure Energy Trading in Industrial Internet of Things. In: *IEEE Transactions on Industrial Informatics* 14 (2018), August, Nr. 8, S. 3690–3700. http://dx.doi.org/10.1109/TII.2017.2786307. – DOI 10.1109/TII.2017.2786307. – ISSN 1551–3203

[48] Dinh, T. T. A. ; Liu, R. ; Zhang, M. ; Chen, G. ; Ooi, B. C. ; Wang, J.: Untangling Blockchain: A Data Processing View of Blockchain Systems. In: *IEEE Transactions on Knowledge and Data Engineering* 30 (2018), Juli, Nr. 7, S. 1366–1385. http://dx.doi.org/10.1109/TKDE.2017.2781227. – DOI 10.1109/TKDE.2017.2781227. – ISSN 1041–4347

[49] Seidl, Matthias: *Implementation of Blockchain-Based and Grid-Friendly Local Energy Markets.* Erlangen-Nürnberg, Friedrich-Alexander-Universität Erlangen-Nürnberg, Master Thesis, Oktober 2018

[50] Wang, Naiyu ; Zhou, Xiao ; Lu, Xin ; Guan, Zhitao ; Wu, Longfei ; Du, Xiaojiang ; Guizani, Mohsen: When Energy Trading Meets Blockchain in Electrical Power System: The State of the Art. In: *Applied Sciences* 9 (2019), April, Nr. 8, S. 1561. http://dx.doi.org/10.3390/app9081561. – DOI 10.3390/app9081561. – ISSN 2076–3417

[51] Laszka, Aron ; Dubey, Abhishek ; Walker, Michael ; Schmidt, Doug: Providing Privacy, Safety, and Security in IoT-Based Transactive Energy Systems Using Distributed Ledgers. In: *Proceedings of the Seventh International Conference on the Internet of Things*. New York, NY, USA : ACM, 2017 (IoT '17). – ISBN 978–1–4503–5318–2, S. 13:1–13:8

[52] Valenta, Martin ; Sandner, Philipp: Comparison of Ethereum, Hyperledger Fabric and Corda / FSBC Working Paper. Frankfurt School of Finance & Management gGmbH, Juni 2017. – Technical report

[53] microgidmedia: *TenneT, IBM Launch Pilot Distributed Energy-Blockchain Projects in Germany and the Netherlands*. Mai 2017

[54] gtreview: *Seven Banks to Go Live with Hyperledger Trade Finance Platform in 2017*. https://www.gtreview.com/news/europe/seven-banks-to-go-live-with-hyperledger-blockchain-trade-finance-platform-in-2017/, Juni 2017

[55] Fujitsu: *Fujitsu Enables Blockchain Proof of Business in Just a Week - Fujitsu CEMEA&I*. http://www.fujitsu.com/fts/about/resources/news/press-releases/2018/fujitsu-enables-blockchain-proof-of-business-in-just-a-week.html, 2018

[56] IBM: *Maersk and IBM Unveil Supply Chain Solution on Blockchain*. https://www-03.ibm.com/press/us/en/pressrelease/51712.wss, März 2017

[57] SWIFT: *SWIFT Completes Landmark DLT Proof of Concept*. https://www.swift.com/news-events/news/swift-completes-landmark-dlt-proof-of-concept, März 2018

[58] Ferris, Christopher: *Does Hyperledger Fabric Perform at Scale?* https://www.ibm.com/blogs/blockchain/2019/04/does-hyperledger-fabric-perform-at-scale/, April 2019

[59] Thakkar, Parth ; Nathan, Senthil ; Vishwanathan, Balaji: Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform. In: *arXiv:1805.11390 [cs]* (2018), Mai

[60] Androulaki, Elli ; Barger, Artem ; Bortnikov, Vita ; Cachin, Christian ; Christidis, Konstantinos ; De Caro, Angelo ; Enyeart, David ; Ferris, Christopher ; Laventman, Gennady ; Manevich, Yacov ; Muralidharan, Srinivasan ; Murthy, Chet ; Nguyen, Binh ; Sethi, Manish ; Singh, Gari

; Smith, Keith ; Sorniotti, Alessandro ; Stathakopoulou, Chrysoula ; Vukolić, Marko ; Cocco, Sharon W. ; Yellick, Jason: Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In: *Proceedings of the Thirteenth EuroSys Conference*. New York, NY, USA : ACM, 2018 (EuroSys '18). – ISBN 978–1–4503–5584–1, S. 30:1–30:15

[61] Lukka, Kari: The Constructive Research Approach. In: *Case Study Research in Logistics*. 2003, S. 83–101

[62] Peffers, Ken ; Tuunanen, Tuure ; Rothenberger, Marcus ; Chatterjee, Samir: A Design Science Research Methodology for Information Systems Research. In: *J. Manage. Inf. Syst.* 24 (2007), Dezember, Nr. 3, S. 45–77. http://dx.doi.org/10.2753/MIS0742-1222240302. – DOI 10.2753/MIS0742–1222240302. – ISSN 0742–1222

[63] Hevner, Alan R. ; March, Salvatore T. ; Park, Jinsoo ; Ram, Sudha: Design Science in Information Systems Research. In: *MIS Quarterly* Vol. 28 (2004), Nr. No. 1, S. 32

[64] Neureiter, C. ; Uslar, M. ; Engel, D. ; Lastro, G.: A Standards-Based Approach for Domain Specific Modelling of Smart Grid System Architectures. In: *2016 11th System of Systems Engineering Conference (SoSE)*, 2016, S. 1–6

[65] CEN-CENELEC-ETSI Smart Grid Coordination Group: Smart Grid Reference Architecture. (2012), November

