LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE


# CAURSPĪDĪGAS FINANŠU TRANSAKCIJU IMPLEMENTĒŠANAI PIEEJAMĀS TEHNOLOĢIJAS

MAĢISTRA DARBS

Autors: **Jaroslavs Rogačs**

Stud. apl. Nr.: jr08038

Darba vadītājs: Dr. dat. Ivo Odītis

RĪGA 2019

# ANOTĀCIJA

Maģistra darbs sniedz sadalītās virsgrāmatas tehnoloģiju risinājumu *Corda*, *Ethereum* un *Hyperledger Fabric* salīdzinošu analīzi, izvēloties optimālāko risinājumu un veicot tā integrācijas dizaina izstrādi.

Maģistra darba uzdevums ir izpētīt un salīdzināt caurspīdīgas finanšu transakciju implementēšanai pieejamās tehnoloģijas, kas palīdzētu risināt transakciju caurspīdīguma, apstrādes ātrdarbības un izmaksu problēmas.

Maģistra darba mērķis ir atrast optimālāko risinājumu *B2B* nozarei, izpētot piedāvātos un nākotnē plānojamos tehnoloģiju kopumus gan *B2B*, gan *B2C* darījumiem, un piedāvāt jaunu produktu finanšu tehnoloģiju uzņēmumiem.

Veicot izvēlēto tehnoloģiju salīdzinošo analīzi, tika secināts, ka optimālākais *B2B* risinājums ir *Corda*, kas ir platforma, kuras jaunākās versijas mērķis ir aizstāt programmatūras, kas tiek izmantotas finanšu transakcijām, ļaujot organizācijām digitalizēt dažādus biznesa procesus. Pamatojoties uz šo secinājumu, tika veikta *Corda* integrācijas dizaina izstrāde.

*Atslēgvārdi: maksājumu sistēmas, finanšu transakcijas caurspīdīgums, blokķēde, decentralizētas lietojumprogrammas, sadalītās virsgrāmatas tehnoloģija, Corda integrācija*

# ABSTRACT

**Technology Availability for Transparent Financial Transactions**

The Master's thesis provides a comparative analysis of Corda, Ethereum, and Hyperledger Fabric, a distributed ledger technology solution, selecting the most optimal solution and developing its integration design.

The task of the Master's thesis is to study and compare technologies available for transparent financial transactions, which would help to solve the transparency, speed of processing and costs issues of transactions.

The goal of the Master's thesis is to find the optimal solution for the B2B industry by exploring the proposed and future technologies for both B2B and B2C transactions and offering a new product to financial technology companies.

By comparing selected technologies, it was concluded that the best B2B solution is Corda, a platform whose latest version aims to replace software used for financial transactions, allowing organizations to digitize various business processes. Based on this conclusion, the development of Corda integration design was carried out.



*Keywords: payment systems, financial transactions transparency, blockchain, decentralized applications, distributed ledger technology, Corda integration*

# AUTOREFERĀTS

Maģistra darba praktiskās daļas rezultātā tika piedāvāta tehnoloģija, kas šobrīd ir ļoti aktuāla un inovatīva, un tai ir labi panākumi pārrobežu maksājumu sistēmu darbībā. Tādi pasaules līmeņa tehnoloģiju uzņēmumi kā "Accenture" un "IBM" ir veikuši pētījumus, kas pierāda sadalītās virsgrāmatas tehnoloģiju efektivitāti caurspīdīgu transakciju veikšanai. Darba autora piedāvātais integrācijas risinājums var palielināt transakciju drošību, izsekojamību un caurspīdīgumu.

Darba teorijas un praktiskās daļas īstenošanai tika izmantots diezgan apjomīgs attiecīgās tēmas literatūras izpētes darbs. Autors ir balstījies uz diviem darba svarīgākajiem zinātniskajiem avotiem - grāmatām par blokķēdi. Šo grāmatu autori ir eksperti, kuri reālajā dzīvē ir ieviesuši autora izpētītās tehnoloģijas. Tā kā pētāmās tehnoloģijas ir salīdzinoši jaunas, tad pamatā darbā tika izmantota šo tehnoloģiju dokumentācija, kas pieejama interneta avotos. Lai gūtu papildu informāciju par pētāmajām tehnoloģijām, tika izmantotas arī profesionālās blogu vietnes.

