FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Bakalářská práce

Elektronická mýtná brána pod kapotou Blockchain

Studijní program: B2646 Informační technologie
Studijní obor: Informační technologie

Autor práce: **Michal Kukla**
Vedoucí práce: Ing. Jan Hybš
Ústav nových technologií a aplikované informatiky

Liberec 2023

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání bakalářské práce

Elektronická mýtná brána pod kapotou Blockchain

Jméno a příjmení:	**Michal Kukla**
Osobní číslo:	M18000086
Studijní program:	B2646 Informační technologie
Studijní obor:	Informační technologie
Zadávající katedra:	Ústav nových technologií a aplikované informatiky
Akademický rok:	2021/2022

Zásady pro vypracování:

1. Seznamte se s transakčním protokolem chytrých kontraktů a technologií decentralizovaných blockchain databází.
2. Navrhněte a realizujte aplikaci demonstrující chytré kontrakty v oblasti elektronických mýtných brán.
3. Demonstrujte výslednou aplikaci na ukázkové trase, definované protokolem GPX.
4. Proveďte bezpečnostní analýzu výsledné aplikace.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30-40 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	čeština

Seznam odborné literatury:

[1] Bhabendu Kumar Mohanta, Panda, S.S. and Jena, D. (2018). *An Overview of Smart Contract and Use Cases in Blockchain Technology*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/328581609_An_Overview_of_Smart_Contract_and_Use_Cases_in_Blockchain_Technology

[2] Maher Alharby and Aad van Moorsel (2017). *Blockchain Based Smart Contracts : A Systematic Mapping Study*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/319603816_Blockchain_Based_Smart_Contracts_A_Systematic_Map

[3] Smart Contracts: 12 Use Cases for Business & Beyond Prepared by: Smart Contracts Alliance -In collaboration with Deloitte An industry initiative of the Chamber of Digital Commerce. (2016). [online] Available at: http://digitalchamber.org/assets/smart-contracts--12-use-cases-for-business-and-beyond.pdf

[4] Bartoletti, M. (2020). Smart Contracts Contracts. *Frontiers in Blockchain*, [online] 3. Available at: https://www.frontiersin.org/articles/10.3389/fbloc.2020.00027/full

Vedoucí práce:	Ing. Jan Hybš Ústav nových technologií a aplikované informatiky
Datum zadání práce:	19. října 2021
Předpokládaný termín odevzdání:	16. května 2022

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

Ing. Josef Novák, Ph.D.
vedoucí ústavu

V Liberci dne 19. října 2021

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

27. prosince 2023 Michal Kukla

Poděkování

Poděkování směřuje vedoucímu práce Ing. Janu Hybšovi za vedené konzultace během závěrečné práce.

Elektronická mýtná brána pod kapotou Blockchain

Abstrakt

Tato bakalářská práce se zabývá vytvořením aplikace pro výběr mýta s využitím blockchain technologie a chytrého kontraktu. Výběr mýta se v Česku týká vozidel nad 3,5 tuny na silnicích I. třídy a dálnicích. Technologie blockchain nabízí možnost uchování záznamů bez vkládání důvěry v třetí stranu. Chytrý kontrakt je spustitelný kód uložený v blockchain. Tyto technologie můžou zásadně ovlivnit trh z důvodu absence důvěry v třetí stranu. V úvodu práce je vysvětlena technologie blockchain a chytrý kontrakt. Dále jsou popsány aktuální metody výběru mýta. Relevantním systémem ve výběru mýta pro tuto práci je systém elektronického mýta používaný v Česku. Podstatná část návrhu se zabývá výběrem typu blockchain databáze. Výsledkem výběru se stal typ private permissioned blockchain. Výběr naopak označil typ permissionless blockchain jako za nevhodný. Výsledkem práce je aplikace, která zpoplatňuje vozidla nad 3,5 tuny za pomoci databáze Hyperledger Fabric. Aplikace a chytré kontrakty jsou napsány v programovacím jazyce Go. Aplikace zpoplatňuje vozidla podle aktuálního sazebníku. Databáze Hyperledger Fabric vede záznamy o palubních jednotkách a jejich zůstatku. Algoritmus pro nalezení mýtného úseku funguje na principu vzdálenosti mezi aktuální polohou vozidla a mýtného úseku. Práce čerpá z konzultací s dodavatelem mýtného systému v Česku a na Slovensku. V závěru je aplikace porovnána s mýtným systémem v Česku.

Klíčová slova: blockchain, chytrý kontrakt, permissioned, hyperledger fabric, gpx, databáze, mýto

Electronic toll gate under the hood Blockchain

Abstract

This bachelor thesis deals with the creation of a toll collection application using blockchain technology and smart contract. Toll collection in the Czechia concerns vehicles over 3.5 tonnes on Class I roads and motorways. Blockchain technology offers the possibility of storing records without placing trust in a third party. A smart contract is an executable code stored in the blockchain. These technologies can fundamentally impact the market because of the lack of trust in a third party. In the introduction of the paper, blockchain technology and smart contract are explained. Next, the current toll collection methods are described. The relevant system in toll collection for this thesis is the electronic toll collection system used in the Czechia. A substantial part of the proposal deals with the selection of the type of blockchain database. The selection resulted in a private permissioned blockchain type. In contrast, the selection identified the permissionless blockchain type as unsuitable. The result of the work is an application that charges vehicles over 3.5 tonnes using the Hyperledger Fabric database. The application and smart contracts are written in the Go programming language. The application charges vehicles according to the current tariff. The Hyperledger Fabric database keeps track of the on-board units and their balance. The algorithm for finding the toll section works on the principle of the distance between the current location of the vehicle and the toll section. The work draws on consultations with toll system supplier in the Czechia and Slovakia. Finally, the application is compared with the toll system in the Czechia.

Keywords: blockchain, smart contract, permissioned, hyperledger fabric, gpx, database, e-toll

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

Seznam zkratek

AES	Advanced Encryption Standard
BFT	Byzantine Fault Tolerance
DLT	Distributed Ledger Technology
ETC	Electronic Toll Collection
GPX	GPS Exchange Format
GSM	Global System for Mobile Communications
Hash	Hash je hexadecimální řetězec s pevnou délkou vytvořený pomocí libovolného řetězce s variabilní délkou. K vytvoření hash se používá funkce např. sha256.
HTTP(S)	Hypertext Transfer Protocol (Secure)
JSON	JavaScript Object Notation
Lat	Latitude zeměpisná šířka, zeměpisné souřadnice, N
Lon	Longitude zeměpisná délka, zeměpisné souřadnice, E
M2	Motorová vozidla, která mají více než osm míst k přepravě osob (nepočítaje místo řidiče) a jejichž nejvyšší přípustná hmotnost nepřevyšuje 5000 kg.
M3	Motorová vozidla, která mají více než osm míst k přepravě osob (nepočítaje místo řidiče) a jejichž nejvyšší přípustná hmotnost převyšuje 5000 kg.
N	Motorová vozidla, která mají nejméně čtyři kola a používají se pro dopravu nákladů.
OBU	On Board Unit
PBFT	Practical Byzantine Fault Tolerance
PoW	Proof of Work
PoS	Proof of Stake
PoA	Proof of Authority
Post-paid	Platba na fakturu (bankovní záruka).
Pre-paid	Platba předem jako „dobíjení“ kreditu u mobilního operátora.
SEM	Systém elektronického mýta
ZKP	Zero-knowledge Proof
UUID	Universally unique identifier
ŘSD	Ředitelství silnic a dálnic (Stát)
3DES	Triple Data Encryption Algorithm - (TDEA, Triple DEA)

1 Úvod

Výběr mýta má za úkol vybrat poplatek za využití komunikace. Poplatek se využívá na údržbu, modernizaci komunikací a stavbu nových silnic a v neposlední řadě také na správu samotného mýtného systému. V Česku platí vozidla s hmotností nad 3,5 tuny poplatek za ujetou trasu. Vozidla do 3,5 tuny platí jednorázový poplatek, za který mohou využívat silnice neomezeně. Dne 1. prosince 2019 se systém mýta pro vozidla nad 3,5 tuny v ČR digitalizoval do podoby, v níž se používá dodnes. Fyzické mýtné brány byly nahrazeny satelitní technologií a mobilní sítí. Už není potřeba stavět fyzické mýtné brány pro výběr mýta na nových silnicích. Stačí už jen přesné souřadnice daných silnic a dálnic. Pro vozidla do 3,5 tuny byl systém také digitalizován. Systémy výběru mýta fungují i v dalších státech EU.

V této práci se zkoumá potenciál blockchain technologie pro další vylepšení systémů elektronického mýta. Metoda řešení práce je následující.

- 2 Seznámení se s technologií blockchain a chytrým kontraktem.
- 3 Provedení rešerše nad existujícími mýtnými systémy.
- 4 Vytvoření návrhu mýtného systému nad existujícím systémem s použitím technologie blockchain.
- 4.4 Výběr vhodného typu blockchain s patřičným okomentováním.
- 5 Realizace návrhu mýtného systému s použitím vybraného blockchain projektu.
- 5.4 Prezentace výsledků výsledné aplikace.

Realizace je k vidění v Github repozitáři[1].

[1] https://github.com/Solamil/etollfabric23

2 Blockchain

Blockchain je decentralizovaná neměnitelná databáze s narůstajícím počtem záznamů. Blockchain se skládá z řetězce bloků. Struktura odpovídá spojovému seznamu, kde každý blok je spojen se svým předchůdcem. Blok obsahuje transakce. [4]

2.1 Historie

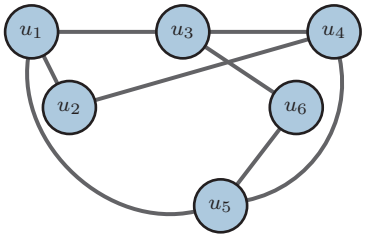
Myšlenka decentralizované databáze pochází pravděpodobně ze 70. let minulého století. Nejblíže byl aktuální podobě blockchain databáze vynálezce David Chaum v roce 1979, kdy popsal podobný systém jako trezorový (angl. vault system). V disertační práci pak v roce 1982 doplnil informace o trezorovém systému. Nicméně blockchain technologie, jak je známa dnes, si musela počkat až do roku 2008, kdy byl zveřejněn Bitcoin whitepaper[1]. V roce 2009 byla zveřejněna síť peer-to-peer s názvem Bitcoin autorem pod pseudonymem Satoshi Nakamoto. Bitcoin se stal základním stavebním kamenem pro další blockchain databáze. V roce 2014 byl zveřejněn blockchain s názvem Ethereum. [30] [22] [39]

2.2 Vlastnosti

V této kapitole jsou popsány vlastnosti blockchain vůči centrální databázi. Mezi základní vlastnosti patří decentralizace, neměnitelnost a anonymita.

Decentralizace neboli peer-to-peer je jedna z klíčových vlastností blockchain. Blockchain databáze je tvořena propojenými klienty. Klient komunikuje s dalšími klienty (uzly) sítě a přeposílají si aktuální stav. Každý klient je server v blockchain. Odpojením jednoho z uzlů ze sítě nedojde k nefunkčnosti blockchain. Záznam blockchain zůstává zachován v ostatních připojených uzlech. [28]

[1] **Whitepaper** – Zpráva pojednávající o řešení problému.



Obrázek 2.1: Ukázka propojených klientů blockchain sítě

Neměnitelnost zaručuje, že historie blockchain nelze odstranit nebo změnit. Neměnitelnost je zaručena strukturou blockchain viz 2.4. Odstranit nebo změnit konkrétní data by znamenalo přepsat celou historii od provedené změny až po současnost. Jakákoliv snaha o změnu historie vyžaduje velké množství energie nebo kapitálu. Ve většině případů se více vyplatí být upřímným členem sítě.

Anonymita je zajištěna asymetrickou kryptografií. Skládá se z páru soukromého a veřejného klíče. Soukromý klíč zůstává v utajení pomocí hesla. Veřejný klíč je přístupný v rámci sítě. Každá transakce je přiřazena k určitému páru. Soukromým klíčem se podepisuje transakce. Veřejným klíčem se ověřuje transakce. Veřejnému klíči se také jinak říká adresa. [32]

2.3 Blockchain botanická

Blockchain databáze se dělí na tzv. permissionless a permissioned. Permissionless databáze jsou například Ethereum a Bitcoin. Jsou to sítě přístupné pro všechny uživatele bez centrální autority. Každý připojený klient může číst nebo upravovat blockchain. Obsah těchto sítí je plně transparentní na internetu. Sítě uchovávající data zašifrovaná jsou např. Zerocash nebo Monero. Data jsou transparetní jen pro uživatele, pro něž jsou určeny.

Permissioned jsou např. Hyperledger nebo Corda. Jsou to sítě přístupné jen s oprávněním. Oprávnění určuje centrální autorita sítě. V tabulce 3.1 jsou shrnuty jejich vlastnosti.

