Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

Využití blockchainu ve výrobních a logistických procesech

Plzeň 2020 Bc. Martin Matas

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 10. srpna 2020

Bc. Martin Matas

Abstract

Blockchain technology is a very widespread concept today, mainly due to its potential use across industries. One such sector is logistics, especially the supply chain. Within this work, the concept of a solution for the exchange of product data between suppliers and customers in the automotive supply chain is designed and implemented. The implemented solution increases the transparency of shared data and has the potential to increase trust between organizations in the supply chain.

Abstrakt

Technologie blockchain je v dnešní době hodně skloňovaný pojem, především kvůli jeho možnostem využití napříč odvětvími. Jedním takovým odvětvím je i logistika, zejména pak dodavatelský řetězec. V rámci této práce je navržen a implementován koncept řešení pro výměnu dat o produktu mezi dodavateli a zákazníky v automobilovém dodavatelském řetězci. Implementované řešení zvyšuje transparentnost sdílených dat a má potenciál zvýšit důvěru mezi organizacemi v dodavatelském řetězci.

Poděkování

Na tomto místě bych rád poděkoval vedoucímu diplomové práce Ing. Jiřímu Dobrému a konzultantovi Ing. Miloslavu Konopíkovi, Ph.D. za podnětné rady, odbornou a všestrannou pomoc, množství cenných a inspirativních rad, podnětů, doporučení, připomínek a zároveň za velkou trpělivost a ochotu při konzultacích poskytnutých ke zpracování této práce.

Dále bych chtěl touto cestou poděkovat svému nejbližšímu okolí a rodině za podporu a poskytnuté zázemí během mého studia na ZČU v Plzni.

Obsah

1 Úvod

Když v roce 2008 představil Satoshi Nakamoto světu síť Bitcoin, většina lidí do té doby nikdy neslyšela o blockchainu - technologii, která se od té doby stala globálním buzzwordem. Koncept však nebyl první svého druhu, vyvinul se jako forma technologie distribuované účetní knihy (Distributed Ledger Technology, DLT), která má bohatou minulost. Ale teprve až po úspěchu, který zaznamenal Bitcoin, se začalo mnoho lidí zajímat, jak tento systém decentralizované měny vlastně funguje. Blockchain brzy začal přitahovat pozornost expertů napříč mnoha odvětvími a zjistilo se, že tuto revoluční technologii lze využít i v jiných odvětvích než je pouze finanční sektor.

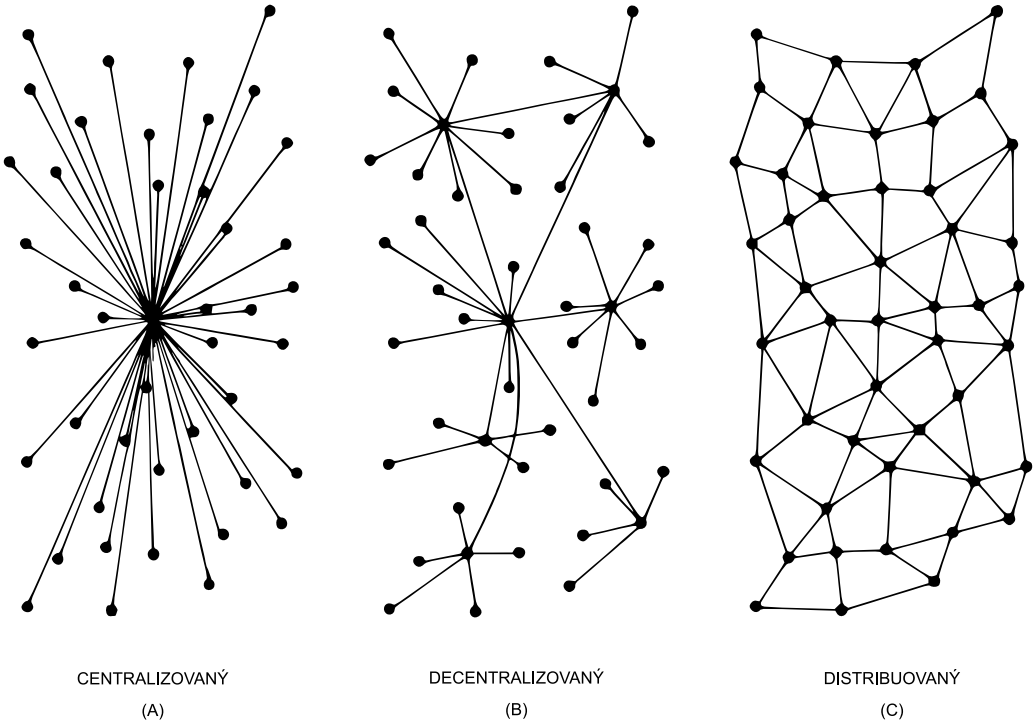
Jedním takovým odvětvím je i logistika, zejména pak dodavatelský řetězec, jelikož problémy s ním spojené jsou podobné těm ve finančním a kryptografickém průmyslu. Například problém kontroly původu zboží a součástí těchto produktů, padělání produktů, atd.

Tato práce se zaměřuje na možnosti implementace dodavatelských řetězců s využitím technologie Blockchain v odvětví logistiky, především pak v oblasti automobilového průmyslu. Toto téma vzniklo ve spolupráci s technologickou firmou AIMTEC a. s., která má v oblasti výrobních a logistických procesů dlouholeté zkušenosti.

Cílem práce je zanalyzovat možnosti využití technologie Blockchain pro řízení dodavatelského řetězce ve výrobních a logistických procesech v automobilovém průmyslu a pro zvolený problém navrhnout a implementovat řešení. Následně proběhne zhodnocení navrhovaného řešení.

2 Architektura systémů

Tato část poskytuje přehled o centralizovaných, decentralizovaných a distribuovaných systémech, zároveň je klíčová k pochopení problematiky distribuovaných výpočtů, na kterých je technologie Blockchain založena.



Obrázek 2.1: Struktura centralizovaných, decentralizovaných a distribuovaných systémů.

V následujících podkapitolách budou uvedeny hlavní rozdíly ve struktuře jednotlivých přístupů, jejich výhody a nevýhody.

2.1 Centralizované systémy

Centralizované systémy jsou systémy, které používají architekturu klient-server, kde jeden nebo více klientských uzlů je přímo připojeno k centrálnímu serveru, viz obr. 2.1(A). Jedná se o nejčastěji používaný typ systému, kde centrální server zpracovává všechny dotazy přicházející z různých uzlů a přiřazuje úkoly různým uzlům v síti. [39]

Centralizovaný systém má několik výhod. Například úplná odpovědnost sítě je plně pod kontrolou centrálního serveru, což usnadňuje správu a řízení celého systému. Kromě toho, použití centrálního serveru šetří náklady na hardwarové vybavení. Jinými slovy, pro vybudování centralizovaného systému je zapotřebí pouze centrální počítač s požadovaným hardwarem a softwarem, zatímco jiné uzly mohou být pouze terminály pro připojení uživatelů k centrálnímu serveru. Pokud je terminál nefunkční, může uživatel jednoduše použít jiný terminál pro přístup k datům uloženým na centrálním serveru. Centralizovaný systém navíc poskytuje lepší fyzickou bezpečnost, protože všechna data jsou uložena na centrálním místě, což usnadňuje ochranu před fyzickým poškozením, snižuje redundanci a zajišťuje integritu dat. [39]

Na druhou stranu, centralizovaný systém má několik nedostatků. Příkladem je centrální server - provádí všechny operace zpracování a řídí všechny uzly, které jsou k němu připojeny. V případě havárie serveru nebude celý systém k dispozici. Tento problém je často označován jako „single point of failure“, takže pokud selže tento síťový uzel (centrální server), celý systém přestane fungovat. Dalším problémem může být nedostatečná výpočetní kapacita centrálního serveru, což může mít za následek, že server nestíhá obsloužit všechny uzly v síti, následkem toho dochází k prodlévám při komunikaci se serverem, v krajních případech i k odmítání požadavků ze strany serveru. Centrální systém není také možné využít ve velkých organizacích, které mají mnoho poboček rozmístěných po celém světě. [39]

2.2 Decentralizované systémy

Alternativou k centralizovaným systémům jsou systémy bez potřeby a existence centrálního prvku, tedy decentralizované, viz obr. 2.1(B). Decentralizovaný systém je založen na peer-to-peer[1] komunikaci mezi různými uzly v síti bez nutnosti centralizovaného serveru pro řízení operací a rozhodování jménem ostatních uzlů. Každý uzel tedy dělá svá vlastní nezávislá rozhodnutí. Konečné chování systému je souhrn rozhodnutí jednotlivých uzlů. [39]

Existují dvě základní struktury pro decentralizovaný systém - čistě decentralizovaná peer-to-peer struktura a master-slave struktura. V čistě decentralizované struktuře jsou všechny uzly odpovědné za přijímání vlastních rozhodnutí, žádný uzel tedy není nadřazený nad ostatními uzly. Zatímco master-slave struktura má řídící prvek tzv. master pro malou část sítě, který

[1] P2P nebo také klient-klient je označení síťové architektury, ve které spolu jednotlivé uzly (uživatelé) komunikují napřímo bez nutnosti zprostředkovatele komunikace.

pomáhá při koordinaci ostatních uzlů (slaves) v dané podsíti. [39]

Oproti centralizovaným systémům toto řešení omezuje možnost řízení a provozu celého systému pouze jedním centrálním serverem, místo toho umožňuje každému uzlu účastnit se rozhodovacích operací. Kromě toho, decentralizovaný systém je podstatně lépe škálovatelný než centralizovaný systém. Každý uzel může být rozšířen o prostředky (hardware, software) pro zvýšení výkonu, což vede ke zvýšení výkonu celého systému. Oproti centralizovanému systému je navíc odolný vůči selhání jednoho či více uzlů. Selhání uzlů v decentralizovaném systému může ovlivnit výkon a omezit přístup k některým datům, ale z hlediska dostupnosti systému přináší velké zlepšení oproti centralizovanému systému. [39, 60]

Naopak, oproti centralizovanému řešení může být obtížné dosáhnout globálního cíle, např. většinové shody. Tento problém je možné řešit mnoha způsoby. Tyto způsoby jsou všeobecně známy jako konsenzuální algoritmy, kterým je v této práci věnována kapitola 4.3.

2.3 Distribuované systémy

Distribuovaný systém je množina vzájemně propojených autonomních uzlů tvořících koherentní síť, viz obr. 2.1(C). Uzly mohou jednat nezávisle na sobě, ale pokud se navzájem ignorují, pak je zbytečné vkládat je do stejného distribuovaného systému. V praxi jsou uzly nastaveny tak, aby dosahovaly společných cílů, které jsou realizovány vzájemnou výměnou zpráv. Kombinace úložných a procesních schopností umožňuje distribuovanému systému provádět komplexní úkoly rychleji než je tomu u jiných systémů tím, že rozdělí úkoly do více dílčích úkolů, které deleguje mezi síťové uzly. Jednotlivé uzly následně zpracovávají obdržené dílčí úkoly a vrací výsledek uzlu dedikovanému ke sběru a vytvoření konečného výsledku. [61]

Distribuovaný systém poskytuje řadu výhod oproti jiným řešením. Například zvyšuje celkový výkon systému rozdělením výpočetní zátěže do různých uzlů, což vede k menší zátěži na jednotlivé uzly. Dále je tento systém odolný vůči selhání jednoho či více uzlů. Pokud některý uzel selže, nebude ovlivněn celý systém jako je tomu u centralizovaných systémů. Navíc je systém velmi dobře škálovatelný, protože umožňuje upravit zdroje jednotlivých uzlů s ohledem na typ prováděných operací. [52]

Na druhé straně má distribuovaný systém určité nedostatky. Jedním z nich jsou problémy s bezpečností a ochranou soukromí, protože systém je založen na sdílení úkolů a dat mezi více uzly. Pokud je jeden z těchto uzlů škodlivý, může způsobit vážné problémy. Systém také komunikuje pomocí zpráv, které lze snadno ztratit. Přenos velkého množství dat vyžaduje široké pásmo a z toho plynou vyšší náklady na dostatečně propustnou síťovou infrastrukturu. [52]

3 Distribuovaná účetní kniha

V tradičních architekturách centralizovaných databází jsou informace umístěny, uloženy a udržovány na jednom místě a jsou řízeny centrální autoritou, která zajišťuje jejich integritu. Data jsou obvykle ukládána v nezpracované formě a existuje nějaký druh zabezpečení, který chrání databázi před vnějšími útoky.

Distribuované databáze poskytují alternativní řešení, kde databáze je rozprostřena po síti a uložena na různých fyzických místech. Přestože je databáze sdílena mezi uzly, její řízení je obvykle centralizované a integrita databáze závisí na centrální autoritě. Ke správě databáze a synchronizaci dat se používá centralizovaná aplikace. [28]

3.1 Definice distribuované účetní knihy

Distribuovaná účetní kniha (Distributed Ledger Technology, DLT) je distribuovaná databáze ve smyslu, že každý uzel má synchronizovanou kopii dat, ale liší se od tradičních distribuovaných databázových architektur třemi důležitými způsoby. Zaprvé, eliminuje potřebu důvěryhodné centralizované autority, centralizovaný server tedy není vyžadován pro zajištění integrity a konzistence dat mezi komunikujícími stranami. Místo toho se využívají mechanismy konsenzu[1] nebo ověřovací protokoly. Zadruhé, spolehlivost v prostředí bez důvěryhodnosti je dosažena již zmíněným mechanismem konsenzu, který zajišťuje konzistenci a integritu databáze, přestože si zúčastněné strany navzájem nedůvěřují. A zatřetí, účetní kniha využívá kryptografické nástroje pro zajištění výše zmíněných bodů. [29, 51]

Velkým benefitem DLT oproti tradičním distribuovaným databázím je schopnost bezpečného provozu v prostředí s nedůvěryhodnými třetími stranami, která je kritickou součástí jakékoli aplikace ve finančním průmyslu. Tradiční distribuované databáze totiž spoléhají na důvěryhodné uzly řízené centrální autoritou, na kterých uchovávají kopie knihy, tudíž nejsou schopny provozu v takovém prostředí. Jednou z nejvýznamnějších aplikací DLT je Bitcoinová síť. [28, 29]

[1] Mechanismům konsenzu je věnována kapitola 4.3.

3.2 Přístupová a ověřovací práva

Podle omezení přístupnosti a typu konsenzuálních mechanismů používaných k zajištění integrity účetní knihy existují různé typy DLT, viz obr. 3.1. Pokud jde o to, kdo má přístup k datům, lze DLT rozdělit na *veřejné* (public), přičemž každý účastník sítě může číst z účetní knihy, a *privátní* (private), kde k údajům mají přístup pouze schválení účastníci.

Obdobně, rozdělíme-li typy DLT v závislosti na tom, kdo má oprávnění k ověření účetní knihy, mohou být DLT rozděleny na typ *s povolením* (permissioned), u kterých může pouze určitá skupina důvěryhodných uživatelů ověřovat nebo upravovat položky účetní knihy, a typ *bez povolení* (permissionless), tzn. kdokoli může sestavit a ověřit účetní knihu. Skutečnost, že k ověřování zápisů do účetní knihy není potřeba žádná ústřední autorita, je účetní kniha odolná vůči cenzuře. Jinými slovy, žádný aktér nemůže zabránit tomu, aby byla transakce přidána do účetní knihy.

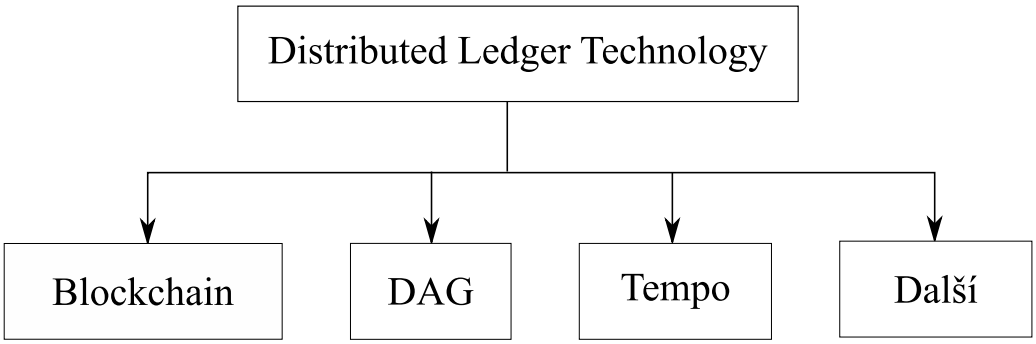
	Přístup pro čtení a vytváření transakcí	
Přístup pro zápis	Kdokoliv	Omezený
Kdokoliv	Public & Permissionless	Private & Permissionless
Omezený	Public & Permissioned	Private & Permissioned

Tabulka 3.1: Přehled přístupových a ověřovacích práv DLT.

3.3 Typy datových struktur

Datové struktury slouží k udržování distribuované účetní knihy, určují způsob jakým jsou data distribuována, strukturována a dohodnuta. DLT lze na základě použité datové struktury rozdělit na několik typů, přičemž nejznámější jsou Blockchain a Orientovaný acyklický graf (Directed Acyclic Graph, DAG). Mezi další typy patří například Tempo či Hashgraph. [27]

Zde také často dochází k záměně významů mezi pojmy DLT a Blockchain, protože v laické rovině jsou tyto pojmy často považovány za totožné. Důvodem této záměny je popularita blockchainu, který se dostal do širšího podvědomí na rozdíl od rodičovské technologie DLT. Přitom mezi DLT a blockchainem existuje řada rozdílů. Například blockchain je posloupnost bloků zřetězených dohromady pomocí hash funkcí a časových razítek, zatímco DLT takové řetězce nepotřebuje. Stručně řečeno, všechny blockchainy jsou DLT, ale ne všechny typy DLT jsou blockchainy (viz přehled datových struktur na obrázku 3.1).



Obrázek 3.1: Typy datových struktur DLT.

Pro potřeby této práce bude pozornost směřována pouze k technologii Blockchain, která je podrobněji vysvětlena v kapitole 4.

4 Blockchain

4.1 Definice blockchainu

Blockchain je specifickým typem distribuované účetní knihy (Distributed ledger technology, DLT), kterou lze definovat jako distribuovanou, sdílenou a šifrovanou databázi sloužící jako nevratné a neporušitelné úložiště informací. [57, 64]

Z teoretického hlediska má technologie Blockchain potenciál nahradit transakce zakořeněné v důvěře transakcemi založenými na pravidlech, která jsou definována matematicky a mechanicky vymáhána. Je důležité poznamenat, že neexistuje jedna univerzálně dohodnutá definice, protože blockchain nachází uplatnění v mnoha oborech.

4.2 Technický koncept

Obecně je Blockchain digitální platforma, která ukládá a ověřuje celou historii transakcí mezi uživateli v síti způsobem chráněným proti neoprávněným zásahům a revizi. Je také základní databázovou strukturou pro transakce v digitální měně jako jsou sítě Bitcoin a Ethereum. [42]

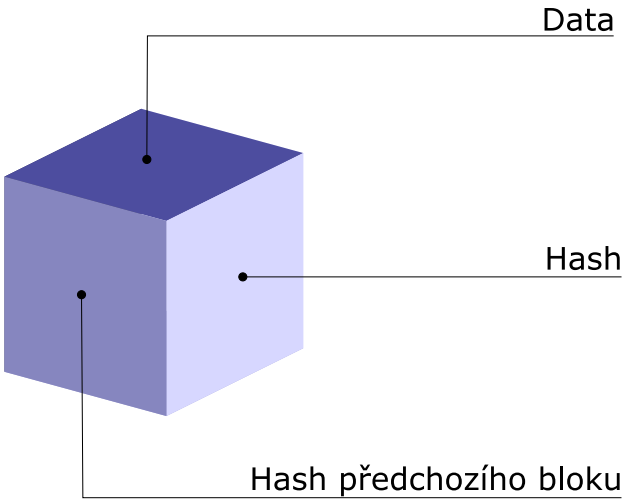
Jednotlivé transakce jsou chronologicky uspořádány do svazků známých jako bloky. Každý blok musí být nejprve ověřen sítí a teprve poté může být přidán do databáze (blockchain ledger). Bloky jsou vzájemně propojeny a tvoří dlouhý řetězec (odtud označení blockchain), který nikdo nemůže změnit, aniž by změnil všechny následujících bloky, k čemuž je potřeba souhlas většiny sítě. [57]

Každý uzel účastnící se sítě má totiž svou vlastní kopii blockchainu, která je synchronizována s ostatními uzly pomocí protokolu peer-to-peer. To odstraňuje potřebu ústředního orgánu, a tedy účastníků, aby měli důvěru v integritu jakéhokoli jednotlivého subjektu. Technologie Blockchain umožňuje více organizacím a skupinám, aby v rámci organizace efektivně zpracovávaly transakce a bezpečně dosáhly konsenzu bez nutnosti třetí strany. [42, 57]

V následujících podkapitolách budou podrobně vysvětleny jednotlivé základní technické koncepty technologie Blockchain.