[66] Rathnayaka, A. J. D. ; Potdar, Vidyasagar M. ; Dillon, Tharam ; Kuruppu, Samitha: Framework to Manage Multiple Goals in Community-Based Energy Sharing Network in Smart Grid. In: *International Journal of Electrical Power & Energy Systems* 73 (2015), Dezember, S. 615–624. http://dx.doi.org/10.1016/j.ijepes.2015.05.008. – DOI 10.1016/j.ijepes.2015.05.008. – ISSN 0142–0615

[67] Maity, I. ; Rao, S.: Simulation and Pricing Mechanism Analysis of a Solar-Powered Electrical Microgrid. In: *IEEE Systems Journal* 4 (2010), September, Nr. 3, S. 275–284. http://dx.doi.org/10.1109/JSYST.2010.2059110. – DOI 10.1109/JSYST.2010.2059110. – ISSN 1932–8184

[68] Schollmeier, R.: A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. In: *Proceedings First International Conference on Peer-to-Peer Computing*, 2001, S. 101–102

[69] Ton, Dan T. ; Smith, Merrill A.: The U.S. Department of Energy's Microgrid Initiative. In: *The Electricity Journal* 25 (2012), Oktober, Nr. 8, S. 84–94. http://dx.doi.org/10.1016/j.tej.2012.09.013. – DOI 10.1016/j.tej.2012.09.013. – ISSN 1040–6190

[70] Marnay, C. ; Chatzivasileiadis, S. ; Abbey, C. ; Iravani, R. ; Joos, G. ; Lombardi, P. ; Mancarella, P. ; Appen, J. von: Microgrid Evolution Roadmap. In: *2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST)*, 2015, S. 139–144

[71] Koponen, Pekka ; Diaz Saco, Luis ; Orchard, Nigel ; Vorisek, Tomas ; Parsons, John ; Rochas, Claudio ; Z. Morch, Adrei ; Lopes, Vitor ; Togeby, Mikael: *Definition of Smart Metering and Applications and Identification of Benefits*. 2008

[72] Buterin, Vitalik: *On Public and Private Blockchains*. https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/, August 2015

[73] Gauba, Alexis ; aparnalocked ; snarkyzk ; MaazUddin11 ; Mechanism Labs: *Alternative Consensus Meta Analysis Chart*. https://docs.google.com/spreadsheets/d/1IW5AuFQtL4z34HgIZrqpcVfZPnOiifsc3xmA9KIef6I/htmlview#gid=658801415, 2019

[74] Conte de Leon, Daniel ; Stalick, Antonius Q. ; Jillepalli, Ananth A. ; Haney, Michael A. ; Sheldon, Frederick T. ; Sheldon, Frederick T.: Blockchain: Properties and Misconceptions. In: *Asia Pacific Journal of Innovation and Entrepreneurship* 11 (2017), Dezember, Nr. 3, S. 286–300. http://dx.doi.org/10.1108/APJIE-12-2017-034. – DOI 10.1108/APJIE–12–2017–034. – ISSN 2398–7812

[75] Walch, Angela: The Path of the Blockchain Lexicon (and the Law) / Social Science Research Network. Rochester, NY, März 2017 (ID 2940335). – SSRN Scholarly Paper

[76] Jackisch, Marcel: *Evaluation of Potential Blockchain Use Cases for the Operation of an Offshore Wind Farm*, Technische Universität Berlin, Master Thesis, Januar 2019

[77] Zheng, Z. ; Xie, S. ; Dai, H. ; Chen, X. ; Wang, H.: An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In: *2017 IEEE International Congress on Big Data (BigData Congress)*, 2017, S. 557–564

[78] Swan, Melanie: *Blockchain: Blueprint for a New Economy*. "O'Reilly Media, Inc.", 2015. – ISBN 978–1–4919–2047–3

[79] Drescher, Daniel: *Blockchain Basics: A Non-Technical Introduction in 25 Steps*. Berkeley, California : Apress, 2017. – ISBN 978–1–4842–2603–2 978–1–4842–2604–9. – OCLC: ocn971341713

[80] Iansiti, Marco ; Lakhani, Karim R.: Blockchain Business. In: *Harvard Business Manager* May 2017 (2017), Mai, S. 62–71. – ISSN 09456570

[81] Schlatt, Vincent ; Schweizer, André ; Urbach, Nils ; al, et: Blockchain: Grundlagen, Anwendungen Und Potenziale / Fraunhofer FIT. Sankt Augustin, 2016. – Technical report

[82] Brainerd, Walter S.: *Theory of Computation*. 1st edition. New York : Wiley, 1974. – ISBN 978–0–471–09585–9

[83] Vujičić, D. ; Jagodić, D. ; Randić, S.: Blockchain Technology, Bitcoin, and Ethereum: A Brief Overview. In: *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2018, S. 1–6

[84] Millsap, Cary: Thinking Clearly About Performance. In: *Queue* 8 (2010), September, Nr. 9, S. 10. http://dx.doi.org/10.1145/1854039.1854041. – DOI 10.1145/1854039.1854041. – ISSN 15427730

[85] Tabora, Vince: *Ethereum 2.0 — The Road To Constantinople And Beyond*. Januar 2019

[86] CoinMarketCap: *Cryptocurrency Market Capitalizations | CoinMarketCap*. https://coinmarketcap.com/, 2019

[87] Narayanan, Arvind: *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton : Princeton University Press, 2016. – ISBN 978–0–691–17169–2

[88] Parity: *What Is a Light Client and Why You Should Care?* https://www.parity.io/what-is-a-light-client/, Juli 2018

[89] Glaser, Florian: Pervasive Decentralisation of Digital Infrastructures: A Framework for Blockchain Enabled System and Use Case Analysis, 2017. – ISBN 978–0–9981331–0–2