Darbā aplūkotās problēmas un risinājumi tika aprakstīti ļoti detalizēti un secīgi. Vispirms tika aplūkoti jēdzieni, pasaulē jau izmantotās tehnoloģijas, kā arī jaunas tehnoloģijas, ko varētu pielietot, lai uzlabotu transakciju caurspīdīguma prasības. Pēc tam tika veikts tehnoloģiju salīdzinājums un piedāvāts optimālākais risinājums. Piedāvātā integrācijas dizaina izstrāde tika izstrādāta un prezentēta pilnā apjomā.

Maģistra darba praktiskais darbs, kas ir integrācijas dizaina projektēšana, tika izstrādāts patstāvīgi, apskatāmo tehnoloģiju salīdzinājums tika veikts, balstoties uz dokumentāciju piemēriem.

Tā kā izstrādātais risinājums ir inovatīvs, par to tiek runāts arī dažādās konferencēs, piemēram, *MoneyConf*, ko ir apmeklējis autors un kur radās ideja par *DLT* tehnoloģiju implementēšanu reālajā dzīvē.

Darba noformējuma kvalitāte atbilst prasībām, izmantojot latviešu valodā oficiāli pieņemto nozares terminoloģiju. Visas idejas, formulējumi, attēli utt., kas aizgūti no citiem autoriem, ir atzīmēti ar attiecīgām literatūras atsaucēm.

SATURS

# APZĪMĒJUMU SARAKSTS

- AML *(Anti-Money Laundering)* - nelikumīgi iegūtu līdzekļu legalizācijas novēršanas politika.
- API (*Application Programming Interface*) - lietojumprogrammas saskarne, iepriekš definētu klašu, procedūru, funkciju, struktūru un konstanšu kopums, kuru iespējams izmantot ārējiem programmatūras produktiem.
- B2B (*Business-to-business)* - situācija, kad viens uzņēmums veic komercdarījumu ar citu uzņēmumu.
- B2C (*Business-to-consumer)* - attiecas uz transakcijām, kas tiek veiktas starp uzņēmumu un patērētājiem, kas ir tā produktu vai pakalpojumu gala lietotāji.
- DLT (*Distributed Ledger Technology*) - sadalītās virsgrāmatas tehnoloģija jeb replicētu, koplietojamu un sinhronizētu cipardatu, kas ģeogrāfiski izkliedēti starp vairākām vietnēm, valstīm vai institūcijām, konsenss.
- FATF (*Financial Action Task Force)* - starpvaldību organizācija, kas dibināta 1989. gadā pēc G7 iniciatīvas, lai izstrādātu politiku cīņai pret naudas atmazgāšanu.
- GDPR (*General Data Protection Regulation*) - vispārīgā datu aizsardzības regula par fizisko personu aizsardzību attiecībā uz personas datu apstrādi un šādu datu brīvu apriti.
- HLF (*Hyperledger Fabric*) - platforma sadalītās virsgrāmatas risinājumiem, kuru pamatā ir moduļu arhitektūra, kas nodrošina augstu konfidencialitātes, elastīguma un mērogojamības pakāpi.
- HTTPS (*Hypertext Transfer Protocol Secure*) - komunikācijas protokols drošai saziņai datortīklā, ar plašu izmantošanu internetā.
- ICO (*Initial Coin Offering*) - sākotnējais virtuālās valūtas piedāvājums jeb investīciju piesaistes veids, pārdodot investoriem noteiktu skaitu jaunu kriptovalūtu vienību.
- JMX (*Java Management Extensions*) - Java tehnoloģija, kas nodrošina rīkus, lai pārvaldītu un pārraudzītu lietojumprogrammas, sistēmas objektus, ierīces (piemēram, printerus) un uz pakalpojumiem orientētus tīklus.
- JVM (*Java Virtual Machine*) - virtuālā mašīna, kas ļauj datoram palaist Java programmas, kā arī programmas, kas rakstītas citās valodās, kuras tiek apkopotas ar Java baitkodu.