4.2.1 Block

Svazek platných transakcí označovaný též jako blok. Blok musí dodržovat předem stanovenou sadu pravidel, aby byl platný. Například, nesmí překročit maximální velikost v bajtech, nesmí obsahovat více než maximální počet transakcí a musí odkazovat na poslední platný blok.



Obrázek 4.1: Jednotlivé složky bloků blockchainu.

Jak je patrné z obrázku 4.1, každý blok je složen ze dvou částí - hlavičky bloku (block header) obsahující metadata a těla bloku (block body) obsahující transakce. Přesná struktura bloku se liší v závislosti na použité technologii. Obecně lze však definovat několik komponent, které se vyskytují v hlavičce bloku vždy.

- číslo verze použitého softwaru
- hash předchozího bloku
- kořen Merkleova stromu[1]
- časové razítko udávající čas vytvoření bloku
- cílová složitost resp. maximální možná hodnota hashe
- nonce (number only used once) neboli libovolné celé číslo, používané k naplnění složitosti bloku pro počítání hashe nového bloku

[1]Hashový strom, který má v listech data a ve všech ostatních uzlech má hodnotu odpovídající výsledku kryptografické hashovací funkce, která na vstup dostane data z potomků. [59]

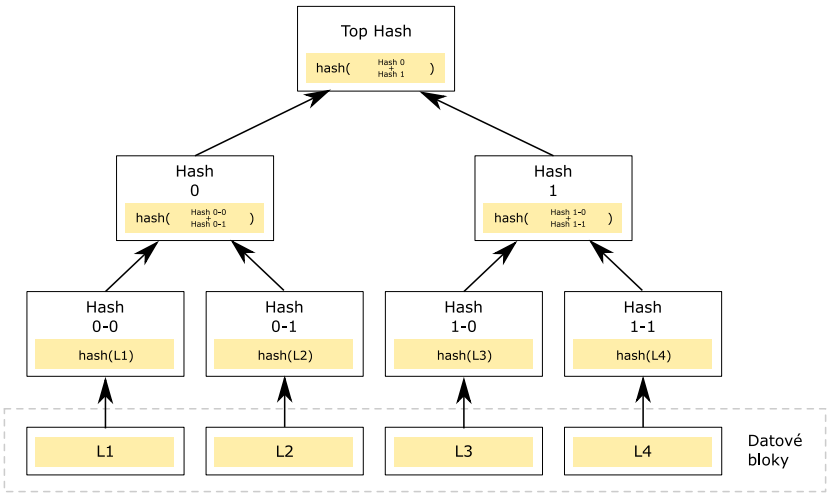
Výše zmíněné komponenty se dále dělí podle užití do tří sad metadat.

První část - Hash předchozího bloku, který spojuje tento blok s dřívějším blokem v blockchainu.

Druhá část - Jsou zde uložena metadata související s ověřením bloku sítí při přidávání nového bloku. [2]

Třetí část - Obsahuje kořen Merkleova stromu. Tento kořen stromu je datová struktura, která je využívána pro efektivní kompilaci všech transakcí v bloku.

Tělo bloku se skládá z čítače transakcí a jednotlivých transakcí. Maximální počet transakcí, které může blok obsahovat, závisí na velikosti bloku a velikosti každé transakce. Transakce jsou uloženy ve struktuře již zmíněného Merkleova stromu, viz obr. 4.2. Pro autentizaci transakcí blockchain používá asymetrický kryptografický mechanismus - digitální podpis založený na asymetrické kryptografii, který bude stručně ilustrován níže. [47, 50, 67]



Obrázek 4.2: Znázornění způsobu uložení transakcí v bloku, kde $L1$ až $L4$ představují uložené transakce a ostatní uzly stromu jsou poté kontrolní součty nad potomky.

[2] Používá se u Proof-of-Work. Více o tomto algoritmu v sekci 4.3.

4.2.2 Hash

Hash je řetězec číslic a znaků vytvořený pomocí kryptografické hashovací funkce[3]. Lze si jej představit jako unikátní neměnné označení každého bloku, které slouží k jednoznačné identifikaci bloku a celého jeho obsahu. Je to jako otisk prstu, který se používá k ujištění příjemce o integritě přijaté zprávy. Pokud by došlo ke změně uvnitř bloku, došlo by zároveň ke změně hashe, který by indikoval, že blok byl změněný.

Názornou ukázkou, jak funguje hashovací funkce je následující příklad, kde jsou zašifrovány zprávy „blo**c**kchain“ a „blokchain“ hashovacím algoritmem SHA-256 používaným jako standardní hashovací funkce techonologie blockchain. Jak je z příkladu možné vidět, i tak drobný rozdíl vstupu ovlivní výpočet natolik, že výstupní řetězec je kompletně odlišný.

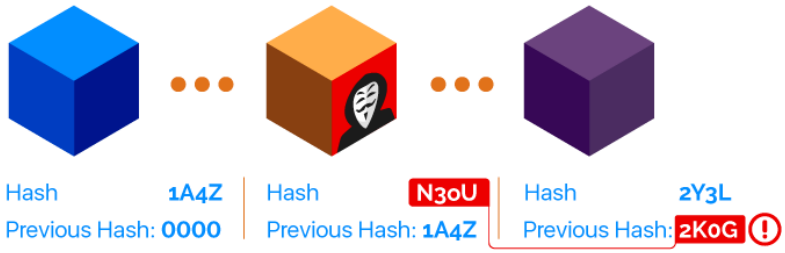
```
sha256("blockchain") =
ef7797e13d3a75526946a3bcf00daec9fc9c9c4d51ddc7cc5df888f74dd434d1

sha256("blokchain") =
f35d18abdcab7e9e7c2358471147fffab9adf1ced327d76ceaa5bfb75f75d775
```

Hash v blockchainu plní ještě jednu důležitou roli, a to, že funguje také jako ukazatel pro vzájemné propojení datových bloků. Z obrázku 4.3 je patrné, že hashový ukazatel (hash předchozího bloku) ukazuje na předchozí blok, takže každý blok zná adresu, na které se nacházejí data bloku předchozího. Navíc hash uložených dat může být veřejně ověřen uživateli, aby bylo prokázáno, že s uloženými daty nebylo manipulováno. [66]

Obrázek 4.3: Názorná ukázka zřetězení datových bloků pomocí hashových ukazatelů.

[3]Matematický algoritmus, který mapuje data o libovolné délce na řetězec pevné délky tzv. hash. Jedná se o jednosměrnou funkci, tzn. funkci, u které není triviální zjistit data na vstupu na základě znalosti výstupních dat, neboť malá změna vstupních dat způsobí velkou změnu na výstupu. [33]

Pokud by někdo změnil data v jednom bloku, změnil by se i hash daného bloku, což by způsobilo neplatnost celého řetězce. Uvedený problém je znázorněn na obrázku 4.4.



Obrázek 4.4: Názorná ukázka pokusu o změnu dat v bloku.

Z výše nastíněného problému vidíme, že hash je ideální nástroj pro identifikaci pokusů o změnu dat v blocích. Pokud by se protivník pokusil změnit data v jakémkoli bloku v celém řetězci, aby zamaskoval manipulaci, bude muset změnit ukazatele hash všech předchozích bloků. Nakonec protivník bude nucen přestat manipulovat s bloky, protože nebude schopen falšovat data na začátku řetězce (tzv. genesis block), který se generuje při vytvoření systému a jehož zaznamenáním společně s jedním kořenovým hashem lze efektivně zajistit, aby celý řetězec byl odolný proti neoprávněné manipulaci. [66]

Samotný hashovací algoritmus však nestačí k zajištění bezpečnosti blockchainu. Pro zmírnění pokusů o zničení blockchainu a zajištění bezpečnosti používá technologie blockchain také proces zvaný Proof-of-Work podrobněji popsaný v sekci 4.3. [66]

4.2.3 Nonce

Nonce je 32bitové číslo, které se nachází v záhlaví bloku spolu s dalšími klíčovými daty, jako je cílová složitost a časové razítko. Když těžaři staví bloky, náhodně si vyberou nonci a vloží ji do záhlaví bloku, čímž vytvoří nový hash záhlaví bloku. Používá se u algoritmu Proof-of-Work. Více o tomto algoritmu v sekci 4.3. [50]

4.2.4 Node

Blockchain je spravován softwarem, který běží na počítači nazývaném uzel (node) nebo také peer. Každý uzel je připojen k síti Blockchain a může

odesílat a přijímat transakce. Dále, každý uzel, který se účastní sítě má například svou vlastní kopii Blockchainu, která je synchronizována s ostatními uzly pomocí protokolu peer-to-peer. Každý uzel kontroluje platnost každé transakce a pokud většina uzlů říká, že transakce je platná, zapíše se do bloku. [42]

4.2.5 Digitální podpis

Digitální podpis stanovuje platnost údajů pomocí kryptografického algoritmu. Je to také způsob, jak lze ověřit, že s údaji nebylo manipulováno.

Každý uživatel vlastní dvojici soukromého a veřejného klíče. Soukromý klíč slouží pro podepisování transakcí a je potřeba jej uchovávat v tajnosti. Typický digitální podpis se skládá ze dvou fází: fáze podpisu a fáze ověření.

Ve fázi podpisu se z dat odesílatele vypočte otisk dat (hash) a zašifruje se soukromým klíčem odesílatele. Zašifrovaný výsledek (podpis) se poté pošle společně s původními daty příjemci. Následně ve fázi ověření příjemce dešifruje podpis veřejným klíčem odesílatele a porovná ho s otiskem původních dat. Tímto způsobem může příjemce snadno zkontrolovat, zda bylo s daty manipulováno. Takto digitálně podepsané transakce jsou posílány v celé síti. [67]

Správně definovaný a bezpečný podpisový algoritmus by měl mít dvě vlastnosti. Zaprvé, vytvářet platné a ověřitelné podpisy. Zadruhé zajistit, že podpisy jsou existenciálně neodpustitelné, tzn. protivník, který získá cizí veřejný klíč, nemůže falšovat podpisy u některých zpráv s velkou pravděpodobností.

Typický algoritmus digitálního podpisu používaný v sítích blockchain je algoritmus digitálního podpisu eliptické křivky (ECDSA[4]). [67]

4.2.6 Chytrý kontrakt

Chytrý kontrakt či inteligentní smlouva (Smart contract) je spustitelný kód, který běží na síti blockchain k usnadnění, provádění a vymáhání podmínek smlouvy. Hlavním cílem chytrého kontraktu je automatické provedení podmínek smlouvy, jakmile jsou splněny stanovené podmínky. Chytré kontrakty tedy slibují nízké transakční poplatky ve srovnání s tradičními systémy, které vyžadují, aby důvěryhodná třetí strana prosadila a provedla podmínky dohody. Navíc, protože jsou chytré kontrakty uloženy v síti blockchain, dědí

[4]https://doi.org/10.6028/nist.fips.186-4

tak některé vlastnosti jako je nezaměnitelnost a distribuovanost. [58] Jinak řečeno, jakmile je smlouva vložena do blockchainu, nelze kód smlouvy změnit. Zároveň je pro potencionálního útočníka nemožné vynutit vykonání kontraktu, protože ostatní účastníci by takový pokus odhalili a označili jej jako neplatný.

4.3 Konsenzuální algoritmy

Konsenzuální algoritmy (nebo také protokoly, consesus protocols) jsou typickým znakem decentralizovaných systémů. Zatímco u centralizovaných systémů určuje pravidla centrální autorita (CA), která zároveň nařizuje jejich dodržování. V decentralizovaných peer-to-peer systémech neexistuje žádný ústřední orgán, který by validoval a verifikoval transakce, přesto je každá transakce v blockchainu považována za zcela zabezpečenou a ověřenou. To je možné pouze díky přítomnosti konsenzuálního algoritmu, který je základní součástí každé sítě blockchain.

Konsenzuální mechanismus je skupinový protokol pro dynamické dosažení dohody ve skupině při aktualizaci jediné datové hodnoty nebo jediného stavu sítě s cílem zajistit rozšíření globální účetní knihy a zamezit škodlivým útokům. [54, 66] Problém dynamického získání konsenzu ve skupině závisí na skupinové spolupráci, která však může být narušena přítomností škodlivých aktérů a chybných procesů. Škodlivý účastník může například tajně vytvářet protichůdné zprávy, aby členové skupiny nejednali jednotně, což narušuje účinnost skupiny při koordinaci jejích akcí. Tento problém se označuje jako Problém byzantských generálů (Byzantine Generals Problem, BGP). Neschopnost dosáhnout konsenzu v důsledku škodlivých aktérů se pak označuje jako byzantská chyba (Byzantine fault). [49]

Aby bylo možné do blockchainu zanést nový blok, musí dojít k souhlasu určitého počtu uzlů sítě ohledně vkládaného bloku. Náhodně zvolený uzel proto představí návrh na vložení bloku ostatním uzlům a pokud určitá část uzlů, která je předem stanovená v konsenzuálním algoritmu, vyjádří souhlas, nový blok je přijat a je zapsán do blockchainu. Dle druhu konsenzu je pak přiřazena i odměna tomuto uzlu. [66] Způsobů, jak vybrat uzel, je hned několik a budou podrobně vysvětleny dále.

4.3.1 Proof of Work

Konsenzuální algoritmus Proof of Work (dále jen PoW) je původní mechanismus konsenzu v síti Blockchain. Poprvé byl použit v Bitcoinových sítích a poté byl rychle implementován mnoha hlavními kryptoměnami.

V decentralizované síti je pro zaznamenání transakcí potřeba určit jeden uzel, který zaznamenání provede. Nejjednodušším způsobem je náhodný výběr. Ten je však náchylný k útokům, takže byl zaveden mechanismus, kdy uzly mezi sebou soupeří o to, kdo jako první spočítá vhodný hash nového bloku. Vytvořený hash je 256bitové číslo a musí začínat velkým počtem nul, tj. má velmi malou hodnotu. Pokud nemá dostatečný počet nul, těžař zahodí hash a zkusí novou nonci. Tento proces se opakuje, dokud těžař nezjistí nonci, která produkuje hash s hodnotou menší nebo rovnou hodnotě cílové obtížnosti. 32bitová velikost nonce znamená, že existují čtyři miliardy možných kombinací (náhodné číslo může být cokoliv v rozsahu 0 až 2^{32}). Nonce je zároveň jediný parametr, který těžař mění, všechny ostatní parametry zůstávají stejné. Pokud těžař najde tzv. zlatou nonci (golden nonce), pošle blok do ostatních uzlů a všechny ostatní uzly musejí vzájemně potvrdit správnost hodnoty hashe. Pokud je blok ověřen, ostatní těžaři přidají tento blok do jejich vlastních blockchainů a těžař získá odměnu ve formě nově vytvořené měny a poplatků za schválené transakce. [5, 54]

Aby tento proces fungoval, PoW předpokládá, že polovina síťových uzlů jsou vždy čestní těžaři, proto pokud by někdo získal více než polovinu hashovací síly, učinil by tak tento konsenzus zranitelným, protože by existovalo riziko narušení sítě, tzv. útok 51%. Při tomto útoku je jedna entita nebo organizace schopna ovládat většinu hashovací rychlosti, tzn. má dostatek těžební síly k úmyslnému vyloučení nebo úpravě řazení transakcí. Zmíněná entita dále může zvrátit transakce, které provedla, zatímco byly kontrolovány, což vede k problému dvojité útraty (double-spending problem), kdy útočník použije stejnou měnu na vstupech více než jedné nové transakce. [66]

Významnou nevýhodou algoritmu PoW jsou náklady na energii a hardware. V současné době neexistuje způsob, jak urychlit proces nalezení správné nonce. To znamená, že těžaři pracují stylem pokus-omyl, dokud nenajdou zlatou nonci, což může být velmi náročné na zdroje. [5, 54] Protokol PoW je tedy poměrně pomalý ve srovnání s jinými protokoly.

4.3.2 Proof of Stake

Dalším nejpoužívanějším algoritmem je Proof of Stake (dále PoS). Jedná se o energeticky úspornou alternativu k PoW. Ve srovnání s PoW, ve kterém může být jakýkoliv účastník sítě těžařem, který ověřuje transakce a vytváří nové bloky, v blockchainech založených na algoritmu PoS jsou definována omezení pro výběr těžařů, tzv. validátorů. Prostřednictvím PoS jsou kvalifi-

kováni jako validátoři pouze účastníci, kteří disponují měnou dané sítě a odeslali speciální typ transakce s jejich kapitálem použitým jako vklad (stake) k uzamknutí do depositu. Poté probíhá aukce a každý validátor se může podílet na navrhování transakcí a ověřování nových bloků pomocí algoritmu konsenzu. Validátoři se střídají a provádí sázky na další blok. Rozhodnutí o tom, kdo bude mít právo navrhnout nový blok/transakci do blockchainu se stanoví na základě velikosti vkladu každého ověřovatele. Pravděpodobnost výběru je úměrná jejich sázce. Ostatní validátoři následně tento blok potvrdí a zanesou ho do své verze blockchainu. Pokud by validátor navrhl chybný blok nebo transakci, vložený vklad by byl zničen jako penalizace. Stejně by tomu bylo v případě, kdyby ostatní validátoři potvrzovali chybné bloky či transakce a navíc by tím snižovali hodnotu měny. [54, 66]

Pokud dojde k připojení bloku, obdrží všichni validátoři odměnu úměrnou jejich sázkám. Neexistuje žádná cena za vytvoření bloku, takže validátoři kromě podílu ze sázek obdrží navíc pouze poplatky za schválené transakce, které vložili do bloku. [66]

Existuje mnoho různých verzí PoS. Z pohledu algoritmů jsou to pak 2 hlavní typy - chain-based Proof of Stake a Byzantine Fault Tolerance (zkráceně BFT) Proof of Stake.

U chain-based PoS algoritmus v intervalech (např. každých 10 sekund) pseudonáhodně vybere validátora a přiřadí tomuto validátoru oprávnění vytvořit blok. Následně se tento blok spojí s některým předchozím blokem (obvykle se jedná o blok na konci dříve nejdelšího řetězce). Postupem času se tak většina bloků sjednotí do jediného neustále rostoucího řetězce. [66]

Prvotní verze řetězových algoritmů PoS byly naivní, protože odměny se používaly pro výrobu bloků bez penalizací, a proto trpí tzv. problémem „nothing at stake“. V této konkrétní situaci může validátor hlasovat a blokovat více konkurenčních řetězců najednou bez obav z jakékoliv penalizace. Pokud by validátoři vytvářeli bloky na obou (nebo více) řetězcích, budou vybírat transakční poplatky podle toho, která větev nakonec vyhraje, čímž by narušovali konsenzus a mohlo by to vést k zranitelnosti sítě vůči útokům s dvojnásobnou útratou. [66]

U algoritmu PoW nedochází k rozdělení řetězců, protože je výhodnější přidat nový blok do delšího řetězce, jelikož žádný těžař nechce ztrácet prostředky na bloku, který bude sítí odmítnut. Existuje tedy implicitní sankce za vytvoření bloku na nesprávném řetězci. [66]

V novějších verzích PoS jsou již zavedeny strategie pro explicitní tresty. Jedním z příkladů je protokol Casper PoS v síti Ethereum, který je založený na sankcích. Konkrétně, validátoři dají část své digitální měny (ETH) jako

podíl za účast na validaci dalších bloků a vsadí část svého vkladu na nový blok. Ověřovatelem se tak stanou pouze ty uzly, které jej chtějí přidat do své kopie blockchainu. Validátoři mohou získat odměnu pouze pokud je blok ověřen a připojen k blockchainu. Pokud validátor nejedná čestně, všechny jeho sázky budou zničeny. Tím se zajistí, že problém „nothing at stake“ nemůže nastat. [66]

Druhým typem Proof of Stake je BFT. Zde je proces navrhování dalšího bloku oddělen od vytváření bloku. Validátoři, kteří mají právo navrhovat bloky, jsou vybíráni zcela náhodně. Následně se provádí několik kol hlasování, kdy každý validátor hlasuje pro určitý konkrétní blok a na konci celého procesu se (čestní a online) validátoři trvale shodují na tom, zda je daný blok součástí řetězce. BFT algoritmy jsou tak schopny dosáhnout konsenzu, i když třetina validátorů přejde do režimu offline nebo se stane škůdcem (za předpokladu, že zbylí účastníci sítě jsou čestní). [21]

4.3.3 Raft

Raft je distribuovaný konsenzuální algoritmus, který byl navržen tak, aby byl snadno srozumitelný. Algoritmus nabízí obecný způsob, jak distribuovat stav v klastru uzlů, kde každý uzel musí souhlasit se stejnou sérií změn stavu. [48]

Raft pracuje na principu volby vedoucího uzlu v klastru. Každý uzel v klastru je buď vůdce, následovník nebo kandidát (v případě voleb, když vedoucí uzel není k dispozici). Vedoucí uzel je zodpovědný za přijímání požadavků klientů a distribuci stavu na ostatní uzly v klastru, tzn. data proudí pouze jedním směrem - od vedoucího uzlu k následovníkům. Raft využívá k provádění svých základních operací dva typy zpráv. [48]

RequestVotes - Typ zprávy používaný kandidáty během voleb. Požaduje hlas od uzlu.