	Permissionless blockchain	**Permissioned blockchain**	**Centrální databáze**
Propustnost	Malá	Vysoká	Velmi vysoká
Odezva	Pomalá	Střední	Rychlá
Počet čtení	Vysoké	Vysoké	Vysoké
Počet zapisování	Vysoké	Nízké	Vysoké
Počet nedůvěryhodných zapisování	Vysoké	Nízké	0
Consensus mechanismus	PoW, PoS, ZKP	BFT protokol (PBFT)	Žádný
Centrálně spravováno	Ne	Ano	Ano

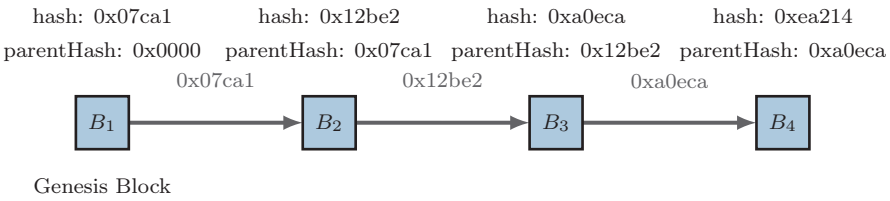
Tabulka 2.1: Vlastnosti sítí Permissionless a Permissioned. [40]

Dále se sítě dělí na veřejné a soukromé. Permissionless sítě jsou veřejné. Permissioned sítě mohou být veřejné, nebo soukromé.

Hybridní blockchain je kombinací permissionless a permissioned blockchain. Část dat je soukromá a některá jsou veřejná vůči všem klientům blockchain. Příkladem hybridní sítě může být Everledger. [20]

2.4 Základní struktura

Základní strukturou blockchain je blok. Blok obsahuje transakce. Blockchain může být definován jako orientovaný graf $G = (B,\ parentHash)$, kde B jsou vrcholy a $parentHash$ jsou hrany viz obrázek 2.2. Hrana $parentHash$ odkazuje na další blok. Blok B_1 je nejstarší, blok B_4 je nejnovější. Kořen v blockchain síti se nazývá Genesis (angl. počátek).



Obrázek 2.2: Graf $G = (B,\ parentHash)$

Hash je vypočítán z obsahu bloku, tudíž nějaká změna v bloku vytvoří rozdílný hash. Hash se poté vloží do dalšího bloku ($parentHash$) a z obsahu (včetně $parentHash$) bloku opět vypočítá hash pro nastávající blok. Změna obsahu některého bloku by znamenala přepočítat všechny hashe v navazujících blocích. Pokud by hash neodpovídal v některém z bloků, ani další bloky by na sebe nenavazovaly.

2.4.1 Timestamp server

Timestamp (v překladu časový záznam) server zajišťuje, že data musela existovat v uvedeném čase nebo před uvedeným časem. Timestamp server může představovat důkaz o existenci (angl. proof of existence). Z dat a aktuálního času se vytvoří hash řetězec. Pokud se změní data nebo čas, výsledný hash řetězec nebude odpovídat. Timestamp je obvykle přidělen důvěryhodnou třetí stranou neboli timestamp authority (TSA).

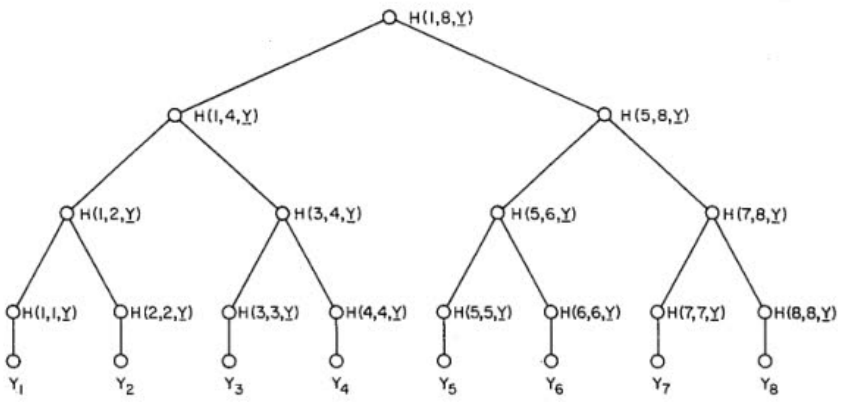
V blockchain se timestamp server používá při vytváření bloků a transakcí. Každý timestamp bloku je poté obsažen v následujícím bloku jako hash řetězec. [17]

2.4.2 Merkle tree

Merkleův strom (angl. Merkle tree) nebo také hash strom (angl. hash tree) je stromová autentizace, ve které je každý vrchol tvořen hash řetězcem. Autorem je Ralph Merkle. Pomocí Merkleova stromu se ověřuje integrita dat, zda nedošlo k poškození nebo jakékoliv změně například při přenosu.

Merkleův strom je definován následovně. Je dán vektor dat $Y = Y_1, Y_2, ...Y_n$. Merkle tree zaručuje ověření náhodně vybraného Y_i. Pro ověření Y_i je definována funkce $H(i, j, Y)$.

$$\begin{aligned} H(i, i, Y) &= F(Y_i) \\ H(i, j, Y) &= F(H(i, \frac{i+j-1}{2}, Y), H(\frac{i+j+1}{2}, j, Y)) \end{aligned} \tag{2.1}$$



Obrázek 2.3: Merkle strom.

Je potřeba ověřit například Y_5 z obrázku 2.3. Příjemci stačí odeslat jen konkrétní

vrcholy tj. kořen $H(1,8,Y)$, data Y_5, $H(6,6,Y)$, $H(7,8,Y)$ a $H(1,4,Y)$. Ostatní vrcholy jsou dopočítané v průběhu cesty grafem.

V blockchain databázi jsou data $Y_1...Y_n$ transakce. Klient pro ověření jedné transakce v bloku nemusí procházet všechny transakce. Jediná změna v transakci vytvoří úplně jiný hash řetězec v kořenu stromu. Náročnost na přenos dat je $\log_2 n$ při použití binárního stromu. [21]

2.5 Consensus

Consensus v překladu znamená dohoda. Základní dohodou je, že minimálně $\frac{2}{3}$ klientů se musí dohodnout na historii transakcí. Menší číslo představuje riziko pro blockchain. Consensus mechanismus je soubor pravidel, se kterými se klienti mezi sebou musejí dohodnout na aktuálním stavu blockchain.

Popsány jsou následující consensus mechanismy

- 2.5.1 Byzantine Fault Tolerance (BFT),
- 2.5.2 Proof of work (PoW),
- 2.5.3 Proof of stake (PoS) a
- 2.5.4 Zero-knowledge proof (ZKP).

BFT slouží ke strategii zachování consensu v blockchain, pokud některý klient nesouhlasí. Permissioned blockchain databáze používají primárně BFT. PoW mechanismus je používán výhradně typem permissionless, např. Bitcoin, Litecoin, Monero...atd. Viz tabulka 2.2.

Typ	**Blockchain**	**Poznámka**
BFT	Hyperledger, Corda	
PoW	Bitcoin, Ethereum, Litecoin, Zcash,...	Energeticky náročné, Ethereum přechází na PoS dne 15. září 2022.
PoS	Cardano, Ethereum	Podle vývojářů je Ethereum s PoS o 99,95% úspornější. [12]
ZKP	Zcash	Používá se pro ověřování transakcí.

Tabulka 2.2: Seznam consensu mechanismů.

2.5.1 Byzantine Fault Tolerance

Byzantine Fault Tolerance (BFT) nebo Byzantine dohoda je tolerance výskytu chyb v systému s nespolehlivou informací o chybě systému. BFT je analogicky označován jako problém generálů. Skupina generálů má zaútočit na pevnost. Pokud všichni zaútočí najednou, bitva bude pro ně vítězná. Pokud někdo složí zbraně, bitva bude prohraná. Pokud někdo složil zbraně, došlo k přenosu chybných informací mezi

generály. Jde o dohodu všech generálů. Buď všichni zaútočí, nebo se všichni stáhnou. BFT tedy toleruje možný výskyt chyb při přenosu a implementuje tím strategii, jak se v takové situaci dohodnout.

Formálně by se BFT dalo definovat následovně. Nechť $P = \{p_1, ..., p_n\}$ je množina hráčů a $\mathcal{D}$ je konečná množina dat. V síti Ξ, kde každý hráč $p_i \in P$ má vstupní hodnotu $x_i \in \mathcal{D}$, při které se rozhodne na výstupu $y_i \in \mathcal{D}$. Dohoda na výstupu y_i je platná za následujících podmínek. Dohoda je platná, pokud všichni hráči se stejnou vstupní hodnotou $x_i = v$ dospěli k výstupní hodnotě $y_i = v$. Zároveň se všichni validní hráči dohodnou na stejném výstupu. Pokud např. $p_i \in P$ a $p_j \in P$ jsou validní, poté platí $y_i = y_j$. [15]

2.5.2 Proof Of Work

Proof of work (PoW) neboli *cost-function* je funkce parametricky drahá na výpočet, zároveň výsledek je efektivně ověřitelný. Výsledek funkce je veřejně ověřitelný jakoukoliv třetí stranou bez nutnosti opakování drahého výpočtu.

Definovaná funkce MINT() vypočítá token $\mathcal{T}$. Parametr w udává průměrnou náročnost, kolik výpočetního výkonu bude potřeba pro výpočet tokenu $\mathcal{T}$. Parametr s je bitový řetězec {0,1}. Indexování začíná od 1, kde hodnota $[s]_1$ je nejvíce vlevo a hodnota $[s]_{|s|}$ je nejvíce vpravo. Funkce VALUE() vyhodnotí správnost tokenu $\mathcal{T}$. [6]

$$\begin{cases} \mathcal{T} \leftarrow \text{MINT}(s, w) \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) \end{cases} \tag{2.2}$$

Definovaný porovnávací operátor $=_b^{left}$, kde b délka části podřetězce zleva.

$$\begin{aligned} x =_0^{left} y \quad & [x]_1 \neq [y]_1 \\ x =_b^{left} y \quad & \forall_{i=1...b} [x]_i = [y]_i \end{aligned} \tag{2.3}$$

$$\begin{cases} \text{PUBLIC:} & \text{hash funkce } \mathcal{H}(\cdot) \text{ s výstupní velikostí } k \text{ bitů} \\ \mathcal{T} \leftarrow \text{MINT}(s, w) & \textbf{find } x \in R\{0,1\}^* \textbf{ st } \mathcal{H}(s||x) =_w^{left} 0^k \\ & \textbf{return } (s, x) \\ \mathcal{V} \leftarrow \text{VALUE}(\mathcal{T}) & \mathcal{H}(s||x) =_v^{left} 0^k \\ & \textbf{return } v \end{cases} \tag{2.4}$$

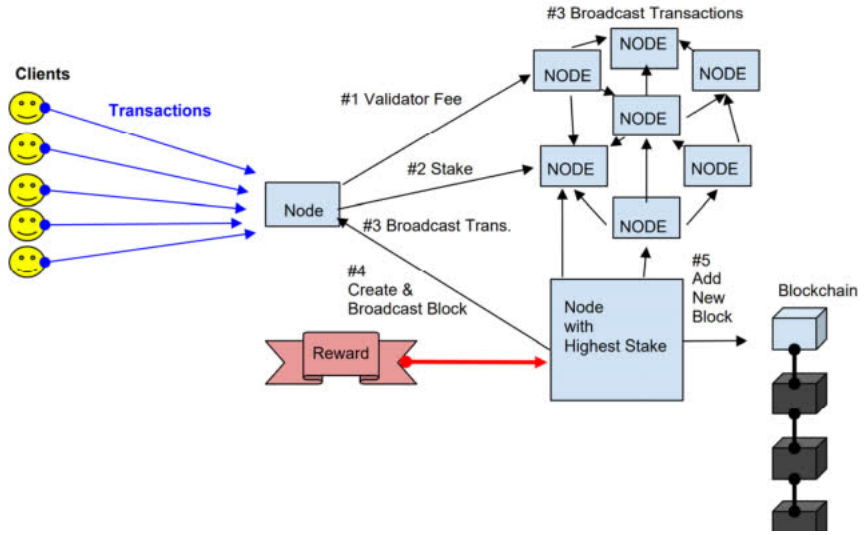
Funkce MINT() hledá shodu všech 0 bitů k-bitového řetězce 0^k. Nejrychlejším algoritmem nalezení shody je hrubá síla (angl. brute force). [7]

2.5.3 Proof Of Stake

Proof of Stake (PoS) nebo také Proof of Ownership je ručení majetkem (kapitálem) za pravost dat. Klient vloží kapitál a ten se v rámci blockchain uzamkne. Klient se poté stane validátorem. Blockchain pseudonáhodně vybere validátora pro vytvoření bloku.

Ostatní validátoři hlasují pro, nebo proti, zda má být blok přidán do blockchain, či nikoliv.

PoS je výrazně méně energeticky náročnější než PoW. Na obrázku 2.4 je znázorněn zjednodušený princip mechanismu PoS.