[90] Ripple: *Ripple - One Frictionless Experience To Send Money Globally*. https://ripple.com/, 2019

[91] Ben Hamida, Elyes ; Brousmiche, Kei L. ; Levard, Hugo ; Thea, Eric: Blockchain for Enterprise: Overview, Opportunities and Challenges. In: *The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017)*. Nice, France, Juli 2017

[92] Mueller, Thomas: *Ethereum, Hyperledger or IOTA for Enterprises — Where Are the Differences?* Juni 2018

[93] Hyperledger Fabric: *Hyperledger Fabric Documentation: Smart Contracts and Chaincode*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/smartcontract/smartcontract.html, 2019

[94] O'Neal, Stephen: *Who Scales It Best? Inside Blockchains' Ongoing Transactions-Per-Second Race*. https://cointelegraph.com/news/who-scales-it-best-inside-blockchains-ongoing-transactions-per-second-race, Januar 2019

[95] Androulaki, Elli ; Cocco, Sharon ; Ferris, Chris: *Private and Confidential Transactions with Hyperledger Fabric*. Mai 2018

[96] UK Government Chief Scientific Adviser: Distributed Ledger Technology: Beyond Block Chain / UK Government Office for Science. London, 2016. – Technical report

[97] Blockchain Monitor: Blockchain for FinTech: Results from a Survey among Professionals from the Financial Services Sector, the Information Technology Sector and Academia / Blockchain Monitor. Hamburg, September 2017. – Technical report

[98] Andoni, Merlinda ; Robu, Valentin ; Flynn, David ; Abram, Simone ; Geach, Dale ; Jenkins, David ; McCallum, Peter ; Peacock, Andrew: Blockchain Technology in the Energy Sector: A Systematic Review of Challenges and Opportunities. In: *Renewable and Sustainable Energy Reviews* 100 (2019), Februar, S. 143–174. http://dx.doi.org/10.1016/j.rser.2018.10.014. – DOI 10.1016/j.rser.2018.10.014. – ISSN 13640321

[99] Dütsch, Gunther ; Steinecke, Neon: Use Cases for Blockchain Technology in Energy and Commodity Trading. 2017. – Technical report

[100] Basden, James ; Cottrell, Michael: How Utilities Are Using Blockchain to Modernize the Grid. In: *Harvard Business Review* (2017), März

[101] Underwood, Sarah: Blockchain Beyond Bitcoin. In: *Commun. ACM* 59 (2016), Oktober, Nr. 11, S. 15–17. http://dx.doi.org/10.1145/2994581. – DOI 10.1145/2994581. – ISSN 0001–0782

[102] Gronholt-Pedersen, Jacob: Maersk, IBM to Launch Blockchain-Based Platform for Global Trade. In: *Reuters* (Tue Jan 16 12:16:34 UTC 2018)

[103] Yermack, David: Corporate Governance and Blockchains / Social Science Research Network. Rochester, NY, November 2016 (ID 2700475). – SSRN Scholarly Paper

[104] Duivestein, Sander ; Bloem, Jaap ; van Doorn, Menno ; van Manen, Thomas: Design to Disrupt – Blockchain: Cryptoplatform for a Frictionless Economy. In: *Sogeti* (2015)

[105] Energy Web Foundation: *EW Origin – Energy Web Foundation*. https://energyweb.org/origin/, 2019

[106] Provenance: *Provenance*. https://www.provenance.org/, 2019

[107] Kshetri, N.: Can Blockchain Strengthen the Internet of Things? In: *IT Professional* 19 (2017), Nr. 4, S. 68–72. http://dx.doi.org/10.1109/MITP.2017.3051335. – DOI 10.1109/MITP.2017.3051335. – ISSN 1520–9202

[108] Conoscenti, Marco ; Vetro, Antonio ; De Martin, Juan C.: Blockchain for the Internet of Things: A Systematic Literature Review, IEEE, November 2016. – ISBN 978–1–5090–4320–0, S. 1–6

[109] Ouaddah, Aafaf ; Abou Elkalam, Anas ; Ait Ouahman, Abdellah: FairAccess: A New Blockchain-Based Access Control Framework for the Internet of Things. In: *Security and Communication Networks* 9 (2016), Dezember, Nr. 18, S. 5943–5964. http://dx.doi.org/10.1002/sec.1748. – DOI 10.1002/sec.1748. – ISSN 1939–0122

[110] Deloitte: Israel: A Hotspot for Blockchain Innovation. 2016. – Technical report

[111] Truby, Jon: Decarbonizing Bitcoin: Law and Policy Choices for Reducing the Energy Consumption of Blockchain Technologies and Digital Currencies. In: *Energy Research & Social Science* 44 (2018), Oktober, S. 399–410. http://dx.doi.org/10.1016/j.erss.2018.06.009. – DOI 10.1016/j.erss.2018.06.009. – ISSN 2214–6296

[112] Kakavand, Hossein ; Kost De Sevres, Nicolette ; Chilton, Bart: The Blockchain Revolution: An Analysis of Regulation and Technology Related to Distributed Ledger Technologies / Social Science Research Network. Rochester, NY, Januar 2017 (ID 2849251). – SSRN Scholarly Paper

[113] Lu, Jack: *The Importance of Blockchain Interoperability.* Oktober 2018

[114] Wood, Gavin: Polkadot: Vision for a Heterogeneous Multi-Chain Framework / Partiy. 2016. – Technical report

[115] Cosmos Network: *Cosmos Network - Internet of Blockchains.* https://cosmos.network/, 2019

