



Blockchain-based Application for Insurance Claims Management

Citation

Gillis, Steven. 2023. Blockchain-based Application for Insurance Claims Management. Master's thesis, Harvard University Division of Continuing Education.

Permanent link

<https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37375031>

Terms of Use

This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at <http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA>

Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. [Submit a story](#).

[Accessibility](#)

Blockchain-based Application for Insurance Claims Management

Steven Gillis

A Thesis in the Field of Software Engineering
for the Degree of Master of Liberal Arts in Extension Studies

Harvard University

May 2023

Abstract

Blockchain technology is a public ledger that offers transparency and trust due to “immutable” (unchangeable) and auditable data. The industry for blockchain-based insurance only first reached scale recently. The ability of applications to automatically execute on the blockchain leads to cost reductions and scale advantages. In this thesis, we demonstrate the claims process is more (cost) efficient and transparent however lacks standardization and trustworthiness when benchmarked against traditional insurance products. To address these limitations, we developed the Blockchain Claims Standard Application (BCSA). The BCSA application is a novel solution that enables a standardized approach to claims management for blockchain-based insurance products. Access to the application is available through a user interface as well as ABIs (Contract Application Binary Interface) for integration with existing blockchain-based insurance products. The solution enables improvements for claims management over existing solutions in terms of standardization, efficiency, and trustworthiness to allow the adoption of blockchain-based insurance.

Acknowledgment

I would like to thank my Thesis Director, professor Daniel Katz for his contribution to this thesis. The theoretical foundation is greatly strengthened by his knowledge of the established practices in the blockchain community and his deep understanding of topics at the intersection of law and technology.

I would like to express my gratitude to professor Hongming Wang, my Research Adviser, for her invaluable guidance and support during my thesis.

Special thanks go out to my wife, Katie Ayling, not only for reviewing the scientific writing of my thesis but mostly for having my back during long days and nights of writing and coursework while our newborn son Kodi is crying for attention.

Table of Contents

| | |
|--|-----|
| Abstract..... | iii |
| Acknowledgement | iv |
| List of figures..... | vii |
| Chapter I. Introduction..... | 1 |
| Chapter II. Background..... | 4 |
| Digitization in insurance claims processing..... | 4 |
| Blockchain technology and decentralized applications | 6 |
| Decentralized Autonomous Organizations (DAO)..... | 8 |
| Emergence of insurance on the blockchain..... | 10 |
| Claims management on the Blockchain..... | 12 |
| Limitations of blockchain-based claims processing | 14 |
| Parametric insurance..... | 16 |
| Prior Work | 18 |
| Chapter III. Validation of prior work..... | 20 |
| Feasibility of automation | 20 |
| Automation along the claims process | 22 |
| Data analysis Nexus Mutual claims..... | 24 |
| UST de-peg case study..... | 27 |
| Summary of validation..... | 29 |
| Chapter IV. Blockchain Claims Standard Application..... | 31 |

| | |
|---|----|
| System Architecture..... | 31 |
| Ethereum blockchain..... | 33 |
| Solidity programming language..... | 34 |
| System components..... | 36 |
| Functional complexity: Creation of a Standard | 38 |
| Scope definition | 42 |
| Chapter V. Implementation..... | 43 |
| Domain Model | 43 |
| Sequence diagram | 48 |
| State Diagram..... | 50 |
| Deployment..... | 51 |
| Application interface..... | 53 |
| Chapter VI. Results..... | 58 |
| Results overview..... | 58 |
| Discussion..... | 60 |
| Recommendations..... | 61 |
| Chapter VII. Conclusion | 65 |
| Glossary | 66 |
| Appendix Code repository and Demo..... | 68 |
| References..... | 69 |

List of figures

| | |
|---|----|
| Figure 1: Process steps in insurance value chain..... | 5 |
| Figure 2: Overview of benefits and limitations in prior work | 19 |
| Figure 3: Overview of possible automation theoretically and existing in the industry | 23 |
| Figure 4: Spread of voting power in Nexus Mutual claims governance..... | 25 |
| Figure 5: Graph of the number of days from claims submission to decision on payout .. | 26 |
| Figure 6: System Architecture | 32 |
| Figure 7: Part of ABI of claims contract..... | 35 |
| Figure 8: Claims management sub-process by Mahlow et al. (2015)..... | 36 |
| Figure 9: Domain model | 44 |
| Figure 10: Example of Solidity code for contract claims.sol..... | 45 |
| Figure 11: Solidity code for contract Events.sol..... | 46 |
| Figure 12: Solidity code for NFT identifyNFT.sol..... | 47 |
| Figure 13: Sequence diagram..... | 48 |
| Figure 14: State diagram of claims object | 50 |
| Figure 15: Script to deploy smart contract code to the blockchain..... | 52 |
| Figure 16: Interface of overview page | 53 |
| Figure 17: Interface of submitting a claim..... | 54 |
| Figure 18: Interface of claims manager submitting an event..... | 55 |
| Figure 19: Interface of Metamask wallet with Identity NFT | 56 |

Chapter I.

Introduction

Insurance is a risk management tool for organizations or individuals to shift the liability for specific events to another party in exchange for a guaranteed payment of premium (e.g. in health insurance you pay a monthly premium and the insurance company will pay the medical bill in case you fall ill). The traditional insurance industry is built on trust in institutions that are hundreds of years old (Swiss Re, the world's largest reinsurance company was founded in 1863). Based on this trust, insurance has become one of the largest industries in the world, roughly consisting of 6% of global GDP (Gross Domestic Product) (*OECD Data, 2022*).

Private insurance companies provide benefits to the policyholders as a central entity. They create value by maintaining historical data and actuarial practices to calculate what premiums will be sufficient to cover expected claims. On top of that, governments can regulate these entities ensuring sufficient funds are available.

Before private insurance companies existed, the peer-to-peer insurance model has proven valuable as early as Babylonian and Roman cultures for burial costs. In the Middle Ages this model was used to insure cargo loss among fellow shipmates and guilds used this method to look after their sick colleagues. This model encourages solidarity between members and cultivates an attitude toward social welfare founded on independence and self-help (Boyle et al., 2021).

Blockchain technology is a fast-growing technology with the potential to disrupt the insurance industry. A blockchain is a distributed ledger where transactions are publicly available for anyone to read. Blockchain technology has the potential to redefine the known concept of trust, thus redefining the trust-based insurance industry: Instead of trusted, centralized institutions trust could come from rules defined in immutable (cannot be changed) computer code. The trust created by immutable code enables a peer-to-peer insurance model where individuals come together and pool their resources for their mutual benefit.

In 2020 a study was published by Popovic et al. (Popovic et al., 2020) in the British Actuarial Journal assessing the opportunities of Blockchain in the insurance industry. The mentioned benefits are immutability (data & code which cannot be changed), auditability, scalability, and increased engagement from the customers. A year later a publication from their American counterpart found that blockchain technology can provide insurance with increased efficiency due to automation between agents (Boyle et al., 2021). The article emphasizes that peer-to-peer insurance, coupled with Blockchain technology, does not require a centralized authority to ensure a trustworthy environment. Boyle et al. conclude that the practical need for an intermediary insurance company is removed when premium payment, claims processing, and payment are processed automatically on the blockchain. The industry is exploring how (partial) automation can replace the trusted central party in judging whether an insurance claim is to be paid. Currently, there is a lack of trust in decentralized insurance and many questions arise about the fairness of claims payout. Also in traditional insurance, claims management has been treated as a necessary part of operations and the competitive advantage in terms of

customer satisfaction and retention has often been overlooked (Mahlow & Wagner, 2016a).

The BCSA application is designed to improve existing solutions to claims management on the blockchain. To define improvements, a review is done of prior (scientific) works. The outcome of this review is an overview of the benefits and limitations of blockchain-based insurance. Using data analysis and case studies of existing blockchain-based insurance products, the overview of benefits and limitations is validated, and improvements are defined. A technical design of the BCSA is included in this thesis as well as details on the implementation.

Chapter II.

Background

Digitization in insurance claims processing

According to the Oxford dictionary, Insurance is "an arrangement which a person or company undertakes to provide a guarantee of specific compensation for specified loss, damage, illness or death in return for payments of a specified premium" (Oxford University Press, 2022). The compensation for the specified loss will only be paid out under specific circumstances which are defined in an insurance policy document. For example, an individual which has a lot of assets in a digital wallet on the blockchain, in case his wallet is hacked, might lose the assets. To mitigate this risk, the individual can decide to buy insurance and in return pay a small fee (typically between 1% and 5% of the value of the assets) and will get their loss refunded in case they are hacked.

For the remainder of this article, the model found in Figure 1 is used to describe the insurance process. Most insurance companies use a slightly different process and hence a generalization is used which is found in literature. To facilitate an insurance contract, the insurance companies undertake 5 phases:

- i. In the Product development phase, the conditions under which a claim is paid out are defined in the insurance policy and actuarial data is used for the pricing of the product.
- ii. In the Sales phase, the insurance company sells the product (through agents) to a client which wants to insure themselves against a certain risk.

- iii. In the underwriting phase potential customers are handled based on their risk profile and during administration, the insurance company facilitates premium payment and is available for requests for data or changes of contract data.
- iv. Risk management ensures that all risks are analyzed, and sufficient capital is available to pay future claims. Part of risk management is to assure the sum of paid claims is not larger than the sum of the insurance premium paid.
- v. The claims management phase involves the decision on the validity of a claim and the payout of this claim (Eling & Lehmann, 2017).

The claims management phase is the focus of this thesis. The next section will define the sub-steps in the claims management process. Mahlow and Wagner interviewed a range of insurance companies to create a theoretic framework for the process steps

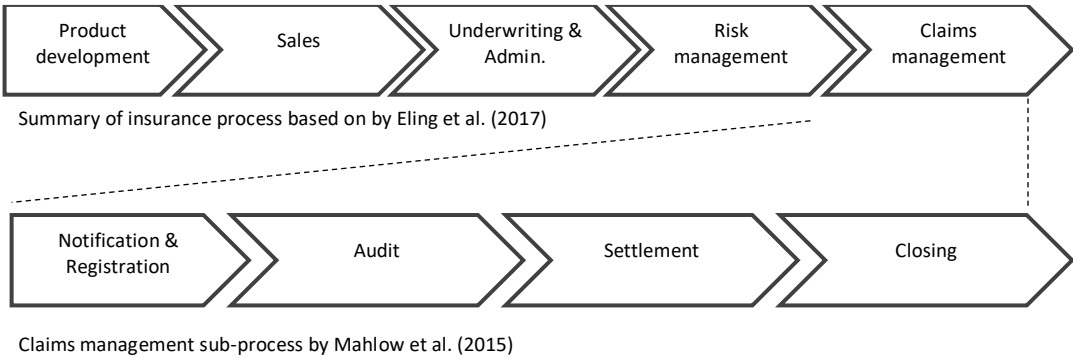


Figure 1: Process steps in insurance value chain

involved in Claim Management. Claims management consists of four phases as shown in Figure 1 (Mahlow & Wagner, 2016b):

- i. Notification and Registration - The clients notify the insurance company of their loss and the insurer registers the claim in the claims system

- ii. Audit - The insurance company determines if the claim is covered by the policy
- iii. Settlement - The insurer determines the settlement amount. This can be done with or without an external Claims Manager
- iv. Closing - The customer receiving the payout

Digitization had an enormous impact on the insurance industry and manifested around enhancements of the customer experience, improvements of business processes, and changes to the insurability due to new and more information created (Eling & Lehmann, 2017). This thesis explores the potential of further digitization with blockchain technology in particular.

Research from 2015 in Germany and Switzerland found that for a typical insurance company, claims are roughly 70% of their annual direct costs, and 10% to 20% of their staff work in claims management (Mahlow & Wagner, 2016b). Historically the claims management process has benefited from digitization due to the rise of big data and AI to support automated audit and settlement steps to calculate the amount of payout and damage and prevention of fraud due to data analytics. Eling et al. also identified that claims management can be improved by blockchain technology due to the storage of information for automated payouts due to smart contracts and customers being able to file claims through their smartphones for immediate payout (Eling & Lehmann, 2017).

Blockchain technology and decentralized applications

The start of blockchain technology was marked by a whitepaper published by Satoshi Nakamoto in 2008 (Nakamoto, 2008) It introduced a peer-to-peer version of electronic cash called Bitcoin. It is a technology that uses a decentralized ledger to store

transactions without the supervision of a centralized authority. Instead, the software is run by multiple nodes each containing a copy of the Blockchain's data which can be read by anyone at any time. This concept of auditability is a great advantage of blockchain technology according to Popovic et al. (Popovic et al., 2020). The data is encrypted on each node and together they validate the data through algorithms, hence no single node can rewrite or delete the history of data on the blockchain making the data “immutable” (Berryhill et al., 2018).