Obrázek 2.4: Zjednodušený princip mechanismu PoS. [19]

2.5.4 Zero-knowledge Proof

Zero-knowledge proof (ZKP) je metoda pro ověření výroku, že je pravdivý. Metoda ZKP je definována 3 parametry.

- Úplnost (angl. Completeness) – pokud je výrok pravdivý, upřímný ověřovatel bude přesvědčen faktem upřímným dokazovatelem.
- Neporušenost (angl. Soundness) – pokud je výrok nepravdivý, nespolehlivý, dokazovatel může přesvědčit upřímného ověřovatele jen s velmi malou pravděpodobností.
- Nulová znalost (angl. Zero-knowledge) – pokud je výrok pravdivý, ověřovatel zjistí jenom fakt, že výrok je pravdivý. Nic víc o výroku nezjistí (zero-knowledge).

ZKP se rozděluje do výzev (angl. challenges), kde pravděpodobnost úspěchu je $0 < q < 1$, že dokazovatel přesvědčí ověřovatele. V k iterací se pravděpodobnost rovná $\lim_{k\to\infty} q^k = 0$. Pravděpodobnost musí být malá, aby přesvědčila ověřovatele, že výrok je pravdivý. [9]

2.6 Chytrý kontrakt

Chytrý kontrakt (angl. smart contracts) je libovolný spustitelný kód uložený v blockchain. Automatizuje spuštění transakcí na základě splněných podmínek vep-

saných v kontraktu. Chytrý kontrakt má vlastní unikátní adresu, zůstatek, úložiště a spustitelný kód. Podle typu blockchain jsou data chytrého kontraktu soukromá, nebo veřejně dostupná. Chytrý kontrakt se nasazuje vytvořením nové transakce. Po nasazení se kód nemůže měnit. Spustit chytrý kontrakt znamená vytvořit transakci. Na základě transakce kontrakt čte, zapisuje do úložiště. Chytrý kontrakt může nasadit další chytrý kontrakt. [4]

Blockchain	**Programovací jazyky**
Ethereum	Solidity, Vyper
Bitcoin	Skriptovací jazyk
Hyperledger	chaincode; Go, Javascript a Java
Corda R3	Kotlin, Java

Tabulka 2.3: Blockchain databáze podporující chytré kontrakty.

2.6.1 Historie

Chytrý kontrakt byl prvně zmíněn v roce 1996 autorem Nick Szabo. V článku se autor nezmiňuje o decentralizovanosti nebo blockchain. Chytrý kontrakt by naopak měl být podle autora spravován jedním centralizovaným serverem. Jde o vnoření chytrého kontraktu do hardwaru nebo softwaru. Příklad, na němž to vysvětluje, je, když si osoba pronajme vozidlo. Pokud splátky splácí, má v chytrém kontraktu nadefinováno, že má právo vozidlo využívat. Pokud ne, vozidlo propadne zpět původnímu majiteli. Prvním chytrým kontraktem podle autora byl prodejní automat. [35]

2.6.2 Dělení chytrého kontraktu

Chytrý kontrakt se dělí na deterministický a nedeterministický. Deterministický kontrakt nepoužívá žádná externí data ze třetí strany. Nedeterministický závisí na externích datech (tzv. oracles) ze třetí strany. Například informace o počasí, výsledcích turnajů, kurz měn ...atd. Tímto ale může být integrita chytrého kontraktu narušena. Propojením externích dat do chytrého kontraktu se zabývá například projekt Chainlink. [10]

3 Výběr mýta

Mýto (angl. road pricing, toll) se platí na určitou dobu[1], nebo za ujetou vzdálenost[2]. Na určitou dobu se platí jednorázový poplatek. Poplatek je přiřazen ke konkrétní SPZ[3]. Kontrola probíhá pomocí automatického rozpoznávání SPZ, nebo silniční kontrolou. Po zaplacení může vozidlo využívat placené komunikace bez omezení na ujetou vzdálenost. Nevýhodou může být menší efektivita než u platby za ujetou vzdálenost. [18]

V podkapitolách jsou popsány systémy pro zpoplatnění za ujetou vzdálenost. Systémy platí většinou pro vozidla nad 3,5 tuny a vyžadují palubní jednotku připevněnou na čelním skle vozidla. V rámci států v EU jsou systémy kompatibilní. Stačí jenom jedna palubní jednotka pro více států. [8] [13] [23]

Mýtný systém	**Typ vozidel**	**Vlastnosti**
3.1 Výběrčí kabiny	všechna vozidla	Snížená propustnost, často automatizované.
Video tolling	všechna vozidla	Rozpoznávání SPZ. Nutnost stavět fyzické mýtné brány.
3.2 Systém LSVA	> 3,5 t	Mikrovlnná komunikace, počítání tunokilometry.
3.3 Systém AGE	> 3,5 t	Mikrovlnná komunikace, odečítání kreditu.
3.4 Německý Toll Collect	> 3,5 t	Komunikace GSM a mikrovlnná, satelitní technologie.
3.5 SEM	> 3,5 t	CZ, SK, Komunikace GSM/GPRS a mikrovlnná, hybridní jednotka, satelitní technologie.
4 **Aplikace**	**> 3,5 t**	**Komunikace IP[4] a HTTP(S), satelitní technologie, blockchain, chytrý kontrakt.**

Tabulka 3.1: Shrnutí mýtných systémů.

[1] Angl. period-based charge
[2] Angl. mileage-based charge
[3] **SPZ** – Státní poznávací značka
[4] **IP** - Internet Protocol

3.1 Výběrčí kabiny

Výběrčí kabiny jsou především vidět v jižních evropských státech jako Španělsko, Itálie a Francie. Platba se provádí v hotovosti s lidskou obsluhou, nebo automatem. Nedostatkem tohoto řešení je snížení propustnosti vozidel o více než polovinu. Výběrčí kabiny se proto automatizují, aby umožnily plynulejší provoz. Automatizace může být zajištěna palubní jednotkou. Palubní jednotka se při příjezdu k výběrčí budce detekuje pomocí mikrovlnné komunikace. Závora se po úspěšné transakci automaticky zvedne. Při detekci z větší vzdálenosti nemusí vozidlo zastavovat, jen nezbytně sníží rychlost pro provedení transakce a zvednutí závory.

3.2 Systém LSVA

Systém LSVA (Liestungsabhängige SchwerVerkehrsAbgabe) funguje na všech švýcarských silnicích. Poplatek je odvozený od hmotnosti nákladního vozidla, tzv. tunokilometry. Palubní jednotka se nazývá Tripon. Zařízení Tripon detekuje vjezd a výjezd ze zpoplatněné zóny. Ve zpoplatněné zóně počítá tunokilometry. Výjezd ze zpoplatněné zóny ukončí počítání. Tripon komunikuje prostřednictvím mikrovlnného systému DSRC (Dedicated Short Range Communication) s mýtnými bránami.

3.3 Systém AGE

Systém AGE (Automatische GebührenErhebung) detekuje vozy prostřednictvím mikrovlnné komunikace DSRC. Tento systém funguje v Rakousku. Palubní jednotka se nazývá GO-Box. V GO-Box jsou načteny informace o kategorii vozidla, počet náprav, hmotnost, emisní třída atd. Z GO-Box jsou odečítávany poplatky mýtnými bránami. Do GO-Box je možné vložit kartu, která se „dobíjí“ jako kredit u mobilního operátora. Systém musí dodržovat platné předpisy pro ochranu osobních údajů. Pokud mýtná brána zaznamená neproběhlou transakci, je pořízen videozáznam vozu s SPZ. Na konci účetního období je vytvořen daňový doklad obsahující vyúčtování mýtných poplatků.

3.4 Německý Toll Collect

Německý Toll Collect systém nebo také ETC (Electronic Toll Collection) používá pro zpoplatnění tras za použití navigačního systému GPS[5]. K přenosu dat do centrály používá mobilní síť GSM. Palubní jednotka se označuje jako OBU (OnBoard Unit). OBU obsahuje uloženou elektronickou mapu zpoplatněných silnic. K nalezení shody se používá systém diferenciální GPS. Při použití GPS pro identifikování polohy dochází k nepřesnosti stanovení polohy. Systém ETC k tomu používá terestrické

[5] **GPS** - Global Positioning System

vysílače. Vysílače redukují nepřesnost na několik centimetrů. Přesto v pohybujícím se vozidle dosahuje nepřesnost až 36 m.

Algoritmus v OBU neustále porovnává polohu vozidla s mapou zpoplatněných silnic. Poté se určí počet ujetých kilometrů na zpoplatněných silnicích a pomocí údajů o vozidle se stanoví poplatek. Vypočtený poplatek je odeslán GSM sítí do centrály.

Pro kontrolu slouží 300 mýtných bran umístěných na německých dálnicích, dále se kontroly provádějí celní správou. Brány mají laserové senzory, které rozpoznají, do jaké kategorie patří podle rozměru vozidla a počtu náprav. Dále se vytvoří celkový snímek vozidla včetně SPZ. Pomocí strojového vidění se rozpozná SPZ a porovná se s daty v databázi. Pokud data souhlasí, pořízená data o vozidle se ihned vymažou.

3.5 Systém elektronického mýta

Systém elektronického mýta (SEM) funguje pro vozidla v ČR také nad 3,5 tuny. Dne 1. prosince 2019 byl nahrazen mikrovlnný systém za satelitní technologii (GNSS[6]). Mikrovlnný systém slouží pro kontrolu. K zpoplatnění se používá poloha určená GNSS. Palubní jednotka se nazývá Billien OBU. Jedná se o hybridní jednotku. To znamená, že je schopno pracovat v režimu tenkého klienta, anebo chytrého klienta. Tenký klient sbírá data a odesílá je rovnou do centrály. Chytrý klient sám rozpozná zpoplatněný úsek pomocí algoritmu. Palubní jednotka komunikuje přes mobilní síť GSM přenosovou rychlostí od 85,6 kbit/s. Modernější jednotky umožňují až 1119 kbit/s skrze mobilní sítě EDGE a LTE. Billien OBU používá pro určení pozičních dat systémy GPS, Gallileo a GLONASS. [36] [37]

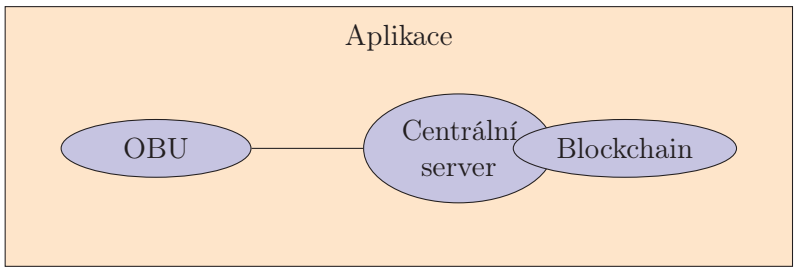
V palubní jednotce je uložen geografický model všech zpoplatněných komunikací. V případě chytrého klienta porovnává algoritmus aktuální polohu s modelem. Při shodě dojde k tzv. mýtné události. Událost se odešle GSM sítí serveru pro zpracování. Kontrola se provádí mikrovlnnou technologií 5,8 GHz. Stejný systém se používá také na Slovensku.

[6] **GNSS** - Global Navigation Satellite System

4 Návrh aplikace

Cílem práce je vytvořit aplikaci pro zpoplatnění silnic za použití blockchain technologie. Dalším cílem této práce je zpoplatnit jen vozidla nad 3,5 tuny. Tato práce vychází ze SEM používaný v Česku. Je nutné zmínit, že není možné v rámci této práce zpracovat celou problematiku výběru mýta se všemi jejími detaily, které dělají systém efektivním. Z hlediska času, dostupnosti informací a celkové rozsáhlosti.

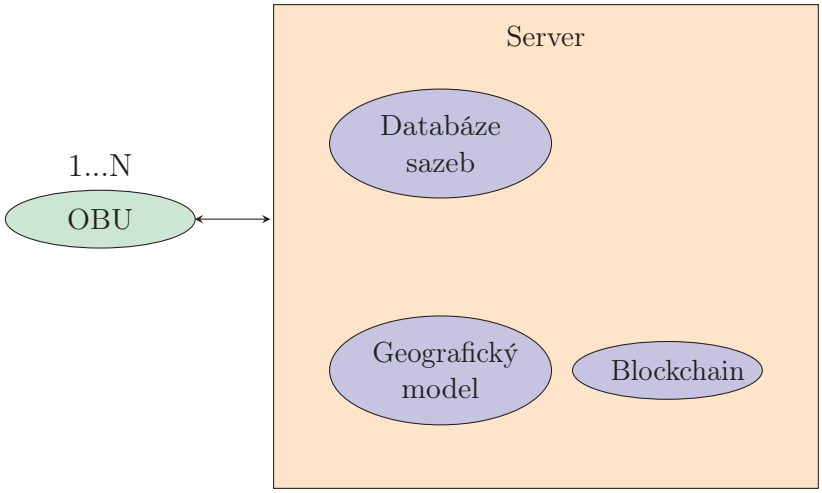
Celá aplikace se skládá z palubní jednotky (OBU) ve vozidle a centrálního serveru. Centrální server používá blockchain databázi k evidování kreditu pro jednotlivá vozidla zpoplatněné komunikace. Centrální server si uchovává kopii blockchain databáze. Každé vozidlo má svůj kredit. Vozidlo je jednoznačně identifikováno v databázi pomocí SPZ.