- KYC (*Know Your Customer)* - uzņēmējdarbības process, kas pārbauda savu klientu identitāti un novērtē iespējamos riskus, kas saistīti ar nelikumīgiem nodomiem uzņēmējdarbības attiecībās.
- OSN (*Ordering Service Node*) - pasūtītāja servisa mezgli, kas veic klienta autentifikāciju, ļauj klientiem rakstīt ķēdē vai nolasīt no tās, izmantojot vienkāršu saskarni, kā arī veic transakciju filtrēšanu un atlasi.
- PBFT (*Practical Byzantine Fault Tolerance*) - konsensa algoritms uzņēmumu konsorcijiem, kuros dalībnieki ir daļēji uzticami.
- PCI DSS *(Payment Card Industry Data Security Standard)* - datu drošības standarts karšu maksājumu sistēmās.
- PoA (*Proof-of-authority*) - algoritms, ko izmanto ar blokķēdēm, kas nodrošina salīdzinoši ātras transakcijas, izmantojot konsensa mehānismu.
- PoW (*Proof-of-work*) - sākotnējais konsensa algoritms blokķēdes tīklā.
- SDK (*Software Developer's Kit*) - palīgprogrammu kopa, kas lietojumprogrammu izstrādātājam atvieglo programmu veidošanu, ievērojot konkrētās to darbības vides īpatnības.
- SEPA (*Single Euro Payments Area*) - Eiropas vienoto maksājumu telpa, kas ir Eiropas Savienības maksājumu integrācijas iniciatīva, lai vienkāršotu eiro pārskaitījumus.
- SWIFT (*Society for Worldwide Interbank Financial Telecommunication*) - bankas identifikācijas koda formāts, kas tiek izmantots, lai bankas un uzņēmumi apmainītos ar informāciju par starptautiskiem pārskatījumiem.
- TLS (*Transport Layer Security*) - šifrēšanas protokols, kuru izmanto lai šifrētu datortīklos (piem., internetā) pārsūtīto informāciju.

# IEVADS

Maģistra darbā tika apskatītas vairākas tēmas, tostarp caurspīdīgas transakcijas maksājumu sistēmās, pieejamās tehnoloģijas caurspīdīgu transakciju īstenošanai, sadalītās virsgrāmatas tehnoloģiju risinājumi, tostarp blokķēdes tehnoloģija. Tāpat darbā tika veikts apskatīto risinājumu salīdzinājums un izvēlētā risinājuma integrācijas dizains.

Darba gaitā tika apkopota teorija par maksājumu sistēmām, transakciju caurspīdīguma prasībām, *AML* priekšrocībām un izaicinājumiem, kā arī tika apskatīta *SWIFT* ziņojumapmaiņas sistēma - tās priekšrocības un trūkumi, sadalītās virsgrāmatas tehnoloģiju jeb *DLT* un blokķēdes priekšrocības un risinājumi caurspīdīgu transakciju prasību ievērošanai.

Maģistra darba uzdevums ir izpētīt un salīdzināt caurspīdīgas finanšu transakciju implementēšanai pieejamās tehnoloģijas, kas palīdzētu risināt transakciju caurspīdīguma, apstrādes ātrdarbības un izmaksu problēmas.

Maģistra darba mērķis ir atrast optimālāko risinājumu *B2B* nozarei, izpētot piedāvātos un nākotnē plānojamos tehnoloģiju kopumus gan *B2B*, gan *B2C* darījumiem, un piedāvāt jaunu produktu finanšu tehnoloģiju uzņēmumiem. Lai sasniegtu darba mērķi, nepieciešams veikt dotos apakšuzdevumus:

- apskatīt maksājuma transakciju, maksājumu sistēmu un transakciju caurspīdīguma jēdzienus;
- izpētīt *AML* ietekmi uz transakcijām;
- apskatīt un izpētīt tehnoloģijas, saistītas ar caurspīdīgām transakcijām;
- veikt pasaulē pieejamo tehnoloģiju izpēti pārrobežu maksājumu sistēmu ietvaros;
- apskatīt un veikt *DLT* risinājumu - *Corda*, *Ethereum* un *Hyperledger Fabric* - salīdzinājumu, izvēloties optimālāko risinājumu un veicot tā integrācijas dizaina izstrādi.

Mūsdienu risinājumi nodrošina elastību maksājumu opcijās, vieglu maksājumu saskaņošanu, caurspīdīgu reālā laika piekļuvi transakciju informācijai. Izvēlētā temata aktualitāte ir saistīta ar esošo risinājumu trūkumiem un iespējām izmantot jaunas tehnoloģijas šo trūkumu novēršanai.

# 1. CAURSPĪDĪGAS TRANSAKCIJAS MAKSĀJUMU SISTĒMĀS

Šajā nodaļā tiks apskatīti maksājuma transakciju, maksājumu sistēmu un transakciju caurspīdīguma jēdzieni, kā arī *AML* jēdziens, tā ietekme un izaicinājumi.