It is often misunderstood that the concept of a Smart Contract came into existence due to blockchain technology. Nick Szabo provided the first definition of a smart contract in 1997 (Szabo, 1997): “The smart contract is a secure, machine-readable and executable program that can automate specified procedures, including those used in legal contexts.” These contracts will be highly efficient as they are fully automated. They are also highly secure when executed on the blockchain because encryption mechanisms which protect the stored coins from attackers.

In 2014, Vitalik Buterin introduced a blockchain-based technology called “Ethereum” which enabled Decentralized Applications (dApp) (Buterin & others, 2014). On top of blockchain’s ability to store data in a decentralized fashion, now there are decentralized software applications: these smart contracts-based software programs have one strong advantage: There is the possibility to ensure no single entity controls the application and can perform changes to the data and code, this ensures it is encrypted and tamper-free. Encryption is important because many customers are afraid of insurance companies getting their hands on their personal data (Lorenz & Münstermann, 2016). For example, health insurance companies are known to reject customers or increase the price

in case there is a previous medical history. You can protect access to a user's data from insurance companies by storing the data encrypted on a public blockchain and allowing it only to be accessed with explicit consent.

Often the scalability, performance, and energy consumption are mentioned as limitations of blockchain technology (Gatteschi et al., 2018). These factors were indeed a problem for the implementation of the blockchain at the time of writing of those articles. Yet, they are not a limitation of the blockchain concept and future implementations of the blockchain. In 2022, the main computation blockchain called Ethereum moved to a proof-of-stake consensus algorithm that will address energy consumption. Also, many extensions to the Ethereum infrastructure are being introduced (so-called "layer-2") which will increase the computational and storage capabilities of Ethereum. These solutions will address the scalability and performance issues. Another often-mentioned downside of anonymous access to the blockchain is that losing access to your address or wallet means you will also lose your insurance products (Gatteschi et al., 2018). In the case of blockchain, there is no customer service agent which can retrieve your address for you which would have been available at a traditional insurance company.

Decentralized Autonomous Organizations (DAO)

With the emergence of decentralized technologies, also a new way of collaborating is emerging which also tries to avoid centralized decision-making. Decentralized Autonomous Organizations (DAOs) are a way of collaborating without a centralized hierarchy. In a DAO, the law is defined by code. All the management and operational rules come from group decision-making and are encoded in a tamper-resistant

blockchain (Wang et al., 2019). The DAO enables stakeholders with different interests to negotiate their rights and obligations and then program them into contracts on the blockchain for distribution and auto-execution. The DAO is different from a traditional legal entity in that it is not confined to a single geography or county. According to Wright et al., the hope is that DAOs improve existing legal entities by being digitally native, easy to join, and global in reach, making the organization as scalable as the technology (Law Aaron Wright, 2021).

DAOs can give their members access to specific rights, like access to (future) profits and engaging in the organization's decision-making process. Governance often is less hierarchical and does rely more on group consensus. Due to transparency, there should be less contested decision-making, mistakes, and fraudulent behavior.

The first DAO was for bitcoin itself. Bitcoin miners vote on protocol upgrades in a way resembling community-based management of Open-Source software such as Linux(Hsieh & Vergne, 2018). An interesting development is that those who contribute to the operation of bitcoin (a.k.a. miners) are paid an incentive automatically for their contribution. Miners consent to play by the rule book, but they can vote to change the rules using the influence they get from their contribution.

The DAO system of voting works if the members of the DAO have the best (long-term) interest of the DAO and their customers in mind. Creating the right incentives for participants in the DAO is important (Wang et al., 2019). This can be done by an incentive scheme which is a win-win situation for all participants. In many cases, the DAO will issue its token to manage the incentives with so-called “token economics”. The

end-goal of the (insurance) DAO should be to reach low-cost or near-zero-cost of transactions.

Wright et al. define two types of DAOs: Algorithmic and participatory (with voting) (Law Aaron Wright, 2021). Currently, insurance protocols use a combination of both models. Premium payments and pricing are done algorithmically. Processes that are more ambiguous like making governance decisions and on claims payout are done participatory. In Chapter III, this thesis will explore whether insurance protocols have successfully embraced the DAO governance model and avoided centralized decision-making, and capture the promised efficiencies in the claims process by creating an effective incentive system.

The emergence of insurance on the blockchain

Since the rise of Ethereum and the blockchain, insurance has been identified as a potential major area of disruption: In a report from Goldman Sachs in 2016, it was estimated that blockchain technology could generate \$2B to \$4B in annual cost savings for real estate title insurance in the US alone due to reduction of errors and manual work (Schneider et al., 2016). A study from 2018 predicted it would take at least 3 to 5 years before it can be determined if investments in blockchain are paying off for businesses (Gatteschi et al., 2018).

Nexus Mutual launched in 2019 as one of the first decentralized insurance protocols and the first to reach commercial viability. They have been the market leader since and in May 2021, Nexus Mutual had over \$1.1B in insured assets. They were different than other blockchain-based insurance products because their first product -

insurance against hacks - solved a problem experienced by a crypto-native community that understands the benefits of blockchain technology already. They utilize the peer-to-peer insurance model where users can buy insurance against their crypto currencies being hacked. The capital for the insurance is provided by peers who lock their crypto currencies in a smart contract in exchange for a small fee (called staking). If no hack happens, the "staker" walks away with his initial capital plus a fee. In case there is a hack, the user which bought the insurance will get a claims payout equal to the loss of the hack from the tokens locked in the smart contract. The processes of Sales, Underwriting, and Risk Management (see Figure 1) at Nexus Mutual are automated on the blockchain and do not require any intermediaries.

Some of the non-core processes (pricing, claims research, sales, software development, business development, risk management) are not automated on the blockchain (yet) and are performed by members of the Nexus Mutual DAO which are rewarded for their contribution with the NXM crypto currency. Owners of this NXM currency are also allowed to participate in governance votes of the DAO. To date, Nexus Mutual has paid \$8M in claims (*Nexus Mutual Tracker*, n.d.)

After the initial success of Nexus Mutual, other insurance protocols started to emerge with slightly different business models. The most notable runner-up is InsurAce which is not only available on the Ethereum blockchain but also at various alternatives (Polygon, Binance Smart Chain, Avalanche) which - in the summer of 2021 - reduced the gas cost of buying an insurance policy from over \$300 to a few dollars. They also greatly increased the capital efficiency in the Underwriting phase by improving the staking process. InsurAce also introduced a new type of insurance product called "stablecoin de-

pegging": Some crypto currencies – called “stablecoins”- promise that they will always be worth 1 US dollar. Now users can insure themselves against this promise being broken. In June 2022, InsurAce had to pay over \$12M in claims, for the UST stablecoin losing its value of 1 dollar. Although this payout was a quarter of all the capital available to InsurAce, it was proof that decentralized insurance is living up to their expectations and mitigating the risks for the buyers of the insurance.

In 2021 and 2022 a range of new insurance protocols emerged: Bridge Mutual, Tidal, Ease, Unslashed, UnoRe, Solace. Each of them provides a new smart contract with some innovation compared to the way of working from Nexus Mutual. Most of them still embrace the peer-to-peer insurance model supported by smart contracts and a claims process based on a voting mechanism. To date, no insurance protocol has been able to build all processes (from Figure 1) on the blockchain in a fully automated way.

Claims management on the Blockchain

Claims processing was mentioned by Popovic et al. as a large opportunity for automation on the blockchain: “The terms of the insurance product are written into a smart contract which automatically pays out claims upon receiving the right parameters from publicly available data: for example, flight delay, extreme weather, natural catastrophes or death of a person”. A key advantage is that “Claims are recorded on the blockchain for auditability to prevent multiple claims on the same insured event” (Popovic et al., 2020). According to authors from the North American Association of Actuaries, there are efficiency gains from running the claims processing on the blockchain, especially when claims can be approved automatically when appropriate

conditions are met and the claims can be automatically enabled, executed, and recorded by the blockchain technology (Boyle et al., 2021).

Nexus Mutual has been able to partly automate the claims management process (see Figure 1). The process is started by the user in case they are at loss due to a hack event which is covered under their insurance policy.

- i. The user must notify Nexus Mutual that they believe they are entitled to a payout. As part of the notification, you need to provide proof that you owned the hacked. Your claim is automatically registered on the blockchain and publicly available.
- ii. The Audit and Settlement phase are combined into one step: The members of the DAO (which are claim assessors and owners of the DAO at the same time) will vote whether the hack has taken place fall within the insurance policy and whether your proof is strong enough.
- iii. Once the claim is approved the member can trigger the claim payout from a Smart Contract. This is fully automated on the blockchain.

The voting process has not been automated because there is some ambiguity on whether the claim should be paid out. Before the voting takes place, experts of the DAO will research the hack and make a non-binding recommendation on whether the conditions under which the hack took place fall within the policy conditions of the insurance contract which was bought. In case there is a strong consensus of more than 70% of the votes, the outcome is clear, and they claim will be approved or denied. In case there is no consensus, all the members of the DAO are allowed to vote based on a simple majority (>50%). It should be noted that in the last few months of 2021, it was very

expensive to vote, to participate in approving a claim, the DAO members had to pay over \$100 in gas fees to operate the Ethereum blockchain. When a claim is paid out this will negatively impact the capital of the DAO. Because claims managers also own the token of the DAO they have a vested interest in both the success of the protocol and the capital of the DAO. This might be seen as a conflict of interest because the claims manager would benefit by reducing the number of claims being paid.

Until the end of 2022, Nexus Mutual has paid just over \$9M in claims (and received \$27M in premium) (*Nexus Mutual Tracker*, n.d.) and their competitor InsurAce paid nearly \$12M in claims (and received just over \$2M in premium) (InsurAce, n.d.).

Limitations of blockchain-based claims processing

To determine which data is “true”, blockchains have multiple nodes trying to reach a consensus. An important limitation is the possibility of bad actors purposely voting against what they know is the truth (Katsh & Rabinovich-Einy, 2017). The crypto community has coined the term 51% attack for a scenario where it is beneficial to own over 50% of the capital (for a very short time) to turn a vote to your own benefit. Gencer et al. show that the top four miners at one point in time owned 53% of the total mining capacity of Bitcoin and the top three miners for Ethereum hold 63% of the Ethereum mining capacity (Gencer et al., 2018). The top 90% of the total mining power for bitcoin is owned by 16 miners and by the top-11 for Ethereum. This shows that even the most well-known blockchains are not as decentralized as they seem. The same scenario can also happen in a voting-based insurance system, especially since the claims manager is not independent and could be impacted financially by the claim being accepted or rejected. By design, the claims manager is also an investor in the insurance provider.

They are incentivized against doing a payout as it will result in a personal loss. Another case is where the claims manager submits a claim himself and makes sure to own sufficient capital in the staking pool to also approve the claim.

According to the founder of Nexus Mutual, a strong incentive scheme needs to be developed for governance decisions, so that independence is not required: In a peer-to-peer model there needs to have sufficient incentive to report and a strong disincentive to prevent fraudulent reporting (Karp & Melbardis, 2017). This can be troublesome in insurance since you can buy a policy for a low amount and receive a very high amount when the insurance pays out. Their suggested approach is to ensure that the claims assessors have a high enough stake in the mutual itself in the form of a membership fee and therefore have an incentive for the mutual to succeed long-term. There will be an advisory board that can punish dishonest claims assessors by burning their crypto assets. Another incentive to act honestly is voting with the consensus entitles the claims assessor to a part of the insurance premium. It is to be noted that in traditional insurance there also is a strong incentive for claims managers to reject a claim as they are employed by the insurance company which will pay-out the claim. Hence the good practice of external audits has been introduced.

An alternative view is provided by Katsh, arguing that there are three conditions for fair decentralized decision-making: It is voluntary, it is a paid activity and the jurors are picked at random (Katsh & Rabinovich-Einy, 2017). The amount which has to be paid needs to compensate for the skill and time required of the claims managers and can differ for each type of claim. In their model, each round of appeal will more than double the number of jurors or claims managers to look at the case and increase the arbitration fee

by the same amount. It is up to the losing party to decide whether they accept the appeal. In the case of the insurance claim the fees should scale with the size of the claim as they typically vary between 1000 dollars to millions of dollars. The biggest difference with Nexus Mutual is that claims assessors are selected based on their skill level instead of their ownership of the DAO.

Popovic et al. mention several challenges for claims management on the blockchain: There are no proven standards or platforms for claims management across the industry (this will be addressed in the recommendations). Also, there needs to be trustworthy data available to trigger the events, the unavailability of data is often a limitation to being able to automatically execute decisions on the blockchain (Popovic et al., 2020).