[116] Rashed Mohassel, Ramyar ; Fung, Alan ; Mohammadi, Farah ; Raahemifar, Kaamran: A Survey on Advanced Metering Infrastructure. In: *International Journal of Electrical Power & Energy Systems* 63 (2014), Dezember, S. 473–484. http://dx.doi.org/10.1016/j.ijepes.2014.06.025. – DOI 10.1016/j.ijepes.2014.06.025. – ISSN 0142–0615

[117] Jogunola, Olamide ; Ikpehai, Augustine ; Anoh, Kelvin ; Adebisi, Bamidele ; Hammoudeh, Mohammad ; Son, Sung-Yong ; Harris, Georgina: State-Of-The-Art and Prospects for Peer-To-Peer Transaction-Based Energy System. In: *Energies* 10 (2017), Dezember, Nr. 12, S. 2106. http://dx.doi.org/10.3390/en10122106. – DOI 10.3390/en10122106

[118] Sorin, E. ; Bobo, L. ; Pinson, P.: Consensus-Based Approach to Peer-to-Peer Electricity Markets With Product Differentiation. In: *IEEE Transactions on Power Systems* 34 (2019), März, Nr. 2, S. 994–1004. http://dx.doi.org/10.1109/TPWRS.2018.2872880. – DOI 10.1109/TPWRS.2018.2872880. – ISSN 0885–8950

[119] Parag, Yael ; Sovacool, Benjamin K.: Electricity Market Design for the Prosumer Era. In: *Nature Energy* 1 (2016), April, Nr. 4, S. 16032. http://dx.doi.org/10.1038/nenergy.2016.32. – DOI 10.1038/nenergy.2016.32. – ISSN 2058–7546

[120] Morstyn, T. ; Teytelboym, A. ; Mcculloch, M. D.: Bilateral Contract Networks for Peer-to-Peer Energy Trading. In: *IEEE Transactions on Smart Grid* 10 (2019), März, Nr. 2, S. 2026–2035. http://dx.doi.org/10.1109/TSG.2017.2786668. – DOI 10.1109/TSG.2017.2786668. – ISSN 1949–3053

[121] Akter, M. N. ; Mahmud, M. A. ; Oo, A. M. T.: A Hierarchical Transactive Energy Management System for Microgrids. In: *2016 IEEE Power and Energy Society General Meeting (PESGM)*, 2016, S. 1–5

[122] Olivella-Rosell, P. ; Viñals-Canal, G. ; Sumper, A. ; Villafafila-Robles, R. ; Bremdal, B. A. ; Ilieva, I. ; Ottesen, S.: Day-Ahead Micro-Market Design for Distributed Energy Resources. In: *2016 IEEE International Energy Conference (ENERGYCON)*, 2016, S. 1–6

[123] Ilieva, I. ; Bremdal, B. ; Ottesen, S. ; Rajasekharan, J. ; Olivella-Rosell, P.: Design Characteristics of a Smart Grid Dominated Local Market. In: *CIRED Workshop 2016*, 2016, S. 1–4

[124] Verschae, Rodrigo ; Kato, Takekazu ; Matsuyama, Takashi: Energy Management in Prosumer Communities: A Coordinated Approach. In: *Energies* 9 (2016), Juli, Nr. 7, S. 562. http://dx.doi.org/10.3390/en9070562. – DOI 10.3390/en9070562

[125] Moret, F. ; Pinson, P.: Energy Collectives: A Community and Fairness Based Approach to Future Electricity Markets. In: *IEEE Transactions on Power Systems* (2018), S. 1–1. http://dx.doi.org/10.1109/TPWRS.2018.2808961. – DOI 10.1109/TPWRS.2018.2808961. – ISSN 0885–8950

[126] Morstyn, Thomas ; McCulloch, Malcolm: Multi-Class Energy Management for Peer-to-Peer Energy Trading Driven by Prosumer Preferences. In: *IEEE Transactions on Power Systems* (2018), S. 1–1. http://dx.doi.org/10.1109/TPWRS.2018.2834472. – DOI 10.1109/TPWRS.2018.2834472. – ISSN 0885–8950, 1558–0679

[127] Long, Chao ; Wu, Jianzhong ; Zhang, Chenghua ; Cheng, Meng ; Al-Wakeel, Ali: Feasibility of Peer-to-Peer Energy Trading in Low Voltage Electrical Distribution Networks. In: *Energy Procedia* 105 (2017),

Mai, S. 2227–2232. http://dx.doi.org/10.1016/j.egypro.2017.03.632. – DOI 10.1016/j.egypro.2017.03.632. – ISSN 1876–6102

[128] Liu, T. ; Tan, X. ; Sun, B. ; Wu, Y. ; Guan, X. ; Tsang, D. H. K.: Energy Management of Cooperative Microgrids with P2P Energy Sharing in Distribution Networks. In: *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2015, S. 410–415

[129] Alvaro-Hermana, R. ; Fraile-Ardanuy, J. ; Zufiria, P. J. ; Knapen, L. ; Janssens, D.: Peer to Peer Energy Trading with Electric Vehicles. In: *IEEE Intelligent Transportation Systems Magazine* 8, Nr. 3, S. 33–44. http://dx.doi.org/10.1109/MITS.2016.2573178. – DOI 10.1109/MITS.2016.2573178. – ISSN 1939–1390

[130] Ilic, D. ; Silva, P. G. D. ; Karnouskos, S. ; Griesemer, M.: An Energy Market for Trading Electricity in Smart Grid Neighbourhoods. In: *2012 6th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, 2012, S. 1–6

[131] Kang, J. ; Yu, R. ; Huang, X. ; Maharjan, S. ; Zhang, Y. ; Hossain, E.: Enabling Localized Peer-to-Peer Electricity Trading Among Plug-in Hybrid Electric Vehicles Using Consortium Blockchains. In: *IEEE Transactions on Industrial Informatics* 13 (2017), Dezember, Nr. 6, S. 3154–3164. http://dx.doi.org/10.1109/TII.2017.2709784. – DOI 10.1109/TII.2017.2709784. – ISSN 1551–3203