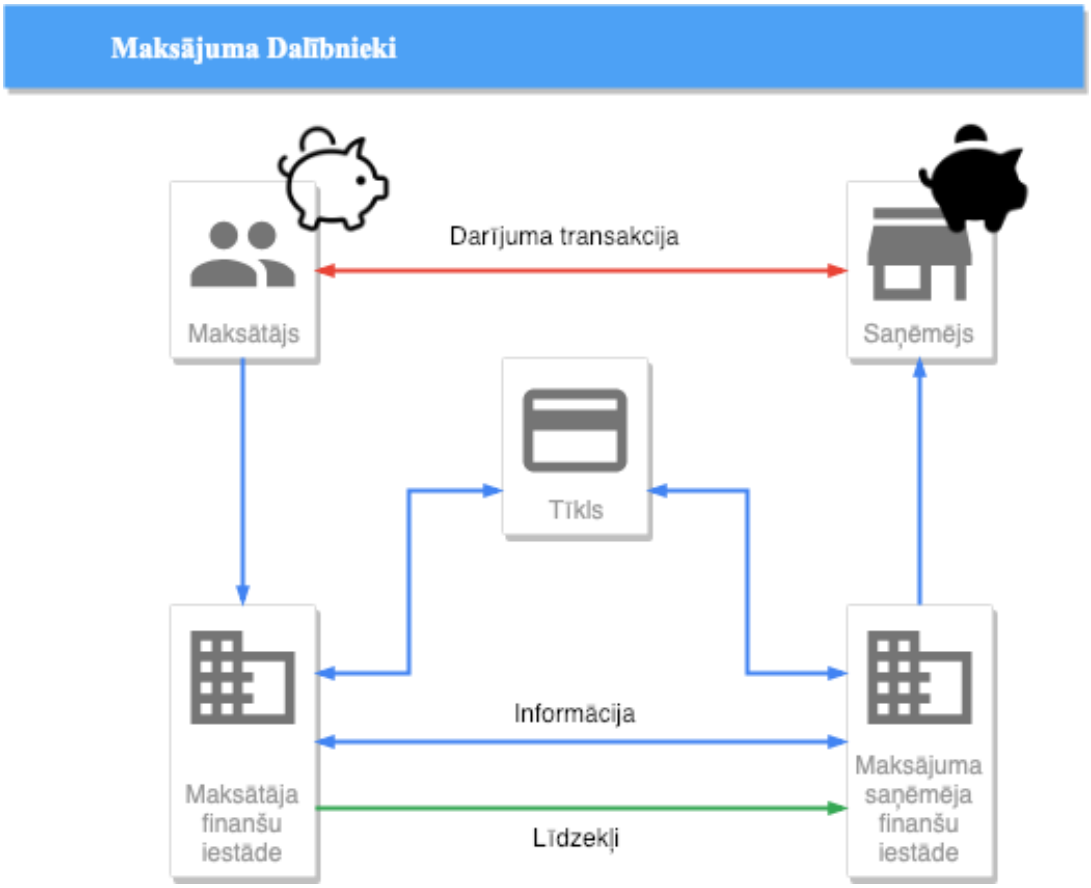
Maksājuma transakcijas ir naudas pārvedums no pircēja apmaiņā pret komersanta sniegtajām precēm vai pakalpojumiem. Maksājuma transakcija, atšķirībā no citiem naudas apmaiņas veidiem, piemēram, naudas pārveduma vai aizdevuma, ietver vismaz vienu komersantu un pircēju, kur abām pusēm ir bankas konti.

Neatkarīgi no maksājumu sistēmas vai tās papildinformācijas, pastāv kopīgs elementu kopums, kas regulē maksājuma transakcijas. Bankas izmanto metodes, kā savstarpēji apmainīties ar maksājuma instrukcijām un pārskaitīt naudu savā starpā. Tās ir nepieciešamas maksājumu sistēmas darbības vai sistēmu garantijas nodrošināšanai.