Obrázek 4.1: Diagram aplikace

4.1 Centrální server

Centrální server komunikuje s jednotkami OBU přes protokol HTTP(S). Struktura posílaných dat je ve formátu JSON. Server posílá geografický model a informace o stavu kreditu jednotce OBU. OBU posílá serveru informace o jízdě na zpoplatněné komunikaci.



Obrázek 4.2: Diagram centrálního serveru

4.1.1 Databáze sazeb

V Česku se sazba mýta počítá za ujetý 1 km v českých korunách. Poplatek závisí na následujících parametrech

- dálnice/silnice I. třídy,
- denní období (denní sazba od 05:00:00 – 21:59:59 hodin, noční sazba od 22:00:00 – 04:59:59 hodin),
- kategorie vozidel N, M2 a M3,
- emisní třída,
- počet náprav a
- hmotnost vozidla.

Obecně platí, že silnice I. třídy jsou levnější než dálnice. Denní sazba je levnější než noční sazba. Čím novější je emisní třída, tím je nižší sazba (poplatek za znečištěné ovzduší). Mezi hmotností vozidla, počtem náprav a sazbou je přímá úměra (poplatek za hlukové znečistění). [33]

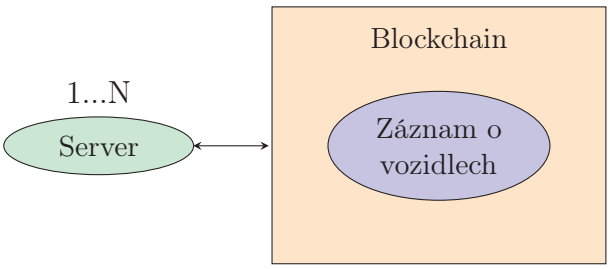
4.1.2 Geografický model

Geografický model je databáze zpoplatněných úseků. Pro model postačí jako databáze GPX soubory. Model je z velké části zjednodušen o reálnou situaci – vedlejší souběžné cesty, překřížené cesty, mosty a tunely. Model je vytvořen z internetové stránky mapy.cz. Každý mýtný úsek je označen GPS bodem každých 20 metrů.

Ve skutečnosti ŘSD dodává přesné souřadnice zpoplatněných úseků. Následně se zpracují pro model externí firmou. Poté se data ještě speciálně upravují pro

algoritmus a jednotku OBU, tak aby bylo možné efektivně detekovat mýtný úsek a počítat ujetou vzdálenost. Veškeré další informace o fungování modelu podléhají tajemství. [3]

4.1.3 Blockchain



Obrázek 4.3: Diagram blockchain

Blockchain má za úkol uchovat záznamy o vozidlech a jejich platbách za mýto. Záznam o vozidle má následující atributy.

- SPZ, Státní příslušnost
- Stav kreditu – platba předem, nebo fakturou
- Kategorie vozidla – N, M2 a M3
- Třída emisí – Euro 0, 1, 2, 3, 4, 5, EEV[1], 6, a CNG[2]/BIO[3]
- Počet náprav – 2, 3, 4, 5 a víc
- Hmotnost – $>$ 3,5 tuny [33]

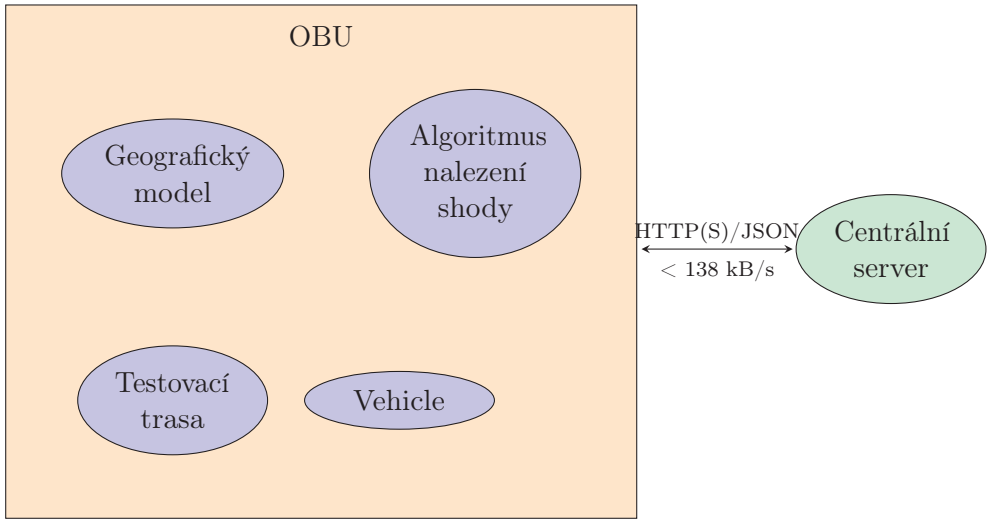
Atributy SPZ, státní příslušnost a kategorie vozidla jsou neměnitelné a zároveň tvoří identifikátor vozidla. V databázi nemůžou být dva záznamy se stejnou SPZ, státní příslušností a kategorií vozidla. Ostatní atributy se mohou měnit.

Mýto je možné v SEM platit předem (angl. pre-paid), nebo na fakturu (angl. post-paid). Platba předem funguje jako „dobíjení“ kreditu u mobilního operatora. Dopravce vloží částku, která se po projetí po mýtném úseku odečítá. Platba na fakturu funguje tak, že dopravci je účtováno mýto. Na konci účetního období je dopravci poté vystavena faktura na zaplacení mýta. Nejpoužívanější je platba na fakturu z důvodu bankovní záruky. [3]

[1] **EEV** – Enhanced environmentally friendly vehicle - Emisní norma mezi Euro 5 a 6.
[2] **CNG** - Stlačený zemní plyn (metan)
[3] **BIO** - Bioplyn se vytváří biologickým rozkladem. Po zpracování identický s CNG. [5]

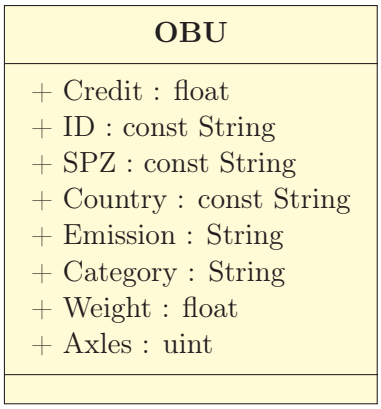
4.2 Palubní jednotka – OBU

OBU je program, který simuluje jízdu vozidla po trase určené GPX protokolem. OBU je nepřenositelná a zároveň vázaná na konkrétní SPZ. OBU pracuje v režimu chytrého klienta. Chytrý klient sám rozpozná zpoplatněnou komunikaci pomocí geografického modelu. Model se aktualizuje z centrálního serveru při startu jednotky.



Obrázek 4.4: Diagram OBU

OBU je definováno strukturou viz diagram 4.5. Tato struktura je kompatibilní také pro vozidla do 3,5 tuny.



Obrázek 4.5: Struktura OBU

4.2.1 Testovací trasa

Testovací trasa je definována podle zadání protokolem GPX. GPS Exchange Format (GPX) je XML[4] formát pro zápis GPS souřadnic. Na GPX se nevztahuje placená licence. XML je rozšiřitelný značkovací jazyk. GPX Je definován pomocí párových a nepárových tagů. Tag má název a hodnotu. Hodnota se může skládat z dalších tagů. Tag může také obsahovat parametry. Zdrojem souborů pro tuto práci je internetová stránka mapy.cz.