With the emergence of many more blockchain-based insurance products in 2021 and 2022 there also where many improvements to the claims process. The first major improvement is that a dispute process is introduced during which a user can fight the conclusion of the voting process. This is to mitigate the fear that claims managers can maliciously vote against a claim which is valid (for example because they are also staking in the insurance protocol and might lose a lot of capital). Having a dispute process in place is also one of the best practices for claims management suggested by the OECD (*OECD Guidelines for Good Practice for Insurance Claim Management*, 2004).

Parametric insurance

Another innovation is the introduction of fully automated insurance (also called parametric insurance), in this case, the voting process is removed in favor of a fully

automated claims assessment based on data. This is for example possible for the "de-pegging insurance" from InsurAce. In case a stablecoin is not traded for 1 US dollar for an extended amount of time, the claim should be automatically paid out. Multiple insurance providers have created examples of how to create fully parametric insurance. The key advantage here is that no claim needs to be submitted by the holder of the insurance. A great practical example is provided by Chainlink (an organization providing data on the blockchain) (Papacharissiou, 2020), where a practical solution is offered to automatically payout insurance to farmers in case certain weather conditions have taken place which negatively impact their crop earnings. Another great opportunity for parametric insurance is the emergence of IOT (Internet Of Things) consisting of millions and millions of sensors. For instance, homes can be equipped with sensors to notify a contract in case of damages (e.g. a damp sensor in the roof checking leakages) As mentioned by Popovic et al., in most cases there is no trustworthy data available to trigger the events. Some insurance types like flight delay or weather-based insurance can be well established from public sources, but in many cases, it also concerns non-public and even private data (e.g. in health insurance) (Popovic et al., 2020).

Prior Work

This section creates an overview of all benefits and limitations within prior work. These are summarized in Figure 2 on the next page. The figure contains a count of how often the benefits and limitations were mentioned across all sources: The remainder of this thesis is on all benefits which are mentioned by four or more authors:

- i. Lower Cost/efficiency
- ii. Decentralization (of technology and governance)
- iii. Transparency / Auditability
- iv. Decision speed

The other benefits mentioned by two or more sources are Improved community engagement, higher scalability & interoperability, and reduction of fraud, security, and privacy concerns. Limitations of blockchain technology found in the literature are misalignment of incentives (in governance), lack of trustworthy data (which needs to be checked by humans) to automate processes, and the absence of standards.

| Author | Year | Title | Benefits | | | | | | | | Limitations | | | | | | | | | | |
|---------------------------|------|---|-------------------|--------------------------------|--------------------------------------|-------------------------|------------------------------|--------------------------|--------------|----------------------------|-------------|------------------------|--------------------------|----------------------------|---------------------------------------|---|--------------------------------|--------------------|-----------------|-------------------|---|
| | | | Cost / efficiency | Decentralized / permissionless | Transparent / Auditable / Verifiable | Decision speed / claims | Improve community engagement | Scalable / Interoperable | Reduce fraud | Security / Trust immutable | Privacy | Missaligned incentives | Lack of trustworthy data | Oversight / appeal process | Sufficiently trained claims assessors | Loss of insurance policy if address is lost | Bad Actors & false information | Energy consumption | Lack of privacy | Lack of standards | |
| Mahdi Farnaghi | 2020 | Blockchain and smart contracts for insurance: Is the | x | x | | | | | | | | | | | | | | | | | |
| Szabo | 1997 | The idea of smart contracts | x | x | | | | | | | | | | | | | | | | | |
| Subramanian | 2017 | Decentralized blockchain-based electronic marketplaces | x | x | | | | | | | | | | | | | | | | | |
| Wang | 2019 | Decentralized Autonomous organization: Concept, | x | x | | | | | | | | | | | | | | | | | |
| Chainlink | 2020 | Build a Parametric Insurance Smart Contract With | x | x | | | | | | | | | | | | | | | | | |
| sleeth | 2018 | Blockchain and contract theory: Modeling smart | x | x | | | | | | | | | | | | | | | | | |
| Gatteschi | 2018 | Blockchain and smart contracts for insurance: Is the | x | x | | | | | | | | | | | | | | | | | |
| Popovic et al | 2020 | Understanding blockchain for insurance use cases | x | x | | | | | | | | | | | | | | | | | |
| Karp | 2020 | Nexus Mutual, A peer-to-peer discretionary mutual on | x | x | | | | | | | | | | | | | | | | | |
| Kleros.io | 2020 | Dispute Revolution, handbook of decentralized justice | x | x | | | | | | | | | | | | | | | | | |
| McKinsey | 2016 | Blockchain in insurance – opportunity or threat? | x | x | | | | | | | | | | | | | | | | | |
| Boyle Et Al | 2021 | P2Pinsurance: Blockchain implications | x | x | | | | | | | | | | | | | | | | | |
| Martin et al | 2018 | The impact of Digitalization on the insurance Value Chain | x | x | | | | | | | | | | | | | | | | | |
| sleeth | 2020 | Blockchain and contract theory: modeling smart | x | x | | | | | | | | | | | | | | | | | |
| Wright et Al | 2021 | The Rise of Decentralized Autonomous Organizations: | x | x | | | | | | | | | | | | | | | | | |
| Davis J. | 2018 | Peer to peer insurance on an Ethereum blockchain | x | x | | | | | | | | | | | | | | | | | |
| Number of mentions | | | 11 | 9 | 8 | 7 | 4 | 4 | 4 | 3 | 3 | 1 | 5 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 |

Figure 2: Overview of benefits and limitations in prior work

Chapter III.

Validation of prior work

This chapter will validate the potential benefits and limitations of insurance from prior work. The validation will consist of three parts: (i) The first part will use a theory from legal literature to assess the feasibility of automating insurance contracts on the blockchain. (ii) The second section is a data analysis of insurance claims submitted to Nexus Mutual in the period from January 2021 to August 2022. (iii) The third part is a case study of an event that led to the largest insurance claims in the history of blockchain technology to validate the benefits mentioned in prior work.

Feasibility of automation

This section will explore the feasibility of automating the insurance claims process on the blockchain. An important principle for the blockchain community is to fully automate processes on the blockchain without any human intervention. An example of this is Uniswap, a “decentralized money exchange”. When digital money (tokens) are exchanged for other digital tokens, the price is automatically set by a mathematical calculation that is completely executed on the blockchain. The exchange of digital money on Uniswap is simple and no external data is required.

Opposingly, a famous quote in the legal Contract Theory from Nobel prize winner Oliver Hart is: "all but the simplest contracts are incomplete" (Hart, 2017). In a Complete Contract, all that can ever happen is written in the contract. In reality, (legal) contracts are poorly worded, and ambiguous and leave important things out. In addition, the world in which the contract is used is complex and dynamically changing (Hart, 2017). In the case, a scenario plays out which is not described in the

contract a decision still needs to be made: This decision is made by a person who has Residual Control Rights. The question of who this person is has no straightforward answer and has an entire body of literature dedicated to it. What is clear is the person that holds the Residual Control Rights should be clearly stated in the contract.

Some examples of this ambiguity are the hacks that have taken place and were not paid out because it was not the blockchain part of the application being hacked but the interface which was shown to the user. An example of this is Curve Finance which halted operations because its application was hacked and \$570K was stolen. The contract did not specify that front-end hacks were not included in the insurance policy (and in the smart contract). Bourgeon et al. are going even further by stating that the completeness of a contract is undesirable due to the cognitive load for the buyer to understand the policy and the cost to the insurer to audit and underwrite the contract (Bourgeon & Picard, 2020). It can be more costly to research the conditions of how the hack occurred, instead, it could be more cost-efficient for the insurance company to generalize the policy rules and have a specialist deal with the ambiguous scenarios.

As described by Jesse Walden, we will define the specific use cases in the insurance value chain whether they are complete or incomplete (Jesse Walden, n.d.). An example of an incomplete contract in the blockchain space is MakerDAO. This DAO manages a stablecoin called DAI and its goal is to keep the 1:1 value to the US dollar. To do so, a lot of decisions need to be made in an ambiguous situation. One of these decisions is the interest that needs to be paid to borrow or lend DAI from MakerDAO. Because there is too much ambiguity, community votes are used to make these decisions.

Sheth et al., go even further to state that -based on the incomplete contract theorem- incomplete contracts are the reason why economic organizations and

governance exist (Sheth & Subramanian, 2020) (Hart, 2017). If contracts were complete, and transaction costs were zero, all transactions would be efficient and there is no need for economic organizations. The cost of transactions most often comes from 1) information uncertainty or unforeseen contingencies 2) and the cost of contract creation, storage, retrieval, and enforcement. Automation on the blockchain can address the second category but not the first. On top of this, issues due to information asymmetry such as adverse selection and moral hazard will not disappear due to the introduction of the blockchain (Sales for a certain insurance product increase just before a hack takes place).

Automation along the claims process

This section will explore the extent to which the insurance claims process could theoretically be automated and compare this against the existing process in the industry. This analysis is done for the two main use cases now seen in the industry: 1) insurance against stablecoins de-pegging and (2) insurance against hacks. An overview is found in Figure 3. In de-pegging insurance, the buyer gets refunded automatically when a token loses its 1-1 link with the dollar. It is an example of a parametric insurance product that is fully data-driven. Data is available to make decisions therefore it is certain whether the claims can be paid out, and what the amount should be. In this case, the insurance contract can be regarded as a “complete contract” as by the definition by Hart et al. (Hart, 2017). Hence, fully blockchain-based insurance is theoretically possible but the case study later in this chapter will show that the implementation is not fully automated.

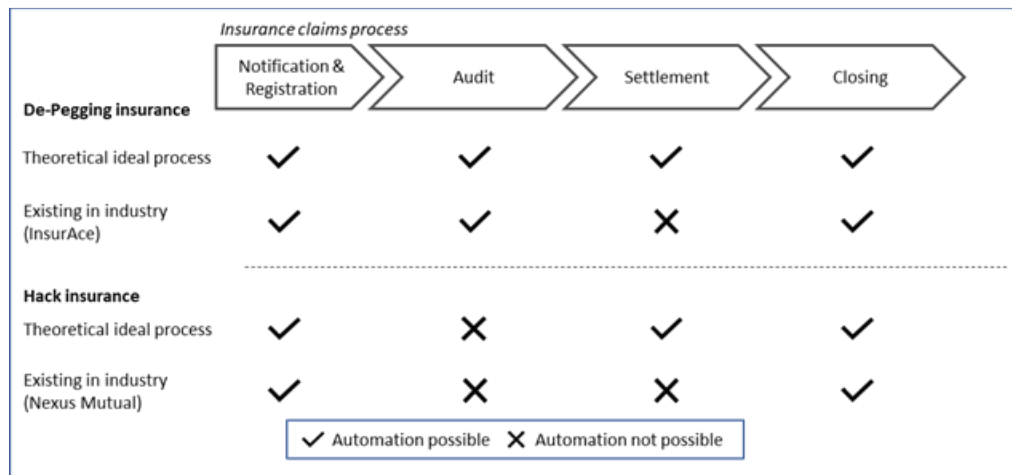


Figure 3: Overview of possible automation theoretically and existing in the industry

The existing industry process for hack insurance (for Nexus Mutual) is described in a previous section “Claims Management on the Blockchain”. Nexus Mutual already automated the Notification & registration and closing steps. Theoretically, it should be possible to automate the settlement process step in which the loss to the buyer of the insurance product is established. This can be done by always paying the full amount for which the insurance policy has been bought, this is described in recommendation number 4 of this thesis. Unfortunately, with existing technology, it will not be possible to establish automatically whether a hack has taken place and what the root cause of this hack is. In traditional insurance, establishing the root cause of hacks is done by accredited cyber security experts. As part of the recommendations of this thesis, this expert-based approach is also recommended for blockchain-based insurance as the reliability is higher than for vote-based decisions.