## 1.1. Maksājumu sistēmas pārskats

Maksājumu sistēma ir procesu un tehnoloģiju kopums, kas veic naudas pārskaitījumu no viena uzņēmuma vai personas citam uzņēmumam vai personai. Maksājumi parasti tiek veikti apmaiņā pret precēm vai pakalpojumiem, vai juridisku saistību izpildi. Tos var veikt dažādās valūtās, izmantojot vairākas metodes, piemēram, skaidru naudu, čekus, elektroniskos maksājumus un kartes. Maksājumu sistēmas būtība ir tāda, ka tā izmanto skaidras naudas aizstājējus, piemēram, elektroniskus ziņojumus, lai izveidotu debetus un kredītus. [1]

Pārskaitāmā vērtība parasti tiek glabāta depozitāriju kontos bankās vai cita veida finanšu iestādēs. Savukārt bankas ir savienotas ar maksājumu sistēmu kopumu, ko tās izmanto, lai apstrādātu maksājumus savu klientu vai noguldītāju vārdā. Bankas, kas darbojas vairākās valstīs, izmanto maksājumu sistēmas katrā no valstīm, kurās tās darbojas tieši vai ar korespondentbankas starpniecību. Bankas parasti uztur kontus valsts centrālajā bankā un piedalās centrālās bankas maksājumu sistēmās. Eirozonā valsts iestādes ir izveidojušas *SEPA* - Eiropas vienoto maksājumu telpu, kas ir pakļauta Eiropas Centrālai bankai jeb *ECB*. *SEPA* tika izveidota, lai nodrošinātu standartizētu maksājumu apstrādi starp dažādām eirozonas valstīm.




*1.1. att.* **Maksājuma dalībnieki**

Vienkāršākajā gadījumā, iesaistot tradicionālo banku sistēmu, maksājumi ietver četrus dalībniekus: [1]

- maksātājs: veic maksājumu un bankas konta debetēšanu par transakcijas vērtību;
- maksātāja finanšu iestāde: apstrādā darījumu maksātāja vārdā;
- maksājuma saņēmēja finanšu iestāde: apstrādā transakciju maksājuma saņēmēja vārdā;
- saņēmējs: saņem kredīta maksājumu savā kontā.

Divas finanšu iestādes var izvēlēties pa tiešo pārskaitīt maksājuma norādījumus un līdzekļus viena otrai. (sk. 1.1. att.) Tāpat bankas var izmantot dažādus starpniekus, kuri atvieglo transakciju veikšanu. Diagrammā starpnieks tiek attēlots kā "Tīkls". Reālajā pasaulē tīklā ietilpst tādas centrālās bankas kā ASV Federālā rezerve, Eiropas Centrālā banka un Japānas Banka, kā arī klīringa iestādes.

Modeļa darbību bieži sauc par maksājumu procesu, un tajā ietilpst četri galvenie soļi:

1. Maksājuma instrukcijas ir informācija, kas ietverta pārskaitījumā vai čekā. Šīs ir maksātāja instrukcijas, kas tiek paziņotas maksātāja bankai, lai tā veiktu pārskaitījumu saņēmējam, izmantojot tīklu un saņēmēja banku.
2. Maksājumu ģenerēšana norisinās tad, kad instrukcijas tiek ievadītas sistēmā.
3. Klīrings ir process, kurā bankas izmanto maksājumu informāciju, lai pārskaitītu naudu savā starpā maksātāja un saņēmēja (maksājuma saņēmēja) vārdā.
4. Norēķins ir pēdējais solis pamatprocesā un notiek, kad saņēmēja bankas konts ir kreditēts un maksātāja bankas konts ir debetēts. Galīgais norēķins notiek tad, kad bankas neatsaucami nodod valūtu savā starpā. Faktiskais maksāšanas process būs atkarīgs no maksāšanas līdzekļa veida, ko maksātājs un maksājuma saņēmējs izvēlas izmantot, vai ko ir izvēlējušās to finanšu iestādes.

Pastāv arī informācijas pārraides mehānismi, piemēram, Pasaules Starpbanku finanšu telekomunikāciju biedrība jeb *SWIFT* un maksājumu sistēmas, kas ietver informācijas pārraides sistēmas. Maksājumu procesā piedalās arī tādi dalībnieki kā čeku printeri, sistēmu nodrošinātāji un karšu sistēmas, piemēram, *Visa* un *MasterCard*. Netradicionālās maksājumu sistēmas, piemēram, *Bitcoin*, gandrīz pilnībā apiet banku sistēmu, izpildot finanšu iestādes, valūtas un tīkla funkcijas.