[132] Tushar, W. ; Chai, B. ; Yuen, C. ; Huang, S. ; Smith, D. B. ; Poor, H. V. ; Yang, Z.: Energy Storage Sharing in Smart Grid: A Modified Auction-Based Approach. In: *IEEE Transactions on Smart Grid* 7 (2016), Mai, Nr. 3, S. 1462–1475. http://dx.doi.org/10.1109/TSG.2015.2512267. – DOI 10.1109/TSG.2015.2512267. – ISSN 1949–3053

[133] Moret, F. ; Baroche, T. ; Sorin, E. ; Pinson, P.: Negotiation Algorithms for Peer-to-Peer Electricity Markets: Computational Properties. In: *2018 Power Systems Computation Conference (PSCC)*. Dublin, Ireland : IEEE, Juni 2018. – ISBN 978–1–910963–10–4, S. 1–7

[134] Prognos AG ; Boos Hummel & Wegerich: "Mieterstrom – Rechtliche Einordnung, Organisationsformen, Potenziale Und Wirtschaftlichkeit von Mieterstrommodellen". 2017. – Technical report

[135] Schäfer-Stradowsky, Simon ; Bachmann, Sandra: Rechtspolitische Rahmenbedingungen für Prosumer. In: *Ökologisches Wirtschaften - Fachzeitschrift* 31 (2016), Mai, Nr. 2, S. 21. http://dx.doi.org/10.14512/OEW310221. – DOI 10.14512/OEW310221. – ISSN 1430–8800

[136] Legifrance: *Code de l'énergie*. 2019

[137] European Commission: Clean Energy For All Europeans / European Commission. Brussels, November 2016 (COM(2016) 860 final). – Technical report

[138] Harris, Jacob: Power... With a Little Help from Your Friends. In: *Electrical Connection* , Nr. Autumn 2019, S. 34

[139] sonnen GmbH: *Smart Local Grids with Blockchain - Sonnen Is Participating in the NEMoGrid Project*. https://sonnengroup.com/smart-local-grids-blockchain-sonnen-participating-nemogrid-project/, 2018

[140] Roth, Alvin E.: The Economist as Engineer: Game Theory, Experimentation, and Computation as Tools for Design Economics. In: *Econometrica* 70 (2002), Nr. 4, S. 1341–1378. http://dx.doi.org/10.1111/1468-0262.00335. – DOI 10.1111/1468–0262.00335. – ISSN 1468–0262

[141] Klemperer, Paul: Auction Theory: A Guide to the Literature / Social Science Research Network. Rochester, NY, Mai 1999 (ID 172650). – SSRN Scholarly Paper

[142] Parsons, Simon ; Marcinkiewicz, Marek ; Niu, Jinzhong ; Phelps, Steve: Everything You Wanted to Know about Double Auctions, but Were Afraid to (Bid or) Ask. (2006), Januar

[143] Myerson, Roger B. ; Satterthwaite, Mark A.: Efficient Mechanisms for Bilateral Trading. In: *Journal of Economic Theory* 29 (1983), April, Nr. 2, S. 265–281. http://dx.doi.org/10.1016/0022-0531(83)90048-0. – DOI 10.1016/0022–0531(83)90048–0. – ISSN 0022–0531

[144] Satterthwaite, Mark A. ; Williams, Steven R.: Bilateral Trade with the Sealed Bid K-Double Auction: Existence and Efficiency. In: *Journal of Economic Theory* 48 (1989), Juni, Nr. 1, S. 107–133. http://dx.doi.org/10.1016/0022-0531(89)90121-X. – DOI 10.1016/0022–0531(89)90121–X. – ISSN 0022–0531

[145] Parkes, David: *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency*, Univesity of Pennsylvania, Ph.D. Dissertation, Mai 2001

[146] Babaioff, M. ; Nisan, N.: Concurrent Auctions Across The Supply Chain. In: *Journal of Artificial Intelligence Research* 21 (2004), Mai, S. 595–629. http://dx.doi.org/10.1613/jair.1316. – DOI 10.1613/jair.1316. – ISSN 1076–9757

[147] Vytelingum, Perukrishnen ; Ramchurn, Sarvapali D. ; Voice, Thomas D. ; Rogers, Alex ; Jennings, Nicholas R.: Trading Agents for the Smart Electricity Grid. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*. Richland, SC

: International Foundation for Autonomous Agents and Multiagent Systems, 2010 (AAMAS '10). – ISBN 978–0–9826571–1–9, S. 897–904

[148] Securities and Exchange Commission: *Self-Regulatory Organizations; New York Stock Exchange LLC; Notice of Filing and Immediate Effectiveness of Proposed Rule Change of New Rule 7.44 To Operate Its Retail Liquidity Program on Pillar, the Exchange's New Technology Trading Platform.* Mai 2019

[149] Blouin, Max R. ; Serrano, Roberto: A Decentralized Market with Common Values Uncertainty: Non-Steady States. In: *The Review of Economic Studies* 68 (2001), April, Nr. 2, S. 323–346. http://dx.doi.org/10.1111/1467-937X.00171. – DOI 10.1111/1467–937X.00171. – ISSN 0034–6527

[150] Mihaylov, Mihail ; Jurado, Sergio ; Moffaert, Kristof V. ; Avellana, Narcís ; Nowé, Ann: NRG-X-Change - A Novel Mechanism for Trading of Renewable Energy in Smart Grids. In: *Proceedings of the 3rd International Conference on Smart Grids and Green IT Systems*. Barcelona, Spain : SCITEPRESS - Science and and Technology Publications, 2014. – ISBN 978–989–758–025–3, S. 101–106