AppendEntries - Typ zprávy používaný vedoucími uzly pro distribuci stavu a také jako tzv. tepová zpráva (neobsahuje žádný stav, vedoucí uzel takto informuje následovníky o své existenci)

Algoritmus Raft dělí problém konsenzu na tři dílčí problémy popsané níže.

Volba vedoucího uzlu

Pokud dojde k výpadku stávajícího vedoucího uzlu nebo když se inicializuje algoritmus, musí být zvolen nový vedoucí.

V tomto případě začíná v klastru nové období. Období je libovolně dlouhý časový úsek, pro který musí být zvolen nový vedoucí. Každé funkční období začíná volbou vedoucího uzlu. Pokud je volba úspěšně dokončena (tj. je zvolen jediný vedoucí), termín pokračuje v běžných činnostech organizovaných novým vedoucím. Pokud je volba neúspěšná, začíná nové období s novými volbami. [48]

Volbu vedoucího začíná kandidující uzel. Uzel se stane kandidátem, pokud po určitý časový limit neobdrží od vedoucího uzlu tepovou zprávu, takže předpokládá, že žádný vedoucí uzel není. Volby zahajuje kandidát zvýšením počitadla funkčních období, hlasováním pro sebe jako nového vůdce a odesláním zprávy RequestVotes všem ostatním uzlům. Hlasování může skončit třemi způsoby: [48]

- Kandidát dostane hlasy od většiny uzlů a stává se vedoucím uzlem. Poté vyšle tepovou zprávu ostatním v klastru, aby si vytvořil autoritu.
- Pokud další kandidáti obdrží zprávu AppendEntries, zkontrolují číslo funkčního období u zprávy. Pokud je číslo větší než jejich vlastní, přijmou uzel jako vůdce a změní si status zpět na následovníka. Pokud je číslo menší, odmítnou požadavek a zůstávají kandidáty.
- Kandidovalo-li více uzlů současně, hlasy se rozdělily bez jasné většiny. V tomto případě se budou konat nové volby po vypršení časového limitu jednoho z kandidátů.

Raft používá náhodný časový limit pro volby, aby zajistil, že problémy s nejasným vítězem budou rychle vyřešeny. Následovníci se nestanou kandidáty současně, protože vedoucí uzel, který vyhrál volby si stihne vytvořit autoritu dříve, než následovníkům vyprší limit. [48]

Distribuce stavu

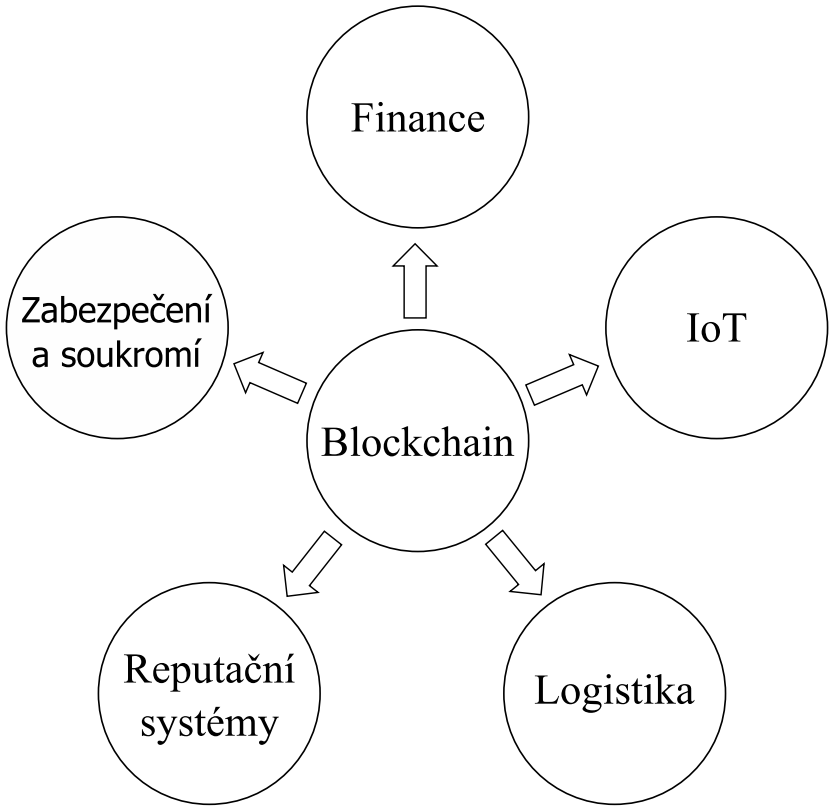
Vedoucí uzel je zodpovědný za distribuci stavu a příjem žádostí klientů. Každý požadavek klienta obsahuje příkaz, který má být proveden uzly v klastru. Když vedoucí uzel dostane požadavek, přidá jej do svého stavu v podobě nového záznamu a rozešle ho do všech následovníků pomocí zprávy AppendEntries, aby bylo možné zachovat konzistentnost stavu. Každý záznam obsahuje příkaz, index pozice ve stavu a období, ve kterém byl záznam proveden. V případě nedostupnosti následovníků, vůdce zprávu odesílá tak dlouho, dokud nebude stav všech následovníků aktualizován. [48]

Když je záznam vedoucím uzlem distribuován na většinu uzlů, považuje se požadavek za provedený a všechny předchozí záznamy (včetně těch vytvořených dřívějšími vůdci) jsou rovněž považovány za provedené. Vedoucí uzel následně provede příkaz v záznamu a vrátí výsledek klientovi. Jakmile následovníci zjistí, že položka byla potvrzena, provedou příkaz také. [48]

Při odesílání zprávy AppendEntries vedoucí uzel ke zprávě přidává číslo aktuálního období a index záznamu, který bezprostředně předchází novému. Pokud následovník nemůže najít shodu pro předchozí záznam ve svém vlastním stavu, odmítne žádost o přidání nového záznamu. Tato kontrola konzistence umožňuje vedoucímu uzlu dojít k závěru, že kdykoli se zpráva AppendEntries vrací úspěšně od následovníka, má vedoucí uzel stejný stav jako následovník. [48]

4.4 Využití

Existuje celá řada aplikací technologie blockchain. V této části bude představeno několik typických odvětví, kde se blockchain využívá, viz obr. 4.5.



Obrázek 4.5: Kategorizace typických aplikací technologie blockchain.

4.4.1 Finance

Blockchainové systémy, jako je Bitcoin[5] či framework Hyperledger[6] mají obrovský dopad na tradiční finanční a obchodní služby. Tato technologie může být použita v mnoha oblastech, včetně zúčtovacích a vypořádacích finančních aktiv. Kromě toho studie ukazuje, že existují skutečné obchodní případy, jako je zajištění finančních derivátů, které by mohly využít blockchain ke snížení nákladů a rizik. [46] Blockchain také upoutal obrovskou pozornost v očích velkých softwarových společností. Microsoft Azure a IBM začínají nabízet blockchain jako službu. [1, 3]

4.4.2 Logistika

Kromě vývoje finančních a obchodních služeb se blockchain využívá také v oblasti logistiky, především pak v systémech dodavatelského řetězce, který bude podrobněji vysvětlen v kapitole 6. [38]

Mezi hlavní způsoby využití patří monitorování cesty produktu od primárního dodavatele surovin až k dodání finálního výrobku na konci dodavatelského řetězce. Dodavatelský řetězec se v dnešní době snaží transformovat v plně transparentní systém, kde každý zákazník může vidět původ zboží, ekologické certifikáty dodavatelů nebo může zkontrolovat, kde se jeho zboží v daném řetězci nachází a může si být jist pravostí uvedených informací. Jedním z příkladů takového systému může být digitální platforma pro sledování potravin, na které pracuje společnost IBM ve spolupráci s obchodním řetězcem Walmart[7]. [23]

Jedná se o implementaci permissioned blockchainu pomocí platformy Hyperledger Fabric (viz kapitola 4.6.1), která umožňuje snazší sdílení informací o původu potravin. [23] Dále v případě otrav způsobených jídlem platforma umožňuje snáze najít zdroj vadné šarže zboží a odstranit jej z celého dodavatelského řetězce. Tím se nejen sníží náklady na stažení zboží, ale i dopady, které by další prodané zkažené zboží způsobilo. Obchodní řetězec Walmart již takto úspěšně využívá blockchainovou platformu pro sledování původu vepřového masa ve svých obchodech v Číně. [23]

Podobně lze blockchain využít pro zpřehlednění dodavatelského řetězce léčiv, čímž chrání spotřebitele před padělky a zdravotními problémy v případě vadných léčiv. Během hackathonu v Heidelbergu tým studentů vyvinul

[5] https://bitcoin.org/
[6] https://www.hyperledger.org/
[7] https://www.walmart.com/

prototyp systému s názvem LifeCrypter, který přináší integritu, sledovatelnost a transparentnost do globálního dodavatelského řetězce léčiv. [56]

Systém je navržen tak, že každý lék je svázán s jednoznačným identifikačním štítkem, který umožňuje převádět virtuální a fyzické vlastnictví z dodavatelů na spotřebitele. Celý proces probíhá paralelně prostřednictvím důvěryhodné sítě ověřené chytrými kontrakty v blockchainu a globálním dodavatelským řetězcem léků. Chytrý kontrakt umožňuje volný obchod bez potřeby důvěry pro všechny zúčastněné strany s možností zavést pravidla, která jsou ve všech stádiích transparentní a vynucující. [56]

Další zajímavý případ užití v systému dodavatelského řetězce nalezneme v námořní přepravě. Zde společnosti IBM a Maersk[8] vytvořily projekt, ve kterém blockchain používaly k digitalizaci transportu květin z Keni do Nizozemska. Běžně je do každého takového transportu zapojeno mnoho zúčastněných stran od státních autorit přes přístavy až po zákazníky a je to velmi náročné na administrativu. Pro posun výrobku v řetězci do dalšího kroku je nutné poskytnout mnoho dokumentů opatřených razítky, které se v lepším případě posílají přes e-mail, v horším se předávají ve fyzické formě z ruky do ruky. Pro základ nového systému byl využit blockchain spolu s chytrými kontrakty (viz 4.2.6), které pomáhají s hladkým přechodem informací mezi jednotlivými zúčastněnými stranami. Například při platbě cla produkt automaticky dostane povolení k přesunu z přístavu do kontejneru. V blockchainu pak jsou zapsány všechny informace jako například certifikace dodavatele o ekologickém ošetření květin. [20]

Hlavní výhody plynoucí z využití technologie blockchain jsou například lepší ověřitelnost toho, že dané zboží splňuje mezinárodní standardy, snížení ztrát způsobených černým trhem nebo redukce papírování a administrativních nákladů. Dle průzkumu světové obchodní organizace by zavedení technologie blockchain do světového dodavatelského řetězce námořní přepravy mohlo zvýšit světové HDP o 5 % a objem globálního obchodu o 15 %. [24]

4.4.3 Internet věcí

Internet věcí (Internet of Things, IoT) umožňuje vývoj aplikací a služeb tím, že různým zařízením a objektům umožňuje sdílet data přes internet, což zvyšuje kvalitu života lidí digitalizací služeb. Jedná se o jednu z nejslibnějších informačních a komunikačních technologií, která odemyká nové příležitosti v různých doménách. Hlavní překážkou masového rozšíření zařízení internetu věcí je dnes nedostatek důvěry. Důvodem je to, že stávající architektura IoT

[8] https://www.maersk.com/

se spoléhá na centralizovaný systém, ve kterém třetí strana nebo poskytovatel služeb spravuje a řídí všechna data shromážděná ze zařízení IoT bez jasných hranic o tom, jak se shromažďovaná data používají. Centralizovaný server funguje jako černá skříňka a účastníci sítě nemají jasnou představu o tom, kde a jak jsou jejich data využívána. [36]

Jak ukazují některé studie, integrace IoT s blockchainem může vyřešit mnoho problémů, zejména pak problémy se zabezpečením. [35, 53]

4.4.4 Reputační systémy

Reputace je důležitým měřítkem důvěry komunity. Čím větší je reputace jednotlivce, tím důvěryhodnějším se stává z pohledu ostatních. Pověst jednotlivce lze hodnotit na základě jeho předchozích transakcí a interakcí s komunitou. Přibývá totiž počet případů, kdy dochází k falšování záznamů za účelem zlepšení pověsti. Například v elektronických obchodech mnoho poskytovatelů služeb generuje obrovské množství falešných zákazníků pro dosažení vysoké reputace. V tomto ohledu může Blockchain tento problém vyřešit. [55]

Důležitou roli hrají reputační systémy také v oblasti průmyslu při posilování vzájemné důvěry mezi průmyslovými subjekty a při budování důvěry u spotřebitelů. [44]

4.4.5 Zabezpečení a soukromí

Blockchain našel uplatnění i v oblasti internetu k ukládání záznamů DNS. Příkladem může být služba Ethereum Name Service (ENS), což je decentralizovaná jmenná služba postavená na technologii Ethereum, která existuje pouze ve formě chytrých kontraktů, které zajišťují soukromí a bezpečnost dat. Díky decentralizovanému řešení je ENS mnohem lépe odolná vůči útokům, jelikož eliminuje problém jediného napadnutelného cíle. [41]

4.5 Běžné způsoby zajištění důvěry

Technologie blockchain zajišťuje důvěru dat pomocí distribuované šifrované databáze, která slouží jako nevratné a neporušitelné úložiště. V praxi se ale častěji můžeme setkat s jinými způsoby, např. časovými razítky nebo certifikáty.

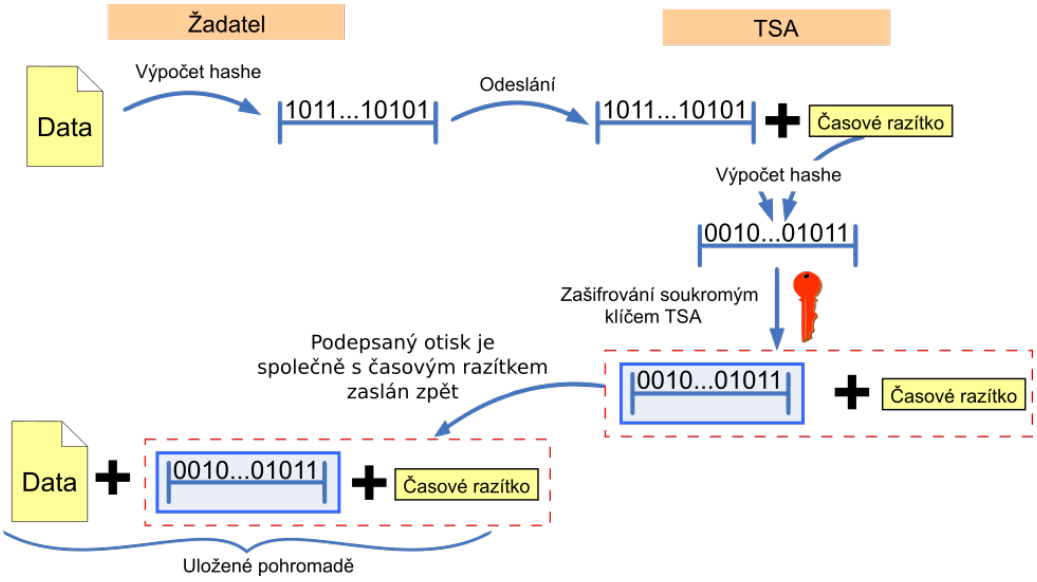
4.5.1 Časové razítko

Časové razítko (Timestamp) je způsob, jak lze bezpečně sledovat čas vytvoření a úpravy dokumentu. Jelikož otisk je vytvořen tak, aby jednoznačně identifikoval dokument, lze zaručit, že i původní dokument v určitém čase existoval.

Důvěryhodné časové razítko (Trusted timestamp) je takové časové razítko, které je vydané důvěryhodnou nezávislou autoritou pro vydávání časových razítek (Time Stamping Authority, TSA). Slouží k prokázání existence určitých údajů před určitým okamžikem (např. smlouvy, výzkumných údajů, lékařského záznamu, atd.) bez možnosti změnění razítka vlastníkem. Pro zvýšení spolehlivosti a snížení zranitelnosti lze použít vícero TSA.

Vytvoření razítka

Tato metoda je založena na digitálních podpisech a hašovacích funkcích. Nejprve žadatel vytvoří otisk dokumentu a požádá TSA o vydání časového razítka pro tento otisk. TSA připojí k obdrženému otisku dat časové razítko a vypočítá otisk tohoto zřetězení. Tento otisk je následně digitálně podepsán soukromým klíčem TSA. Takto podepsaný otisk (digitální podpis) je společně s časovým razítkem zaslán zpět žadateli, který jej uloží s původními daty. Celý postup je znázorněn na obrázku 4.6. [22]



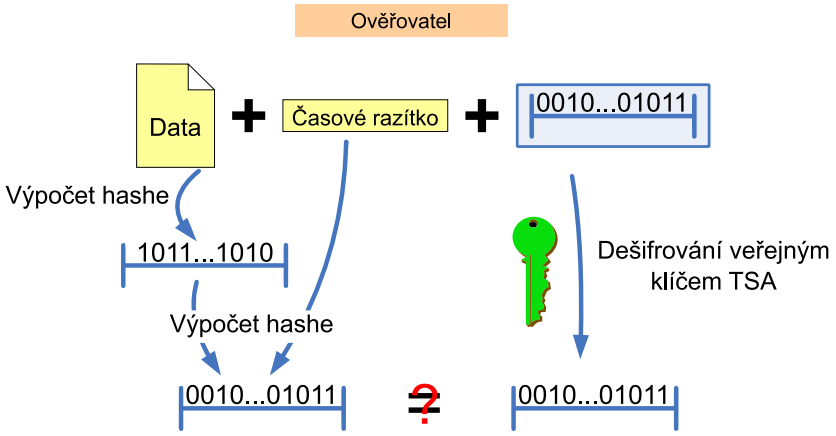
Obrázek 4.6: Postup vytvoření časového razítka.

O vydání časového razítka k otisku může požádat kdokoliv, protože TSA nijak nezkoumá totožnost žadatele ani jeho vztah k dokumentu, takže vy-

dané razítko neobsahuje jeho identifikaci. Navíc, TSA pracuje pouze s otiskem původních dat, což umožňuje použít tuto metodu i pro důvěrná data, protože TSA původní data nikdy neuvidí. [22]

Ověření razítka

Aby bylo možné ověřit časové razítko, je potřeba vypočítat otisk původních dat a k nim následně připojit časové razítko získané od TSA. Vypočtený otisk z tohoto zřetězení označíme jako A. Nyní je třeba ověřit digitální podpis TSA. To lze provést tak, že podepsaný otisk poskytnutý od TSA dešifrujeme veřejným klíčem TSA a otisk označíme jako B. Následně jsou otisky A a B porovnány, aby se potvrdilo, že jsou stejné, čímž se potvrdí, že časové razítko a data jsou nezměněny a byly vydány TSA. Celý postup ověření je uveden na obrázku 4.7. Pokud tomu tak není, pak bylo časové razítko změněno nebo TSA nevydal časové razítko. [22]



Obrázek 4.7: Postup ověření časového razítka.

4.5.2 Digitální certifikát

Jedná se o mechanismus zabezpečení, se kterým se setkáváme v každodenním životě. Digitální certifikát (public key certificate nebo digital certificate) je elektronický dokument, který spojuje předmět certifikátu s veřejným klíčem. Předmětem certifikátu může být fyzická osoba, instituce nebo třeba server. Jedná se o jeden ze základních nástrojů používaných v rámci infrastruktury veřejných klíčů (Public Key Infrastructure, PKI), která tvoří základ digitální důvěry. [4]

Existuje celá řada formátů certifikátů. Struktura, obsah a náležitosti certifikátů jsou specifikovány technickým standardem RFC 5280. Nejběžnějším formátem certifikátů pro PKI je standard X.509. Certifikáty X.509 se používají v mnoha internetových protokolech, včetně protokolu SSL/TLS a v offline aplikacích pro elektronické podpisy. [25]

Certifikát X.509 obsahuje informace o veřejném klíči a předmětu a je podepsán certifikační autoritou nebo podepsán sám sebou. Když je certifikát podepsán důvěryhodnou certifikační autoritou nebo ověřen jinými prostředky, lze veřejný klíč použít pro navázání bezpečné komunikace s jinou stranou nebo pro ověření digitálně podepsaných dokumentů. [25]

Vydání certifikátu

V typickém PKI schématu je vydavatelem certifikátů certifikační autorita (Certificate authority, CA), která ručí za certifikáty jí vydané a jejímž úkolem je ověřit splnění konkrétních podmínek, aby certifikát mohl být vydán. Tyto podmínky se liší podle toho, k jakému účelu je certifikát určen. Například pro certifikáty určené pro SSL/TLS šifrování je potřeba prokázat, že osoba žádající o certifikát je skutečným vlastníkem domény. [4]

Ověření certifikátu

Každý certifikát je digitálně podepsán soukromým klíčem vystavitele. Aby mohl být tento podpis ověřen, je třeba získat podepisující certifikát. Pokud tento certifikát není tzv. kořenovým certifikátem (certifikát CA podepsaný jejím soukromým klíčem), má také svého vydavatele a je třeba ověřit pravost i jeho certifikátu. To je opět možné pouze certifikátem jeho vydavatele, a tak se proces opakuje dokud nebude nalezen kořenový certifikát. Sestavená posloupnost certifikátů se označuje jako tzv. certifikační cesta a všechny certifikáty v této cestě je nutné verifikovat. Ověření, že certifikát skutečně vydala uvedená CA, se provede verifikací digitálního podpisu. Zkoumají se i další atributy - zda je certifikát odpovídajícího typu, zda vydavatel měl právo vydat takový certifikát atd. U všech certifikátů v certifikační cestě musí být zjištěno, zda nebyly odvolány, tzn. zda nejsou neplatné. [6, 25]

4.6 Blockchain platformy

Technologie Blockchain nachází v dnešní době uplatnění napříč různými odvětvími. Již existuje mnoho společností, které se rozhodly pro tuto techno-

logii a vyvíjejí aplikace založené na jejích principech, které pomohou zvýšit transparentnost a účinnost obchodních operací, viz sekce 4.4. Zde je důležité zmínit různé platformy Blockchain, které mohou podniky používat.