## 1.2. Transakciju caurspīdīgums

Caurspīdīgums darbojas tādā veidā, ka citiem ir viegli redzēt, kādas darbības tiek veiktas, veicot maksājumus. Transakciju caurspīdīgums parasti attiecas uz sūtītāju, zinot, kāds būs iznākums - kopējās komisijas maksas, cik daudz naudas saņems saņēmējs, kāda kompensācija tiks saņemta, ja kāds no šiem solījumiem netiks izpildīts.

Maksājumu sistēmu var uzskatīt par jebkuru sistēmu, kas ietilpst maksājumu inicializācijas, pārveidošanas vai apstrādes ķēdē. Tālāk tiks aplūkots, kā tiek veicinātas maksājumu caurspīdīguma vajadzības attiecībā uz pārskaitījumiem. [2]

### *1.2.1. Maksājumu uzsākšana*

Šajā posmā ir nepieciešams pārliecināties, ka ir nodrošināta visa nepieciešamā informācija par sūtītāju un saņēmēju. Nepieciešamās detaļas atšķiras atkarībā no dažādiem maksāšanas

modeļiem. Informācijai jābūt formatētai standarta ziņojumā, ko var pieņemt turpmākās apstrādes sistēmas.

### *1.2.2. Maksājumu transformācija*

Finanšu iestādēm parasti ir starpnieks validācijas un transformācijas mērķiem, kas ietver sistēmas, kas pieņem maksājumu ziņojumus, izmantojot kanālus kā *SWIFT*, pircēju vārtejas un iekšējās maksājumu iniciēšanas sistēmas, pirms *back-office* sistēma pārņem ziņojumus tālākai maršrutēšanai vai norēķiniem. Vairumā gadījumu no saņemtajiem datiem tiek veikta konvertēšana uz izejošo formātu, kas ir raksturīgi *back-office* sistēmai.

Prasība šādām sistēmām būtu nodrošināt, ka transformācijas laikā netiek zaudēta informācija un tiktu nodrošināta visa maksājumu caurspīdīguma informācija, kas ietver maksājumu ziņojumu glabāšanu katrā transformācijas posmā. Sistēmām būtu jānoraida saņemtie maksājumi bez nepieciešamajiem datiem.

### *1.2.3. Maksājumu norēķini un maršrutēšana*

Norēķinu sistēmām ir jānodrošina, lai informācija par sūtītāju un saņēmēju būtu pārbaudīta pirms faktiskajiem norēķiniem vai maršrutēšanas tālāk maksājumu ķēdē.

Gadījumos, kad ziņojumi tiek novirzīti tālāk, jānodrošina, lai nekāda informācija netiktu mainīta vai izlaista. Tehnisku problēmu gadījumā jābūt mehānismiem, kas vienojas ar nākamo ķēdes dalībnieku par to, kā informācija tiks saglabāta izmeklēšanas nolūkā, kas varētu rasties nākotnē.

### *1.2.4. Caurspīdīguma prasības*

Tālāk tiks sniegts apraksts par kartēšanas prasībām, ko veic finanšu iestāde, lai veiktu kredīta pārskaitījumu. Šos maksājumus var iedalīt divos galvenajos veidos, lai gan atšķirības starp tiem joprojām ir iespējamas, pamatojoties uz maksājuma veidu un prasībām. Abi veidi ir *own* maksājumi un *on-behalf-of* maksājumi. Atšķirība ir tāda, ka maksātājs ir bankas konta īpašnieka klients *on-behalf-of* maksājumu gadījumā, kas redzams 1.1. tab. [2]

*1.1. tabula*

| | |
|---|---|
| Maksājuma iniciators | *Own-payment* – Jāiekļauj bankas konta īpašnieka vārds, konta numurs un adrese.<br><br>*On-behalf-of payment* – Jāiekļauj galīgā iniciatora nosaukums, konta numurs un adrese. Vienā no atribūtiem ir jābūt bankas konta īpašnieka konta numuram. |
| Maksājuma saņēmējs | Ir jābūt iekļautam saņēmēja vārdam, konta numuram un adresei. Tāpat jābūt saņēmēja finanšu iestādei. |
| Atsauces informācija | Maksājuma atsauces iekļaušanai ir vairāki atribūti. Viens no tiem var saturēt sūtītāja atsauci, kā arī gala *ID*, kas atrodas galīgajā pārskatā. Cits atribūts var saturēt pārskaitījuma informāciju un sūtītāja-saņēmēja informāciju saskaņošanas nolūkos. |
| Finanšu iestāde - starpnieks | Šie lauki ir obligāti jāturpina maksājumu ķēdē. |
| Summa, valūta un valūtas kurss | |