[151] Clearwater, SH̃: *Market-Based Control*. WORLD SCIENTIFIC, 1996. http://dx.doi.org/10.1142/2741. – ISBN 978–981–02–2254–3

[152] Kok, Koen: *The PowerMatcher: Smart Coordination for the Smart Electricity Grid*, Diss., Juli 2013

[153] Lamparter, Steffen ; Becher, Silvio ; Fischer, Jan-Gregor: An Agent-Based Market Platform for Smart Grids. In: *AAMAS*, 2010

[154] Mengelkamp, Esther ; Gärttner, Johannes ; Weinhardt, Christof: Intelligent Agent Strategies for Residential Customers in Local Electricity Markets. In: *Proceedings of the Ninth International Conference on Future Energy Systems*. New York, NY, USA : ACM, 2018 (E-Energy '18). – ISBN 978–1–4503–5767–8, S. 97–107

[155] Zubi, Ghassan ; Dufo-López, Rodolfo ; Carvalho, Monica ; Pasaoglu, Guzay: The Lithium-Ion Battery: State of the Art and Future Perspectives. In: *Renewable and Sustainable Energy Reviews* 89 (2018), Juni, S. 292–308. http://dx.doi.org/10.1016/j.rser.2018.03.002. – DOI 10.1016/j.rser.2018.03.002. – ISSN 1364–0321

[156] Albadi, M. H. ; El-Saadany, E. F.: A Summary of Demand Response in Electricity Markets. In: *Electric Power Systems Research* 78 (2008), November, Nr. 11, S. 1989–1996. http://dx.doi.org/10.1016/j.epsr.2008.04.002. – DOI 10.1016/j.epsr.2008.04.002. – ISSN 0378–7796

[157] Mengelkamp, Esther ; Bose, Samrat ; Kremers, Enrique ; Eberbach, Jan ; Hoffmann, Bastian ; Weinhardt, Christof: Increasing the Efficiency of Local Energy Markets through Residential Demand Response. In: *Energy Informatics* 1 (2018), August, Nr. 1, S. 11. http://dx.doi.org/10.1186/s42162-018-0017-3. – DOI 10.1186/s42162–018–0017–3. – ISSN 2520–8942

[158] Liebau, Björn: *Der deutsche Strommarkt: Marktdesign und Anbieterverhalten.* Münster, Germany, Westfälische Wilhelms-Universität Münster, Diss., Januar 2012

[159] David, A. K. ; Fushuan Wen: Strategic Bidding in Competitive Electricity Markets: A Literature Survey. In: *2000 Power Engineering Society Summer Meeting (Cat. No.00CH37134)* Bd. 4, 2000, S. 2168–2173 vol. 4

[160] Kahn, Alfred E. ; Cramton, Peter C. ; Porter, Robert H. ; Tabors, Richard D.: Uniform Pricing or Pay-as-Bid Pricing: A Dilemma for California and Beyond. In: *The Electricity Journal* 14 (2001), Juli, Nr. 6, S. 70–79. http://dx.doi.org/10.1016/S1040-6190(01)00216-0. – DOI 10.1016/S1040–6190(01)00216–0. – ISSN 1040–6190

[161] Cramton, Peter ; Stoft, Steven: Why We Need to Stick with Uniform-Price Auctions in Electricity Markets. In: *The Electricity Journal* 20 (2007), Januar, Nr. 1, S. 26–37. http://dx.doi.org/10.1016/j.tej.2006.11.011. – DOI 10.1016/j.tej.2006.11.011. – ISSN 1040–6190

[162] Chitchyan, Ruzanna ; Murkin, Jordan: Review of Blockchain Technology and Its Expectations: Case of the Energy Sector. In: *arXiv:1803.03567 [cs]* (2018), März

[163] Dorri, A. ; Kanhere, S. S. ; Jurdak, R. ; Gauravaram, P.: Blockchain for IoT Security and Privacy: The Case Study of a Smart Home. In: *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2017, S. 618–623

[164] Knirsch, Fabian ; Unterweger, Andreas ; Engel, Dominik: Privacy-Preserving Blockchain-Based Electric Vehicle Charging with Dynamic Tariff Decisions. In: *Computer Science - Research and Development* 33 (2018), Februar, Nr. 1, S. 71–79. http://dx.doi.org/10.1007/s00450-017-0348-5. – DOI 10.1007/s00450–017–0348–5. – ISSN 1865–2042

[165] Pop, Claudia ; Cioara, Tudor ; Antal, Marcel ; Anghel, Ionut ; Salomie, Ioan ; Bertoncini, Massimo: Blockchain Based Decentralized Management of Demand Response Programs in Smart Energy Grids. In: *Sensors* 18 (2018), Januar, Nr. 1, S. 162. http://dx.doi.org/10.3390/s18010162. – DOI 10.3390/s18010162

[166] Wang, Jian ; Wang, Qianggang ; Zhou, Niancheng ; Chi, Yuan: A Novel Electricity Transaction Mode of Microgrids Based on Blockchain and Continuous Double Auction, 2017

[167] Sikorski, Janusz J. ; Haughton, Joy ; Kraft, Markus: Blockchain Technology in the Chemical Industry: Machine-to-Machine Electricity Market. In: *Applied Energy* 195 (2017), Juni, S. 234–246. http://dx.doi.org/10.1016/j.apenergy.2017.03.039. – DOI 10.1016/j.apenergy.2017.03.039. – ISSN 0306–2619

[168] Schlund, Jonas ; Ammon, Lorenz ; German, Reinhard: ETHome: Open-Source Blockchain Based Energy Community Controller. In: *Proceedings of the Ninth International Conference on Future Energy Systems*. New York, NY, USA : ACM, 2018 (E-Energy '18). – ISBN 978–1–4503–5767–8, S. 319–323