4.6.1 Hyperledger Fabric

Hyperledger Fabric[9] (dále jen Fabric) je modulární a rozšiřitelný systém s otevřenými zdrojovými kódy pro nasazení a provozování privátních blockchainů a jeden z projektů Hyperledger nadace Linux Foundation[10]. Jedná se o první skutečně rozšiřitelný systém blockchain pro provoz distribuovaných aplikací, který podporuje modulární konsenzuální protokoly, jež umožňují přizpůsobení systému konkrétním případům použití.

Fabric je také první distribuovaná účetní kniha, která podporuje chytré kontrakty[11] vytvořené v běžných programovacích jazycích jako jsou Java, Go a JavaScript (Node.js) spíše než v omezených doménově specifických jazycích (DSL). To znamená, že většina podniků již má potřebné dovednosti pro vývoj a není třeba dalšího školení.

Platforma Fabric dále podporuje oprávnění, což znamená, že na rozdíl od veřejných permissionless blockchainů jsou si účastníci navzájem známí, nikoli anonymní a tedy nedůvěryhodní. To znamená, že i když si účastníci nemusí zcela navzájem důvěřovat (mohou to být například konkurenti ve stejném odvětví), systém může být provozován podle modelu řízení, který je postaven na důvěře mezi účastníky. [9]

4.6.2 Ethereum

Ethereum[12] je veřejná, distribuovaná výpočetní platforma s otevřenými zdrojovými kódy založená na technologii blockchain, která umožňuje uživatelům sítě vytvářet a používat decentralizované aplikace, tzv. DApps (Decentralized Applications). Tyto decentralizované aplikace jsou programovány jako chytré kontrakty a je možné je následně nasadit a spouštět virtuálním strojem Ethereum Virtual Machine (EVM) běžícím v každém uzlu sítě.

Takto napsané decentralizované aplikace pak získávají výhody technologie blockchain. To znamená, že mohou být důvěryhodné a nezávislé na centrální autoritě (jakmile budou nasazeny do Ethereum sítě, budou vždy fungovat tak, jak byly naprogramovány).

[9] https://www.hyperledger.org/projects/fabric

[10] https://www.linuxfoundation.org/

[11] Chytrý kontrakt umožňuje dvěma stranám bezpečnou výměnu peněz, akcií, majetku a jiných aktiv přímo bez účasti zprostředkovatelů.

[12] https://ethereum.org/

Pro kompenzaci prováděných výpočtů a pro platby se využívá kryptoměna ether (ETH), která je generovaná platformou Ethereum. [8]

Ačkoliv je platforma Ethereum založená na veřejném blockchainu, existují implementace zaměřené především na oblast podnikání, které vycházejí z této platformy, ale využívají privátní blockchain. Mezi takové patří například Quorum[13], což je distribuovaný protokol privátní účetní knihy s povoleními založený na Ethereu, který je využívaný v průmyslových odvětvích jako např. finance, dodavatelský řetězec, maloobchod, nemovitosti.

4.6.3 Insolar

Insolar[14] je novější, stále ještě vyvíjená, blockchain platforma s otevřenými zdrojovými kódy. Platforma se zaměřuje především na podnikovou sféru, kde distribuovaná účetní kniha a inteligentní smlouvy jsou schopny zlepšit propojení a interakce mezi podniky. Mezi tato odvětví patří např. dodavatelský řetězec a logistika, maloobchod a spotřební zboží, výroba, atd.

Platforma Insolar nabízí flexibilní správu, která uživatelům umožní vybrat si mezi použitím veřejného blockchainu vytvořením sítě s vlastními pravidly nebo kombinace veřejné a privátní sítě. Tato flexibilita umožňuje budovat distribuované podnikové sítě, kde si každý může vybrat konfigurace a funkce, které nejlépe vyhovují jeho potřebám.

Platforma dále podporuje chytré kontrakty navržené s ohledem na potřeby podnikových vývojářů, které usnadňují práci s obchodní logikou a snadno se vyvíjejí pomocí jazyků Go a Java. Platforma Insolar navíc poskytuje funkce pro zabezpečení dat splňující předpisy (jako je GDPR) a kryptografické standardy specifické pro jednotlivé země. [40]

4.6.4 Srovnání blockchain platforem

Následující tabulka 4.1 přehledně srovnává parametry výše uvedených typů blockchain platforem.

[13] https://www.goquorum.com/

[14] https://insolar.io/

[15] Tx\s = počet provedených transakcí za sekundu.

	Insolar	Hyperledger Fabric	Ethereum	Quorum
Veřejné sítě	Ano	Ne	Ano	Ne
Privátní sítě	Ano	Ano	Ne	Ano
Hybridní sítě	Ano	Ne	Ne	Ne
Otevřené zdrojové kódy	Ano	Ano	Ano	Ano
Stav vývoje	Ve vývoji	Plně vyvinutý	Plně vyvinutý	Plně vyvinutý
Zaměření	Průmysl obecně	Průmysl obecně	Finanční sektor	Finanční sektor
Programovací jazyk chytrých kontraktů	Java, GoLang	Java, GoLang, Javascript	Solidity	Solidity
Konsenzus	upravený BFT	Raft, BFT	PoW, PoS	Raft, BFT
Propustnost (Tx\s)[15]	> 10000	~ 1200	~ 260	~ 650

Tabulka 4.1: Srovnání blockchain platforem. Hodnoty Ano/Ne vyjadřují podporu dané vlastnosti konkrétní platformou. [8, 9, 30, 34, 40]

5 Hyperledger Fabric

Tato kapitola se zabývá popisem klíčových funkcionalit frameworku Hyperledger Fabric a způsobu, jak framework funguje jako celek.

Hyperledger Fabric je framework pro distribuovaná řešení účetní knihy pod licencí Linux Foundation. Jedná se o modulární implementaci technologie DLT (viz kapitola 3) postavené na Docker[1] kontejnerech, která nabízí řešení pro zabezpečení podnikových sítí, škálovatelnost a důvěrnost. [12] Hyperledger Fabric dále poskytuje následující funkce sítě blockchain:

Správa identit - Na rozdíl od otevřeného systému bez oprávnění, kde se každý může účastnit sítě, je Hyperledger Fabric privátní systém s oprávněními, který vyžaduje protokoly (jako např. „PoW“) k ověření transakcí a zabezpečení sítě. Hyperledger Fabric registruje nové členy sítě prostřednictvím důvěryhodného poskytovatele služeb členství (Membership Service Provider, MSP). Hyperledger Fabric dále používá certifikáty, které jsou vázány na organizace, síťové komponenty a koncové uživatele nebo klientské aplikace. Kromě toho mohou být použity seznamy řízení přístupu (Access Control List, ACL) pro přidání dalších vrstev oprávnění. [12]

Efektivní zpracování - V síti Hyperledger Fabric má každý uzel jednu nebo více rolí. Pro zlepšení paralelismu v síti je provádění transakce odděleno od schvalování a potvrzení transakce. Toto souběžné provádění zvyšuje efektivitu zpracování u každého partnera a urychluje doručení transakcí řadící službě. [12]

Chaincode - Jedná se o chytré kontrakty sítě Hyperledger Fabric, které kódují logiku, jež může být vyvolána externími aplikacemi. Pokud externí aplikace potřebuje komunikovat s účetní knihou, vyvolá speciální typ transakce v komunikačním kanálu. Například chytrý kontrakt, který definuje parametry pro změnu vlastnictví, zajišťuje, aby všechny transakce, které převádějí vlastnictví, podléhaly stejným pravidlům a požadavkům. Chytré kontrakty lze psát v různých programovacích jazycích jako jsou Go, JavaScript (Node.js) a Java. [12]

[1]Docker je open source software pro automatizaci nasazení aplikací jako přenosných, soběstačných kontejnerů, které lze spustit v cloudu nebo v místním prostředí. [7]

Soukromí a důvěrnost - Hyperledger Fabric poskytuje možnost vytvářet kanály. Soukromé kanály jsou omezené komunikační cesty pro zasílání zpráv, které lze použít k zajištění soukromí a důvěrnosti transakcí pro konkrétní podmnožiny členů sítě. Všechna data v kanálu, včetně transakcí, informací o členech a kanálech, jsou neviditelná a nepřístupná pro členy sítě, kterým nebyl výslovně udělen přístup k tomuto kanálu. [12]

Modulární design - Architektura Hyperledger Fabric nabízí několik možností propojení týkajících se konsenzuálních protokolů a formátů. Například, do jakékoli sítě Hyperledger Fabric mohou být zapojeny specifické algoritmy pro identifikaci, konsenzus a šifrování. [12]

5.1 Architektura sítě

Síť blockchain s oprávněními je infrastruktura, která poskytuje služby účetní knihy spotřebitelům a správcům aplikací. Ve většině případů se více organizací spojí jako konsorcium[2] za účelem vytvoření sítě a jejich oprávnění jsou určena sadou zásad, které jsou dohodnuty konsorciem při původní konfiguraci sítě (síťové zásady mohou být v průběhu času změněny, pokud s tím organizace v konsorciu souhlasí). [16]

Síť Hyperledger Fabric se skládá z následujících komponent:

- Uzel sítě
- Řadící služba
- Kanál
- Certifikačních autorita
- Účetní kniha (pro každý kanál jedna)
- Chytrý kontrakt (chaincode)

Uživatelé síťových služeb jsou:

- Správci sítě blockchain
- Klientské aplikace vlastněné organizacemi

[2]Konsorcium je sdružení podnikatelů založené za účelem dosažení nějakého společného obchodního cíle. [19]

5.1.1 Uzel

Uzel je síťová entita, která udržuje účetní knihu a spouští kontejnery s chaincodem, aby mohla provádět operace čtení a zápisu do účetní knihy. Uzly jsou vlastněny a spravovány členy sítě. [13]

Jednotlivé uzly v síti mohou mít různé role. Uzly se dělí dle činností, které vykonávají, následovně:

Potvrzovací uzel (Commiting peer) - Uzel, který má na starosti ověřování bloků seřazených transakcí a potvrzuje (zapisuje / připojuje) bloky ke kopii účetní knihy, kterou udržuje. Jeho funkce je podrobněji popsána v kapitole 5.1.2.

Schvalovací uzel (Endorsing peer) - Specifický uzel definovaný zásadami sítě, který provádí simulaci chytrých kontraktů na své kopii účetní knihy a vrací navrhovanou odpověď (potvrzení) klientské aplikaci. Definování politiky schvalování pro chytré kontrakty znamená výběr podmnožiny organizací, jejichž uzly jsou povinny transakci digitálně podepsat před jejím odevzdáním do účetní knihy. [16] Celý proces schvalování je podrobně popsán v kapitole 5.1.7.

Protože všechny uzly udržují kopii účetní knihy pro každý kanál, ke kterému jsou připojeny, všechny uzly jsou potvrzovací. Avšak pouze uzly definované schvalovací politikou chytrých kontraktů mohou být schvalovacího typu. Uzel může být dále definován níže uvedenými rolemi:

Opěrný uzel (Anchor peer) - Definován v konfiguraci kanálu a je prvním uzlem, který bude objeven v síti jinými organizacemi v rámci kanálu, ke kterému jsou připojeny. [16]

Vedoucí uzel (Leading peer) - Uzel slouží pro komunikaci s řadící službou jménem organizace, která má více uzlů. [16]

5.1.2 Řadící služba

Mnoho distribuovaných blockchainů, jako jsou Ethereum a Bitcoin, nemá oprávnění, což znamená, že se může na procesu konsenzu účastnit kterýkoli uzel, přičemž transakce jsou uspořádány a seskupeny do bloků. Hyperledger Fabric funguje jinak. Vyčleňuje speciální uzel tzv. orderer (také označovaný jako ordering node), který je pověřen prováděním transakčního řazení a spolu s dalšími uzly tvoří řadící službu. Protože Hyperledger Fabric je založen na

deterministických konsenzuálních algoritmech, je zaručeno, že jakýkoli blok vygenerovaný řadící službou, který uzel validuje, bude konečný a správný.

Řadící služba existuje pro všechny kanály v síti nezávisle na vzájemných procesech a řadí transakce způsobem „kdo dřív přijde, je dřív na řadě“. Řadící služba je společnou vazbou pro celou síť, obsahuje certifikáty svázané s každým členem sítě. [13, 17]

5.1.3 Kanál

Kanál je komunikační prostředek používaný pro připojení komponent sítě. Kanály jsou vytvářeny generováním konfiguračního bloku v řadící službě, která vyhodnocuje platnost konfigurace kanálu. Kanály jsou užitečné, protože umožňují izolaci a důvěrnost dat. Zúčastněné organizace musí být nejprve ověřeny, aby s kanálem mohly spolupracovat. Kanály se řídí zásadami, s nimiž jsou nakonfigurovány.

Každý kanál má vlastní kopii účetní knihy sdílenou všemi uzly v kanálu, proto je důležité, aby s ní mohly interagovat pouze ověřené organizace. [16]

5.1.4 Certifikační autorita

Certifikační autorita (CA) slouží k vydávání certifikátů členským organizacím v síti a jejich uživatelům. V síti může být jedna nebo více certifikačních autorit a organizace se mohou rozhodnout, jestli chtějí používat vlastní certifikační autority nebo zabudovanou CA (nazývanou Fabric-CA).

Certifikáty vydávané CA jsou velmi důležité. Klientské aplikace vlastněné organizacemi v konsorciu používají certifikáty k ověřování transakčních návrhů a uzly je používají ke schvalování návrhů a k přidávání platných transakcí do účetní knihy.

Certifikáty se dále používají k vytváření digitálních podpisů v transakcích, ve smyslu, že organizace souhlasí s výsledkem transakce. [13, 16]

Identita

Účastníci sítě blockchain mohou být řadící služby, uzly, administrátoři, klientské aplikace a další. Každý účastník musí mít digitální identitu v digitálním certifikátu X.509. Identity definují oprávnění ke zdrojům a přístup k informacím, které mají účastníci v síti blockchain.

Aby byla identita ověřitelná, musí pocházet od důvěryhodné autority, kterou definuje poskytovatel služeb členství (viz odstavec níže). [14]

Poskytovatel služeb členství

Poskytovatel služeb členství (Membership Service Provider, MSP) je komponenta, která umožňuje definovat, které CA jsou důvěryhodné pro definování členů domény (např. organizace), a to buď tím, že uvedou totožnost svých členů nebo určí, které CA jsou oprávněny vydávat platné identity pro jejich členy nebo kombinace obou. MSP dále umožňuje definovat konkrétní role, které účastník má v rámci organizace, kterou MSP reprezentuje a nastavuje základ pro definování přístupových oprávnění v kontextu sítě a kanálu. V síti se může vyskytovat jedna nebo i více MSP.

Přes své jméno tato komponenta ve skutečnosti nic nenabízí. Ve své podstatě se jedná o sadu složek, které se přidávají do konfigurace sítě a používají se k definování organizace jak směrem dovnitř (organizace rozhodují, kdo jsou její správci), tak směrem ven (povolením jiným organizacím ověřit, že entity mají oprávnění dělat to, co se pokoušejí udělat). Výchozí implementace MSP používá jako identitu certifikáty X.509, přičemž používá tradiční infrastrukturu veřejných klíčů PKI. [15]

5.1.5 Účetní kniha

V Hyperledger Fabric se účetní kniha skládá ze dvou částí:

Světový stav (World state) - Databáze, která obsahuje aktuální hodnoty atributů business objektu jako jedinečný stav účetní knihy. Tento tzv. world state usnadňuje programu přímý přístup k aktuální hodnotě stavu, aniž by jej musel vypočítat procházením celého protokolu transakcí. Stavy účetních knih jsou ve výchozím nastavení vyjádřeny jako páry klíč - hodnota, ale pro větší flexibilitu je možné využít jiného typu databáze (např. CouchDB). [10]

Blockchain - Protokol transakcí, který zaznamenává všechny změny, které vedly k současnému stavu. Transakce jsou shromažďovány uvnitř bloků, které jsou připojeny k blockchainu (což umožňuje pochopit posloupnost změn, které vedly k současnému stavu). [10]

5.1.6 Chaincode

Chytré kontrakty obecně definují logiku transakcí, které řídí životní cyklus business objektu obsaženého ve stavu světa. Poté jsou zabaleny do chaincodu, který je nasazen do sítě blockchain. Chytré kontrakty lze považovat za řídící transakce, zatímco chaincode určuje, jak jsou chytré kontrakty baleny pro nasazení, tzn. ve stejném chaincodu lze definovat více chytrých kontraktů

a při nasazení chaincodu jsou všechny chytré kontrakty v něm zpřístupněny aplikacím. [11]

Nasazení chaincodu

Nasazení chaincodu probíhá ve dvou krocích:

Instalace - Po vytvoření chytrého kontraktu jej musí administrátor v organizaci nainstalovat do uzlu. Pokud má organizace v kanálu více uzlů, může si administrátor zvolit uzly, na které instaluje chytré kontrakty, protože není třeba instalovat chytré kontrakty na každý uzel. [11]

Vytvoření instance - Přestože je chytrý kontrakt nainstalován v uzlu, ostatní komponenty připojené ke kanálu o tom nevědí. Z toho důvodu musí administrátor ještě vytvořit instanci chytrého kontraktu na kanálu. Po vytvoření instance si je každý uzel, který je součástí kanálu, vědom existence tohoto chytrého kontraktu a lze jej nyní vyvolat klientskou aplikací. [11]

Interakce s účetní knihou

Chytré kontrakty mohou programově přistupovat k oběma částem účetní knihy. Mohou provádět operace vložení, získání, modifikace a odstranění stavu. Dále se také mohou dotazovat na neměnný záznam transakcí v blockchainu, jelikož blockchain neměnně zaznamenává transakce, které aktualizují stavy v účetní knize (tzn. ve všech případech, kdy transakce vytvářejí, čtou, aktualizují nebo odstraňují businness objekty ve světovém stavu). [11]

5.1.7 Průběh transakce

Průběh vykonávání transakce lze popsat v několika krocích uvedených níže.

Návrh transakce a schválení

Aplikace odešle návrh transakce každému uzlu, který je součástí zásady schvalování. Každý z těchto uzlů provede nezávisle na sobě transakci a změny stavu účetní knihy jsou zachyceny jako tzv. „odpověď na návrh transakce“, která obsahuje stavy, které již byly přečteny, a nové stavy, které mají být zapsány, pokud bude transakce platná. Tyto uzly neprovádí žádné změny do účetní knihy, pouze simulují transakci a odesílají zpět do aplikace odpověď na návrh transakce. Krok končí, když aplikace má všechny odpovědi

na návrhy transakcí vyžadované zásadou schvalování. V případě, že uzly pošlou zpět do aplikace různé a nekonzistentní odpovědi na návrhy transakcí, aplikace požádá o nové odpovědi. [11]

Uspořádání transakcí

Jakmile má aplikace všechny požadované odpovědi na návrh transakce, odešle návrh transakce spolu se všemi zásadami řadícímu uzlu. Jakmile řadící uzel nashromáždí dostatek transakcí, uspořádá je a vytvoří blok připravený k odeslání zpět všem potvrzujícím uzlům. [11]

Validace

Řadící uzel pošle připravený blok každému potvrzujícímu uzlu v síti. Každý uzel nezávisle zpracovává každou transakci v rámci bloku a kontroluje, že je splněna zásada schválení a že nedošlo ke změnám stavu účetní knihy pro sadu přečtených proměnných od doby, kdy byla tato sada generována při provádění transakce. Pokud transakce splňuje oba požadavky, je označena za platnou. V případě, že je jeden z uvedených požadavků nesplněn, transakce je neplatná. [11]

5.1.8 Privátní data

V případech, kdy skupina organizací na kanálu potřebuje uchovat data soukromá od jiných organizací na stejném kanálu, lze vytvořit nový kanál obsahující pouze organizace, které potřebují přístup k těmto datům. Nicméně, vytváření samostatných kanálů v každém z těchto případů vytváří další administrativní režii. Navíc, není možné, aby všichni účastníci kanálu viděli transakci a zároveň část dat byla soukromá.