Data analysis Nexus Mutual claims

This analysis uses industry data from the largest insurance provider to validate the potential benefits of Chapter II. The data from 107 insurance claims is used from Nexus Mutual in 2020. Three of the potential benefits identified in Chapter II will be addressed in this section: decentralization (of governance), transparency/auditability, cost/efficiency/scalability, and decision speed.

i. Efficiency/cost

The cost of claims management in traditional insurance companies often includes the salaries of back-end employees, claims auditors, cost of external experts, and IT investment and operations cost. In total, these costs are typically between 5% to 15% of the overall premiums (Mahlow & Wagner, 2016a). In the period 2020-2021, there were \$7M in premiums annually which would lead to claims cost of \$350K-\$1.05M when using the benchmarking numbers of the traditional insurance industry (*Nexus Mutual Tracker*, n.d.). The total cost of handling the 107 claims would be around \$100-\$500 in cost each (gas costs and incentives for claims managers) totaling \$10K to \$50K in total cost. Hence it seems that the claims management activity performed by Nexus Mutual is significantly cheaper than the traditional insurance industry.

ii. Decentralization & incentive system

An important metric for decentralized governance is how much the decision power is spread across all voters. The data shows that in 30 cases (28%) only a single person voted (which likely is an employee of Nexus Mutual). For 82% of the votes, a single person (or address on the blockchain) was responsible for more than 50% of the voting power, basically deciding the direction of the vote themselves. There is no example where the person with the most voting power had less than 20% of the total

votes. It can be concluded that the decision-making of Nexus Mutual has not been “decentralized” in the period 2020-2021. Two reasons are often mentioned as potential causes for this: The cost of voting on Ethereum can be quite costly (>\$100 in December 2021) and the community is not always aware of the voting taking place. This is understandable as 83% of the payouts are done within 3 days of the claim’s submission.

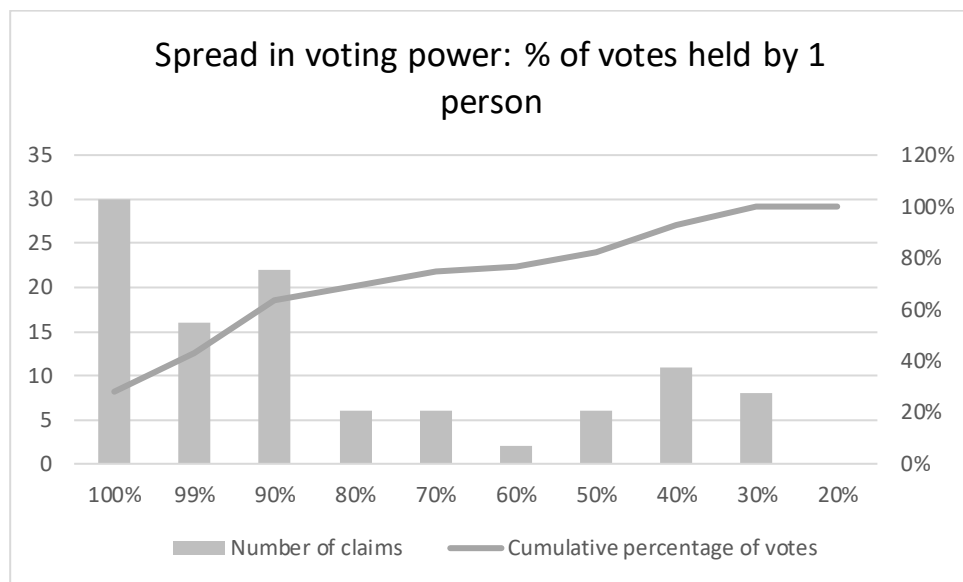


Figure 4: Spread of voting power in Nexus Mutual claims governance

iii. Transparency & auditability

All the claims data of Nexus Mutual is found on the Ethereum blockchain and they have also made a graphical interface to analyze the data (<https://nexustracker.io/claims>). Most of the blockchain insurance providers have done the same and it is unique that everyone can see the sales and claims data. The traditional insurance industry is very secretive about its data as they regard this as its competitive advantage.

iv. Decision speed

Most claims (61%) at Nexus Mutual have been resolved within the first or second day. 92% of the claims have been resolved within 3 days. The claim with the longest time has taken 7 days to be resolved. It should be noted that users are requested to submit their claims only once the experts have given their informal advice.

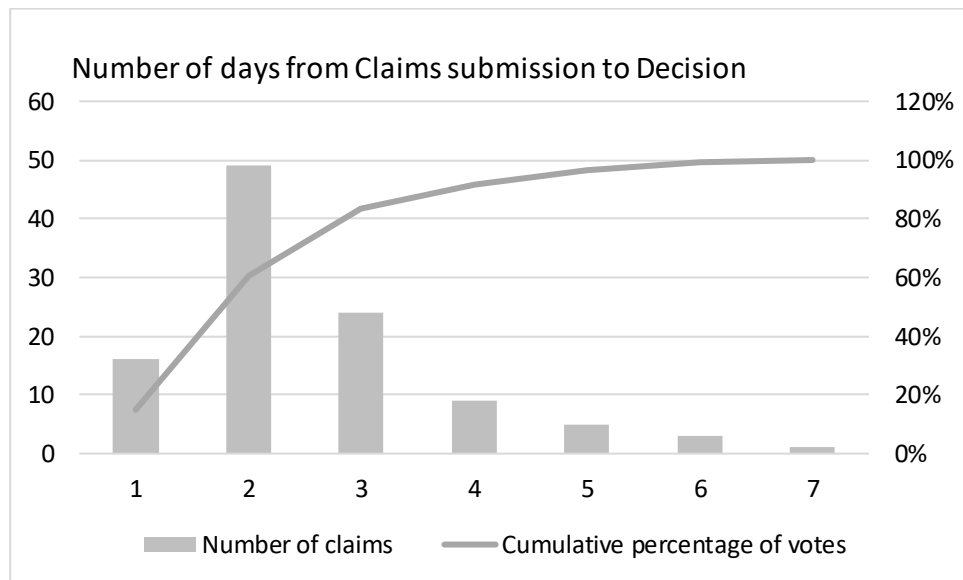


Figure 5: Graph of the number of days from claims submission to the decision on the payout

When compared to a benchmarking study done by Mahlow and Wagner amongst insurance companies in Germany and Switzerland, Nexus seems very quick (Mahlow & Wagner, 2016b). Standard claims in car, property, and liability insurance would take 60 to 70 days to be approved. More complex cases can take over 100 days to a year. Claims at Nexus Mutual can be compared to complex cases as there are millions of dollars involved and independent experts are required to assess the

damage. In the same study, only the most straightforward claims (with low amounts) which are automatically approved are paid on average within 2 days (property insurance) or 3 days (liability insurance). Hence, we can conclude that Nexus Mutual can live up to the expectation where the decision speed for claims payout is improved compared to traditional insurance companies.

UST de-peg case study

On 10th May 2022, the largest claimable event in the history of crypto happened, and decentralized insurance was put to the test. \$18B worth of UST tokens – which were supposed to cost \$1 each – lost their value due to a digital bank run. A total of 182 policies have been sold insuring 14 million dollar worth of UST through InsurAce, one of the largest decentralized insurance parties.

On the 9th of May the first worries about UST were shown in the market. 10th of May the de-peg started and by the 12th, it was clear that the token was beyond saving. On the 13th of May, the official event was triggered on InsurAce, claims submission was allowed until May 20th and the team used the weekend of 20th – 23rd to monitor all the claims.

From June 6th until June 8th final vote took place to decide on the claims payout. InsurAce paid out \$12M in insurance claims to over 150 users which were seen in the industry as proof that decentralized insurance is a mature industry. This example is the slowest a claim in web3 insurance has been paid out and is well below the 60 days (for standard claims) and 100 days (for complex claims) found in the traditional insurance industry (Mahlow & Wagner, 2016a).

One of the principles behind decentralized applications is that there is no central authority that can change the rules or stop a smart contract from running (Steis et al., 2022). The pre-defined rules in the smart contract stated that funds can be withdrawn after 7 days, but this was not possible as the time from the hack until the payout was almost a month later. The InsurAce team performed at least 3 days of hard manual labor to validate whether the insurance claims were in line with their policy (mention the reasons). The labor intensity was further increased because the blockchain on which the claimable event took place (Terra) was taken down as it had lost its value. The scenario described had so catastrophic that the InsurAce team could not anticipate what happened. There seemed to be a lot more ambiguity than expected (e.g. would people still be covered if they already sold their UST token at a loss?) and it proves Hart's theory that it is near impossible to capture all possible scenarios in a complete contract (Hart, 2017). One of the topics of dispute that were raised was, what happened to policyholders who sold their UST tokens already. In the process of UST losing, its value from \$1 to less than a cent, they might have sold for example at 90 or 80 cents.

Another topic of dispute was that one of the voters also owned a big stake in the investment pool of InsurAce once he voted against some claims made by the member. Many twitter threads were dedicated to the fact that there might be a misalignment of incentives. In the preparation for the insurance claim, there was a lot of fear that misalignment of the incentives will drive to claims process to produce false outcomes. Investors in InsurAce's risk pools where to lose \$12M, which provides them with sufficient incentive to buy their way into becoming a claims assessor and vote against the claim being paid out.

Summary of validation

Table 1 contains a summary of the validation for each of the potential benefits identified in prior work. Based on case studies and industry data analysis it is validated that existing blockchain insurance products live up to their expectation in most areas. Blockchain-based insurance comes especially close to its potential in case the entire process can be automated based on available data but does not reach its full potential in cases where ambiguity exists. The unique capability of the blockchain is regarding transparency and auditability of the data and processes. In the traditional industry, you will trust a brand, and the legal system to ensure that the service you buy from a company is the same service that is advertised. On the blockchain, it can be validated because both the code and the data are publicly available. This becomes especially powerful once multiple parties are collaborating and will not trust a single entity to manage their IT infrastructure and data.

True decentralization of technology and governance seems to be still far away. The DAOs are still able to update the code and voters in decentralized governance seem to be driven by conflicts of interest. Blockchain-based solutions seem promising for cost, efficiency, and decision speed. Teams of blockchain-based insurance providers are typically able to run their processes end-to-end with between 5-10 employees. Nevertheless, comparable results might be able to be achieved with automation using non-blockchain technology.

| Potential benefits | Validation | Summary |
|---|---------------------|---|
| Cost & efficiency | Partially Validated | Claims management in hack insurance is more cost-efficient than in traditional insurance companies. Yet, there is more potential for efficiency gains as existing solutions are not yet near the theoretical ideal situation. |
| Decentralization (of technology and governance) | Invalidated | Decentralization of technology has not been sufficiently implemented and a central entity still has sole access to the code. Decentralization of governance with a community vote is not implemented well and raises concerns about conflicts of interest and fairness. |
| Transparency & auditability | Validated | Data on sales and underwriting is readily available for blockchain insurance providers and can be regarded as one of the innovations of blockchain technology |
| Decision speed | Partially validated | The decision speed of blockchain-based insurance products is faster than that of traditional insurance companies. Yet, further steps can be made to automate more steps of the process to further increase the decision speed. |

Table 1: Validation outcome for benefits of running insurance on the blockchain

Chapter IV.

Blockchain Claims Standard Application

This chapter will define the specifications and scope of the proposed Blockchain Claims Standard Application (BCSA). The system is divided into components based on the claims management business process which is defined based on the Background chapter. The specifications contain a detailed description of the functional complexity tackled by the application: It will define how a single judgment for a claim assessor can be used by all insurance providers. The third section in this chapter will define the scope which is implemented for this thesis.

System Architecture

To implement a blockchain-based insurance system the application will need to communicate with a public blockchain. For this thesis, we selected the Ethereum network, which is used by most of the large insurance protocols (so the data of these protocols will be available on Ethereum. To write smart contracts, the Solidity language is used. For the users, an interface will be required which can be used in a web browser that has MetaMask installed. MetaMask is a software tool that allows users to send transactions on the blockchain. The web application in the browser will be implemented using JavaScript and the React framework. This is a light front-end framework that is optimal for building quick prototypes.

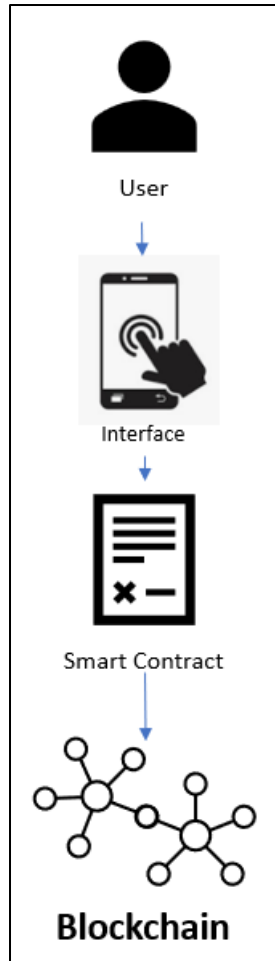


Figure 6: System Architecture

Ethereum blockchain

Ethereum is a decentralized open-source blockchain that can run decentralized applications (dApp) based on Smart Contracts (Buterin & others, 2014) A new block of data is committed to the blockchain every 11-13 seconds which is sufficiently fast for buying and selling insurance covers (Wood, 2014). Data on the Ethereum blockchain is validated by a Proof of Stake consensus mechanism by network participants which own at least 32 Ether. In 2022 there were roughly 1 to 1.5 million daily transactions and Ethereum was the most popular blockchain for financial applications.

| | |
|----------------------|-------------------------------|
| Year of introduction | 2014 |
| Native currency | Ether (ETH) |
| Consensus mechanism | Proof of Stake (since 2022), |
| Time between blocks | 11-13seconds |
| Daily transactions | 1million-1.5million (in 2022) |
| Active addresses | 329.900 (Q1 2022) |
| Main coding language | Solidity |

Table 2: Specifications of the Ethereum Blockchain

The EVM (Ethereum Virtual Machine) is the runtime environment of smart contracts introduced by the Ethereum blockchain. The EVM stores data on the blockchain and executes smart contracts. It does so by defining the rules for the nodes running the blockchain and changing the state from block to block. In the case of Ethereum the EVM runs across all nodes supporting the network but can also be run

locally on a single device. The EVM technology is also used by some of the blockchains competing with Ethereum (Binance Smart Chain, Polygon).