Finanšu iestādēm maksājumu caurspīdīgums ir jāaplūko ne tikai kā vienkārša regulējuma prasība, bet drīzāk jāizmanto tas, lai izvairītos no situācijas, kad to izmanto kā finanšu noziegumu izplatīšanas līdzekli. Kvalitatīvu maksājumu datu nodrošināšana ir primāra, lai efektīvi izmantotu sankciju, krāpšanas un nelikumīgi iegūtu līdzekļu legalizācijas novēršanas sistēmas.

## 1.3. AML

Ātrāku maksājumu globālā parādība ietekmē operatīvos jautājumus, kas saistīti ar nelikumīgi iegūtu līdzekļu legalizēšanas novēršanu. Ātrāki maksājumi ir atbilde uz nepieciešamību modernizēt pašreizējos maksājumu norēķinu tīklus. Ātrāki maksājumu tīkli sniedz priekšrocības trim galvenajām dalībnieku grupām: korporācijām, patērētājiem un finanšu iestādēm.

*AML* operācijas tradicionāli ir izstrādātas pārskatīt potenciāli aizdomīgas transakcijas. Ir iespējams uzlabot *AML* procedūras, pārskatot izmantotos procesus, personāla pieejas un

optimizējot tehnoloģiju rīkus. Tālāk tiks aplūkota *AML* ietekme uz transakcijām un *AML* izaicinājumi attiecībā uz finanšu iestādēm.

### *1.3.1. AML ietekme*

*AML* parasti ir lineārs process, kas pieprasa transakciju pārbaudi, lai noteiktu, vai ir potenciāli augsta riska aizdomīga darbība saistībā ar naudas atmazgāšanu. Visām finanšu iestādēm ir juridisks pienākums veikt pienācīgu kontroli attiecībā uz savu klientu kontiem, ievērojot iepriekš noteiktas atbilstības pārbaudes procedūras. Eiropā ir ieviestas dažādas direktīvas par nelikumīgi iegūtu līdzekļu legalizēšanu, pieprasot plašam finanšu iestāžu lokam ziņot par terorisma finansēšanu un apkarot nelikumīgi iegūtu līdzekļu legalizēšanu. Daudzās valstīs visā pasaulē ir līdzīgas klientu identifikācijas / zināšanas par jūsu klientu *KYC* programmas, kas paredzētas līdzīgu mērķu sasniegšanai.

Programmas veic klientu un transakciju informācijas atbilstības meklējumus sankciju un augsta riska sarakstos, lai noteiktu iespējamās atbilstības. Ja ir atrasta atbilstība, transakcija tiek apturēta, tiek bloķēts konts un paziņots attiecīgajām iestādēm. Finanšu iestādes var arī veikt atbilstības meklējumus augsta riska sarakstā, kurā iekļautas politiskas personas vai valdības amatpersonas, kuras ir pakļautas iespējamajām korupcijas darbībām. [3]

### *1.3.2. AML izaicinājumi*

Bankas un finanšu iestādes saskaras ar nopietnām *AML* atbilstības problēmām. Uzņēmumi, kas nespēj novērst nelikumīgi iegūtu līdzekļu legalizēšanu, saskaras ar nopietniem rezultātiem: ieņēmumu samazināšanās, klientu neapmierinātība, lieli sodi, reputācijas zudums un akciju cenu kritums. Lai ievērotu *AML* noteikumus, bankas visā pasaulē izmanto dažādus tehnoloģiski pamatotus produktus un risinājumus. Finanšu rīcības darba grupa *FATF* ir izstrādājusi noteikumu kopumu, kas ir atzīts par starptautisku *AML* standartu sistēmu. [3] Šie *AML* noteikumi palīdz atklāt, ziņot un novērst aizdomīgas darbības finanšu iestādēs. Finanšu iestādes saskaras ar vairākām problēmām, pārvaldot riskus, kas saistīti ar pašreizējā *AML* statusa novērtēšanu un vājo vietu identificēšanu. Situācija ir saistīta ar krāpšanas un kibernoziegumu pieaugumu. Lai risinātu šīs problēmas, finanšu iestādēm ir jānodrošina datu aizsardzība, ievērojot *FATF* noteikumus.