Alternativním řešením je vytváření tzv. privátních datových kolekcí. Tento způsob umožňuje definované podskupině organizací na kanálu schvalovat, potvrzovat nebo dotazovat se na soukromá data, aniž by bylo nutné vytvářet samostatný kanál. Další nespornou výhodou tohoto řešení je možnost definovat životnost uložených dat, takže data po určitém (zvoleném) počtu provedených transakcí zmizí. Lze tak snáze sdílet větší objemy dat, protože z dlouhodobého hlediska nedochází k zaplňování úložiště jako by tomu bylo v případě uložení těchto dat do blockchainu, kde by záznam těchto dat zůstal a zaplňoval tak úložiště. [18]

Každá kolekce se skládá ze dvou částí - dat a kontrolního součtu (hashe) těchto dat.

Data

Tato data jsou uložena v soukromé stavové databázi (někdy nazývané „postranní databáze“ nebo „SideDB“) na uzlech autorizovaných organizací, k nimž lze na těchto autorizovaných uzlech přistupovat z chaincodu. Řadící služba zde není zapojena a nevidí soukromé údaje. [18]

Hash dat

Kontrolní součet dat je schvalován, uspořádán a zapsán do kopie účetní knihy každého uzlu na kanálu. Hash slouží jako důkaz existence transakce a je používán pro ověření stavu a může být použit pro účely auditu.

Autorizované organizace se mohou rozhodnout sdílet soukromé údaje s ostatními stranami. Třetí strana pak může vypočítat hash soukromých dat a zjistit, zda odpovídá stavu v účetní knize kanálu, což dokazuje, že stav existoval mezi členy kolekce v určitém časovém okamžiku. [18]

6 Dodavatelský řetězec

6.1 Definice dodavatelského řetězce

Dodavatelský řetězec (Supply chain, SC) je definován jako síť organizací, které jsou přímo či nepřímo zapojeny do uspokojování požadavků zákazníka. SC může obsahovat všechny činnosti, které přeměňují suroviny na konečné výrobky a dodávají je zákazníkům. Zahrnuje tedy nejen výrobce a dodavatele, ale také skladové prostory, distribuční centra, maloobchodníky i samotné zákazníky. [32]

Na dnešním vysoce konkurenčním a komplexním trhu může mít společnost s efektivnějším dodavatelským řetězcem více výhod než její konkurenti. Řízení dodavatelského řetězce jako zdroj konkurenční výhody se tak stalo velkou výzvou pro společnosti v různých průmyslových odvětvích. [45]

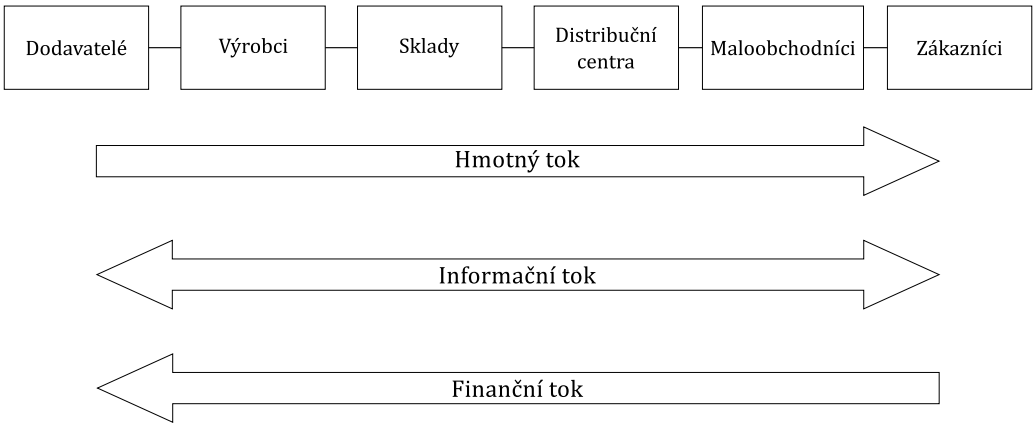
6.2 Definice řízení dodavatelského řetězce

Profese v oblasti řízení dodavatelského řetězce (Supply Chain Managment, SCM) se neustále mění a vyvíjí tak, aby vyhovovaly potřebám rostoucího globálního dodavatelského řetězce. Vzhledem k tomu, že dodavatelský řetězec pokrývá širokou škálu oborů, může být definice toho, co je dodavatelským řetězcem, nejasná. SCM může být často zaměňováno s termínem logistika.

Podle oficiální definice se řízení dodavatelského řetězce definuje jako úloha zahrnující plánování a řízení všech činností souvisejících se získáváním a zadáváním zakázek, konverzí a všemi činnostmi v oblasti řízení logistiky. Důležité je také to, že zahrnuje koordinaci a spolupráci s partnery, kterými mohou být dodavatelé, zprostředkovatelé, poskytovatelé služeb třetích stran a koncoví zákazníci. [62]

6.2.1 Toky v dodavatelském řetězci

Jak lze vidět na obrázku 6.1, dodavatelský řetězec je soubor organizací přímo propojených jedním nebo více toky produktů, financí nebo informací mezi dodavateli a zákazníky. Řízení dodavatelského řetězce je řízení takového řetězce. [45]



Obrázek 6.1: Struktura dodavatelského řetězce.

Hmotný tok

Tok fyzického produktu od dodavatele až po zákazníka. Tento tok je obvykle jednosměrný, to znamená, že proudí pouze jedním směrem od dodavatele k zákazníkovi. V některých případech, když zákazník vrátí produkt, může být tok obousměrný. Typický tok materiálů obvykle začíná dodavateli surovin výrobcům do skladů a distribucí ke konečnému zákazníkovi. [26]

Informační tok

Informační tok je tok informací od dodavatele k zákazníkovi a od zákazníka zpět k dodavateli. Tento tok je obousměrný, to znamená, že jde oběma směry v dodavatelském řetězci. Mezi informace, které se pohybují mezi zákazníky a dodavateli, patří nabídky, objednávky, stav doručení, faktury, stížnosti zákazníků atd. [26] Jedním ze způsobů komunikace mezi dvěma subjekty je elektronická výměna dat (EDI, Electronic Data Interchange)[1].

V mnoha případech jsou do informační sítě zapojeni další partneři, jako jsou distributoři, obchodníci, maloobchodníci a poskytovatelé logistických služeb. [26]

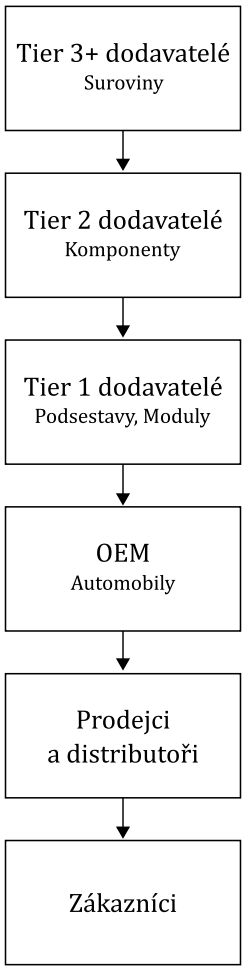
Finanční tok

Finanční tok zahrnuje pohyb peněz od zákazníka k dodavateli. Obvykle, když zákazník obdrží produkt a ověří jej, zaplatí a peníze putují zpět k dodavateli. Finanční prostředky někdy obíhají jiným směrem (od dodavatele k zákazníkovi), např. v případě vrácení peněz při reklamaci. [26]

[1] https://web.archive.org/web/20080511043940/https://www.itl.nist.gov/fipspubs/fip161-2.htm

6.3 SCM v automobilovém průmyslu

V případě automobilového průmyslu se struktura dodavatelského řetězce liší. SC v automobilovém průmyslu se dělí na dodavatele, výrobce originálního vybavení (OEM[2]), prodejce a distributory, viz obr. 6.2. Další část, která není v dodavatelském řetězci pro výrobu automobilových vozidel, ale přesto je s ní spojená, je trh s náhradními díly (tzv. aftermarket). [2]



Obrázek 6.2: Struktura dodavatelského řetězce v automobilovém průmyslu dle statistiky trhu od Standard & Poor's.

[2]OEM (zkratka anglického Original Equipment Manufacturer) je obchodní termín, který označuje výrobce zařízení, jehož výrobek je prodáván a propagován jinou obchodní značkou. [63]

Hierarchie dodavatelů je typicky rozdělena do 3 úrovní, ale obecně těchto úrovní může být i více. V hierarchii jsou dodavatelé obvykle odstupňováni z pohledu výrobce:

Tier 1 dodavatelé - Celosvětoví výrobci hotových modulů a dílů pro konečnou montáž (např. palubní desky, motory, sedadla atd.) s vlastní výrobní nebo montážní kapacitou nacházející se převážně v blízkosti závodu automobilek s ohledem na způsob zásobování JIS (just-in-sequence) nebo JIT (just-in-time).

Tier 2 dodavatelé - Globální nebo regionální společnosti s vlastní výrobou nebo montážními závody, které se zřizují poblíž dodavatelů první úrovně. Dodávají komponenty a menší moduly dodavatelům první úrovně (např. svařená konstrukce sedadla).

Tier 3 dodavatelé - Výrobci surovin a společnosti s výrobními kapacitami pro malé jednoduché díly a jednotlivé komponenty (např. plastové díly, kovové díly, hliníkové díly), které splňují podmínky kvality a objemu u dodavatelů druhé úrovně. Někteří dodávají materiál přímo pro dodavatele první úrovně (např. svinutý plech).

V případě trhu s náhradními díly je rozdíl v tom, že servisní díly a příslušenství mají oddělené dodavatelské řetězce, přesto se ale prodávají v autorizovaných prodejnách. Díly tak mohou pocházet od dodavatelů součástek, jakož i od OEM. [2]

6.3.1 Řízení zásob

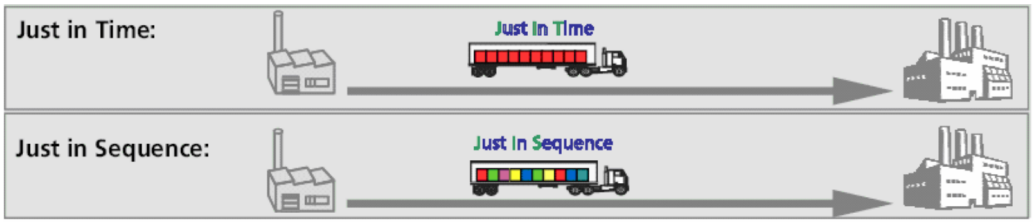
Jak bylo již popsáno obecně na obr. 6.1, tak zde to platí obdobně. Každý člen automobilového dodavatelského řetězce je spojen s ostatními tokem materiálů v jednom směru, tokem objednávek a peněz v druhém směru a koordinován tokem informací. Změna v jakémkoli přenosovém článku řetězu obvykle vytváří vlny vlivu, které se šíří v celém dodavatelském řetězci. Tyto vlny vlivu se projevují v cenách (za suroviny, práci, součástky a hotové výrobky), tocích materiálů, produktů a zásobách. Dodavatelské řetězce se tak stávají řízené především poptávkou než předpovědí (na základě předešlých objednávek), aby mohly účinně reagovat na poptávku výrobců automobilů v reálném čase. Výrobci se tak snaží využívat různé přístupy řízení skladových zásob, aby mohli zajistit, že mají dostatek zásob a splnit současné a očekávané požadavky zákazníků. [43]

Just-in-Time (JIT)

Just-in-Time (JIT) je strategie řízení zásob, která umožňuje zajistit dostatek zásob pro současné i očekávané požadavky zákazníků. V automobilových dodavatelských řetězcích mají komponenty tendenci postupovat vpřed v „pull“ systému[3], který dopravuje zásoby do místa spotřeby, pouze pokud je to potřeba. Po dodání budou komponenty ve skladu výrobců až do finální montáže. Metoda JIT primárně snižuje zásoby a tím i náklady. Relevantními faktory jsou zde například méně vázaný kapitál, menší náklady na skladování, menší administrativní náklady pro sklady, menší opotřebení materiálu a zastarávání. [43]

Just-in-Sequence (JIS)

V moderním automobilovém průmyslu nestačí, aby dodavatelé byli schopní dodávat komponenty včas, ale je také důležitá posloupnost dodávek. Kvůli tomuto požadavku je třeba dodávat díly v součinnosti s procesem montáže. Tento způsob dodávek se nazývá Just-in-Sequence (JIS).



Obrázek 6.3: Srovnání dvou strategií pro řízení zásob v dodavatelském řetězci. [43]

JIS strategie funguje tak, že komponenty od dodavatelů dorazí na montážní linku v konkrétním okamžiku, kdy jsou potřeba, nikoli dříve, což je případ JIT. Každá komponenta dorazí ve správný čas, ve správném pořadí a v odpovídající verzi. Vzhledem k tomu, že komponenty dorazí ve správný okamžik, pracovníci na montážní lince berou komponenty přímo z kontejnerů nebo palet a instalují do finálního produktu, bez skladování nebo třídění.

[3]Strategie tahu souvisí se správou inventury Just-in-Time, která minimalizuje zásoby na skladě, se zaměřením na dodávky na poslední chvíli. V rámci této strategie vstupují produkty do dodavatelského řetězce, pokud to odůvodňuje poptávka zákazníků.

To umožňuje, aby byl v místě montáže k dispozici malý počet součástek, s malým množstvím bezpečnostních zásob pro použití v nouzových situacích (např. poškození součástky během montáže). Bez sekvencování by bylo potřeba skladovat každou možnou variantu komponenty ve výrobní oblasti závodu OEM. [43] Obrázek 6.3 ukazuje rozdíl v organizaci procesu produkce mezi strategií JIT a JIS.

7 Návrh řešení

Tato kapitola se zabývá návrhem vlastního řešení systému založeného na technologii blockchain pro správu dodavatelského řetězce. Vlastní řešení se zaměřuje na oblast logistiky v automobilovém průmyslu s cílem vytvořit prototyp systému, který umožní prokázat kvalitu dodávaných součástek a zvýšit důvěru mezi zástupci v dodavatelském řetězci.

Existuje mnoho způsobů, jak měřit kvalitu vyrobených součástek, např. formou měření výsledných rozměrů součástky nebo zaznamenáváním použité teploty a tlaku při výrobě, atd. To všechno jsou faktory ovlivňující výslednou kvalitu. Takovéto hodnoty jsou ale problematické, protože je lze snadno zaměnit za korektní, které splňují požadavky na kvalitu součástky. Z toho důvodu bude navržený systém ukládat kromě naměřených hodnot také fotografie výsledných výrobků jako důkaz dobré výstupní kvality. Z datových souborů fotografií bude systém počítat kontrolní součty a ty následně ukládat v síti blockchain. Tímto způsobem bude následně možné zpětně fotografii zkontrolovat a ověřit její pravost vypočtením kontrolního součtu a jeho porovnání se součtem uloženým v síti blockchain.

7.1 Související práce

Většina souvisejících prací byla již zmíněna v oddíle 4.4. V této části jsou zmíněné další práce, které jsou relevantní pro toto řešení.

První ze zmíněných prací je projekt automobilky BMW a poskytovatele logistických služeb DHL zaměřující se na koncept řešení sítě blockchain pro řízení dodavatelského řetězce v Asii a Tichomoří s cílem poskytnout lepší trasovatelnost součástek dodávaných z Malajsie. [37] Bližší technické detaily nejsou veřejně známé.

Další studie se zabývá návrhem řešení pro řízení kvality dodavatelského řetězce s využitím technologie blockchain. [31] Autoři uvádějí víceúrovňové řešení, kdy na nejnižší úrovni dochází ke sběru dat z výroby pomocí IoT sensorů. Data se následně ukládají do vyšší vrstvy do sítě blockchain. Nad touto vrstvou je poté vrstva chytrých kontraktů, které zajišťují autorizaci, řízení a kontrolu kvality. Celý systém zastřešuje business vrstva pro rozhodovaní o nákupních a výrobních aktivitách.

Podobná studie byla provedena pro obecnější problém týkající ověření obsahu videa pomocí technologie blockchain. [65] Autoři uvedli dva přístupy k ověření přijatých videí z mobilních zařízení. První navrhovaný algoritmus používá postup zadávání kódu pomocí pohybu mobilního fotoaparátu, druhý pak informace z různých mobilních senzorů jako akcelerometr, gyroskop, barometr a GPS. Následně je hash vypočtený z video souboru uložen do blockchainu. Pro ověření pravosti video souboru uživatele se vypočítá jeho hash, který se odešle do blockchainu. Chytrý kontrakt ověří hash a vrátí informace uložené pro tento hash nebo upozorní uživatele, že takový hash nebyl načten. Na základě obdržených informací lze dojít k závěru, zda se jedná o stejné video či nikoliv.

7.2 Volba platformy

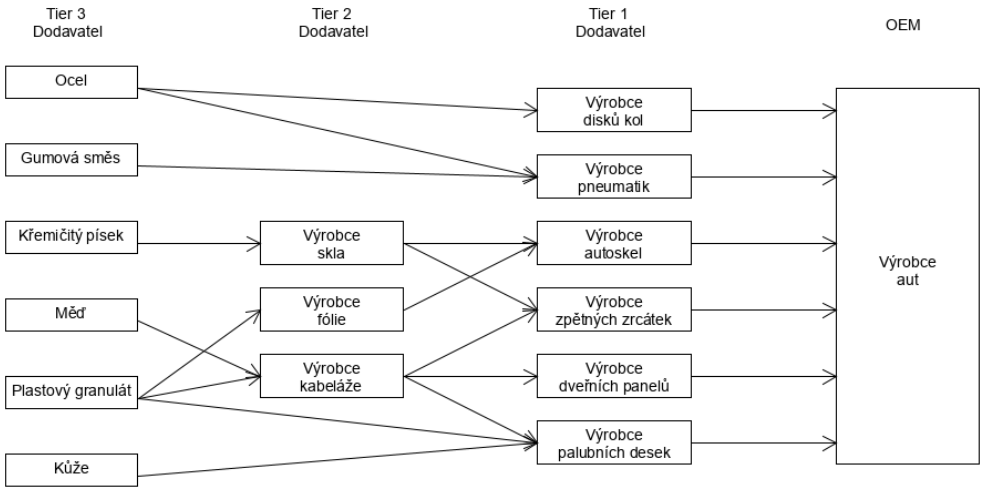
Volba platformy pro navrhovaný systém závisí na několika faktorech. V prvé řadě na typu podporovaných sítí. Je potřeba, aby systém umožňoval konfiguraci oprávnění pro jednotlivé uzly (účastníky dodavatelského řetězce) a zároveň měl co nejmenší režii při vytváření konsenzu, jinými slovy, aby podporoval privátní typ blockchain sítě. Dále je nutné, aby zvolená platforma umožňovala vývoj v programovacím jazyce Java, který je velice rozšířený a není potřeba znalosti jiného/nového jazyka. V neposlední řadě je potřeba, aby licence zvolené platformy umožňovala bezplatný vývoj komerčních aplikací. A na závěr je důležité také zvážit, v jaké fázi vývoje se daná platforma nachází.

Podíváme-li se na všechny tyto faktory a porovnáme je s dostupnými široce využívanými platformami pro vývoj podnikových blockchain aplikací viz tabulka 4.1, docházíme k závěru, že platformy Insolar a Hyperledger Fabric splňují potřebné parametry. Nicméně, přestože platforma Insolar nabízí větší možnosti využití, platforma je stále ve fázi vývoje a v současné době ještě neobsahuje všechny deklarované vlastnosti. Z tohoto důvodu se jako nejvhodnější alternativa nabízí platforma Hyperledger Fabric.

7.3 Modelový dodavatelský řetězec

V této sekci bude popsán návrh dodavatelského řetězce, který slouží k modelování skutečné struktury dodavatelského řetězce v automobilovém průmyslu. Hlavním důvodem k tomuto kroku je potřeba redukovat počet organizací zainteresovaných v dodavatelském řetězci, protože není výjimkou, že automobilky mají stovky až tisíce dodavatelů.

Modelový dodavatelský řetězec uvedený na obrázku 7.1 byl vytvořen na základě veřejně dostupných výrobních procesů různých dodavatelů komponent do aut. Navržený dodavatelský řetězec zahrnuje oblasti výroby disků kol, pneumatik, autoskel, zpětných zrcátek, dveřních panelů a palubních desek.



Obrázek 7.1: Modelový dodavatelský řetězec pro automobilový průmysl. Orientace hrany v modelu udává směr proudění hmotného toku výrobků od výrobce k zákazníkovi.