Solidity programming language

Solidity is a high-level programming language to write smart contracts on the Ethereum blockchain. It is statically typed and can support both object-oriented and functional programming making it a good fit for financial applications. Because Solidity supports the concept of events it is easy to track data of activity taking place on the blockchain and create transparent data structures (Modi, 2018). Solidity is released under the MIT license making it a popular and free-to-use option for development on the Ethereum blockchain.

One advantage of the Solidity language is the ease of integration with other applications through the ABI (Contract Application Binary Interface). ABIs are a standard way of interacting from contract-to-contract and to interact from outside to a blockchain. It is assumed that interface functions of a contract are strongly typed (the type is defined already) and known at compilation time. The ABI supports several standard entities like Address and Huge Integer. Figure 7 contains part of the ABI of the claims contract in JSON (Java Standard Object Notation) format. This ABI is included as a file in the React Front-end Application to allow communication between the interface and the solidity contracts.

```
{
  "_format": "hh-sol-artifact-1",
  "contractName": "Claims",
  "sourceName": "contracts/Claims.sol",
  "abi": [
    {
      "inputs": [],
      "stateMutability": "nonpayable",
      "type": "constructor"
    },
    {
      "inputs": [
        {
          "internalType": "uint256",
          "name": "insurerId",
          "type": "uint256"
        },
        {
          "internalType": "uint256",
          "name": "rootCauseId",
          "type": "uint256"
        },
        {
          "internalType": "bool",
          "name": "covered",
          "type": "bool"
        }
      ],
      "name": "AddInsurerRootCause",
      "outputs": [],
      "stateMutability": "nonpayable",
      "type": "function"
    }
  ]
}
```

Figure 7: Part of ABI of claims contract

System components

This section describes the 4 components of the application based on the insurance process. The claims process as described in Chapter II by Mahlow et al. will be used for this purpose (Mahlow & Wagner, 2016b). The implementation will mostly focus on the Audit component of the claims process as this contains the most novelty compared to existing solutions.



Figure 8: Claims management sub-process by Mahlow et al. (2015)

i. Notification & registration:

A user interface will be made available for the users to submit a claim for an insurance policy they bought earlier. In the first version of the BCSA, the user will select which insurance provider they have bought a cover for and what asset they are insuring. In later versions of the application the blockchain address of the user will be scanned to identify automatically which insurance policies they own so, the data can be read from there. This is currently not in scope as it would require integration with the insurance providers.

ii. Audit

A user interface will be made available for a claims manager to submit a claimable event (hack) that has taken place. The claims manager will not be able to interact with individual claims but the system will automatically decide which claims are to be approved based on the root cause of a hack.

iii. Settlement

The settlement phase will be fully automatic. In line with the fourth recommendation, the size of the settlement should be calculated in an automated way to capture the benefits of cost and decision speed. Hence it is recommended to always pay the full amount for which a cover is bought. This data point can be retrieved from the insurance policy inside the wallet of the user.

iv. Closing

The system should automatically pay the claims to all outstanding policies bought for the insurance (not only for the claims which are submitted). An integration interface should be made available for the insurance providers to use. The Closing process step is out of the scope of the current application as integration with the insurance providers is required.

Functional complexity: Creation of a Standard

One important requirement for the Blockchain Claims Standard Application is to create a standard for insurance claims which can be utilized across multiple competing insurance providers”. To create a standard, the way of communicating and storing claims data should be generic so that it would work for all insurance providers. This section contains an overview of the top-4 insurance providers to understand the requirements to enable them to do their claims management using a single standard.

Table 3 contains an overview of the data points used by each insurance provider to determine the payout. An assumption is made that the new standard should be usable for the current top-4 insurance providers and the remaining smaller providers should adjust to the standard.

| | Nexus Mutual | InsurAce | Unslashed | Ease |
|--------------------------------|---|--|---|---|
| Date | Should hold the cover at the moment of the exploit | Should hold the cover on the moment of the exploit and should not be within 10 days of the purchase date | Should hold the cover at the moment of the exploit | Should hold the cover at the moment of the exploit |
| Capital lost | The user should proof the actual value is lost | For hacks: the user should prove actual value is lost. Stablecoins are parametric | The user should proof the actual value is lost | Fully parametric |
| Hacked protocol | The ID of the protocol is set by the insurance protocol | The ID of the protocol is set by the insurance protocol | The ID of the protocol is set by the insurance protocol | The ID of the protocol is set by the insurance protocol |
| Root Causes included: | | | | |
| Code vulnerabilities & Hacks | Y | Y | Y | Y |
| Code used in an unintended way | Y | N | Y | Y |
| Governance attack | Y | N | N | Y |
| Front-end-hack | N | N | N | N |
| Rug Pull | N | N | N | N |

Table 3 Overview of requirements of top insurance providers

One novelty implemented in the BCSA is to ensure the Claims Manager identifies the root cause of each hack. This will allow a single assessment to be done across all insurance companies and still allow a different outcome for different insurance providers. As mentioned earlier, if the root cause of a hack is that an insider stole the money, Nexus Mutual will not pay the claim but Ease will. Table 3 provides an overview of the most important root causes mentioned in the contracts of the top-4 insurance providers. The proof-of-concept app will contain at least a few examples of events that will be assessed which will lead to payout at some insurance providers but not at others.

Another data point required is the date a hack has taken place. This is important because the insurance will only pay the claim if the hack was during the period a policy was bought for. This is to avoid the situation that clients will buy an insurance policy after it has become public knowledge that a hack has taken place. InsurAce goes even further for their de-pegging insurance to state that they only pay claims for which the claimable event takes place at least 10 days after the policy has been bought. All insurance providers will be helped if the date is stored in the contract.

The Background chapter describes how proof of loss needs to be uploaded for claims at Nexus Mutual which are confirmed manually for each claim. For the BCSA, all processes are as automated as possible. Hence any amount will be paid out as long as it is within the limit of the insurance policy bought and the total value of the hack is larger than the total value of the insurance claims.

The third data point is the protocol or crypto company that is hacked (e.g. Uniswap). Currently, there is no standard across insurance providers to identify the underlying assets. Various factors make this complex: There can be various versions of a

digital asset (e.g. Uniswap version 1 and Uniswap version 2) which are covered under a different policy. There can be different sub-products of the digital asset which are covered under a different policy (e.g. Curve finance has different vaults with their insurance policies) or multiple digital assets can be combined inside a single policy (e.g. InsurAce offered a cover for USDT, Luna, and Mirror combined). The solution to this problem is outside the scope of the thesis as the mapping of different insurance products is done by the company at which the author of this paper is advising (Bright Union).

Scope definition

Table 2 contains an overview of the scope of the BCSA.

| | Specification | Scope |
|----|--|--------------------------------------|
| 1 | As an Insured user, I would like to submit an insurance claim | In Scope |
| 2 | As an Insured user, I would like my wallet to be scanned for the insurance policies that I own to automatically supply the required data during claims submission | Out of Scope (requires integrations) |
| 3 | As a blockchain system, I will see if a newly submitted claim belongs to a “hack event” which has already happened. | In Scope |
| 4 | As a Claims manager, I can submit an event for which I know a hack has taken place | In Scope |
| 5 | As a Claims manager, I can select a root cause that I believe to have belonged to a hack, triggering claims to be accepted or denied based on the insurance policies | In Scope |
| 6 | The system checks the identity of the Claims manager using an NFT | In scope |
| 7 | As a blockchain system, I will automatically establish which claims belong to the “hack event” based on the date of the hack and parameters of the insurance contract and validate or invalidate the claim | Out of scope (already exists) |
| 8 | As a blockchain system, I will automatically pay the Insured in case the claim or disputed claim is validated and provide the Claims manager with an incentive | Out of scope (already exists) |
| 9 | As an Insured, I can see the status of my claims and whether it is validated or invalidated and what amount is paid | In Scope |
| 10 | As an Insured, I can submit a dispute for a claim which has been invalidated | Out of scope |
| 11 | As a blockchain system, I will automatically select Claims managers to validate the claim | Out of scope |

Table 2: Specifications that are in scope.

Chapter V. Implementation

This chapter describes the detailed technical design of the BCSA. The domain model will describe the important concepts in the domain of the application. The sequence diagram describes the interaction between all the different layers of the application from the user to the front-end through the various smart contracts.

Domain Model

The domain model (Figure 8) contains the most important entities and concepts of the claims management application. The domain model will be leading in identifying how to break down the application into distinct smart contracts. The main logic of the BCSA will be mapping the Events to the Claims. The scope of the BCSA is the blue box labeled “proof of concept”. The figure also contains the logic which exists inside most existing insurance products as integration will be required in the future.

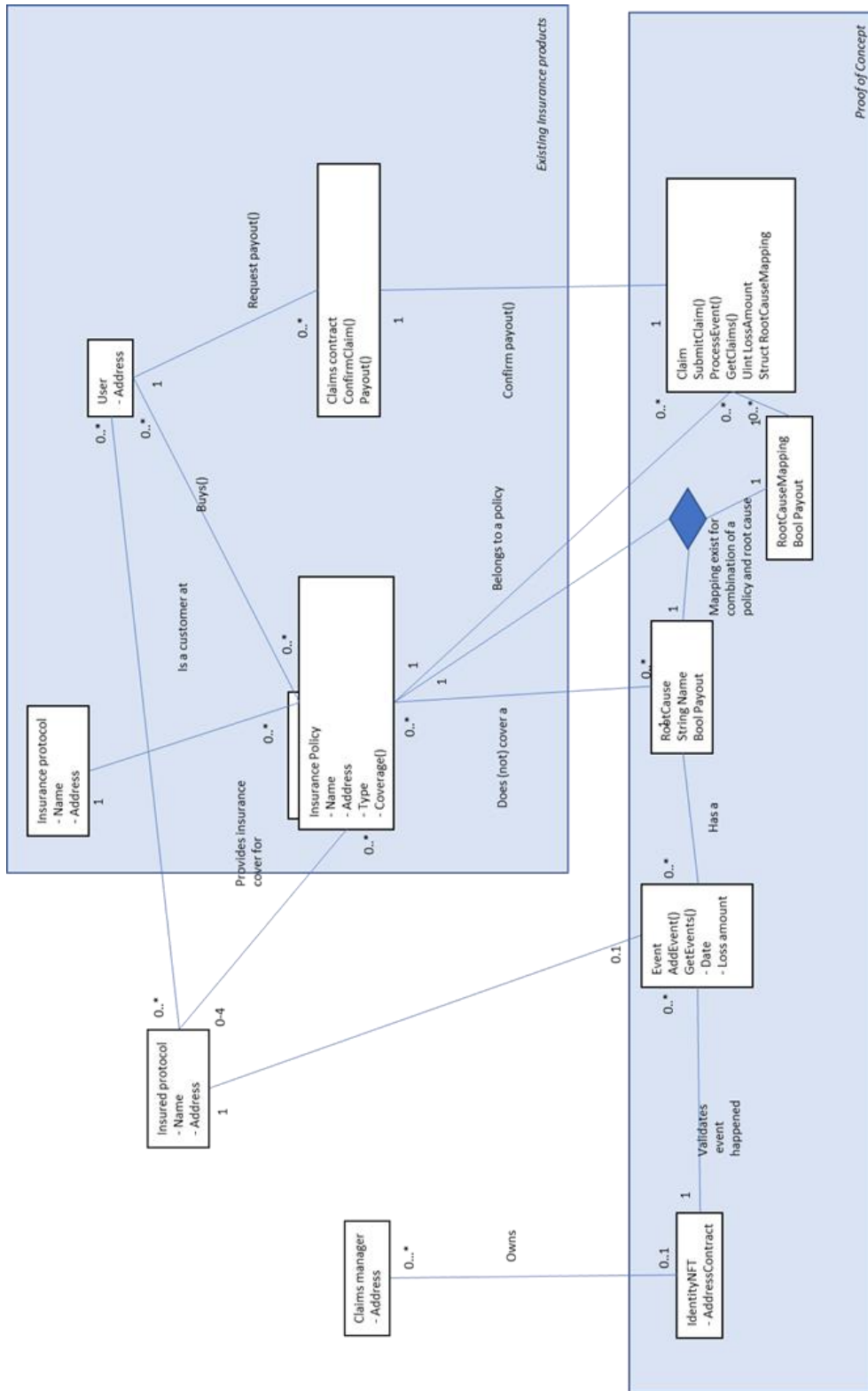


Figure 9: Domain model

i. Claim

The Claim entity (see code in Figure 9) represents an insurance claim which can be submitted by the users, or which can be used by existing insurance providers to determine whether a payout should be made. The claim contains a LossAmount and a link to an Insurance Policy entity.