Jednotka očekává GPX soubor v následujícím formátu viz kód 4.1. Soubor obecně začíná hlavičkovými tagy, které řeší kompatibilitu mezi zařízeními. Soubor obsahuje jednu souvislou cestu. Z důvodu nepřesnosti určení polohy za jízdy je nutné do testovacího souboru vložit chybu.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <gpx xmlns="http://www.topografix.com/GPX/1/1" version
      ="1.1" creator="https://mapy.cz/">
3        <wpt lat="50.615327" lon="15.111531">
4              <ele>318.000000</ele>
5        </wpt>
6        <wpt lat="50.615275" lon="15.111810">
7              <ele>317.712708</ele>
8        </wpt>
9        .
10       .
11       .
12 </gpx>
```

Zdrojový kód 4.1: Ukázka GPX souboru

Cesta je tvořena jednotlivými GPS body označenými jako `wpt`. Tag `wpt` je jednotný bod a má parametry zeměpisné souřadnice `lat` a `lon`. Přesnost zeměpisných souřadnic se vyskytuje kolem 6 desetinných míst. To odpovídá určení polohy s přesností kolem desítky centimetrů viz tabulka 4.1. [11]

[4] **XML** - eXtensible Markup Language

Počet desetinných míst	Stupně (°)	Vzdálenost
0	1	111 km
1	0.1	11.1 km
2	0.01	1.11 km
3	0.001	111 m
4	0.0001	11.1 m
5	0.00001	1.11 m
6	0.000001	11.1 cm
7	0.0000001	1.11 cm
8	0.00000001	11.1 mm

Tabulka 4.1: Přesnost zeměpisných souřadnic na počet desetinných míst. [27]

4.2.2 Geografický model

Geografický model zpoplatněných komunikací si OBU stáhne ze serveru. Následně dojde k jeho uložení v OBU. Stahování se provádí při startu OBU. Nejdříve se zjistí, jestli model byl před posledním použitím uložen. Poté se zjistí lokální verze modelu a porovná se s verzí na serveru. Pokud je lokální verze starší, stáhne se ze serveru novější model. Zkontroluje se také kontrolní součet z dat, zda nedošlo k lokální modifikaci modelu.

4.2.3 Algoritmus pro nalezení shody

Cílem algoritmu je detekovat mýtný úsek a spočítat počet ujetých kilometrů po zpoplatněném úseku. Vstupem pro algoritmus je aktuální pozice a model. Pozice i model musejí mít přesnost GPS souřadnic alespoň na 4 desetinná místa viz tabulka 4.1. Pro detekci úseku se počítá vzdálenost ke každému bodu k modelu. Hraniční vzdálenost nebo také práh bodu testovací trasy k modelu je mezi 20 až 30 m. To vyplývá z rozměrů silnic a dálnic. Silnice I. třídy mají šířku od 9,5 až 24,5 m. Dálnice mají šířku od 26,5 do 27,5 m dle ČSN. [38]

Algoritmus pracuje na základě vzdálenosti dvou bodů. K výpočtu vzdálenosti se použije Haversinův vzorec. Vzorec se používá pro výpočet vzdálenosti dvou bodů na kulové ploše. Vzorec pracuje se souřadnicemi v radiánech. [24]

- r - poloměr země
- $lat1$, $lon1$ - zeměpisné souřadnice 1. bodu
- $lat2$, $lon2$ - zeměpisné souřadnice 2. bodu
- $dlat$, $dlon$ - rozdíl zeměpisných souřadnic dvou bodů
- d - vzdálenost mezi dvěma body

$$dlat = lat2 - lat1 \tag{4.1}$$

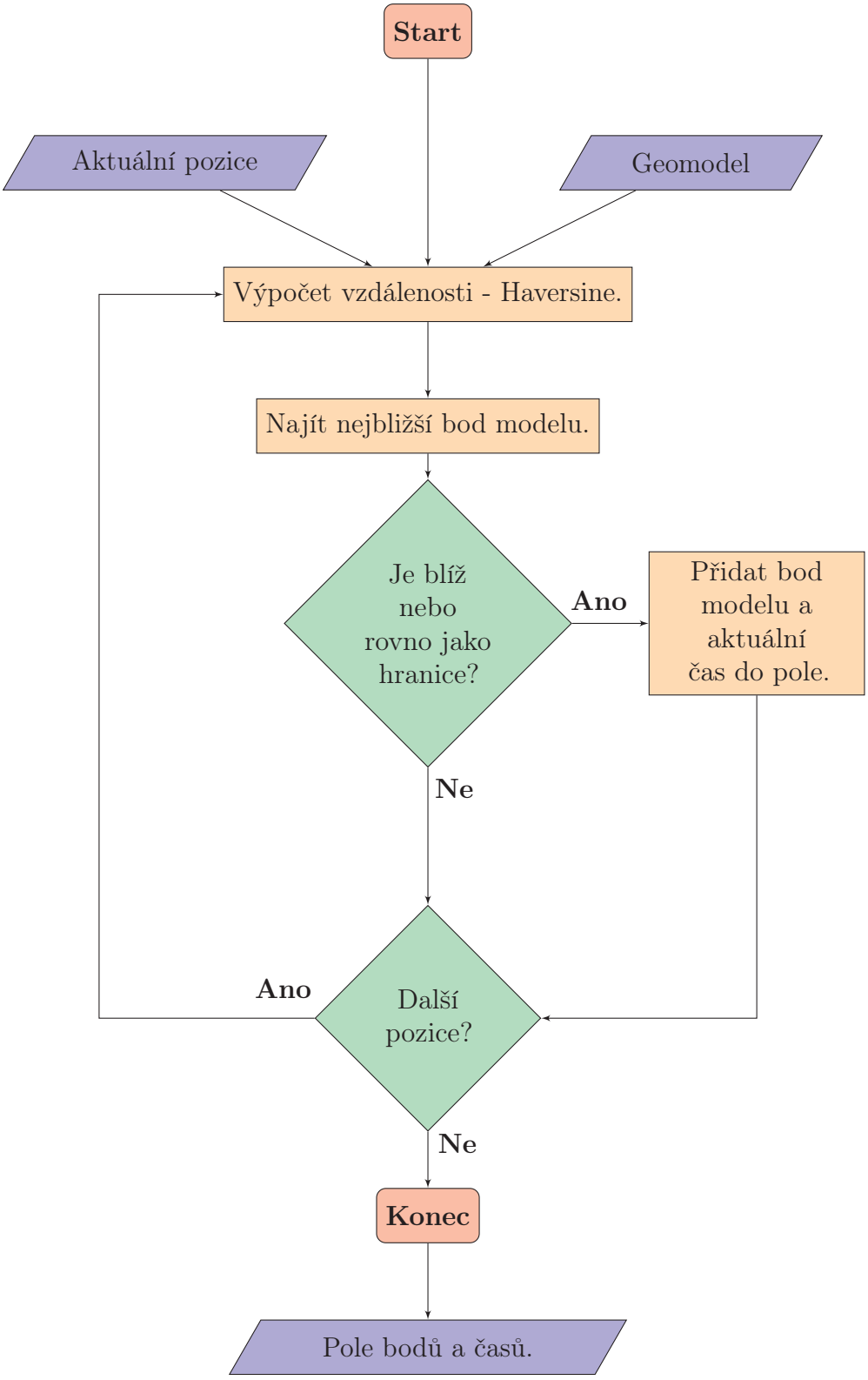
$$dlon = lon2 - lon1 \tag{4.2}$$

$$d = 2r \arcsin(\sqrt{\sin^2(\frac{dlat}{2}) + \cos(lat1) \cdot \cos(lat2) \cdot \sin^2(\frac{dlon}{2})}) \tag{4.3}$$

Pro převod ze stupňů na radiány se používá vzorec viz 4.4, kde *degrees* jsou stupně.

$$rad = degrees \cdot \frac{\pi}{180} \tag{4.4}$$

Kroky algoritmu jsou následující viz diagram 4.6.



Obrázek 4.6: Vývojový diagram algoritmu pro nalezení shody

Ještě před přidáním bodu do pole je dobré zkontrolovat, zda se bod už nachází v poli. Tímto se může zamezit duplikacím a výrazně snížit velikost pole. I když na samotný výpočet vzdálenosti to nemá vliv, z důvodu vzdálenosti dvou identických bodů je prakticky 0.

Výstupem algoritmu jsou uložené body v poli. K jednotlivým bodům náleží také datum a časový záznam s přesností na sekundy. Časový záznam se hodí u přechodu z denní na noční sazbu a naopak. Dále se hodí při změně tarifu od konkrétního data. Po ukončení algoritmu se body s časovým záznamem odešlou na server.

4.3 Bezpečnost

Bezpečnost má za cíl, aby server výhradně komunikoval s jednotkami OBU a zároveň nedovolil nikomu neoprávněnému přístup na server. Nejdříve je nutné zanalyzovat, o co se může potenciální útočník pokoušet. V rámci této práce by se mohl pokoušet o následující:

- Změna údajů v OBU.
- Změna geografického modelu v OBU, tzn. vyhnout se zpoplatněným úsekům.
- Úprava stavu kreditu.

Údaje jsou kontrolovány s údaji na serveru při startu OBU. Údaje se musí shodovat. Server má vždy správné údaje o OBU. Geografický model se kontroluje s verzí na serveru při startu OBU. Úprava stavu kreditu by musela nastat přímo na serveru, protože stav kreditu v OBU se jen čte ze serveru.

4.4 Výběr blockchain databáze

Pro aplikaci je nutné rozhodnout, jaký typ blockchain databáze použít, zda je vhodné použít permissionless, nebo permissioned, a dodatečně rozhodnout, zda použít soukromý, nebo veřejný permissioned blockchain. K tomu je použit vývojový diagram 7.2 viz příloha.

- **První rozhodovací blok** se ptá, zda je potřeba ukládat stav. Odpověď je „ano", protože aplikace chce mít přehled o aktuálním stavu kreditu, hmotnosti, počtu náprav a emisní třídy. Tímto je odkazováno na další rozhodovací blok. Pokud by odpověď byla „ne", rozhodnutí by vedlo nepoužívat blockchain.
- **Druhý rozhodovací blok** se ptá, zda existuje více než jeden zapisovatel. Odpověď je „ano" z důvodu, že k databázi může přistupovat více serverů, ať už z hlediska škálovatelnosti, nebo interkompatibility s ostaními zeměmi v rámci EU. Pokud by odpověď byla „ne", rozhodnutí by vedlo nepoužívat blockchain.

- **Třetí rozhodovací blok** se ptá, zda se může vždy použít online třetí strana. Odpověď je „ne" z důvodu povahy zadání této práce. Pokud by nezáleželo na zadání, odpověď by byla „ano", protože je to reálně častější řešení. Pokud by odpověď byla „ano", rozhodnutí by vedlo nepoužívat blockchain.

- **Čtvrtý rozhodovací blok** se ptá, zda všichni zapisovatelé jsou známí. Odpověď je „ano" z důvodu, že se jedná o službu, která musí splňovat zákony a směrnice EU. Pokud by odpověď byla „ne", rozhodnutí by vedlo použít permissionless blockchain. Tímto se stává nevhodným použít blockchain typu permissionless jako např. Ethereum. Kredit všech vozidel by byl veřejný. To narušuje ochranu soukromí uživatelů. Tento krok rozhodl, že bude vhodnější použít permissioned blockchain. V dalších krocích se rozhodne, zda použít veřejný, nebo soukromý blockchain, pokud se odmyslí varianta nepoužívat blockchain z důvodu povahy zadání práce.

- **Pátý rozhodovací blok** se ptá, zda všichni zapisovatelé jsou důvěryhodní. Odpověď je „ano". Pro jednoduchost by měl být centrální server předem ověřitelný. Podle diagramu je tímto doporučeno nepoužívat blockchain. Blockchain je vhodnější pro nedůvěryhodné strany z důvodu decentralizovanosti a vepsané dohody.

- **Šestý rozhodovací blok** se ptá, zda je požadována veřejná ověřitelnost. Odpověď je „ne". Kredit vozidel by měl zůstat v soukromí každého provozovatele vozidla – ochrana soukromí.

Výsledkem diagramu pro tuto práci je použít soukromý blockchain typu permissioned. Na výběr jsou blockchain Hyperledger, nebo Corda. Tyto projekty se také nazývají Distributed Ledger Technology – DLT. Hybridní blockchain nebyl ve výběru zahrnut z důvodu absence zveřejnění některých dat. Každý by měl mít přehled jenom o svém kreditním zůstatku.

Databáze	**Vhodná/Nevhodná**
Permissionless	Nevhodná
Public permissioned	Nevhodná
Private permissioned	Vhodná
Centrální databáze	Vhodná[5]

Tabulka 4.2: Shrnutí analýzy

4.4.1 Škálovatelnost

Škálovatelnost určuje, kolik požadavků je schopno systém zpracovat za určitou dobu. Škálovatelnost se u blockchain určuje počtem transakcí za sekundu – TPS.

[5]Povaha zadání tuto možnost neumožňuje.

Permissionless sítě se pohybují kolem 10 TPS (aktuálně v roce 2023). Permissioned sítě se pohybují kolem 100 TPS a víc. Záleží na infrastruktuře, počtu klientů a jakým způsobem se TPS měří. TPS se může měřit přímo u klienta, kde transakce vznikají, nebo až se transakce dostane ke každému klientovi v síti. U DLT není definován standard pro měření škálovatelnosti. Je doporučeno sledovat škálovatelnost systému jako celek. [26]

4.4.2 Hyperledger

Hyperledger je skupina projektů zabývající se DLT. První Hyperledger byl zveřejněn roku 2016. Hyperledger je vyvíjen primárně společnostmi Linux Foundation a IBM. Jejich kód je otevřený (angl. open source)[6]. Hyperledger je složen z několika projektů, např. Aries, Besu, Cacti, Fabric ... atd. Každý z těchto projektů řeší něco jiného. Relevantním projektem pro tuto aplikaci je Fabric. Fabric je blockchain framework určený pro všeobecné účely jako základní stavební kámen pro blockchain databáze.

Fabric se skládá z blockchain a globálního stavu tzv. world state. World state je non-SQL[7] databáze typu LevelDb, nebo CouchDb. World state uchovává informace, které samotný blockchain upravuje. Blockchain obsahuje historii transakcí, kterými mění World state. Každý připojený klient si uchovává celou kopii blockchain a world state. [1]

4.4.3 R3 Corda

R3 Corda je DLT framework. Kód je otevřený[8]. První verze vyšla roku 2017. Corda nabízí komunitní a komerční verzi. R3 si získal oblibu primárně u světových bank.

Corda funguje v prostředí JVM[9]. Databáze se nazývá trezor (angl. vault). Data jsou ukládaná do relační databáze. V komunitní verzi je možné použít databázi H2 nebo Postgres. Komerční verze má na výběr ještě z SQL[10] Server a Oracle. Je tedy možné využít funkcionalitu relační databáze ke komplexnímu dotazování (výběr, spojení, průnik, ...). Corda nemá globální stav celé databáze. Každý klient si uchovává jen tu část, kterou potřebuje. Podporující programovací jazyky Java a Kotlin. [25]

[6] https://github.com/hyperledger
[7] **non-SQL** - Not only Structured Query Language
[8] https://github.com/corda
[9] **JVM** – Java Virtual Machine
[10] **SQL** - Structured Query Language

4.4.4 Shrnutí

	Fabric	**R3 Corda**
Platforma	Linux	Multiplatformní
Virtualizace	docker	JVM
Databáze	Level-Db, Couch-Db	H2, Postgres, SQL Server, Oracle
Chytré kontrakty	Chaincode - Go, Javascript, Java	JVM, Java nebo Kotlin

Tabulka 4.3: Srovnání mezi DLT

Fabric a R3 Corda jsou DLT , které řeší stejnou věc různými způsoby. Fabric je univerzální DLT pro širokou škálu využití. R3 Corda se více zaměřuje na finanční sektor a bankovnictví.

Faktory pro rozhodnutí jsou následující. Každý klient obsahuje celou kopii blockchain. Nejsou v této práci identifikovány další strany. Jedinou stranou je provozovatel – stát. V tomto se ztrácí potenciál využití R3 Corda, kde si každý klient uchovává jen potřebnou část. Tímto je World state pro aplikaci vhodnějším řešením. Z těchto důvodů je vybrán pro tuto práci projekt **Fabric**.

5 Realizace

Aplikace byla realizována na platformě Linux. Server a OBU byly naprogramovány v programovacím jazyce Go[1] ve verzi 1.19. Jedná se o kompilovací jazyk. Framework Fabric byl použit ve verzi LTS[2] 2.5. Realizace je dostupná v Git repozitáři[3]. V terminologii databáze Fabric se chytrý kontrakt nazývá chaincode. Tato práce zůstává kvůli zjednodušení u termínu chytrý kontrakt.

5.1 Server

Server komunikuje s OBU jednotkami přes HTTP(S) protokol. Výchozí `PORT` je nastaven na 8905. Server má následující rozhraní:

- `localhost:PORT/` - Vypíše nápovědu.
- `localhost:PORT/obu` - Kontrola údajů při startu OBU.
- `localhost:PORT/geomodel[?v]` - Vrací geografický model v JSON formátu. S volitelný parametrem `v` vrací jen aktuální verzi modelu.
- `localhost:PORT/ticket` - Přijímá detekované úseky jednotkou OBU. Má za úkol spočítat vzdálenost z detekovaného úseku a následně vypočítat částku za detekovaný úsek. K výpočtu částky se používá vypočtená vzdálenost a informace o vozidle. Na závěr vrací údaje o OBU s aktualizovaným zůstatkem.

5.1.1 Geografický model

Geografický model je rozdělen do souborů formátu GPX podle mýtných úseků. Pro testování byly vytvořeny dva mýtné úseky o délce přibližně 200 metrů. Mýtný úsek je tvořen body po cca 20 metrech. To znamená, že soubory obsahují 10 tagů `wpt`. Úseky byly vytvořeny na stránkách mapy.cz tzv. od ruky z dostupných nástrojů na webu. Souřadnice jsou v jednotkách stupňů.

Název souboru začíná označením kategorie silnice I. třídy `i`, nebo dálnice `d`. Poté následuje identifikační číslo silnice. Celý název souboru může vypadat následovně, např. `i35.gpx`, `i55.gpx`, `d1.gpx`, `d2.gpx`, ...atd.

[1] https://go.dev/
[2] **LTS** - Long Term Support
[3] https://github.com/Solamil/etollfabric23

Do souborů jsou vloženy tagy `title` a `version`. Tag `title` je název mýtného úseku a `version` je verze mýtného úseku viz kód 5.1.