7.3.1 Definice procesů

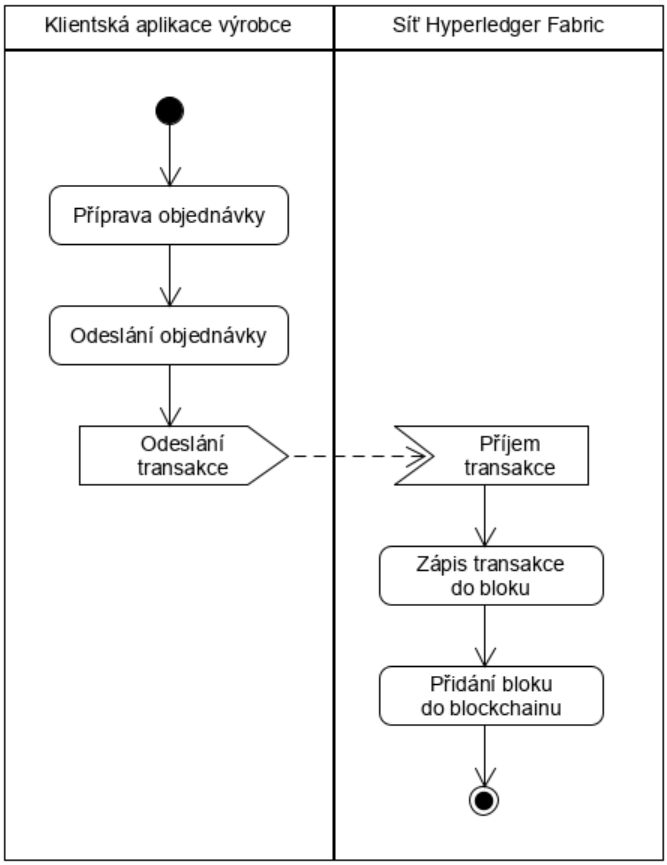
Součástí modelu dodavatelského řetězce bylo i nadefinování procesů mezi zúčastněnými stranami v dodavatelském řetězci. Modelované procesy se soustředí především na výměnu informací mezi organizacemi za účelem ověření kvality.

Pro zjednodušení, jakýkoliv dodavatel nižší úrovně (např. Tier 3) bude dále označen jako „výrobce“ a jakýkoliv dodavatel vyšší úrovně (např. Tier 2) nebo automobilka bude označen jako „zákazník“.

Odeslání objednávky

Proces popisuje způsob výměny informací týkajících se objednávek mezi výrobcem a zákazníkem z pohledu výrobce, viz obrázek č. 7.2. Proces zápisu transakcí byl pro znázornění zjednodušen, kompletní průběh je popsán v kapitole 5.1.7.

Výrobce při odeslání objednávky vytváří transakce s informacemi o produktech, které souvisejí s objednávkou. Tyto transakce jsou následně zapsány do blockchainu. Sdílené informace musejí kromě metadat o produktu obsahovat také identitu výrobce a kontrolní součet (hash) dat pro možnost jejich pozdějšího ověření (např. zákazníkem).

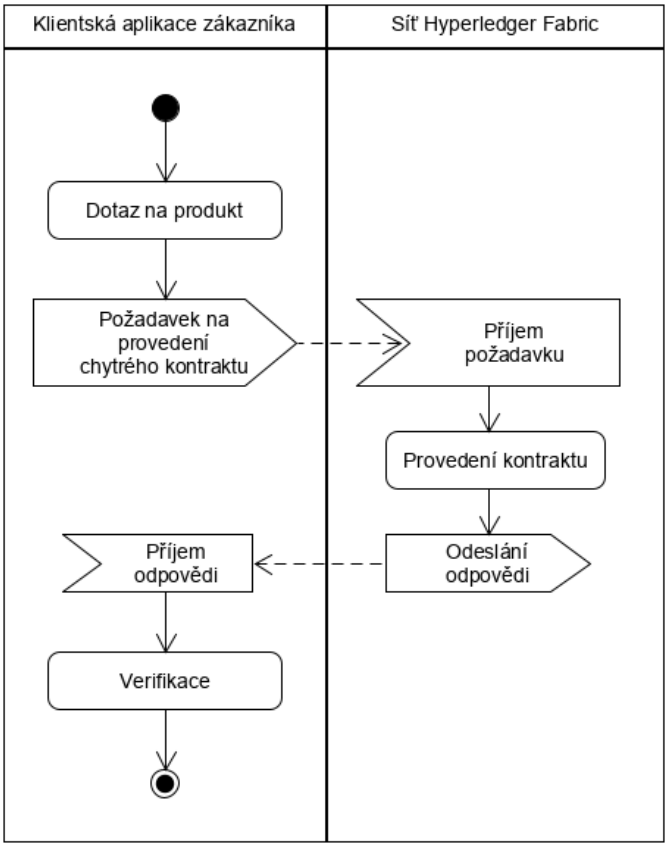


Obrázek 7.2: Proces výměny informací týkajících se objednávek mezi výrobcem a zákazníkem z pohledu výrobce.

7.3.2 Verifikace objednávky

Proces popisuje způsob, jakým může zákazník ověřit věrohodnost a konzistenci dat poskytovaných výrobcem, viz obrázek 7.3.

Zákazník při obdržení objednaných produktů může provést kontrolu objednávky, aby se ujistil, že kvalita produktů odpovídá hodnotám udávaným výrobcem. To lze provést pomocí dotazu na produkty, jejichž metadata jsou uložena výrobcem v blockchainu, a následnému vypočítání jejich hashe, který musí být totožný.



Obrázek 7.3: Proces verifikace objednávek z pohledu zákazníka.

7.4 Architektura systému

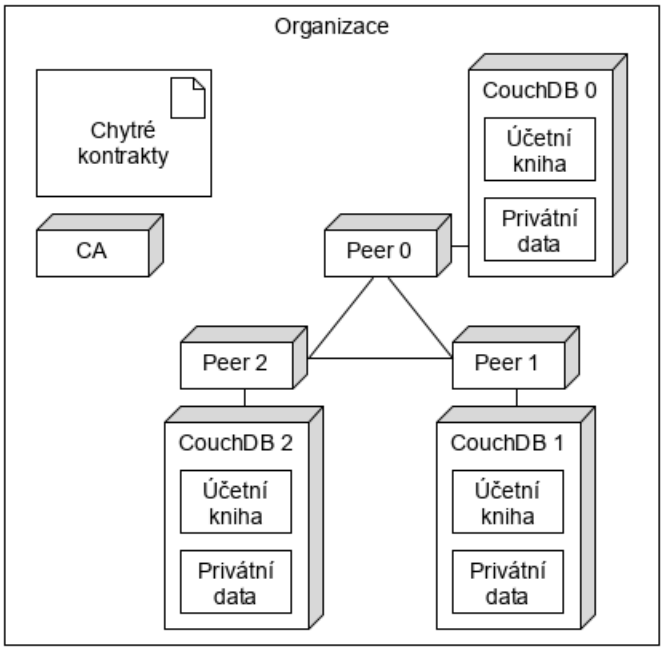
Architektura vytvářeného systému se skládá především ze sítě Hyperledger Fabric a klientské aplikace, pomocí které se lze k síti připojit.

7.4.1 Konfigurace sítě

Jedná se o klíčovou část při návrhu sítě blockchain. Při konfiguraci sítě je potřeba zvážit několik faktorů, které mají vliv na výslednou strukturu sítě. Mezi ty klíčové patří zaměření sítě, počet uzlů v organizaci, použitý algoritmus konsenzu, použití kanálů a privátních datových kolekcí, zvolený typ databáze.

Zaměření sítě a počet uzlů

Struktura sítě blockchain je do velké míry ovlivněna jejím zaměřením, v tomto případě dodavatelským řetězcem v automobilovém průmyslu. Navrhovaná síť bude tedy vycházet z modelového dodavatelského řetězce představeného v kapitole 7.3, kdy každá organizace je vlastníkem hned několika typů uzlů - potvrzovacího, schvalovacího, certifikačního a databázového. V síti se dále vyskytují uzly řadící služby, které jsou popsány v sekci 7.4.1, a může tam být také uzel příkazové řádky pro konfiguraci sítě.



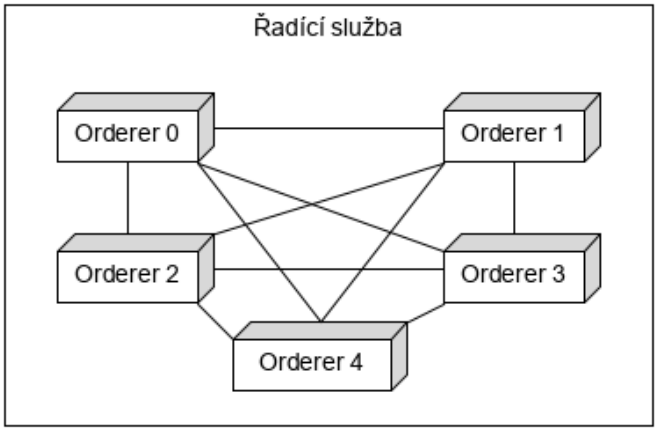
Obrázek 7.4: Ukázka struktury uzlů organizace.

Pro účely této práce bude každá organizace vlastnit 3 potvrzovací uzly, které zároveň budou sloužit pro schvalování transakcí. Organizace tak bude schopna schvalovat transakce při výpadku až 2 uzlů. Zároveň ke každému

uzlu bude náležet jeden databázový uzel, v němž bude uložena kopie účetní knihy a privátní data. Organizace dále bude vlastnit ještě jeden certifikační uzel, který bude sloužit ke generování a spravování certifikátů organizace, viz obrázek 7.4.

Algoritmus konsenzu a řadící služba

Součástí konfigurace sítě blockchain je i volba konsenzuálního algoritmu. Platforma Hyperledger Fabric aktuálně podporuje dvě varianty implementace řadící služby, které jsou vhodné do produkčního prostředí - Raft a Kafka. Tyto implementace vychází z distribuovaných konsenzuálních algoritmů, přičemž implementace Raft protokolu je mnohem vyspělejší a snadněji konfigurovatelná než je tomu v případě Kafky, proto byla tato implementace použita pro účely této práce.



Obrázek 7.5: Ukázka struktury uzlů řadící služby.

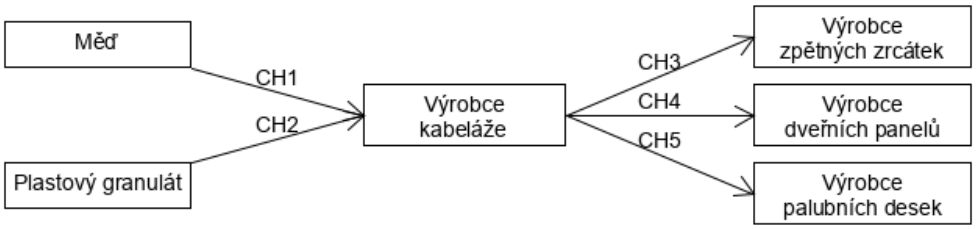
Řadící služba je v sítí reprezentována (na doporučení vývojářů platformy[1]) jako samostatná organizace s vlastními uzly, aby se předešlo situaci, ve které bude potřeba vymezit část uzlů pro potvrzování/schvalování a část pouze pro řazení transakcí. Aby byla řadící služba odolná proti poruchám, navrhované řešení počítá s pěti řadícími uzly, jak je možné vidět na obrázku 7.5.

[1]https://hyperledger-fabric.readthedocs.io/en/release-1.4/Fabric-FAQ.html

Sdílení a ukládání dat

Dalším důležitým krokem při návrhu sítě Hyperledger fabric je definování způsobu sdílení a ukládání dat. Platforma Hyperledger Fabric umožňuje dvojí způsob sdílení dat v rámci kanálu (viz kapitola 5.1.8). Každý způsob má své výhody a nevýhody. V případě dodavatelského řetězce v automobilovém průmyslu lze předpokládat, že dvě organizace, které spolu mají uzavřený kontrakt, budou mít vyhrazený kanál pro komunikaci. Data postačí sdílet skrze účetní knihu v blockchainu, kde provedené transakce budou v rámci kanálu dostupné všem (oběma organizacím). Nicméně mohou existovat výjimky, kdy více organizací chce mít vyhrazen společný kanál nebo jen organizace chtějí sdílet větší objemy dat. V této situaci je vhodné, aby síť umožňovala využití privátních kolekcí, které jsou pro tyto účely mnohem lépe uzpůsobené. Zároveň bude potřeba změnit typ používané databáze na CouchDB, jelikož původní databáze LevelDB, kterou platforma Hyperledger Fabric využívá pro ukládání párů klíč-hodnota, nepodporuje privátní data.

V rámci návrhu je dále potřeba definovat zásady schvalování transakcí, které jsou potřebné při definování kanálů a privátních kolekcí. Jejich účel je popsán v kapitole 5.1.1 u schvalovacího uzlu.



Obrázek 7.6: Ukázka způsobu sdílení dat v kanálech. Jeden kanál slouží výrobci kabeláže pro ověření údajů o přijatých výrobcích od dodavatele mědi a plastového granulátu a zbylé kanály slouží pro sdílení dat o jeho výrobcích určených pro zákazníky vyšší úrovně.

V případě modelového dodavatelského řetězce představeného v kapitole 7.3 je navrhované řešení takové, že každé dvě organizace budou mít vyhrazený vlastní kanál pro sdílení dat. Každá organizace bude tak součástí nejméně jednoho kanálu, viz obrázek 7.6. Vedle kanálu budou mít organizace k dispozici také privátní kolekce dat, které jsou v tomto případě navržené především pro dočasná a velkoobjemová data, aby se předešlo problému s ukládáním větších objemů dat do účetní knihy v blockchainu.

Aby bylo možné označit transakci jako validní, budou mít oba způsoby sdílení dat nastavené zásady vyžadující podpis od obou organizací, jinak bude transakce při potvrzování označena za nevalidní.

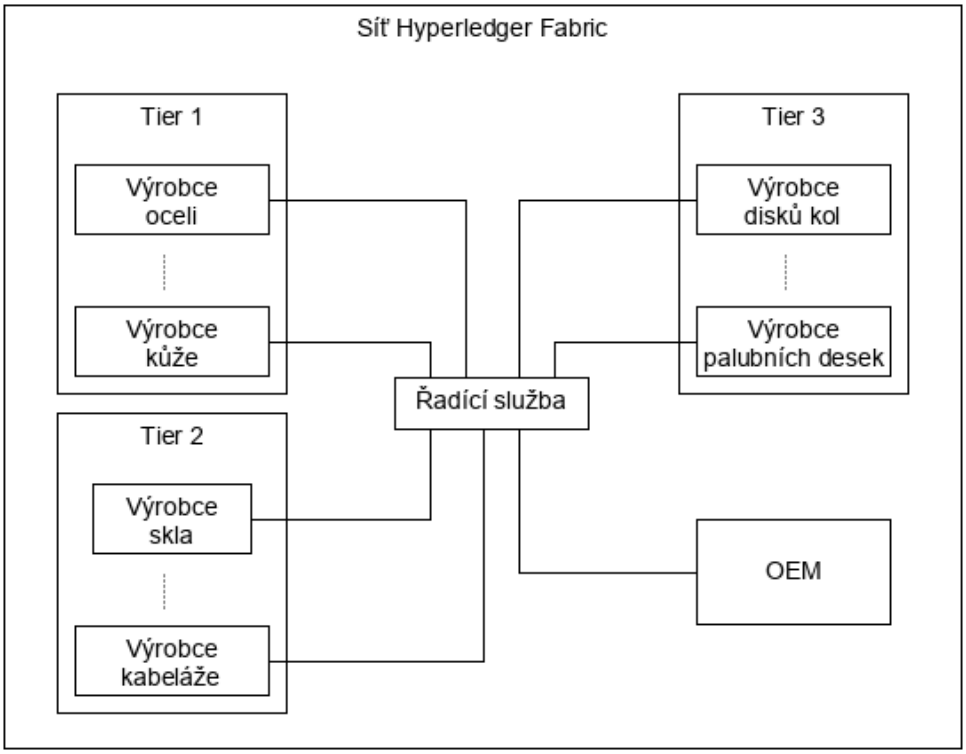
7.4.2 Klientská aplikace

Klientská aplikace aplikace slouží uživatelům k připojení do sítě blockchain skrze uzly organizace a umožňuje uživatelům spouštět chytré kontrakty, viz kapitola 7.3.1. Navrhovaná klientská aplikace bude fungovat ve 2 režimech - režimu příkazové řádky a režimu webové aplikace. První režim bude sloužit především jako pomocný nástroj pro hromadné provádění transakcí a druhý jako uživatelská aplikace. Implementační detaily budou popsány v kapitole 8.3.

8 Implementace

Tato kapitola má za cíl popsat implementaci systému, který byl v rámci této práce vytvořen. Systém se skládá ze tří částí - sítě Hyperledger Fabric, chytrých kontraktů a klientské aplikace, viz obrázek 8.1. Součástí implementace bylo také vytvoření pomocného generátoru konfigurace, který usnadňuje vytvoření takto rozsáhlé sítě. Při implementaci byl využit framework Hyperledger Fabric ve verzi 1.4.7, který je na rozdíl od verze 2.x lépe odladěný a má lepší podporu ze strany komunity.

V rámci implementace bude framework Hyperledger Fabric dále označován pouze jako framework.



Obrázek 8.1: Vysokoúrovňová architektura vytvořeného systému. Jednotlivé organizace reprezentují soubor vlastněných uzlů znázorněných v diagramu 7.4. Stejně tak řadící služba představuje soubor řadících uzlů uvedených v diagramu 7.5.

8.1 Síť Hyperledger Fabric

V této části bude zaměřena pozornost na implementaci samotné sítě blockchain pomocí zvoleného frameworku s přesahem do chytrých kontraktů v rámci spuštění sítě. Veškeré soubory týkající se implementované sítě jsou umístěny v adresáři `network/` na přiloženém DVD.

Na začátek je potřeba připomenout, že framework je založen na principu Docker kontejnerů, které je nutné stáhnout společně s binárními soubory frameworku, aby bylo možné síť spustit a nastavit. Celý postup přípravy, instalace a následného spuštění sítě je popsán v příloze B.

8.1.1 Konfigurace sítě

Jak již bylo řečeno, použitý framework je postaven na Docker kontejnerech. Tyto kontejnery představují základní stavební kameny sítě blockchain a využívají se při její konfiguraci. Konfigurace sítě blockchain se provádí definováním jednotlivých služeb do konfiguračních YAML souborů. Pod službou si lze představit nastavení konkrétního uzlu, CA nebo databáze. Následně je možné nadefinované služby vytvořit a spustit nástrojem Docker Compose[1].

V případě implementovaného řešení byly konfigurovány služby pro uzly organizací, řadící služby, certifikační autority, databáze a příkazovou řádku pro nastavení sítě po spuštění. Součástí konfigurace jsou i soubory s nastavením zásad sítě a nastavením pro generování kryptografických materiálů (klíčů a certifikátů). Následující seznam shrnuje potřebné konfigurační soubory pro spuštění sítě blockchain:

base/peer-base.yaml Základní konfigurační soubor uzlů sítě, který je děděn a rozšiřován dalšími konfiguračními soubory.

base/docker-compose-base.yaml Rozšiřující konfigurační soubor uzlů sítě, který je využíván zbylými konfiguračními soubory.

configtx.yaml - Konfigurační soubor definující zásady sítě, kanály, MSP, konsenzuální algoritmus a řadící službu.

crypto-config.yaml - Konfigurační soubor pro nástroj Cryptogen, který generuje kryptografický materiál pro entity definované v souboru. Je

[1] https://docs.microsoft.com/cs-cz/azure/cognitive-services/containers/docker-compose-recipe

určen především do testovacího prostředí a generuje certifikáty pro CA před tím než je CA spuštěna.

docker-compose-ca.yaml Konfigurační soubor s nadefinovanými certifikačními autoritami pro všechny organizace. Při spouštění jsou do konfigurace přidány vygenerované certifikáty pomocí spouštěcího skriptu.

docker-compose-cli.yaml Konfigurační soubor pro konfiguraci pomocného nástroje, který usnadňuje konfiguraci spuštěné sítě. Bez této služby by bylo nutné se připojit na jednotlivé uzly a provést nastavení zvlášť (např. vytvoření kanálu).

docker-compose-couch.yaml Konfigurační soubor definující CouchDB databázi pro každý potvrzovací/schvalovací uzel.

docker-compose-etcdraft2.yaml Konfigurační soubor obsahující konfiguraci řadících uzlů.

Pro ukázku je přiložena část konfiguračního souboru z počátečního konceptu sítě, viz ukázka 8.1.

```
peer0.org1.example.com:
 container_name: peer0.org1.example.com
 extends:
   file: peer-base.yaml
   service: peer-base
 environment:
 - CORE_PEER_ID=peer0.org1.example.com
 - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
 - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
 - CORE_PEER_CHAINCODEADDRESS=peer0.org1.example.com:7052
 - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
 - CORE_PEER_LOCALMSPID=Org1MSP
 volumes:
 - /var/run/:/host/var/run/
 ports:
 - 7051:7051
```

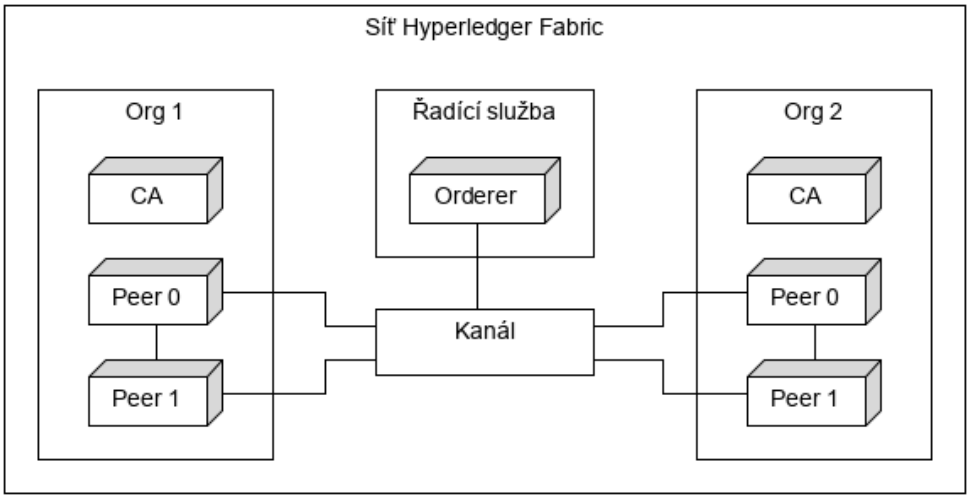
Listing 8.1: Ukázka konfigurace prvního uzlu organizace Org1. Jedná se o část konfiguračního souboru `docker-compose-base.yaml` z počátečního konceptu sítě. Rozšíření zděděné konfigurace přidává nastavení portu, mapování adresářů a adresy, na kterých budou služby uzlu dostupné.