Once an Event is created the processEvent() function on the claim can be used to define whether a claim should be paid out. The processEvent() uses the RootCauseMapping entity, which is a Boolean defining whether a payout is required for a combination of a root cause and an insurance policy. The RootCauseMapping exists in the claims contract.

```
struct Claim {
}
address public owner; Claim[] internal Claimslist; struct insurerRootCause {
    uint insurerId;
    uint rootCauseId;
    bool covered;
} insurerRootCause[] internal insurerRootCauseList; constructor() {
    owner = msg.sender;
    initializeCovers();
} function addClaim(uint coverId, uint insurerId, address coverBuyer) external {
    console.log('reached claims contract');
    // If a claim already exist for the same cover and has been approved
    Claimslist.push(Claim(coverId, insurerId, coverBuyer, block.timestamp, checkIfExists(coverId, insurerId)));
    console.log('pushed claims');
} function checkIfExists(uint coverId, uint insurerId) public view returns (uint) {
    for (uint i = 0; i < Claimslist.length; i++) {
        // If a claim already exist for the same cover and has been approved
        if (Claimslist[i].coverId == coverId && Claimslist[i].status == 3 && Claimslist[i].insurerId==insurerId) {
            return 3;
        }
        //Reject if another claim from the same insurance provider has been rejected
        //ToDo: Case where it's the first claim for that specific provider
        if (Claimslist[i].coverId == coverId && Claimslist[i].status == 4 && Claimslist[i].insurerId==insurerId) {
            return 4;
        }
        //If a claims for the same cover already exists make it class action
        if (Claimslist[i].coverId == coverId) {
            return 2;
        }
    }
    return 1;
} function processEvent(uint coverId, uint rootCauseId) public{
    //Check that this is called by the Event contract only
    for (uint i = 0; i < Claimslist.length; i++) {
        //Approve claim if it's for the same cover ID as the event AND checkIfcovered = true
        if (Claimslist[i].coverId == coverId && checkIfCovered(Claimslist[i].insurerId, rootCauseId))
        {
            Claimslist[i].status = 3;
        } if (Claimslist[i].coverId == coverId && !checkIfCovered(Claimslist[i].insurerId, rootCauseId))
        {
            Claimslist[i].status = 4;
        }
    }
}
```

Figure 10: Example of Solidity code for contract claims.sol

ii. Events

The Event entity represents a hack or claimable event. An event is linked to an Insured Protocol and is assigned a Root Cause and a date on which the hack took place. The event is created by a Claims Manager using the AddEvent() function. In case the caller of this function owns an IdentityNFT the event will be added. After the event is added, the claims contract will be called to process the event.

```
pragma solidity ^0.8.9;
import "./Claims.sol";
import "./IdentityNFT.sol";contract Events {
    // Claims status 0 = submitted-FirstClaim, 1 = submitted-ClassAction, 2 = Rejected, 3 = Accepted
    //Root cause 0=hack, 1=governance attack, 3=front-end
    struct Event {
        uint coverId;
        uint lossAmount;
        uint rootCauseId;
    }
    address public owner;
    Claims private claimsContract;
    IdentityNFT private identityNFTContract; Event[] internal Eventlist; constructor(address _claimsContractAddress, address
    _identityNFTContractAddress) {
        owner = msg.sender;
        claimsContract = Claims(_claimsContractAddress);
        identityNFTContract = IdentityNFT(_ide
    ntityNFTContractAddress);
    } function addEvent(uint coverId, uint LossAmount, uint rootCauseId) external {
        console.log('reached Event contract');
        require(ownsNFT(msg.sender),"Only claims managers can add an event");
        Eventlist.push(Event(coverId, LossAmount, rootCauseId));
        console.log('Updating claims contract');
        claimsContract.processEvent(coverId, rootCauseId);
    } function ownsNFT(address eventCreatorAddress) private view returns (bool) {
        if (identityNFTContract.balanceOf(eventCreatorAddress)>=1) {
            return true;
        } else {
            return false;
        }
    } function getEvents() public view returns(Event[] memory) {
        return Eventlist;
    }
}
```

Figure 11: Solidity code for contract Events.sol

iii. IdentityNFT

The Identity NFT will represent proof of identity for the claims manager. When the application is deployed for the first time the NFT will be minted(created) to the claims managers based on the addresses which are specified in the deployment script (see code in Appendix 2). Any user is only able to create an event when they own an Identity NFT.

```
pragma solidity ^0.8.9;import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
import "hardhat/console.sol";contract IdentityNFT is ERC721 {
    address public owner;
    address[] public ClaimsManagers; constructor(
        string memory tokenName,
        string memory tokenSymbol
    )
    ERC721(tokenName, tokenSymbol){
        owner = msg.sender;
        console.log("created NFT");
        //Address of creator on hardhat network
        mint(0xf39Fd6e51aad88F6F4ce6aB8827279cFfFb92266);
    } function mint(address _to) public {
        _mint(_to, 1);
        console.log("minded NFT to");
        ClaimsManagers.push(_to);
    } function getClaimsManagers() external view returns (address[] memory) {
        return ClaimsManagers;
    }
}
```

Figure 12: Solidity code for NFT identifyNFT.sol

Sequence diagram

The sequence diagram in Figure 13 describes the sequence of the most important interactions between the users and the smart contracts: adding an event by the claims manager and adding a claim by a user. The functions in the sequence diagram align with those implemented in the code.

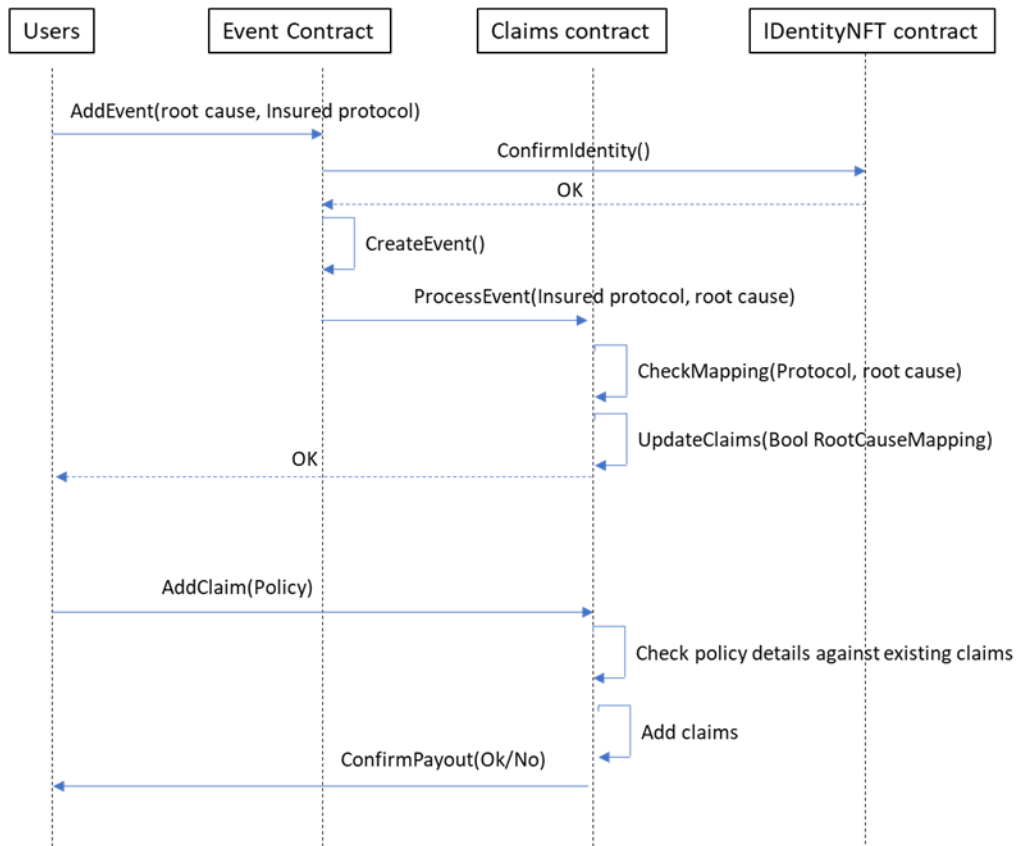


Figure 13: Sequence diagram

When the claims manager creates an event. Their identity is confirmed with the Identity NFT contract. In case it is valid, an event is created in the Event Contract after which the claims contract is called to process the event. Then all outstanding claims for

the insured protocol will be updated using the RootCauseMapping. In case the combination of root cause and policy returns true, a payout should be made. In the first version of the application, only the status of the claim will be updated.

Any user can create a claim based on their insurance policy. The Claims contract will check the data on the policy and check against existing claims. If existing claims - with the same policy & root cause - have already been accepted or canceled, the new claim will get the same status. In future versions of the app, this functionality can be used by any blockchain insurance provider to poll whether a claim should be paid out so they can automate this process.

State Diagram

The claim is the central entity in the domain diagram. In the system, the claim can be in 4 different “states” based on the choices the user makes. State transitions 1.1 to 1.4 are for claims which have just been created. If the claim is the first for an asset, it will be submitted with the status “submitted – first claim” (1.1). If there is already at least one claim submitted for the asset the claim will go in status “submitted – class action” (1.4). In case a claims manager already accepted or rejected the claim with a root cause, any new claim will be automatically Accepted (1.2) or Rejected (1.3).

Steps 2.1 to 2.4 are for claims which are already submitted at the time when the claims manager submits an event. At that moment all submitted claims will be automatically accepted (2.1 and 2.3) or rejected (2.2 or 2.4).

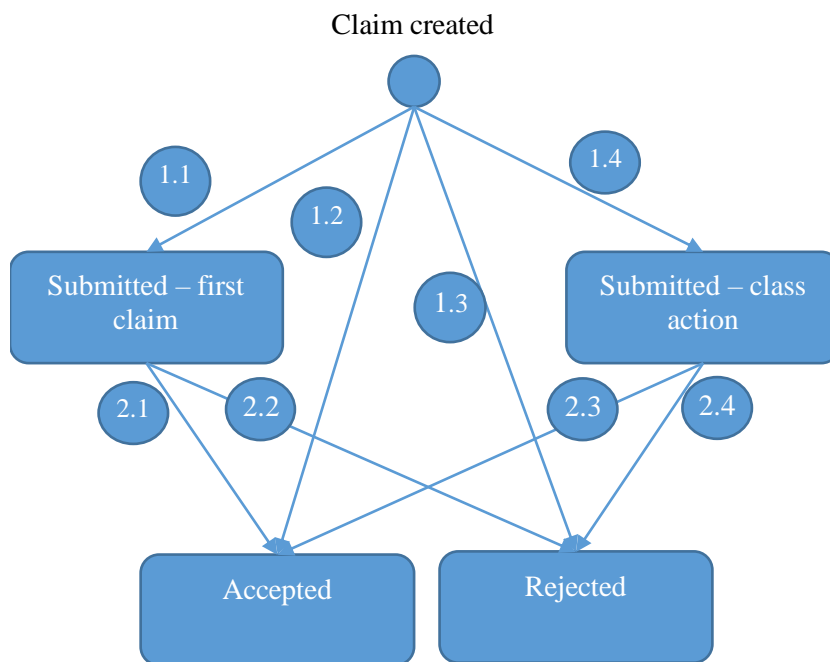


Figure 14: State diagram of claims object

Deployment

Hardhat will be used to run a replica of the Ethereum blockchain on a laptop because it is cheap and easy to use. For this thesis, the application is run on a Lenovo Thinkpad laptop (Intel i7-8550, 180Ghz processor, 16GB RAM) and Macbook pro (2021, M1 Pro 8-core processor, 16GB RAM). Hardhat is available for Windows, Linux, and IOS and should work with any laptop with over 8GB of RAM memory.