[169] Biggelaar, D.E. van d.: *Towards Decentralized Grids – EnergyBazaar: Decentralized Free-Market Energy-Trade within an Isolated Community Micro-Grid*, TU Delft, Diss., April 2018

[170] Aitzhan, N. Z. ; Svetinovic, D.: Security and Privacy in Decentralized Energy Trading Through Multi-Signatures, Blockchain and Anonymous Messaging Streams. In: *IEEE Transactions on Dependable and Secure Computing* 15 (2018), September, Nr. 5, S. 840–852. http://dx.doi.org/10.1109/TDSC.2016.2616861. – DOI 10.1109/TDSC.2016.2616861. – ISSN 1545–5971

[171] Ferreira, Joao ; Martins, Ana: Building a Community of Users for Open Market Energy. In: *Energies* 11 (2018), September, Nr. 9, S. 2330. http://dx.doi.org/10.3390/en11092330. – DOI 10.3390/en11092330. – ISSN 1996–1073

[172] Reekers, Jasper N.: *Analyse Und Optimierung Eines Demonstrators Für Elektrisches Netzmanagement Und Energiehandel Auf Blockchains*. Hamburg, Germany, HWI Hamburg, Bachelor Thesis, April 2018

[173] Thut, Andreas: *Development and Evaluation of a DLT-Based Marketplace for Sector Coupling in Quarters*. München, Germany, Technische Universität München, Diss., Juni 2018

[174] De Angelis, Stefano ; Aniello, Leonardo ; Baldoni, Roberto ; Lombardi, Federico ; Margheri, Andrea ; Sassone, Vladimiro: PBFT vs Proof-of-Authority: Applying the CAP Theorem to Permissioned Blockchain. In: *Italian Conference on Cyber Security (06/02/18)*, 2018

[175] Ethermint: *Ethermint - Cosmos Zone*. https://ethermint.zone/, 2019

[176] Linux Foundation: *A Blockchain Platform for the Enterprise — Hyperledger-Fabricdocs Master Documentation*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/, Februar 2019

[177] Paul, Moses S.: *Hyperledger — Chapter 6 | Hyperledger Fabric Components — Technical Context*. Mai 2018

[178] Rilee, Kynan: *Understanding Hyperledger Fabric — Byzantine Fault Tolerance*. Februar 2018

[179] Hyperledger Fabric: *[FAB-33] BFT-Based Ordering Service - Hyperledger JIRA*. https://jira.hyperledger.org/browse/FAB-33, 2019

[180] Hyperledger Fabric: *Hyperledger Fabric Documentation: Ledger*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html, März 2019

[181] Rouse, Margaret: *X.509 Certificate*. https://searchsecurity.techtarget.com/definition/X509-certificate, 2019

[182] Hyperledger Fabric: *Hyperledger Fabric Documentation: Blockchain Network*. https://hyperledger-fabric.readthedocs.io/en/latest/network/network.html, 2019

[183] Hyperledger Fabric: *Hyperledger Fabric Documentation: Cross-Chain Access*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/developapps/chaincodenamespace.html#cross-chaincode-access, 2019

[184] van Deursen, Arie ; Klint, Paul ; Visser, Joost: Domain-Specific Languages: An Annotated Bibliography. In: *ACM SIGPLAN Notices* 35 (2000), Juni, Nr. 6, S. 26–36. http://dx.doi.org/10.1145/352029.352035. – DOI 10.1145/352029.352035. – ISSN 03621340

[185] Booch, Grady ; Rumbaugh, James ; Jacobson, Ivar: *Unified Modeling Language User Guide, The, 2nd Edition*. 2nd. Addison-Wesley Professional. Part of the Addison-Wesley Object Technology Series series., 2005. – ISBN 978–0–321–26797–9

[186] Rouse, Margaret: *What Is Privilege Escalation Attack? - Definition from WhatIs.Com*. https://searchsecurity.techtarget.com/definition/privilege-escalation-attack, 2019

[187] Frankenfield, Jake: *Eavesdropping Attack*. https://www.investopedia.com/terms/e/eavesdropping-attack.asp, 2019

[188] Rouse, Margaret: *What Is Denial-of-Service Attack? - Definition from WhatIs.Com*. https://searchsecurity.techtarget.com/definition/denial-of-service, 2019

[189] ENTSO-E: The Harmonised Electricity Market Role Model. 2018. – Technical report

[190] SGAM Toolbox: *SGAM Toolbox – Modelling Aid for the Smart Grid Architecture Model*. 2019

[191] Lin, Jason: *Analysis of Blockchain-Based Smart Contracts for Peer-to-Peer Solar Electricity Transactive Markets*. Arlington, Virginia, Virginia Polytechnic Institute and State University, Diss., Dezember 2018

[192] Morey, Mathew J.: Power Market Auction Design – Rules and Lessons in Market-Based Control for the New Electricity Industry / Edison Electric Institute. 2001. – Technical report. – 96 S.