Pro představu, pokud by síť obsahovala 3 organizace a každá z nich by vlastnila 2 potvrzovací/schvalovací uzly, je potřeba definovat explicitně 6 těchto služeb (pro každý uzel zvlášť) pouze v tomto konfiguračním souboru. Podobnou úvahou vznikla myšlenka vývoje sítě ve 2 fázích. Nejprve byl vytvořen koncept sítě v malém měřítku (viz kapitola 8.1.2) a následně, když bylo vše odladěno, bylo implementované řešení aplikováno na modelový dodavatelský řetězec z kapitoly 7.3.

Za tímto účelem byl v průběhu vývoje vytvořen nástroj pro generování konfigurace, který bude blíže specifikován v kapitole 8.1.4. Tato iniciativa vznikla na základě potřeby automatizace procesu konfigurace sítě a spouštěcích skriptů, jelikož manuální úprava se postupně stala velmi zdlouhavou a často nekonzistentní (např. změna názvu uzlu nebyla provedena ve všech konfiguračních souborech a spouštěcích skriptech).

8.1.2 Koncept sítě

V rámci implementace byl kvůli snazší konfiguraci a rychlejšímu spouštění sítě nejprve vytvořen menší koncept sítě, jehož základem byly 2 organizace (výrobce a zákazník). Každá z těchto organizací vlastnila 2 potvrzovací/schvalovací uzly. Síť dále obsahovala jeden řadící uzel a pomocnou službu pro nastavení sítě. Diagram původní sítě je možné vidět na obrázku 8.2.



Obrázek 8.2: Počáteční koncept sítě bez databáze a záložních řadících uzlů.

Konfigurace CouchDB

Po zprovoznění konceptu sítě byl do sítě nainstalován chytrý kontrakt, který umožňoval zápis a čtení řetězců z účetní knihy kanálu (podrobnosti ohledně vývoje a nasazení chytrých kontraktů budou popsány v kapitole 8.2). Zapsané údaje byly uchovávány v původní databázi LevelDB, která umožňovala uchovávat data v páru klíč-hodnota, což vedlo k problémům, protože nebylo možné vyhledávat údaje jinak než podle klíče. Dalším krokem tak bylo nahrazení původní databáze NoSQL databází CouchDB. Jedná se o alternativu k LevelDB databázi, která je též podporována ze strany frameworku a byla by stejně potřeba pro privátní datové kolekce.

Aby databáze fungovala korektně, bylo potřeba vytvořit již zmíněný konfigurační soubor `docker-compose-couch.yaml` a pro každý uzel definovat jednu databázi, viz ukázka kódu 8.2.

```
couchdb0:
  container_name: couchdb0
  image: hyperledger/fabric-couchdb
  environment:
  - COUCHDB_USER=admin
  - COUCHDB_PASSWORD=admin
  ports:
  - 5984:5984
  networks:
  - network
peer0.org1.example.com:
  environment:
  - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
  - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=
                                        couchdb0:5984
  - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=admin
  - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=admin
  depends_on:
  - couchdb0
```

Listing 8.2: Ukázka konfigurace databáze pro jeden uzel organizace Org 1 v počátečním konceptu sítě blockchain.

Konfigurace privátních datových kolekcí

Dalším krokem při konfiguraci konceptu sítě bylo přidání podpory pro privátní datové kolekce. Jedná se o poměrně jednoduchý krok, kdy je potřeba definovat konfigurační soubor `collections_config.json`, ve kterém jsou uložené konfigurace jednotlivých privátních kolekcí, viz ukázka kódu 8.3. U každé kolekce je možné definovat:

- název kolekce
- zásady pro uchovávání dat
- potřebný počet uzlů, na které musí být data distribuována
- maximální počet uzlů, na které mohou být data distribuována
- životnost dat
- pokud je pravda, umožňuje přístup pouze uzlům definovaných organizací

```
[
  {
    "name": "productShared",
    "policy": "OR('Org1MSP.member', 'Org2MSP.member')",
    "requiredPeerCount": 1,
    "maxPeerCount": 3,
    "blockToLive": 0,
    "memberOnlyRead": true
  }
]
```

Listing 8.3: Ukázka konfigurace privátních datových kolekcí z počátečního konceptu sítě.

Když jsou kolekce nadefinovány, je možné z nich číst a zapisovat do nich prostřednictvím chaincode API, které bude popsáno v kapitole 8.2.

Konfigurace konsenzu

Posledním krokem při implementace konceptu sítě byla změna konsenzuálního algoritmu. Síť do té doby využívala nejmenovaný algoritmus Solo (nejde o skutečný konsenzuální algoritmus), který sloužil pouze jako náhrada pro testovací účely a umožňoval rychlejší průběh transakcí. Algoritmus využíval

pouze jeden řadící uzel a uzly nemuseli schvalovat transakce, ty byly rovnou potvrzeny a zapsány do účetní knihy. Místo něj byl použit algoritmus Raft, který je použitelný v produkčním prostředí a používá další řadící uzly pro větší odolnost proti výpadkům. Tyto dodatečné uzly se definují v konfiguračním souboru `docker-compose-etcdraft2.yaml`.

8.1.3 Spouštěcí skripty

Pro účely snadného spuštění sítě bylo vytvořeno několik shellových skriptů. Tyto skripty obsahují řadu příkazů pro spuštění sítě, generování kryptografického materiálu, práci s kanály a pro nasazení chaincodu. Kompletní přehled skriptů je uveden níže:

network.sh - Řídící skript, který nástrojem Cryptogen generuje kryptografický materiál a pomocí příkazu Docker Compose vytváří a startuje služby sítě. Následně spouští konkrétní skripty pro tvorbu kanálů a nasazení chaincodu.

start-network.sh - Skript spouští síť předáním argumentů řídícímu skriptu.

stop-network.sh - Ukončovací skript, který pomocí řídícího skriptu ukončí služby a uklidí pracovní prostor od nepotřebných souborů, které byly při spuštění vytvořeny.

scripts/globals.sh - Obsahuje vygenerované globální proměnné, které využívají skripty při generování kryptografického materiálu, konfiguraci kanálů, nasazení chaincodu.

scripts/chaincode.sh - Skript slouží pro instalaci chytrých kontraktů na uzly organizací dle definovaných zásad a pro vytváření jejich instancí.

scripts/channel.sh - Slouží pro vytváření kanálů a připojování uzlů organizací do kanálů dle zásad. Zároveň definuje opěrné uzly jednotlivých organizací.

scripts/utils.sh - Pomocný skript s utilitami pro ověření výsledku příkazu a nastavení globálních proměnných.

8.1.4 Generátor konfigurace

Konfigurace sítě blockchain se provádí definováním jednotlivých služeb do konfiguračních YAML souborů. Tyto soubory tak obsahují nastavení pro

konkrétní strukturu sítě a jakákoliv změna v samotné struktuře sítě znamená změnu konfiguračních souborů a rozšíření spouštěcích skriptů. V určitém bodě už jsou změny konfigurace tak rozsáhlé, že manuální změna se nevyplatí (především kvůli velké pravděpodobnosti zanesení chyby do konfigurace). Za tímto účelem byl vytvořen generátor konfigurace, aby automatizoval vytváření konfiguračních souborů na základě specifikace struktury sítě.

Generátor byl implementován v jazyce Python ve verzi 3.7 a slouží pro generování YAML konfiguračních souborů představených v kapitole 8.1.1 a pro generování globálních proměnných, viz kapitola 8.1.3. Celý generátor je obsažen v adresáři `generator/` na přiloženém DVD.

Jako předloha pro implementovaný generátor posloužily konfigurační soubory konceptu sítě. Po vzoru těchto konfiguračních souborů byly implementovány jednotlivé generátory souborů. Každý generátor generuje konkrétní konfigurační soubor od základu pouze s využitím informací ohledně struktury sítě, která je popsána v konfiguračním souboru generátoru `config.cfg`, viz ukázka kódu 8.4.

```
[Orderer]
Name = Orderer
ExposedPort = 7050
HiddenPorts = 8050
              9050
              10050
              11050
Count = 5

[Peer]
Names = Org1
        Org2
Ports = 7051
        8051
        9051
        10051
AnchorPeer = peer0
PeersCount = 2
UsersCount = 1
```

Listing 8.4: Ukázka konfigurace řadících a potvrzovacích/schvalovacích uzlů počátečního konceptu sítě v konfiguračním souboru `config.cfg`.

Dodatečně byl implementován generátor globálních proměnných. Současně s tímto rozšířením byly modifikovány spouštěcí skripty, které byly do té doby pouze statické, aby umožňovaly (na základě globálních proměnných) dynamicky generovat kryptografický materiál, konfigurovat kanály a instalovat chytré kontrakty.

Jednotlivé generátory jsou implementovány jako samostatné moduly a každý konfigurační soubory je reprezentován v programu jako JSON objekt sestavený z dílčích částí konfigurace. V případě globálních proměnných je výsledný soubor v programu reprezentován jako seznam vygenerovaných řádků s proměnnými.

Pro generování konfiguračních souborů byl implementován hlavní skript `config_generator.py`, který importuje moduly konkrétních generátorů a postupně je spouští. Výstupy jednotlivých generátorů jsou následně zapsány do příslušných souborů. K zápisu do konfiguračních souborů se využívá balík `PyYaml`, který umožňuje vytvořený objekt uložit ve formátu YAML. V případě globálních proměnných jsou řádky zapsány do souboru pomocí standardních funkcí jazyka.

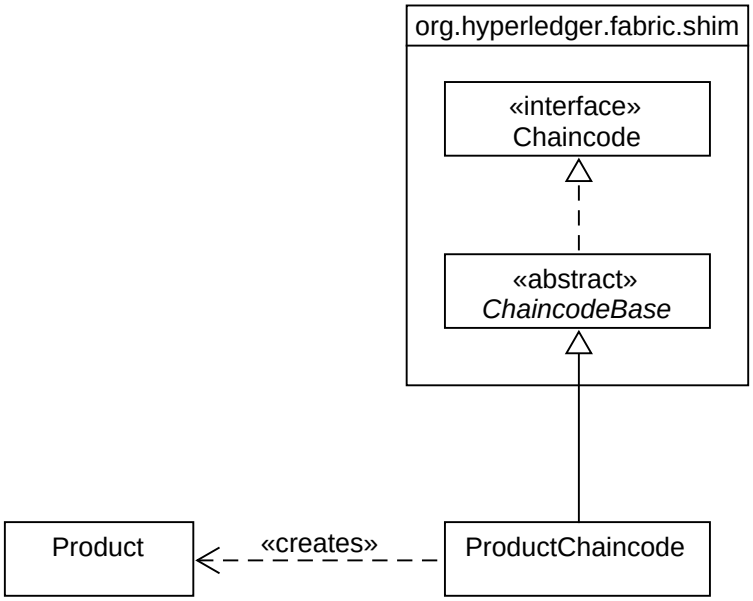
Výstupem programu jsou kompletní konfigurační soubory sítě a soubor s globálními proměnnými pro shellové skripty.

8.2 Chytré kontrakty

Přestože jsou chytré kontrakty instalovány při konfiguraci sítě, z hlediska vývoje jde o samostatný projekt, jehož zdrojové soubory jsou obsaženy v adresáři `chaincode/` na přiloženém DVD. Chytré kontrakty byly implementovány v programovacím jazyce Java ve verzi 1.8.0_212.

Projekt sestává z doménových objektů, které jsou ukládány do účetní knihy sítě blockchain nebo privátních datových kolekcí, a samotných kontraktů. V rámci této práce byl implementován jeden kontrakt pro výměnu informací o produktu. Implementace vychází z návrhu popsaného v kapitole 7.3.1 a konkrétní způsob realizace je představen v diagramu tříd na obrázku 8.3. K implementaci chytrých kontraktů byla využita knihovna `fabric-chaincode-shim`, která poskytuje API pro chytré kontrakty k přístupu ke stavovým proměnným účetní knihy, privátním datovým kolekcím, transakčním kontextům a volání dalších chytrých kontraktů. Implementovaný chytrý kontrakt `ProductChaincode` definuje řadu metod, které prová-

dějí zápis nebo čtení objektů typu `Product`. Metody jsou definovány jak pro interakci s účetní knihou, tak privátními datovými kolekcemi.



Obrázek 8.3: Diagram tříd implementovaného kontraktu pro výměnu informací o produktu.

Zároveň, aby bylo možné chytrý kontrakt nainstalovat, projekt musí být řádně sestaven a mít nadefinovaný manifest, ve kterém je specifikována třída s chytrým kontraktem. To je zde provedeno pomocí nástroje Gradle[2].

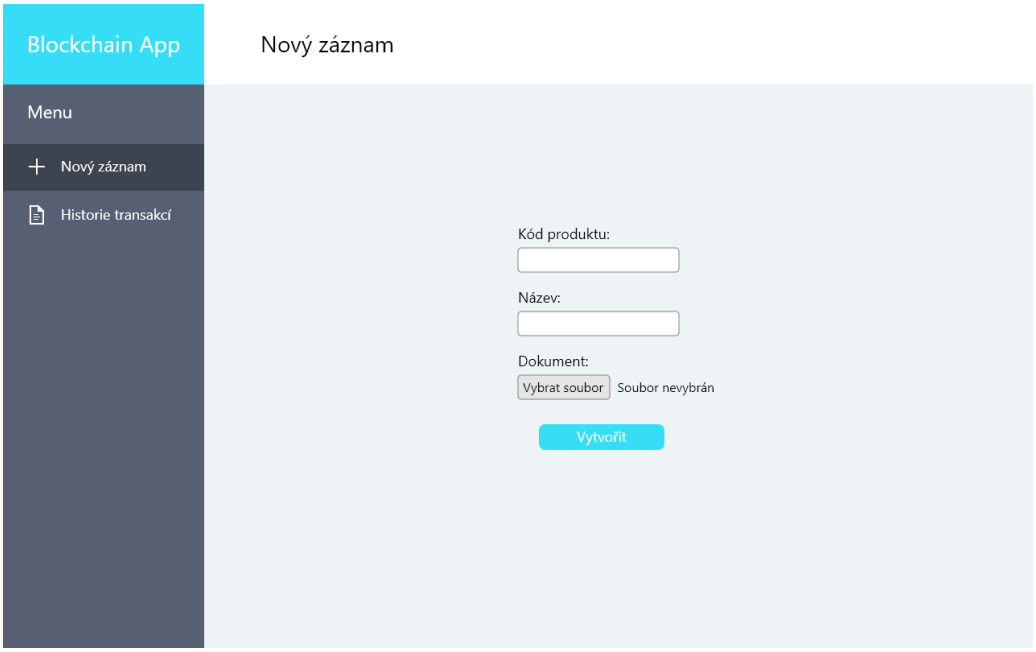
8.3 Klientská aplikace

Klientská aplikace byla implementována v programovacím jazyce Java ve verzi 1.8.0_212. Jedná se o samostatný projekt, který se nachází v adresáři `app/` na přiloženém DVD.

Implementovaná aplikace podporuje 2 režimy spuštění - režim příkazové řádky a režim webové aplikace. V režimu příkazové řádky nemá aplikace žádné grafické rozhraní a je ovládána pomocí parametrů. Tento režim umožňuje vytvářet hromadně transakce a slouží především pro simulaci aktivit ostatních organizací.

[2]https://gradle.org

Druhý režim byl implementován jako webová aplikace pomocí frameworku Spring Boot[3]. V tomto režimu aplikace nabízí jednoduché grafické rozhraní, které bylo navrhnuto v grafickém editoru a následně vytvořeno pomocí nástroje Bootstrap[4]. Návrh designu grafického rozhraní je znázorněn na obrázku 8.4. Webová aplikace umožňuje zápis údajů o produktech, procházení historie transakcí a ověřování údajů proti otisku dat v účetní knize.



Obrázek 8.4: Návrh designu grafického rozhraní. Obrazovka pro přidání nového záznamu.

Při implementaci webové aplikace byl použit architektonický vzor MVC (Model View Controller), který rozděluje aplikaci do tří částí - datového modelu aplikace, uživatelského rozhraní a řídící logiky. Pro připojení do sítě Hyperledger Fabric byla implementována služba `ConnectionManager`, která zprostředkovává přístup k uzlu organizace, pod kterou daný uživatel spadá. K implementaci této služby byla použita knihovna `fabric-gateway-java`, jež umožňuje aplikaci komunikovat se sítí Hyperledger Fabric. Knihovna poskytuje prosté API pro odesílání transakcí do účetní knihy a dotazování se na obsah účetní knihy.

[3]https://spring.io/projects/spring-boot
[4]https://getbootstrap.com/

8.4 Testování

Součástí implementace systému bylo vytvoření testů pro ověření funkčnosti. Byly implementovány tři sady testů - testy pro generátor konfigurace, chytré kontrakty a klientskou aplikaci. Ve všech případech šlo o jednotkové testy, které pokrývaly základní funkčnost.

V případě generátoru konfigurace testování vycházelo z funkčního konceptu sítě, jehož kopie konfiguračních souborů byly použity pro kontrolu správnosti generátorů. K porovnání byla využita knihovna `deepdiff`, která umožňuje porovnat 2 objekty a vrací výsledek v podobě rozdílu. Pokud při porovnání byl výsledkem prázdný objekt, generátor fungoval správně. V opačném případě bylo z výsledného objektu patrné, v jaké části generátoru se stala chyba.

Pro otestování chytrých kontraktů a klientské aplikace byl využit mechanismus mockování objektů, aby bylo možné dílčí metody otestovat bez potřebných objektů, které zajišťovaly komunikaci.

9 Zhodnocení

V rámci této práce bylo cílem zanalyzovat možnosti využití technologie blockchain ve výrobních a logistických procesech. Za tímto účelem bylo vytvořeno PoC (Proof-of-Concept) řešení pro kontrolu kvality dodávaných produktů, které zahrnovalo 3 dodavatele. Následně bylo toto řešení aplikováno na modelový dodavatelský řetězec. Cílem této kapitoly je analýza dosažených výsledků a srovnání s alternativními způsoby (uvedenými v kapitole 4.5), které se v praxi běžně používají.

Dosažené výsledky budou analyzovány z několika hledisek: podle způsobu zajištění důvěry, podle transparentnosti dat a použitelnosti frameworku.

9.1 Zajištění důvěry

Jak již bylo zmíněno výše, implementované řešení přichází se způsobem sdílení dat, který umožňuje výrobcům důvěryhodným způsobem sdílet informace o produktu se zákazníkem. Ať už jsou samotná data sdílena prostřednictvím sítě blockchain nebo prostřednictvím emailu, výrobce může kdykoliv prokázat jejich existenci v určité době a platnost tím, že uloží jejich otisk se základními údaji do účetní knihy sítě blockchain. Stejně tak zákazník může kdykoliv potřebný záznam dohledat a ověřit tak platnost sám.

Jde tak o alternativu k dnes již rozšířeným službám, které poskytují časová razítka nebo certifikáty, kde důvěra v nějaký dokument či data vychází z důvěry v samotnou autoritu, která je vydáváním certifikátů a časových razítek pověřena. V případě blockchainu důvěra v uložené informace plyne z distribuované povahy sítě, která je nezávislá na autoritě, a principu kryptografického zabezpečení účetní knihy a samotných bloků (viz kapitola 4.2). Toto řešení tak má v praxi mnohem větší potenciál, jelikož ke schválení transakce a uložení dat do účetní knihy je potřeba dojít ke konsenzu. To je přístup, který v případě centralizovaných certifikačních autorit chybí a jsme tak odkázáni na důvěru v samotnou autoritu, případně nadřazenou autoritu, která za ni ručí.

9.2 Transparentnost dat

Z hlediska transparentnosti přináší řešení organizacím větší přehled nad sdílenými daty v dodavatelském řetězci. To je umožněno především díky kryptografickému zabezpečení jednotlivých bloků. Nikdo tak nemůže zpětně přepsat část databáze, aniž by se o tom ostatní nedozvěděli, protože každá změna dat je uložena ve formě další transakce. Nelze tedy například zpětně falšovat údaje o již proběhlých transakcích.