```
<?xml version="1.0" encoding="utf-8"?>
<gpx xmlns="http://www.topografix.com/GPX/1/1" version
   ="1.1" creator="https://mapy.cz/">
      <title>D10</title>
      <version>0.1</version>
      <wpt lat="50.612128" lon="15.113993">
            <ele>312.000000</ele>
      </wpt>
      <wpt lat="50.611962" lon="15.114114">
            <ele>311.718323</ele>
      </wpt>
      .
      .
      .
</gpx>
```

Zdrojový kód 5.1: Ukázka souboru d10.gpx

Server si při startu načte model do vlastní struktury pro rozesílání k jednotkám OBU. Při načítání se souřadnice převedou do radiánů. Výsledný výstup rozhraní `/geomodel` vypadá následovně, viz příloha 7.1.

5.1.2 Databáze sazeb

Databáze sazeb je vůči reálné situaci v poměru 1 : 10. To znamená místo poplatku za 1 kilometr je poplatek počítán za 100 metrů z důvodu kratších testovacích tras.

Sazebník je rozdělen do souborů podle typu vozovky silnice I. třídy `i`, dálnice `d` a denního období `day`, `night`. Soubory jsou ve formátu JSON. Název souborů vypadá např. `i-day.json`, `d-night.json`. Formát JSON byl vybrán z důvodu snadnější úpravy struktury v důsledku možných změn sazebníku, např. nová kategorie vozidel, nová emisní třída, sjednocení starší emisních tříd... apod., v neposlední řadě v důsledku také různých datových typů v sazebníku, např. intervaly, čísla, znaky, řetězce... atd.

Struktura je vnořená v následujícím pořadí – typ vozidla, emisní třída, hmotnost a počet náprav viz příloha 7.2. Hodnota poplatku je typu `float`. Nutné upozornit, že sazebník u kategorie vozidel M2 a M3 rozlišuje na rozdíl od kategorie N nápravy jen 2, 3 a více.

5.2 Hyperledger Fabric

Databáze Fabric byla vytvořena za pomoci git repozitáře[4]. Repozitář obsahuje návody, pomocné skripty, příklady od instalace až po vývoj aplikace. Repozitář také obsahuje testovací síť pro otestování funkcionalit. Fabric běží v `docker`[5] kontejnerech.

Fabric používá consensus algoritmus Raft. Jedná se o tzv. crash fault tolerance. I navzdory chybě některé z komponent by síť měla dojít k dohodě. Používá se jako výchozí algoritmus. Algoritmus BFT se plánuje plně podporovat ve verzi 3.0.

5.2.1 Princip

Hyperledger Fabric se skládá z několika komponent. Mezi základním stavebním kamenem je kanál (angl. channel), který umožňuje propojit organizace. Síť může provozovat více kanálů než jeden. Každý kanál má vlastní blockchain databázi – World state. World state je přístupný jen pro organizace uvnitř kanálu. Organizace může mít přístup do více kanálů. Organizace má svůj X.509 certifikát, kterým prokazuje svoji totožnost v síti.

Samotný kanál je tvořen z klientů, kteří se nazývají peer a orderer service. Peer je klient, který má kopii World state a dokáže spouštět uvnitř kanálu chytrý kontrakt (chaincode). Kanál může obsahovat více než jeden chytrý kontrakt. Orderer service je klient pro správu transakcí a patří vždy k některé organizaci. Stará se o tvorbu, potvrzení a rozesílání transakcí k peer klientům. [2]

5.2.2 LevelDb, nebo CouchDb

Je nutné rozhodnout, jaký typ databáze použít pro ukládání dat. Fabric nabízí uložení World state do databáze LevelDb, nebo CouchDb. Ve výchozím nastavení se používá databáze LevelDb. Obě databáze jsou non-SQL a jsou postaveny na architektuře klíč-hodnota. Hlavním rozdílem je podpora indexů v databázi CouchDb. Indexy se používají pro rychlejší vyhledávání v databázi. Naopak indexy mohou zpomalovat zápis do databáze z důvodu setřídění nově přidaných dat do indexů. [14]

Aplikace se dotazuje vždy na konkrétní OBU v databázi identifikátory `ID`, `SPZ` a `Státní příslušnost`. Co řádek, to záznam o jedinečné OBU. Z tohoto důvodu nepřijde výhodné použít jakýkoliv index, protože by nepřinesl žádnou další selektivnost záznamů. Další důvod je jednoduchost databázového modelu. Pro účely této aplikace bude postačovat databáze typu LevelDb. [34]

5.2.3 Testovací síť

Aplikace používá testovací síť z dostupného repozitáře. Konkrétněji používá síť z adresáře `fabric-samples/test-network`. Adresář obsahuje pomocné skripty pro automatizování spuštění sítě, nasazení chytrého kontraktu... atp. Síť se konfiguruje

[4] https://github.com/hyperledger/fabric-samples
[5] https://www.docker.com/

v souboru `fabric-samples/config/configtx.yaml`. Soubor je dostatečně popsán komentáři. Soubor definuje organizace, klienty a kanály. Jednotlivé parametry mohou zůstat beze změny. Při nasazení ostré verze je pak nutné změnit cestu ke skutečným certifikátům. Pro testovací účely si síť vytváří vlastní certifikáty určené především pro lokální rozhraní.

Ještě před spuštěním sítě je nutné nastavit jmenný prostor proměnných souborem `setOrgEnv.sh`. Proměnné slouží k nalezení programů a certifikátů pro klienty z organizace. Samotná síť se spouští skriptem `network.sh`. K němu náleží konfigurační soubor `network.config`, nebo je možné síť nakonfigurovat skrz parametry v příkazové řádce. Nastavuje se hlavně název kanálu, název chytrého kontraktu, cesta k chytrému kontraktu a typ databáze (LevelDb nebo CouchDb). Ke spuštění sítě byl použit příkaz s parametry viz kód 5.2.

```
1 ./network.sh up createChannel -ca -c channel1
```

Zdrojový kód 5.2: Spuštění testovací sítě Fabric

Příkaz spustí síť, vytvoří kanál s názvem `channel1` a vygeneruje certifikáty pro organizace parametr `-ca`.

5.2.4 Chytrý kontrakt

Chytrý kontrakt je možné psát v programovacím jazyce Javascript, Typescript, Java, nebo Go. Jako programovací jazyk pro psaní chytrého kontraktu byl vybrán jazyk Go. Důvodem je, že ostatní komponenty aplikace jsou také psány v jazyce Go.

Psaní chytrého kontraktu se neliší od psaní jiných aplikací. Kontrakt se ale liší od jiných aplikací při jeho nasazování. Nasazení je nutné provést na jednotlivých peer klientech zvlášť. Většina organizací se musí dohodnout na definici kontraktu s předem nastavenou politikou sítě. Proces je následující.

- Zabalit chytrý kontrakt do balíčku s koncovkou `tar.gz`.
- Nahrát balíček na peer klient z každé organizace.
- Organizace musejí odsouhlasit definici kontraktu dle nastavené politiky sítě.
- Pokud většina organizací souhlasí, je možné nasadit kontrakt do kanálu.

Tento proces je automatizovaný skriptem `test-network/scripts/deployCC.sh`. Kontrakt byl nasazován příkazem viz kód 5.3.

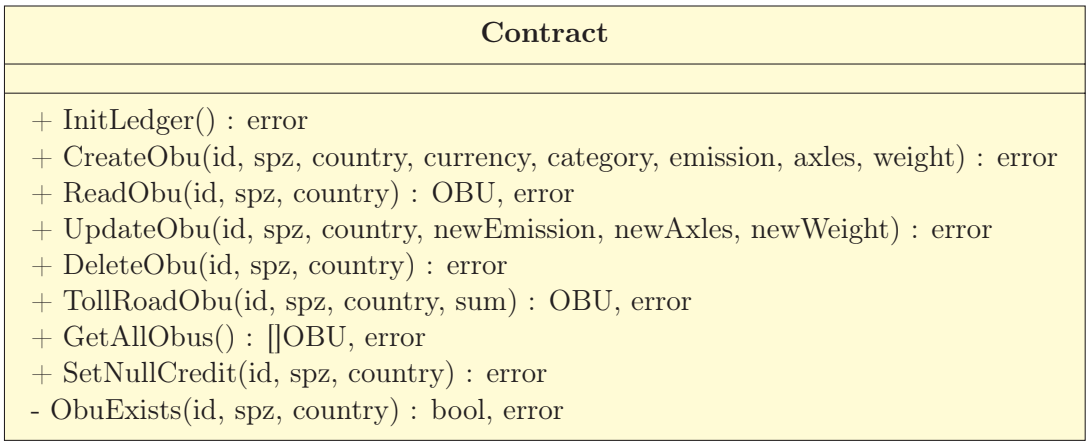
```
1 ./scripts/deployCC.sh channel1 toll ../asset-toll/
    chaincode-go/ go 1 1 "InitLedger"
```

Zdrojový kód 5.3: Nasazení chytrého kontraktu

Příkaz spustí nasazení kontraktu `toll` z adresáře `../asset-toll/chaincode-go/` v jazyce `go` ve verzi `1` a v sekvence `1`. Nakonec je spuštěna inicializační funkce kontraktu názvem `InitLedger`, to vše na kanálu s názvem `channel1`. Verze slouží k popsání chytrého kontraktu. Sekvence určuje, kolikrát byl chytrý kontrakt v kanálu definován. Pro nahrání aktualizovaného chytrého kontraktu je zapotřebí inkrementovat o jedničku parametry verze a sekvence.

5.2.5 Mýtný kontrakt

Mýtný kontrakt má za cíl evidovat jednotky OBU a částku pro zaplacení mýta. Kontrakt je naprogramován na principu post-paid. Zůstatek se při projetí mýtným úsekem přičítá a po zaplacení faktury se zůstatek anuluje. Kontrakt umožňuje CRUD[6] operace nad OBU záznamy. Kontrakt definuje strukturu OBU dle diagramu 4.5. Každý záznam OBU je identifikován složeným klíčem ID, SPZ a státní příslušností. Kontrakt má rozhraní viz diagram 5.1.



Obrázek 5.1: Mýtný kontrakt

Mýtný úsek je evidován funkcí `TollRoadObu`. Zůstatek se anuluje funkcí `SetNullCredit`.

[6] **CRUD** - Operace Create, Read, Update a Delete.

5.2.6 Organizace

Organizace je v této aplikaci jenom jedna – stát. Stát nebo také ŘSD vlastní silnice a vybírá za využívání silnic I. třídy a dálnic poplatek. Organizace má před spuštěním sítě předem nastavená pravidla a oprávnění. Organizace může schvalovat, vkládat a spouštět kontrakty. Organizace má v síti klienty peer a orderer. Testovací síť ve výchozím nastavení automaticky vytváří dvě organizace. Obě se podílejí na schvalování, přidávání a spouštění kontraktů. Pro aplikaci toto zůstalo ve výchozím nastavení.

5.2.7 Brána

Brána (angl. Gateway) se používá pro propojení jednoho programu s druhým skrze rozhraní. Centrální server si stáhne knihovny

- `github.com/hyperledger/fabric-sdk-go/pkg/core/config` a
- `github.com/hyperledger/fabric-sdk-go/pkg/gateway`.

Knihovny umožní připojit se k síti databáze Fabric.

Každá organizace v síti je poskytovatelem členství tzv. MSP (Membership Service Provider). Organizace poskytne uživatelům identitu, skrze níž se v síti prokážou. Konkrétněji centrální server se prokáže identitou v adresáři.

- `test-network/organizations/peerOrganizations/org1.example.com/users/User1@org1.example.com/msp/`.

Centrální server se představuje jako uživatel s názvem `User1` z organizace `org1`.

5.3 OBU

OBU má za cíl simulovat jízdu po testovací trase. OBU má za úkol provést následující kroky:

- Načtení OBU z JSON souboru. Pokud neexistuje, program OBU se ukončí.
- Kontrola údajů OBU se serverem přes rozhraní `/obu`. Rozhraní umožňuje upravit atributy jako počet náprav, hmotnost a třídu emisí. Server odešle OBU potvrzené údaje. Pokud server neidentifikuje jednotku OBU, program OBU se ukončí.
- Načtení testovací trasy z GPX souboru. Pokud soubor s testovací trasou neexistuje, program OBU se ukončí.
- Načtení lokální verze geografického modelu a jeho kontrola s verzí na serveru. Kontroluje se verze a kontrolní součet modelu.
- Simulace jízdy 5.3.2.

- Po ukončení jízdy se odešlou detekované úseky spolu s údaji o OBU serveru na rozhraní `/ticket`. Rozhraní vrátí potvrzené údaje OBU s aktuálním zůstatkem.

Program OBU přijímá parametr `--name NAZEV_OBU`. Pokud název OBU je `obu1`, program OBU si načte soubory typu `obu1.json` a `obu1.gpx`. Soubor typu JSON obsahuje údaje o OBU. Soubor typu GPX obsahuje testovací trasu. Geografický model ze serveru se uloží do adresáře `cache/`. Geografický model je uložen v souboru typu JSON.

5.3.1 Zabezpečení

Ve skutečnosti je OBU Billien identifikován pomocí RSA[7] klíče. RSA klíč slouží k autentizaci a podepisování komunikace se serverem. Komunikace je šifrována metodou 3DES a AES. Na výslednou šifru je aplikována hash funkce SHA. V OBU je vyhrazené chráněné místo pro uložení klíčů. [37]

V rámci této práce je jednotka OBU identifikována pomocí UUID ve verzi 4. Pomocí UUID server pozná, s jakým OBU komunikuje, a zamezí se vydávání se za některé z OBU. Identifikátor UUID ve verzi 4 nabízí 2^{128} počtu kombinací. Uhodnutí některého použitého identifikátoru je výpočetně náročné. Pro testování v lokální, nebo interní síti je toto dostačující řešení. Pro testování ostré verze ve veřejné síti je nutné implementovat komplexní zabezpečení, tj. při nejmenším RSA klíč a AES. Z důvodu rozsáhlosti a časového prostoru toto zde není implementováno. [31]

5.3.2 Algoritmus pro nalezení shody

Algoritmus počítá vzdálenost mezi aktuálním bodem testovací trasy a modelem. Vzdálenost se počítá v jednotkách metrů. K tomu je nutné, aby Haversinův vzorec měl k dispozici poloměr Země v metrech. Algoritmus počítá vzdálenost s poloměrem Země 6 371 000 m. Je to aritmetický průměr poloměru Země z důvodu, že Země je spíše elipsoid než koule. Na rovníku je poloměr až 6 378 km a u polárního kruhu 6 357 km. Algoritmus si vystačí s poloměrem 6 371 km, z důvodu lokace modelu - střední Evropa. [16]

Práh (angl. Threshold) pro detekci úseku byl zvolen na úrovni 20 metrů včetně. Z důvodu, že v reálné situaci se s dobrým signálem přesnost pohybuje kolem 2–5 m, tolerance je přibližně až 20 m. V rozhodování byla zahrnuta také samotná šířka silnice zmíněná v sekci 4.2.3. [3]

Testovací trasa byla vytvořena na stránkách mapy.cz tzv. od ruky z dostupných nástrojů na webu viz příloha 7.1. Body trasy leží 3–15 m od sebe.

Při shodě se uloží indexy konkrétního bodu v poli a časový záznam ve formátu RFC3339[8]. Výstupem algoritmu je pole indexů, následně ke každému indexu v poli náleží časový záznam. Záznam může vypadat následovně viz kód 5.4.

[7] **RSA** - Rivest–Shamir–Adleman kryptografický systém založený na principu veřejný a soukromý klíč.

[8] **RFC3339** - Standard pro zápis data, času a časové zóny např. "2022-01-02T15:04:05Z02:00".