[193] Shamsee, Navaid ; Klebanov, David ; Fayed, Hesham ; Afrose, Ahmed ; Karakok, Ozden: *CCNA Data Center DCICT 640-916 Official Cert Guide*. Cisco Press, 2015. – ISBN 978–0–13–386045–0

[194] Hyperledger Fabric: *Hyperledger Fabric Documentation: Glossary*. https://hyperledger-fabric.readthedocs.io/en/latest/glossary.html, 2019

[195] Hyperledger Fabric: *Hyperledger Fabric Documentation: Endorsement Policies*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/endorsement-policies.html, 2019

[196] Hyperledger Fabric: *Hyperledger Fabric Documentation: Peers*. https://hyperledger-fabric.readthedocs.io/en/release-1.4/peers/peers.html, 2019

[197] Hu, Vincent ; Ferraiolo, David ; Chandramouli, Ramaswamy ; Kuhn, Richard: Attribute Based Access Control. In: *Attribute Based Access Control* (2017), S. 280–280

[198] TypeScript: *TypeScript — JavaScript That Scales*. https://www.typescriptlang.org/, 2019

[199] Montes, Walter: *Convector Smart Contracts Is Now Part of Hyperledger Labs, Hosted by The Linux Foundation*. März 2019

[200] Hyperledger Fabric: *Hyperledger Fabric Documentation: FabToken*. https://hyperledger-fabric.readthedocs.io/en/latest/token/FabToken.html, 2019

[201] IEEE: The Open Group Base Specifications Issue 7, 2018 Edition. (2018), Nr. Issue 7

[202] Sánchez, David C.: Zero-Knowledge Proof-of-Identity: Sybil-Resistant, Anonymous Authentication on Permissionless Blockchains and Incentive Compatible, Strictly Dominant Cryptocurrencies. In: *arXiv:1905.09093 [cs]* (2019), Mai

[203] Mustafa, M. A. ; Zhang, N. ; Kalogridis, G. ; Fan, Z.: DEP2SA: A Decentralized Efficient Privacy-Preserving and Selective Aggregation Scheme in Advanced Metering Infrastructure. In: *IEEE Access* 3 (2015), S. 2828–2846. http://dx.doi.org/10.1109/ACCESS.2015.2506198. – DOI 10.1109/ACCESS.2015.2506198. – ISSN 2169–3536

[204] Guan, Z. ; Si, G. ; Zhang, X. ; Wu, L. ; Guizani, N. ; Du, X. ; Ma, Y.: Privacy-Preserving and Efficient Aggregation Based on Blockchain for Power Grid Communications in Smart Communities. In: *IEEE Communications Magazine* 56 (2018), Juli, Nr. 7, S. 82–88. http://dx.doi.org/10.1109/MCOM.2018.1700401. – DOI 10.1109/MCOM.2018.1700401. – ISSN 0163–6804

[205] Mashima, D. ; Roy, A.: Privacy Preserving Disclosure of Authenticated Energy Usage Data. In: *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2014, S. 866–871

[206] Rial, Alfredo ; Danezis, George: Privacy-Preserving Smart Metering. In: *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*. New York, NY, USA : ACM, 2011 (WPES '11). – ISBN 978–1–4503–1002–4, S. 49–60

[207] Lee, C. H. ; Kim, K.: Implementation of IoT System Using Block Chain with Authentication and Data Protection. In: *2018 International Conference on Information Networking (ICOIN)*, 2018, S. 936–940

[208] Garcia, Flavio D. ; Jacobs, Bart: Privacy-Friendly Energy-Metering via Homomorphic Encryption. In: Cuellar, Jorge (Hrsg.) ; Lopez, Javier (Hrsg.) ; Barthe, Gilles (Hrsg.) ; Pretschner, Alexander (Hrsg.): *Security and Trust Management*, Springer Berlin Heidelberg, 2011 (Lecture Notes in Computer Science). – ISBN 978–3–642–22444–7, S. 226–238

[209] Li, F. ; Luo, B. ; Liu, P.: Secure Information Aggregation for Smart Grids Using Homomorphic Encryption. In: *2010 First IEEE International Conference on Smart Grid Communications*, 2010, S. 327–332

[210] Hyperledger: *Hyperledger Fabric Now Supports Ethereum*. Oktober 2018

[211] Zhu, Nicole: *Simply Explained: Blockchain Scalability Solutions Past, Present, and Future*. Juni 2019

[212] State of the DApps: *State of the DApps — A List of 2,667 Blockchain Apps for Ethereum, Steem, EOS, and More*. https://www.stateofthedapps.com/, 2019

[213] Energy Web Foundation: The Energy Web Chain: Accelerating the Energy Transition with an Open-Source, Decentralized Blockchain Platform. 2018. – Technical report

[214] The Share&Charge Foundation: *Share&Charge - Enabling The Open EV-Economy of Tomorrow*. https://shareandcharge.com/, 2019

[215] Libra Association: An Introduction to Libra / Libra Association. Geneva, Switzerland, 2019. – Whitepaper

[216] Sasson, E. B. ; Chiesa, A. ; Garman, C. ; Green, M. ; Miers, I. ; Tromer, E. ; Virza, M.: Zerocash: Decentralized Anonymous Payments from Bitcoin. In: *2014 IEEE Symposium on Security and Privacy*, 2014, S. 459–474

[217] Kan, L. ; Wei, Y. ; Muhammad, A. H. ; Siyuan, W. ; Linchao, G. ; Kai, H.: A Multiple Blockchains Architecture on Inter-Blockchain Communication. In: *2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, 2018, S. 139–145

[218] Abadi, Joseph ; Brunnermeier, Markus: Blockchain Economics / National Bureau of Economic Research. Version: Dezember 2018. http://dx.doi.org/10.3386/w25407. 2018 (25407). – Working Paper

[219] LO3 Energy: Exergy: Electric Power. 2017. – Technical Whitepaper

[220] Block, C. ; Neumann, D. ; Weinhardt, C.: A Market Mechanism for Energy Allocation in Micro-CHP Grids. In: *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, 2008, S. 172–172

[221] Gochermann, Josef: *Expedition Energiewende*. Springer Spektrum, 2016. – ISBN 978–3–658–09851–3

[222] Longstaff, Ben: *Hyperledger Fabric — the 20 Most Important Terms Made Simple*. Januar 2019