For the application, 3 smart contracts will need to be deployed to the blockchain: Claims contract, Event contract, and IdentityNFT. Figure 14 on the next page contains the code of the deployment script. Most notable is the order in which these contracts are deployed: The Event contract is deployed last because it needs to be aware of the address of the IdentityNFT contract so it can check the identity of the user who is submitted an event. Also, when an event has been created the address of the claims contract needs to be known so it can communicate to the claims contract that the status of several claims needs to be updated.

After deploying the 3 smart contracts the user interface can be deployed. A separate file is automatically created which contains the addresses of the relevant smart contracts. The Ethers library is used to enable communication with smart contracts.

```

const path = require("path"); async function main() {
  if (network.name === "hardhat") {
    console.warn(
      "You are trying to deploy a contract to the Hardhat Network, which" +
      "gets automatically created and destroyed every time. Use the Hardhat" +
      " option '--network localhost'"
    );
  } // ethers is available in the global scope
  const [deployer] = await ethers.getSigners();
  console.log(
    "Deploying the contracts with the account:",
    await deployer.getAddress()
  ); console.log("Account balance:", (await deployer.getBalance()).toString()); const Token = await ethers.getContractFactory("Token");
  const token = await Token.deploy();
  await token.deployed(); console.log("Token contract address:", token.address); const Claims = await ethers.getContractFactory("Claims");
  const claims = await Claims.deploy();
  await claims.deployed(); console.log("Claims contract address:", claims.address); const IdentityNFT = await
  ethers.getContractFactory("IdentityNFT");
  const identityNFT = await IdentityNFT.deploy("ClaimsmanagerIdentity", "CMI");
  await identityNFT.deployed(); console.log("Identity NFT:", identityNFT.address); const Events = await ethers.getContractFactory("Events");
  const events = await Events.deploy(claims.address, identityNFT.address);
  await events.deployed(); console.log("Events contract address:", events.address); // We also save the contract's artifacts and address in the frontend
  directory
  saveFrontendFiles(token, claims, events, identityNFT);
}function saveFrontendFiles(token, claims, events, identityNFT) {
  const fs = require("fs");
  const contractsDir = path.join(__dirname, "..", "frontend", "src", "contracts"); if (!fs.existsSync(contractsDir)) {
    fs.mkdirSync(contractsDir);
  } fs.writeFileSync(
    path.join(contractsDir, "contract-address.json"),
    JSON.stringify({
      Token: token.address,
      Claims: claims.address,
      IdentityNFT: identityNFT.address,
      Events: events.address
    }, undefined, 2)
  ); const TokenArtifact = artifacts.readArtifactSync("Token");
  const ClaimsArtifact = artifacts.readArtifactSync("Claims");
  const EventsArtifact = artifacts.readArtifactSync("Events");
  const IdentityNFTArtifact = artifacts.readArtifactSync("IdentityNFT"); fs.writeFileSync(
    path.join(contractsDir, "Token.json"),
    JSON.stringify(TokenArtifact, null, 2)
  ); fs.writeFileSync(
    path.join(contractsDir, "Claims.json"),
    JSON.stringify(ClaimsArtifact, null, 2)
  ); fs.writeFileSync(
    path.join(contractsDir, "IdentityNFT.json"),
    JSON.stringify(IdentityNFTArtifact, null, 2)
  ); fs.writeFileSync(
    path.join(contractsDir, "Events.json"),
    JSON.stringify(EventsArtifact, null, 2)
  );
}
}main()
.then(() => process.exit(0))
.catch((error) => {
  console.error(error);
  process.exit(1);
});

```

Figure 15: Script to deploy smart contract code to the blockchain

Application interface

This section contains a description of the implemented interfaces.

i. Overview page (Registration phase)

The overview page contains a table with the claims submitted by users(top) and confirmed events submitted by Claims Managers(bottom). The table with the claims data contains all the claims registered and shows the data points presented in the functional requirements: The insured asset (e.g. the user buys insurance against Coinbase being hacked), the insurance provider (e.g. Nexus Mutual), the blockchain address to which the insurance policy belongs, the time at which the claim was submitted and the status of the claim.

The screenshot displays the 'Thesis Demo' application interface for 'Insurance claims management on the blockchains'. The user is logged in as a 'Claims manager'. The interface features two main tables and two action buttons.

Claims Table:

| Insured asset | Insurer | Address client | Submission time | Claim status |
|---------------|--------------|--|--|--------------|
| Coinbase | Nexus Mutual | 0x95E7E68a0275c691fDB3beAEbB5f94265Aec3Fdf | Sun Jan 29 2023 03:39:59 GMT+0100 (Central European Standard Time) | First Claim |
| Coinbase | Ease | 0x5FbDB2315678afecb367f032d93f642f64180aa3 | Sun Jan 29 2023 03:39:59 GMT+0100 (Central European Standard Time) | ClassAction |
| FTX | Nexus Mutual | 0xe711725E7734CE288F8367e1Bb143E90bb3F0512 | Sun Jan 29 2023 03:39:59 GMT+0100 (Central European Standard Time) | First Claim |

Events Table:

| Hacked asset | Lost assets amount USD | Root cause |
|--------------|------------------------|------------|
| USDT | 100000 | Governance |

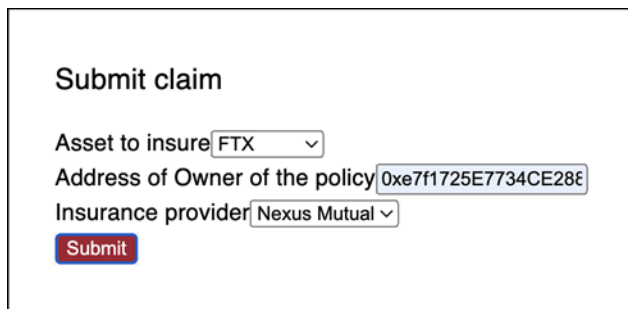
The interface also includes a 'SUBMIT CLAIM' button and a 'SUBMIT EVENT' button.

Figure 16: Interface of overview page

The second table on the overview page contains the events submitted by the claims manager. Each hacked asset contains the total amount lost during the hack (in USD dollars) and the root cause of the hack. In Figure 8 an example is given where the root cause of the hack is an attack on the “governance”. An example of this would be that a bad actor took a very big loan to do a 51% attack (See Limitations section in the background for further explanation) to take a governance decision in their favor and sell the tokens straight after. So, in this case, there has not been a hack of the code which led to the loss for users of the blockchain product.

ii. Submit a claim (Notification phase)

Any user which bought an insurance policy can submit a claim. In the first version of the application, the user has to identify themselves which asset they want to insure with which insurance provider. In the future version of the app, it will not be required to provide any information as this can be read automatically from the insurance policy your address on the blockchain owns.



The screenshot shows a form titled "Submit claim". It contains three input fields: "Asset to insure" with a dropdown menu showing "FTX", "Address of Owner of the policy" with a text input field containing the hexadecimal address "0xe7f1725E7734CE28E", and "Insurance provider" with a dropdown menu showing "Nexus Mutual". Below these fields is a red "Submit" button.

Figure 17: Interface of submitting a claim

To support the standardization across insurance providers a few checks are being done by the claims contract in the back end: If an event has already been created for the same asset and insurance provider, the claim will automatically be approved (and automatically rejected in the opposite situation). If it is the first claim for a specific asset, the claims managers will be notified so they can add an event in case a hack has taken place. In case another claim already exists but no decision has been made by the claims manager, the status is changed to “class action” to identify that multiple users are awaiting the verdict of a claims manager.

In future versions of the app, it will not be required for the user to submit a claim. The insurance providers will be possible to “ask” the claims contract if a hack has taken place and will be able to automatically pay the claim. This is not in the scope of the thesis as it requires collaboration with the insurance providers.

iii. Submit event (Audit phase)

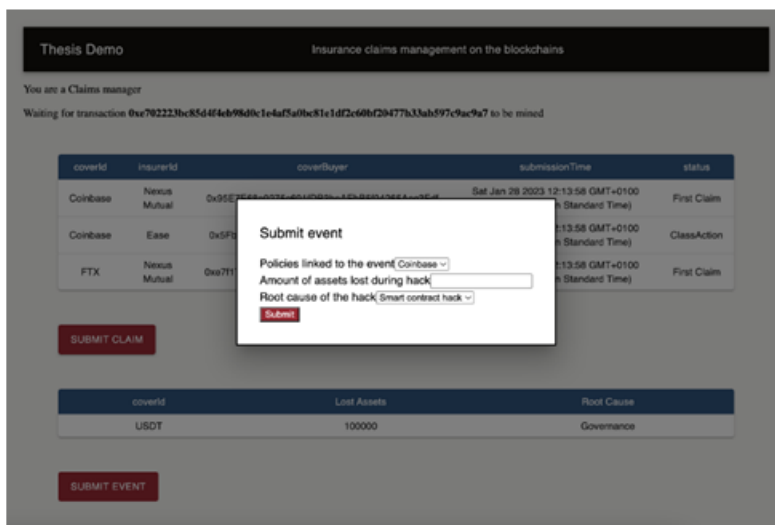


Figure 18: Interface of claims manager submitting an event

Once a first claim I created for an asset, the claims manager is notified to validate whether an event happened which is covered under the insurance policies. In case the claims manager validates that the claims happened, it is submitted together with the number of assets lost and the root cause.

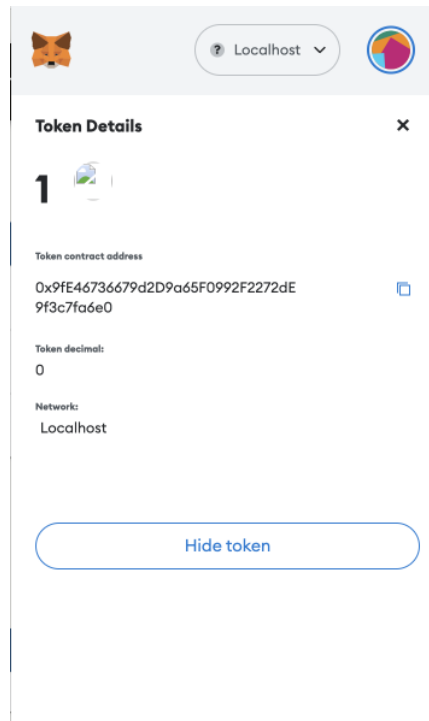


Figure 19: Interface of Metamask wallet with Identity NFT

A user is only able to submit an event in case their blockchain address owns an NFT (Non-Fungible Token) to identify their identity. This is checked by the Events contract (see code in Appendix 2). In case the identity of the claims manager is validated, the event contract will process the event by communicating to the claims contract the coverId and the root cause. The claims contract contains the logic of which insurance

providers cover which root causes. Figure 12 contains the interface of a blockchain wallet which contains an identity NFT. The “token contract address” is the same address on the blockchain to which the IdentityNFT contract (see code in Appendix 2) is deployed. Awarding the NFTs to the claims managers is out of the scope of this thesis.

iv. Settlement & closing

For the BCSA, settlement of the claim will be fully automatic. Based on the amount of insurance coverage, the claims should automatically be paid out. This is currently out of scope because integration with the insurance providers is required.

Chapter VI.

Results

This chapter compares the results of the BCSA with existing blockchain applications and the traditional insurance industry. This is followed by a discussion section that highlights the limitations of the research approach used in this thesis. Five recommendations are made for the blockchain insurance industry to reach the next level of their maturity.

Results overview

The BCSA further improves upon the concept of blockchain-based insurance, especially in the area of cost, efficiency, and decision speed. This is achieved by centralizing the audit process so not all insurance companies have to do an audit of the claim (saving roughly \$2000 per insurance company) and because claims will be paid out automatically (Saving roughly \$10K in gas costs per insurance provider if votes would be required for all 100 claims). The fully automated settlement in the proof of concept further reduces the decision speed by almost a month as the de-pegging use-case showed the settlement process can take almost 30 days for 100 claims.