```
{
       [0 0 0...]
       [0 1 2...]
       [2023-08-22T14:29:47+02:00
        2023-08-22T14:29:47+02:00
        2023-08-22T14:29:47+02:00...]
}
```

Zdrojový kód 5.4: Ukázka výstupu algoritmu pro nalezení shody.

Indexy prvního pole značí typ úseku, např. i55, d1, d2... atd. Druhé pole je index bodu konkrétního úseku. Po ukončení algoritmu se výstup algoritmu a informace o OBU spojí a odešlou se serveru k zaplacení přes rozhraní `/ticket`.

5.4 Výsledek

Výsledkem se stala aplikace, která zpoplatňuje vozidla nad 3,5 tuny za ujetou trasu. Na vstupu jsou dva soubory. První soubor je ve formátu GPX a obsahuje testovací trasu. Druhý soubor je ve formátu JSON a obsahuje informace o jednotce OBU. Program OBU simuluje jízdu po testovací trase. OBU detekuje mýtné úseky podle geografického modelu. Model se stahuje z centrálního serveru. Po skončení jízdy se detekované mýtné úseky a informace o OBU odešlou do centrálního serveru.

Centrální server spočítá ujetou vzdálenost z detekovaných mýtných úseků. Podle údajů OBU se najde sazba v databázi. Sazba se platí za 100 ujetých metrů v korunách českých. Částka se vypočte násobením sazby a vzdálenosti v metrech. Částka za mýto se uloží do databáze Fabric ke konkrétnímu OBU záznamu. Operace nad databází je prováděna chytrým kontraktem.

	SEM v ČR	**Aplikace**
Typ klienta	Chytrý, tenký	**Chytrý**
Detekce úseku	?*[9]	**Haversine vzorec**
Geografický model	?*	**GPX formát**
Vzdálenost detekce	3-5 m, mez 20 m	**do 20 m včetně**
Účtování	na fakturu, předem	**na fakturu**
Platba za	1 km/Kč	**100 m/Kč**
Platba mýta	?*	**Po ukončení jízdy**
Bezpečnost	RSA, 3DES, AES, SHA	**UUID, Certifikát**
Databáze	?*	**DLT, Fabric, non-SQL (LevelDb)**
Ostatní technologie	?*	**Go, JSON**

Tabulka 5.1: Srovnání aplikace

[9]*Closed source - Podléhá tajemství

6 Závěr

V této práci byla vyvinuta aplikace pro výběr mýta za použití technologie blockchain a chytrého kontraktu. Nejdříve byla vysvětlena technologie blockchain a chytrý kontrakt. Blockchain je neměnitelná decentralizovaná databáze se stále rostoucím počtem záznamů. Chytrý kontrakt je spustitelný kód uložený v blockchain. Blockchain se dělí na dva typy databází, tj. permissionless a permissioned. Permissionless je například bitcoin a ethereum. Permissioned je například R3 Corda a Hyperledger. Pro typ permissioned se používá také termín DLT – Distributed Ledger Technology.

Byla vytvořena rešerše existujících mýtných systémů. Většina jich funguje jen pro vozidla nad 3,5 tuny. Mezi primitivní systémy patří výběrčí kabiny s lidskou obsluhou. Mýtný systém se postupně modernizuje mikrovlnnou komunikací a fyzickými mýtnými bránami, které nevyžadují změnu jízdy. Mezi komplexní systémy se řadí systém postavený na satelitní technologii a mobilní síti. Tento systém už nevyžaduje stavbu fyzických mýtných brán na nových komunikacích. Systému stačí jen přesné GPS souřadnice mýtných úseků. Dne 1. prosince 2019 došlo právě k přechodu na satelitní technologii v Česku.

Návrh vznikal díky konzultaci s firmou TollNet a.s.[1], která je dodavatelem systému mýta v Česku a na Slovensku. Konkrétněji komunikace probíhala s p. Fišerou z technického oddělení. Byly především vysvětleny základní principy fungování mýta. Veškeré podrobnější informace podléhají tajemství (Closed-source). V návrhu byla aplikace rozdělena na centrální server a palubní jednotku (OBU) ve vozidle. Centrální server vede evidenci OBU jednotek a jejich zůstatek v blockchain databázi. OBU odesílá centrálnímu serveru údaje o detekovaných mýtných úsecích. Komunikace probíhá přes protokol HTTP(S). Větší část v návrhu byla věnována výběru blockchain databáze. Při výběru byl použit vývojový diagram viz příloha 7.2. Diagram označil za nevhodné použít blockchain typu permissionless. Vedl k použití permissioned, nebo centrální databáze, i když povaha zadání druhou možnost neumožňovala. Proto byla v práci použita soukromá databáze typu permissioned.

V realizaci se použila DLT technologie projekt Hyperledger Fabric oproti R3 Corda z důvodu využití globálního stavu blockchain databáze u Fabric. OBU projede po předem definované GPX trase. Při jízdě detekuje mýtné úseky definované geografickým modelem. Algoritmus pro detekci mýtných úseků funguje na principu výpočtu vzdálenosti mezi dvěma GPS body. K jejímu výpočtu byl použit Haversinův vzorec. Hraniční vzdálenost je nastavena na 20 metrů včetně. Detekované mýtné úseky se po ukončení jízdy odešlou centrálnímu serveru. Centrální server spočítá

[1]Kontakt office@tollnet.cz

ujetou vzdálenost na mýtných úsecích. Částka za mýtné úseky se spočítá z databáze sazeb, která definuje jednotlivé sazby pro konkrétní typ vozidla. Částka se zaeviduje do databáze Fabric. Po úspěšné transakci se odešle OBU aktuální zůstatek.

Pomocí databáze Fabric aplikace dokáže urychlit dohodu mezi více organizacemi najednou díky předem nastavené politice v síti. Autor také vidí nespornou výhodu u DLT využití konvenčních programovacích jazyků oproti typu permissionless. Podle autora to může mít rozhodující fakt v praxi. Aplikace je dostupná v Github repozitáři[2].

[2] https://github.com/Solamil/etollfabric23

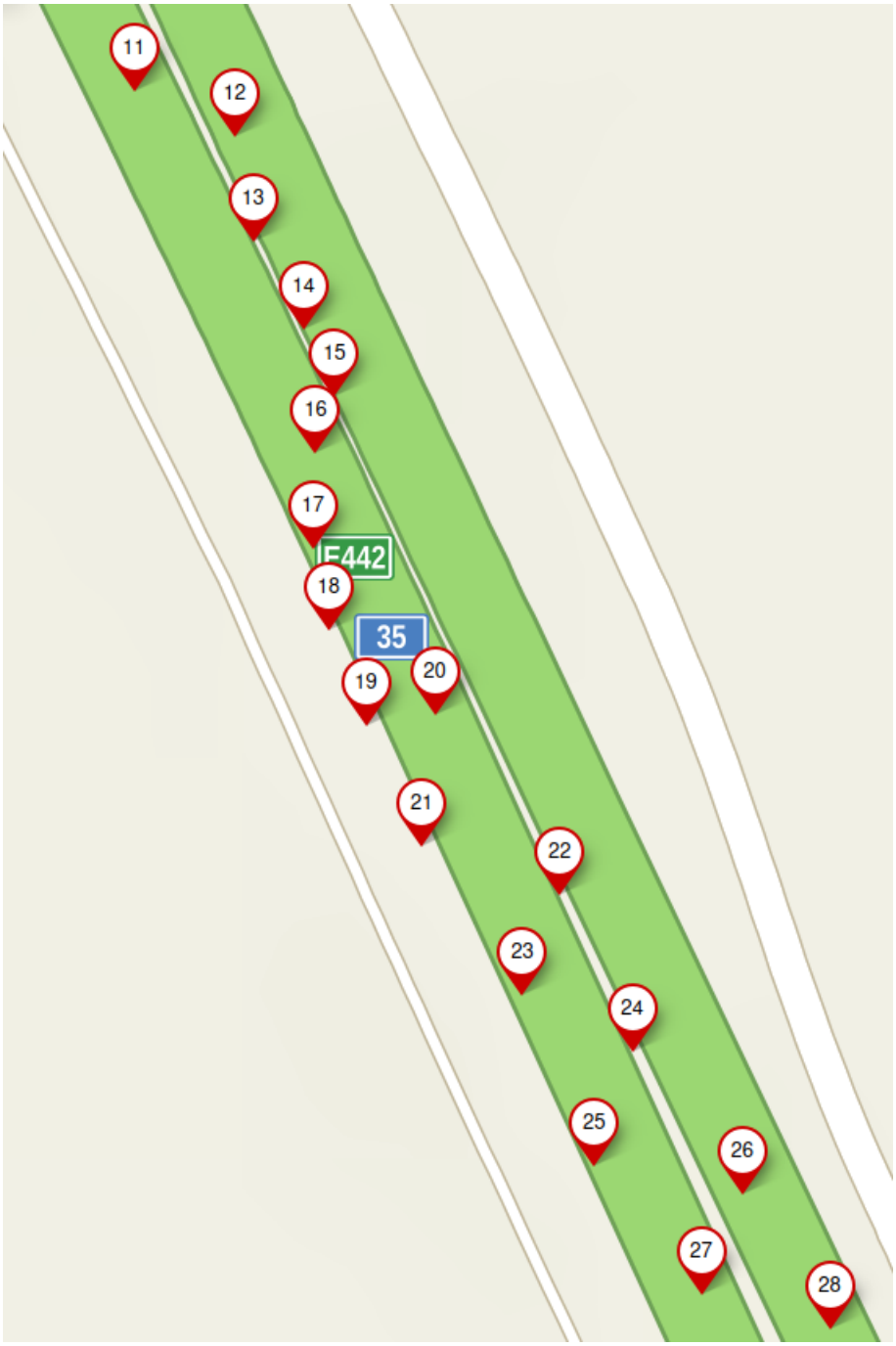
7 Příloha