Ne vždy je ale naprostá transparentnost výhodou. Proto implementované řešení nabízí možnost v podobě privátních datových kolekcí (viz kapitola 5.1.8), které umožňují organizacím sdílet data jen s určitou podmnožinou organizací nebo jen pouze sami se sebou (např. v případě citlivých údajů nebo know-how).

Kromě zmíněných výhod má implementované řešení jednu poměrně velkou nevýhodu v podobě obtížného trasování původu výrobků. Je to dáno tím, že aktuálně implementované řešení využívá ke komunikaci kanály a privátní kolekce. V síti tak existuje hned několik účetních knih, takže kdokoliv potřebuje zjistit původ výrobku a jeho součástek, musí kontaktovat svého dodavatele, který má k dispozici informace k součástkám, ze kterých byl výrobek vyroben. Možným řešením tohoto problému by bylo přidání dalšího kanálu, do kterého budou mít přístup všichni.

9.3 Použitelnost frameworku

Co se týče použitého frameworku, Hyperledger Fabric nabízí rozsáhlé možnosti konfigurace sítě, které umožňují použití technologii blockchain v různých oblastech podnikání.

Framework dále nabízí podporu různých programovacích jazyků, ve kterých je možné implementovat chytré kontrakty a klientskou aplikaci. Nicméně, kromě jazyka Go, ke kterému existuje celá řada návodů, je podpora ostatních jazyků o poznání nižší a to platí i v případě dokumentace pro vývojáře, kdy je občas nutné funkce ručně otestovat, protože k nim neexistuje úplná dokumentace.

V praxi tak může být z počátku problém s frameworkem pracovat, ale po překonání počátečních problémů dokáže být Hyperledger Fabric přínosným při tvorbě sítě blockchain.

9.4 Budoucí práce

Implementovaný systém by bylo možné dále vylepšit o další chytré kontrakty a postupně tak nahradit fyzické dokumenty a část emailové korespondence mezi organizacemi v dodavatelském řetězci. Proces výměny informací by se tak zrychlil a stal se více transparentní.

Součástí další práce by mohla být i úprava architektury za účelem nasazení systému na více strojů, aby bylo možné otestovat reálnou odezvu a propustnost systému. Za tímto účelem by bylo možné pro nasazení použít Docker Swarm, Kubernetes nebo některou z cloudových služeb, která podporuje framework Hyperledger Fabric. Samotné testování by pak bylo možné provést pomocí vlastního programu nebo pomocí nástroje Hyperledger Caliper[1], což je nástroj pro testování výkonnosti implementací sítí blockchain.

V úvahu připadá také možnost integrace implementovaného systému jako nadstavby pro systém EDI.

[1] https://www.hyperledger.org/use/caliper

10 Závěr

Cílem této práce bylo seznámit se s problematikou sítí blockchain a řízením dodavatelského řetězce za účelem prozkoumání možností jak využít technologii blockchain pro výrobní a logistické procesy. Dalším cílem bylo pro vybraný výrobní či logistický proces navrhnout a implementovat prototyp sítě blockchain a zhodnotit jeho přínosy pro zvolený proces v oblasti řízení dodavatelského řetězce.

V rámci této práce došlo k seznámení se s technologií blockchain a jejími možnostmi využití v rámci automobilového dodavatelského řetězce. Byla provedena analýza dostupných blockchain platforem a z nashromážděných informací byl vybrán Hyperledger Fabric pro implementaci vlastního řešení.

V další fázi práce došlo k seznámení s frameworkem Hyperledger Fabric a byl vytvořen návrh řešení, ve kterém jsou sdílená data zabezpečena otiskem, který se ukládá do účetní knihy sítě blockchain. Lze tak zpětně otestovat pravost přijatých dat porovnáním jejich otisků. Za tímto účelem byl vytvořen jednoduchý koncept sítě, na kterém byla ověřena funkčnost řešení. Následně byla síť překonfigurována, aby používala konsenzuální algoritmus Raft a databázi CouchDB. Tato databáze umožnila implementaci privátních datových kolekcí, které umožňují výměnu objemných a citlivých údajů mezi podmnožinou organizací.

Souběžně s vývojem konceptu řešení vznikla potřeba automatizace procesu konfigurace sítě, protože manuální úprava se postupně stala velmi zdlouhavou a nekonzistentní. Z toho důvodu byl implementován pomocný nástroj, který na základě definované struktury sítě generuje konfigurační soubory a spouštěcí skripty. Později byla pomocí tohoto nástroje síť rozšířena o další organizace, aby lépe odrážela skutečný dodavatelský řetězec.

Nakonec byla implementována plnohodnotná klientská aplikace pro přístup k účetní knize a bylo provedeno zhodnocení přínosnosti implementovaného systému pro automobilový dodavatelský řetězec.

Implementované řešení zvyšuje transparentnost sdílených dat a má potenciál zvýšit důvěru mezi organizacemi v dodavatelském řetězci. Slabým místem jsou nejasné výkonnostní vlastnosti při spuštění na více strojích, jelikož tento scénář nebyl v rámci práce testován.

Seznam zkratek

ACL - Access Control List

API - Application Programming Interface

BFT - Byzantine Fault Tolerance

BGP - Byzantine Generals Problem

CA - Certificate authority

DLT - Distributed Ledger Technology

DNS - Domain Name System

DSL - Domain Specific Language

ECDSA - Elliptic Curve Digital Signature Algorithm

EDI - Electronic Data Interchange

ENS - Ethereum Name Service

ETH - Ethereum

EVM - Ethereum Virtual Machine

GDPR - General Data Protection Regulation

GPS - Global Positioning System

JIS - Just in Sequence

JIT - Just in Time

JSON - JavaScript Object Notation

MSP - Membership Service Providers

OEM - Original Equipment Manufacturer

PKI - Public Key Infrastructure

RFC - Request for Comments

SC - Supply Chain

SCM - Supply Chain Managment

SHA - Secure Hash Algorithm

SSL - Secure Sockets Layer

TLS - Transport Layer Security

TSA - Time Stamping Authority

YAML - YAML Ain't Markup Language

Literatura

[1] IBM Blockchain - Enterprise Blockchain Solutions & Services. Dostupné z: https://www.ibm.com/blockchain.

[2] Industry Surveys Autos & Auto Parts, 2005. Dostupné z: http://docshare03.docshare.tips/files/27311/273119750.pdf.

[3] Blockchain Technology and Applications: Microsoft Azure. Dostupné z: https://azure.microsoft.com/en-us/solutions/blockchain/.

[4] Public key certificate, Aug 2020. Dostupné z: https://en.wikipedia.org/wiki/Public_key_certificate.

[5] What Is a Nonce? A No-Nonsense Dive into Proof of Work, Dec 2018. Dostupné z: https://coincentral.com/what-is-a-nonce-proof-of-work/.

[6] Certification path validation algorithm, Jul 2020. Dostupné z: https://en.wikipedia.org/wiki/Certification_path_validation_algorithm.

[7] Docker (software), Feb 2020. Dostupné z: https://cs.wikipedia.org/wiki/Docker_(software).

[8] What is Ethereum?, 2019. Dostupné z: https://ethereum.org/beginners/.

[9] Hyperledger Fabric Documentation, 2019. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/whatis.html.

[10] Ledger, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html.

[11] Smart Contracts and Chaincode, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/smartcontract/smartcontract.html.

[12] Hyperledger Fabric Functionalities, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/functionalities.html.

[13] Glossary, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/glossary.html.

[14] Identity, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/identity/identity.html.

[15] Membership Service Provider (MSP), July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/membership/membership.html.

[16] Hyperledger Fabric Network, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/network/network.html.

[17] The Ordering Service, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/orderer/ordering_service.html.

[18] Private data, July 2020. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.4/private-data/private-data.html.

[19] Konsorcium, July 2020. Dostupné z: https://cs.wikipedia.org/wiki/Konsorcium.

[20] TradeLens blockchain-enabled digital shipping platform continues expansion with addition of major ocean carriers Hapag-Lloyd and Ocean Network Express, 07 2019. Dostupné z: https://www.maersk.com/news/articles/2019/07/02/hapag-lloyd-and-ocean-network-express-join-tradelens.

[21] A Proof of Stake overview, Jun 2018. Dostupné z: https://medium.com/@poolofstake/a-proof-of-stake-overview-445c52558d03.

[22] Trusted timestamping, May 2020. Dostupné z: https://en.wikipedia.org/wiki/Trusted_timestamping.

[23] Walmart Case Study. Dostupné z: https://www.hyperledger.org/resources/publications/walmart-case-study.

[24] Enabling Trade: Valuing Growth Opportunities. Dostupné z: https://www.weforum.org/reports/enabling-trade-valuing-growth-opportunities.

[25] X.509, Jul 2020. Dostupné z: https://en.wikipedia.org/wiki/X.509.

[26] Aluko, N. – Aluko, N. The three flows of supply chain, Sep 2019. Dostupné z: https://kpakpakpa.com/spotlight/the-three-flows-of-supply-chain/.

[27] Anwar, H. Distributed Ledger Technology: Where Technological Revolution Starts, Jan 2019. Dostupné z: https://101blockchains.com/distributed-ledger-technology-dlt.

[28] Benos, E. – Garratt, R. – Gurrola-Perez, P. The Economics of Distributed Ledger Technology for Securities Settlement. *Ledger*. 2019, 4. doi: 10.5195/ledger.2019.144.

[29] Benčić, F. M. – Podnar Žarko, I. Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, s. 1569–1570, July 2018. doi: 10.1109/ICDCS.2018.00171.

[30] Chase, J. jpmorganchase/quorum, Feb 2020. Dostupné z: https://github.com/jpmorganchase/quorum.

[31] Chen, S. et al. A Blockchain-Based Supply Chain Quality Management Framework. 11 2017. doi: 10.1109/ICEBE.2017.34.

[32] CHRISTOPHER, M. Logistics and Supply Chain Management: Strategies for Reducing Cost and Improving Service (Second Edition). *International Journal of Logistics Research and Applications*. 1999, 2, 1, s. 103–104. doi: 10.1080/13675569908901575. Dostupné z: https://doi.org/10.1080/13675569908901575.

[33] Denis, T. S. – Johnson, S. Chapter 5 - Hash Functions. In Denis, T. S. – Johnson, S. (Ed.) *Cryptography for Developers*. Burlington: Syngress, 2007. s. 203 – 250. doi: https://doi.org/10.1016/B978-159749104-4/50008-X. Dostupné z: http://www.sciencedirect.com/science/article/pii/B978159749104450008X. ISBN 978-1-59749-104-4.

[34] Dinh, T. et al. BLOCKBENCH: A Framework for Analyzing Private Blockchains. s. 1085–1100, 05 2017. doi: 10.1145/3035918.3064033.

[35] Dorri, A. et al. Blockchain for IoT security and privacy: The case study of a smart home. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, s. 618–623, March 2017. doi: 10.1109/PERCOMW.2017.7917634.

[36] Fabiano, N. Internet of Things and Blockchain: Legal Issues and Privacy. The Challenge for a Privacy Standard. In *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, s. 727–734, June 2017. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.112.

[37] GmbH, D. I. *DHL and BMW Blockchain Proof of Concept* [online]. 07 2019. Dostupné z: https://www.youtube.com/watch?v=KeMGcKn-EmI.

[38] Hackius, N. – Petersen, M. Blockchain in logistics and supply chain : trick or treat? s. 3–18, 2017. doi: 10.15480/882.1444. Dostupné z: http://tubdok.tub.tuhh.de/handle/11420/1447. ISBN 978-3-7450-4328-0.

[39] Hooda, P. Comparison - Centralized, Decentralized and Distributed Systems, Apr 2019. Dostupné z: https://www.geeksforgeeks.org/comparison-centralized-decentralized-and-distributed-systems/.

[40] Insolar. Insolar Technical Paper, 2019. Dostupné z: https://insolar.io/uploads/Insolar%20Tech%20Paper.pdf.

[41] Johnson, N. Ethereum Name Service Documentation. Dostupné z: https://docs.ens.domains/.

[42] Kakavand, H. – Sevres, N. – Chilton, B. The Blockchain Revolution: An Analysis of Regulation and Technology Related to Distributed Ledger Technologies. *SSRN Electronic Journal*. 01 2017. doi: 10.2139/ssrn.2849251.

[43] Lešková, A., 2010. Dostupné z: http://web2.vslg.cz/fotogalerie/acta_logistica/2012/3-cislo/4_leskova.pdf.

[44] Liu, D. et al. Anonymous Reputation System for IIoT-Enabled Retail Marketing Atop PoS Blockchain. *IEEE Transactions on Industrial Informatics*. June 2019, 15, 6, s. 3527–3537. ISSN 1941-0050. doi: 10.1109/TII.2019.2898900.

[45] Mentzer, J. *Fundamentals of Supply Chain Management: Twelve Drivers of Competitive Advantage*. 01 2004. doi: 10.4135/9781452204604.

[46] Morini, M. From 'Blockchain Hype' to a Real Business Case for Financial Markets. *SSRN Electronic Journal*. 01 2016. doi: 10.2139/ssrn.2760184.

[47] Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system, 2009. Dostupné z: http://www.bitcoin.org/bitcoin.pdf.

[48] Ongaro, D. – Ousterhout, J. In Search of an Understandable Consensus Algorithm. In *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, s. 305–320, USA, 2014. USENIX Association. ISBN 9781931971102.

[49] Pease, M. – Shostak, R. – Lamport, L. Reaching Agreement in the Presence of Faults. *J. ACM*. April 1980, 27, 2, s. 228–234. ISSN 0004-5411. doi: 10.1145/322186.322188. Dostupné z: http://doi.acm.org/10.1145/322186.322188.

[50] Prasanna. What is the Blockchain data structure?, Nov 2018. Dostupné z: https://cryptoticker.io/en/blockchain-data-structure/.

[51] Ray, S. The Difference Between Blockchains & Distributed Ledger Technology, Oct 2018. Dostupné z: https://towardsdatascience.com/the-difference-between-blockchains-distributed-ledger-technology-42715a0fa92.

[52] Rehman, J. What are advantages and disadvantages of distributed operating systems, Sep 2017. Dostupné z: http://www.itrelease.com/2015/09/what-are-advantages-and-disadvantages-of-distributed-operating-systems/.

[53] Salah, K. – Khan, M. IoT Security: Review, Blockchain Solutions, and Open Challenges. *Future Generation Computer Systems*. 11 2017. doi: 10.1016/j.future.2017.11.022.

[54] Sayeed, S. – Marco-Gisbert, H. Assessing Blockchain Consensus and Security Mechanisms against the 51 *Applied Sciences*. 04 2019, 9, s. 1788. doi: 10.3390/app9091788.

[55] Schaub, A. et al. A Trustless Privacy-Preserving Reputation System. In Hoepman, J.-H. – Katzenbeisser, S. (Ed.) *ICT Systems Security and Privacy Protection*, s. 398–411, Cham, 2016. Springer International Publishing. ISBN 978-3-319-33630-5.

[56] Schöner, M. M. et al. Blockchain technology in the pharmaceutical industry. *Frankfurt, Germany: Frankfurt School Blockchain Center*. 2017.

[57] Seibold, S. – Samman, G. Consensus: Immutable agreement for the Internet of value. *KPMG< https://assets. kpmg. com/content/dam/kpmg/pdf/2016/06/kpmgblockchain-consensus-mechanism. pdf.* 2016.

[58] Szabo, N. Formalizing and Securing Relationships on Public Networks. *First Monday*. Sep. 1997, 2, 9. doi: 10.5210/fm.v2i9.548. Dostupné z: https://journals.uic.edu/ojs/index.php/fm/article/view/548.

[59] Szydlo, M. Merkle Tree Traversal in Log Space and Time. In Cachin, C. – Camenisch, J. L. (Ed.) *Advances in Cryptology - EUROCRYPT 2004*, s. 541–554, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-24676-3.

[60] Technologies, B. Centralized vs Decentralized vs Distributed Systems, Jun 2019. Dostupné z: https://berty.tech/blog/decentralized-distributed-centralized/.

[61] Steen, M. – Tanenbaum, A. A brief introduction to distributed systems. *Computing*. 08 2016. doi: 10.1007/s00607-016-0508-7.

[62] Vitasek, K. CSCMP Supply Chain Management Definitions and Glossary. Dostupné z: `https://cscmp.org/CSCMP/Educate/SCM_Definitions_and_Glossary_of_Terms.aspx`.

[63] Wikipedia. OEM produkce - Wikipedia, The Free Encyclopedia, 2020. Dostupné z: `https://cs.wikipedia.org/wiki/OEM_produkce`.

[64] Wright, A. – De Filippi, P. Decentralized Blockchain Technology and the Rise of Lex Cryptographia. *SSRN Electronic Journal*. 01 2015. doi: 10.2139/ssrn.2580664.

[65] Zelensky, A. et al. Video Content Verification Using Blockchain Technology. s. 208–212, 09 2018. doi: 10.1109/SmartCloud.2018.00042.

[66] Zhang, R. – Xue, R. – Liu, L. Security and Privacy on Blockchain. *ACM Comput. Surv.* July 2019, 52, 3, s. 51:1–51:34. ISSN 0360-0300. doi: 10.1145/3316481. Dostupné z: `http://doi.acm.org/10.1145/3316481`.

[67] Zheng, Z. et al. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In *2017 IEEE International Congress on Big Data (BigData Congress)*, s. 557–564, June 2017.

A Obsah DVD

Na přiloženém DVD se nachází následující soubory:

- `app/` - adresář se zdrojovými soubory klientské aplikace a spustitelným .jar souborem
- `bin/` - adresář s binárními soubory frameworku
- `generator/` - adresář se zdrojovými soubory generátoru a spustitelným skriptem
- `chaincode/` - adresář se zdrojovými soubory chytrých kontraktů
- `network/` - adresář s konfiguračními soubory sítě a spouštěcími skripty
- `poster/` - adresář s posterem
- `text/` - adresář se zdrojovými soubory LaTeX pro tento dokument
- `Matas_Martin_DP_2020.pdf` - elektronická verze tohoto dokumentu
- `README.txt` - Návod pro spuštění

B Uživatelská příručka

B.1 Instalace

Prostředí je připravené pro operační systém Ubuntu, následující příkazy jsou otestovány pouze s tímto systémem. Předpokladem je, že příkazy budou spouštěny z kořenového adresáře přiloženého DVD.

```
$ sudo apt install git curl docker.io docker-compose
```

Následně je potřeba se ujistit, že příkazy fungují a v případě příkazů `docker` a `docker-compose` musí být verze vyšší než `1.14.0`.

```
$ docker --version
$ docker-compose --version
```

Nyní je potřeba stáhnout Docker kontejnery. To se provede následujícím příkazem:

```
$ sudo curl -sSL http://bit.ly/2ysbOFE | \
  sudo bash -s -- 1.4.7 1.4.7 0.4.20 -s -b
```

B.2 Spuštění sítě

Pro spuštění sítě je potřeba se přepnou do adresáře `network/`. Zde jsou umístěny konfigurační soubory sítě a spouštěcí skripty.

```
$ cd network/
```

Nyní je potřeba přidat skriptů oprávnění, aby je bylo možné spustit. To se provede sérií následujících příkazů.

```
$ chmod u+x *.sh
$ chmod u+x scripts/*.sh
```

Spuštění sítě se provede příkazem uvedeným níže. Spuštění sítě zabere řádově několik minut až desítek minut dle výkonnosti počítače. Při prvním spuštění se stáhnou ještě další Docker kontejnery, protože prvotní stažení provedené ve fázi instalace obsahuje pouze několik základních kontejnerů.

```
$ sudo ./start-network.sh
```

Obdobně je možné síť ukončit následujícím příkazem.

```
$ sudo ./stop-network.sh
```

B.3 Klientská aplikace

V této části bude popsán způsob spuštění aplikace. Aplikaci je možné spustit ve 2 režimech - režimu příkazové řádky a režimu webového rozhraní.

Pro spuštění aplikace je potřeba se přepnou do adresáře `app/`, který se nachází v kořenovém adresáři přiloženého DVD. Za předpokladu, že byl proveden krok spuštění sítě je příkaz následující:

```
$ cd ../app/
```

B.3.1 Generování transakcí

Nejprve je potřeba vygenerovat transakce následujícím příkazem. Příkaz spouští aplikaci v režimu příkazové řádky a pro každou organizaci vygeneruje několik transakcí.

```
$ chmod u+x generate_transactions.sh
$ ./generate_transactions.sh
```

B.3.2 Spuštění aplikace

Aplikace se spouští z příkazové řádky příkazem. Následně je potřeba si otevřít prohlížeč a přejít na adresu `localhost:8080`.

```
$ java -jar client.jar
```

Aplikace se spouští pod identitou OEM, která je nastavena v konfiguračním souboru `application.properties`. Z úvodní obrazovky aplikace je možné se pomocí menu po levé straně dostat na několik obrazovek popsaných níže.

Přidání záznamu

Na této obrazovce je možné po vyplnění formuláře přidat nový záznam, který bude uložen do účetní knihy a stane se dostupný pro ostatní organizace.

Historie transakcí

Na této obrazovce je možné procházet historii provedených transakcí. Při kliknutí na konkrétní transakci se zobrazí detail transakce a možnost ověření údajů (v případě, že dokument nebyl součástí transakce).