No further improvements have been done to transparency and auditability as these benefits were already validated. Future research is required to establish if users prefer the centralized expert-based event approval which is implemented in the proof of concept application.

| | | | |
|---|---|--|--------------------------------|
| Performance comparison | BCSA | Existing blockchain insurance | Traditional insurance |
| Cost & efficiency | ~\$2.5K-\$5K for 100 claims (0.1%-0.25% of premiums) | ~\$10K-\$50K for 100 claims (~0.5%-1% of premiums) | Five percent-15% of premium |
| Decentralization (of technology and governance) | Decision on claims is performed in a centralized expert governance to increase fairness | Not able to capture the benefits of decentralization | Centralized |
| Transparency & auditability | Data is publicly available | Data is publicly available | Data is not publicly available |
| Decision speed | 2-3 days Only the Audit of events is not automated | 2 days - 1 month Audit and settlement steps not automated | 60days-100days |

Table 4: Performance comparison

Discussion

The results in the previous section use a benchmark in which blockchain insurance projects are compared against traditional insurance companies. This might not be a fair comparison. Often, traditional insurance companies have legacy IT systems, and their staff is performing manual tasks. To validate the benefit of blockchain as a technology, the benchmark should be against insurance companies that have fully automated and digitized their process with other technology than blockchain. These companies would have fully automated their process using software hosted in (a centralized) cloud instead of on the blockchain. In many cases, this could be as efficient or scalable as running the processes on the blockchain. The Ethereum blockchain can even be seen as less scalable and efficient than cloud-hosted software because the calculations and storage are duplicated across many of the nodes.

To validate the benefits of the BCSA more accurately, the application should be released to the wider public and evaluated at scale for many insurance claims. Doing a more accurate validation might take multiple years as sufficient data needs to be collected. Currently, the industry must pay out only 1 or 2 claims per. In addition, the largest barrier to industry-wide adoption is the collaboration of the existing blockchain insurance providers. Although there are significant benefits in terms of cost, efficiency, and fairness, the existing providers would cede some of the decision power they now hold themselves.

Recommendations

To improve blockchain-based insurance, consider the following five recommendations. The first 3 recommendations have been implemented as part of the BCSA.

- i. Class action

Combine the claims of users into a single “class action” claim. This concept from legal practice entails that “one class action that resolves some or all issues for a group of persons is more efficient than numerous individual suits that all raise the same or similar issues. Efficiency arises through the adjudication of numerous claims via a single proceeding” (Legg, 2015). In the event described in the case study, InsurAce received 107 individual claims which according to the existing process would require 107 votes. The InsurAce team decided to combine most claims into a single “group claim” vote. This way of working is comparable to the legal concept of a “class action” lawsuit. During this type of lawsuit, a group of people is represented collectively in a single proceeding. This greatly increases efficiency, especially in a scenario where each vote would cost ~\$100 of gas to be stored on the blockchain. Going forward it should be established once whether an event happened on a certain date, and from there all insurance policies related to that event should be automatically adjudicated.

ii. Industry-standard

Create an industry standard that allows a single judgment on an event to apply to all insurance parties. At the time of writing, there are 5-10 insurance providers which each need to establish whether an insurable event has taken place. This is inefficient but also can lead to unfair situations in which insurance providers come to a different conclusion which can happen especially in ambiguous situations.

The challenge in defining an industry standard comes from the fact that each insurance provider excludes certain conditions from the payout. For example, Nexus Mutual does insure a situation where only a frontend interface is hacked but not the smart contract on the blockchain. Ease – a competing insurance provider – insures the loss of value regardless of how the hack has taken place. Hence the industry standard should define an event in such a way that a root cause is identified on which it can automatically be established which insurance providers should continue to payout and which do not.

iii. Expert based decision making

An independent expert should establish if an event has taken place instead of a community vote. As described in the case study, there often is a (perception) of conflict of interest when using voting as a governance tool. Because the person with the most ownership has the most votes, but also a high stake to lose when a claim is paid out. In line with the recommendations of the OECD for traditional claims management, an independent party should be established to determine if a claimable event has taken place (*OECD Guidelines for Good Practice for Insurance Claim Management*, 2004). Because

recommendation 2 established the root cause of a hack needs to be established, this independent party should also be an expert. For this app, it should be a cyber security expert which can establish the root cause of a hack taking place.

For this thesis, the expert needs to be reliably verified during the approval of the claimable event. How the expert is selected will be out of the scope and part of future work.

iv. Use parametrization wherever possible

Settlement and Closing process steps should be fully automated, even at the expense of paying larger claims. The case study example shows a lot of manual research is required to establish the amount of the damages to be refunded. Mostly this was based on the burden of proof that the user had to submit in a pdf document on whether they owned specific tokens at a specific point in time. Going forward, establishing the amount to be settled should be automated. This is most easily done by establishing the value of the assets at the time the insurance premium is paid. This might mean that higher claims need to be paid in scenarios where the user bought more insurance than the assets they owned. This can be calculated into the price during the underwriting process.

v. Dispute process

Currently, users feel reasons not to be able to trust the governance voting, not only because of the inherent conflict of interest but also due to mistakes that can be made in the process or new information which can surface (more research could identify another root cause for a hack). Hence, a dispute process should be made available to the users in line with the recommendations of the OECD for traditional insurance claims management (*OECD Guidelines for Good Practice for Insurance Claim Management*, 2004)

Existing dispute processes have the owners of the insurance providers as final decision-makers. A well-functioning dispute process should at least escalate to an independent party. A potential solution could be to escalate decision-making to traditional legal authorities and courthouses. This might give the blockchain community additional trust that the owners of the insurance platforms are to be held accountable. The dispute process is not part of the scope of the application as developed as part of this thesis and will be part of future work.

Chapter VII.

Conclusion

The Blockchain Claims Standard Application (BCSA) is a novel solution that enables a standardized approach to claims management for blockchain-based insurance products. Integration with the largest existing blockchain insurance products is enabled using the Solidity programming language on the Ethereum blockchain. The application can be made available to the public through a React-based front-end interface.

The BCSA implements improvements over existing solutions in terms of standardization, cost, efficiency, and decision speed and suggests a fairer but less decentralized claims process as compared to existing blockchain insurance products. This will help the industry to narrow the gap between implemented solutions and the theoretical potential of blockchain-based insurance as in scientific literature. Certain types of insurance are already proven to be more efficient when run on the blockchain as compared to traditional insurance companies. Wide adoption of the BCSA across the blockchain insurance industry would create even more value by automating the entire process for all insurance providers in a single application.

Glossary

| | |
|----------------------|--|
| Actuarial data | Historic data with details on the claims paid for an insured risk. This data helps insurance providers to set the price of their products accurately |
| Complete contract | A legal or smart contract in which it is possible to define all potential scenarios. For an incomplete contract, there are ambiguous scenarios for which the outcome is not defined in the contract |
| dApp | A “Decentralized Application” which is automatically executed on a blockchain |
| DAO | A Decentralized Autonomous Organization is a way of collaborating without a centralized hierarchy relying on the fact that code is law |
| De-pegging | The event in which a stablecoin loses value against the dollar |
| Immutable | Something which cannot be changed (data or code) |
| Insurance provider | In the context of this paper, an insurance provider refers to a party that provides blockchain-based insurance products. Most insurance providers are not listed as companies and are organized as a DAO |
| Parametric insurance | A type of insurance where the payout is automatically defined based on parameters on available data |

| | |
|------------------------|---|
| Peer-to-Peer Insurance | An insurance model where peers come together to share risk and the cost in case of catastrophic events |
| Premium | The (monthly) cost of buying an insurance product |
| Smart Contract | A machine-readable and executable program that can automate specified procedures. Often used executed on a blockchain |
| Stablecoin | A token that is one-to-one linked to fiat currency like the US dollar or Euro |
| Staking | Investing crypto assets by locking them in a smart contract |

Appendix

Code repository and Demo

The code repository is found [here](#)

Demo:

Introduction: <https://www.loom.com/share/b6ac45804ba044ed9453bfdcac638720>

Deployment: <https://www.loom.com/share/4b031c7a0e614c93bea06ae1d87f47e1>

Application: <https://www.loom.com/share/37f3369252e84279b93937d31f14e333>

References

- Berryhill, J., Bourgery, T., & Hanson, A. (2018). *Blockchains Unchained*. 28.
- Bourgeon, J. M., & Picard, P. (2020). Insurance law and incomplete contracts. *The RAND Journal of Economics*, 51(4), 1253–1286.
- Boyle, E., Pesic, S., Jevtic, P., & Boscovic, D. (2021). Peer-to-Peer Insurance: Blockchain Implications. *Society of Actuaries*.
- Buterin, V., & others. (2014). A next-generation smart contract and decentralized application platform. *White Paper*, 3(37).
- Eling, M., & Lehmann, M. (2017). The Impact of Digitalization on the Insurance Value Chain and the Insurability of Risks. *Geneva Papers on Risk and Insurance. Issues and Practice*, 43(3), 359–396.
- Gatteschi, V., Lamberti, F., Demartini, C., Pranteda, C., & Santamaría, V. (2018). Blockchain and smart contracts for insurance: Is the technology mature enough? *Future Internet*, 10(2), 20.
- Gencer, A. E., Basu, S., Eyal, I., van Renesse, R., & Sirer, E. G. (2018). Decentralization in Bitcoin and Ethereum Networks.
- Hart, O. (2017). Incomplete contracts and control. *American Economic Review*, 107(7), 1731–1752.
- Hsieh, Y.-Y., & Vergne, J.-P. (2018). *Bitcoin and the Rise of Decentralized Autonomous Organizations*.

InsurAce. (n.d.). Retrieved January 22, 2023, from <https://app.insurace.io/claims>

Jesse Walden. (n.d.). *Incomplete Contracts (and Scaling Crypto) | Andreessen Horowitz*.

Retrieved January 22, 2023.

Karp, H., & Melbardis, R. (2017). “*Nexus Mutual Whitepaper: A peer-to-peer discretionary mutual on the Ethereum blockchain.*”

Katsh, M. E., & Rabinovich-Einy, O. (2017). *Digital justice: technology and the internet of disputes*. Oxford University Press.

Law Aaron Wright, C. P. of L. at B. N. C. S. of. (2021). The Rise of Decentralized Autonomous Organizations: Opportunities and Challenges. *Stanford Journal of Blockchain Law & Policy*.

Legg, M. (2015). *Efficiency arises through the adjudication of numerous claims via a single proceeding*. Australian Lawyers Alliance.

Lorenz, J.-T., & Münstermann, B. (2016). *Blockchain in insurance-opportunity or threat?*

Mahlow, N., & Wagner, J. (2016a). Evolution of Strategic Levers in Insurance Claims Management: An Industry Survey. *Risk Management and Insurance Review*, 19(2), 197–223.

Mahlow, N., & Wagner, J. (2016b). Process landscape and efficiency in non-life insurance claims management: An industry benchmark. *The Journal of Risk Finance*, 17(2), 218–244.

- Modi, R. (2018). Solidity programming essentials : a beginner’s guide to build smart contracts for Ethereum and blockchain. In 2018.
- Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 21260.
- Nexus Mutual Tracker. (n.d.). Retrieved January 22, 2023, from <https://nexustracker.io/claims>
- OECD data. (2022). Organisation of Economic Co-Operation and Development. <https://data.oecd.org/insurance/insurance-spending.htm>
- OECD guidelines for good practice for insurance claim management. (2004). “<https://www.oecd.org/finance/insurance/33964905.pdf>”
- Oxford University Press. (2022). *Oxford English Dictionary*. <http://www.oed.com/viewdictionaryentry/Entry/11125>
- Papacharissiou, H. (2020). *How to Build a Parametric Insurance Smart Contract*. Chainlink
- Popovic, D., Avis, C., Byrne, M., Cheung, C., Donovan, M., Flynn, Y., Fothergill, C., Hososeinzadeh, Z., Lim, Z., Shah, J., & et al. (2020). Understanding blockchain for insurance use cases. *British Actuarial Journal*, 25, e13.
- Schneider, J., Blostein, A., Brian Lee, C., Steven Kent, C., Robert Boroujerdi, C., Fox Ingrid Groer, J., ichael Lapidés Lara Fourman, C. M., Safa Conor Fitzgerald Hank Elder, P., Beardsley, E., & Barnard, G. (2016). *Blockchain Profiles in Innovation*.

Sheth, A., & Subramanian, H. (2020). Blockchain and contract theory: modeling smart contracts using insurance markets. *Managerial Finance*, 46(6), 803–814.

Steis, M., Morales, R., Baeriswyl, J., Burke, J., & Nordnes, R. A. (2022). *MetaFi: DeFi for the Metaverse*.

Szabo, N. (1997) Formalizing and Securing Relationships on Public Networks. First Monday, 2, No. 9

Wang, S., Ding, W., Li, J., Yuan, Y., Ouyang, L., & Wang, F.-Y. (2019). Decentralized autonomous organizations: concept, model, and applications. *IEEE Transactions on Computational Social Systems*, 6(5), 870–878.

Wood, G. (2014). Ethereum: A secure decentralized generalised transaction ledger. *Ethereum Yellow Paper*.