```
[{
      "latRad":
            [0.8833989070424129,0.8833961319689024,...],
      "lonRad":
            [0.2637513989411649,0.2637535456961449,...],
      "distances":[0,0,...],
      "version":"0.1",
      "len":10,
      "name":"I35"
},
{
      "latRad":
            [0.8833482750408126,0.8833453777942543,...],
      "lonRad":
            [0.26378894097337535,0.2637910528217703,...],
      "distances":[0,0,...],
      "version":"0.1",
      "len":10,
      "name":"D10"
}]
```

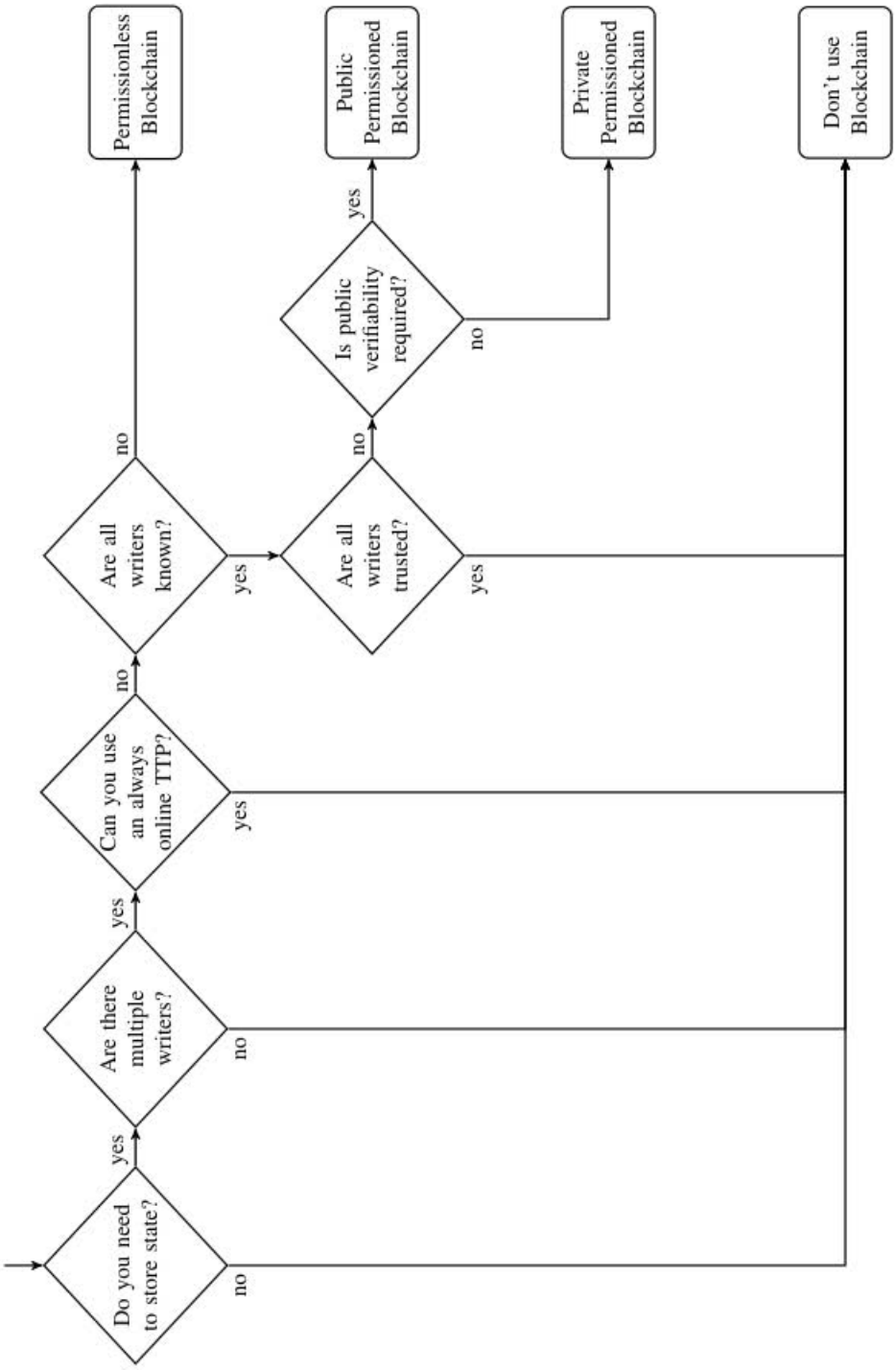
Zdrojový kód 7.1: Ukázka výstupu rozhraní `/geomodel`.

```
{
	"N": {
		"0-4": {
			"35-75": {
				"2": 0.056,
				"3": 0.076,
				"4": 0.096,
				"5": 0.116
			},
			"75-12": {
				"2": 1.163,
				"3": 1.563,
				"4": 1.983,
				"5": 2.408
			},
			"12": {
				"2": 3.045,
				"3": 4.091,
				"4": 5.191,
				"5": 6.295
			}
		},
		"5-EEV": {
			.
			.
			.
		},
		"6": {
			.
			.
			.
		},
		"CNG": {
			.
			.
			.
		}
	},
	"M2": {
		.
		.
		.
	}
}
```

Zdrojový kód 7.2: Ukázka sazebníku.



Obrázek 7.1: Ukázka části testovací trasy.[29]



Obrázek 7.2: Vývojový diagram pro rozhodování jaký typ databáze.[40]

Použitá literatura

[1] Hyperledger 2020-2023. *Documentation Hyperledger Fabric*. [Online]. 2023. URL: https://hyperledger-fabric.readthedocs.io/en/latest/index.html.

[2] Hyperledger 2020-2023. *Fabric - Key Concepts*. [Online]. 2023. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.5/key_concepts.html.

[3] TollNet a.s. *Komunikace s technickým oddělením email: office@tollnet.cz*. 2023. URL: tollnet.cz.

[4] Maher Alharby and Aad van Moorsel. "Blockchain Based Smart Contracts : A Systematic Mapping Study". In: Aug. 2017, pp. 125–140. DOI: 10.5121/csit.2017.71011.

[5] All4car.sk. "Porovnání LPG, CNG, LNG a BIOplynu". In: (2023). URL: https://www.levnevozeni.cz/porovnani-lpg-cng-lng-a-bioplynu/.

[6] Adam Back. "Hashcash - A Denial of Service Counter-Measure". In: (Sept. 2002), pp. 2–.

[7] Adam Back. "Hashcash - A Denial of Service Counter-Measure". In: (Sept. 2002).

[8] Beneš Petr RNDr. *Elektronické mýtné v Evropě*. 2005. URL: https://web.archive.org/web/20070310211832/http://www.stech.cz/articles_print.asp?idk=97&ida=528.

[9] Micali Silvio Blum Manuel Feldman Paul. *Non-Interactive Zero-Knowledge and Its Applications*. 1988. URL: https://apps.dtic.mil/sti/pdfs/ADA222698.pdf.

[10] Chainlink Foundation. *Chainlink oracles*. 2023. URL: https://chain.link/.

[11] Dan Foster. *GPX: the GPS Exchange Format*. [Online]. 2023. URL: https://www.topografix.com/gpx.asp.

[12] Ethereum contributors. *Ethereum blockchain*. 2023. URL: https://ethereum.org/.

[13] Evropský parlament. *Directive 2004/52/EC of the European Parliament and of the Council of 29 April 2004 on the interoperability of electronic road toll systems in the Community*. [Online]. 2004. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32004L0052.

[14] Jimmy Farillo. *The Basics of Database Indexes For Relational Databases*. 2017. URL: https://medium.com/@jimmyfarillo/the-basics-of-database-indexes-for-relational-databases-bfc634d6bb37.

[15] Matthias Fitzi. "Generalized Communication and Security Models in Byzantine Agreement". ETH ZURICH(SWISS FEDERAL INSTITUTE OF TECHNOLOGY), 2002, pp. 37–58.

[16] Frédéric Chambat, Bernard Valette. "Mean radius, mass, and inertia for reference Earth models". In: vol. Physics of the Earth and Planetary Interiors, Volume 124, Issue 3-4, p. 237-253. 2001. DOI: 10.1016/S0031-9201(01)00200-X. URL: https://web.archive.org/web/20200730215818/http://frederic.chambat.free.fr/geophy/inertie_pepi01/article.pdf.

[17] Stuart Haber and W. Scott Stornetta. "How To Time-Stamp a Digital Document". In: *Journal of Cryptology* (1991). URL: https://link.springer.com/article/10.1007/BF00196791.

[18] Státní fond dopravní infrastruktury, ed. *Elektronická dálniční známka*. 2021. URL: https://edalnice.cz/.

[19] Shijie Lin. "Proof of Work vs. Proof of Stake in Cryptocurrency". In: *Highlights in Science, Engineering and Technology* 39 (Apr. 2023), pp. 953–961. DOI: 10.54097/hset.v39i.6683.

[20] Foreverhold Ltd. 2017. URL: https://everledger.io/.

[21] Ralph Merkle. "Method of providing digital signatures". [US4309569A]. 1982.

[22] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Tech. rep. 2008. URL: www.bitcoin.org.

[23] EETS Info Platform. *European Electronic Toll Service*. 2023. URL: https://www.eetsinfoplatform.eu/eets/.

[24] R. I. 'Scibor-Marchochi. *Elementary-Geometry Trigonometry*. [Online]. 1997. URL: https://web.archive.org/web/19991010004728/http://www.geocities.com/ResearchTriangle/2363/trig02.html.

[25] R3. *Corda Key Concepts*. [Online]. 2023. URL: https://docs.r3.com/en/platform/corda/4.10/enterprise/about-corda/corda-key-concepts.html.

[26] R3. *Transactions per second (TPS)*. [Online]. 2018. URL: https://corda.net/blog/transactions-per-second-tps/.

[27] Chethan S. *Measuring accuracy of latitude and longitude*. [Online]. 2011. URL: https://gis.stackexchange.com/questions/8650/measuring-accuracy-of-latitude-and-longitude.

[28] Rüdiger Schollmeier. "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications". In: Proceedings of the First International Conference on Peer-to-Peer Computing, IEEE, 2002.

[29] Seznam a.s. *Mapy.cz*. [Online]. 2023. URL: https://mapy.cz/.

[30] Alan T. Sherman et al. "On the Origins and Variations of Blockchain Technologies". In: *CoRR* abs/1810.06130 (2018). arXiv: 1810.06130. URL: http://arxiv.org/abs/1810.06130.

[31] Soham Kamani. *A Complete Guide to UUID Versions (v1, v4, v5) - With Examples*. [Online]. 2021. URL: https://www.sohamkamani.com/uuid-versions-explained/.

[32] W. Stallings. *Cryptography and Network Security: Principles and Practice*. The William Stallings books on computer and data communications technology. Prentice Hall, 1999. ISBN: 9780138690175. URL: https://books.google.cz/books?id=Dam9zrViJjEC.

[33] Stát, Ředitelství silnic a dálnic ČR. *SEM - Sazby mýta*. [Online; cit. 12.09.2023]. 2023. URL: https://mytocz.eu/cs/emytne/sazby-mytneho-2021.

[34] Inc. Portions Sybase. *Composite indexes*. 2003. URL: http://dev.cs.ovgu.de/db/sybase9/help/dbugen9/00000433.htm.

[35] Nick Szabo. "Smart Contracts : Building Blocks for Digital Markets". In: 1996.

[36] Tollnet a.s. *Billien OBU 5050, Hybrid On-Board Unit for Electronic Toll Collection and More Services*. [Online; cit. 04-September-2023]. 2023. URL: https://www.tollnet.cz/wp-content/uploads/2020/09/Prospekt-Billien-OBU-5050_EN_01_05.pdf.

[37] Tollnet a.s. *Billien OBU 5450, Hybrid On-Board Unit for Electronic Toll Collection and More Services*. [Online; cit. 06-September-2023]. 2023. URL: https://www.tollnet.cz/wp-content/uploads/2022/05/Prospekt-Billien-OBU-5450_EN_01_01.pdf.

[38] Centrum dopravního výzkumu. *Kategorie pozemních komunikací dle ČSN*. [Online]. 2007. URL: https://www.czrso.cz/clanek/kategorie-pozemnich-komunikaci-dle-csn/?id=1205.

[39] Gavin Wood et al. "Ethereum: A secure decentralised generalised transaction ledger". In: *Ethereum project yellow paper* 151.2014 (2014), pp. 1–32.

[40] Karl Wüst and Arthur Gervais. "Do you need a blockchain". In: (2017). URL: https://eprint.iacr.org/2017/375.pdf.