*AML* izaicinājumi ietver pārvaldības prasību paaugstināšanu - bankām un finanšu iestādēm var būt grūti pārvaldīt pārrobežu un vairāku jurisdikciju *AML* atbilstības prasības un arvien pieaugošās klientu uzticamības pārbaudes prasības. Tāpat kvalificētu cilvēku resursu piesaiste ar padziļinātām *AML* zināšanām var būt izaicinājums. Organizācijām ir jāiegulda ievērojams laiks un pūles, lai personāls atbilstu mainīgajām normatīvajām prasībām.

Vēl *AML* atbilstība prasa, lai finanšu iestādes ieviestu daudzus procesus un tehnoloģiju risinājumus, kas konsolidēs *KYC* datus un sistēmas vienā repozitorijā. Tām ir arī jāizveido infrastruktūra aizdomīgu darbību noteikšanai, datu kvalitātes uzlabošanai un datu standartizēšanai, lai nodrošinātu krāpšanas un finanšu noziegumu centralizētu analīzi.

# 2. PASAULĒ PIEEJAMĀS TEHNOLOĢIJAS

Šajā nodaļā tiks aprakstītas jau esošās tehnoloģijas, kas ir saistītas ar caurspīdīgām transakcijām, kā arī tehnoloģijas, kas ir parādījušās pavisam nesen. Konkrētāk, darbā tiks apskatītas ziņojumapmaiņas *SWIFT* un *RippleNet* pārrobežu maksājumu tehnoloģijas, kā arī *SEPA* un *TARGET2* maksājumu sistēmas. Tāpat nodaļā tiks apskatīti sadalītās virsgrāmatas tehnoloģijas un blokķēdes jēdzieni, to aktualitāte un iespējas.

Starptautiskā starpbanku informācijas nodošanas un maksājumu veikšanas sistēma *SWIFT,* kas sastāv no vairāk nekā 10000 organizācijām un veic vairāk nekā miljons transakciju dienā - naudas pārvedumus, starpbanku maksājumus u.c. Šo transakciju galvenās sastāvdaļas ir finanšu (starp sistēmas lietotājiem) un sistēmas ziņojumi (starp lietotāju un sistēmu). Var redzēt, ka šo transakciju ātrums, izmaksas un laiks bieži cieš liela datu apjoma dēļ, kas tiek pārsūtīti noteiktā formātā.

*DLT* (*distributed ledger technology*) jeb sadalītās virsgrāmatas tehnoloģija konkurē ar esošajiem risinājumiem un nodrošina efektīvākus veidus šo problēmu risināšanai. Uz šī pamata veidotās sistēmas nodrošina ātras un caurspīdīgas transakcijas ar minimālām komisijas izmaksām.

## 2.1. Pārrobežu maksājumu tehnoloģijas

Starptautiskie naudas maksājumi būtībā ir starpbanku pārskaitījumi, tikai šajā gadījumā abas bankas atrodas dažādās valstīs. Arī starptautisko maksājumu gadījumā tiek piemērots korespondentbanku princips, kas nozīmē, ka abām bankām ir jābūt izveidotām attiecībām, lai atvieglotu šo pārskaitījumu.

Divi izplatītākie veidi starptautisku maksājumu veikšanai ir *SWIFT* un *SEPA* pārskaitījumi. *SEPA* tika izveidota, lai vienkāršotu pārrobežu maksājumus eiro valūtā, kas ir vienīgā valūta, ko atbalsta *SEPA*. Var teikt, ka *SEPA* pārskaitījumi ir līdzīgs vietējiem pārskaitījumiem. Būtībā bankām, kas atbalsta *SEPA* pārskaitījumus, ir tiešas attiecības vai starpniekbanku tīkls, tādējādi ļaujot pārskaitījumiem darboties pāri valstu robežām.

Tā kā *SEPA* maksājumi tiek īstenoti tikai Eiropas Savienības dalībvalstīs un Eiropas Ekonomikas zonas valstu robežās eiro valūtā, lai pilnībā apskatītu pasaulē pieejamās pārrobežu maksājumu tehnoloģijas, tālāk tiks apskatītas *SWIFT* un *Ripple* tehnoloģijas. Savukārt, lai

padziļinātāk apskatītu Eiropas Savienības maksājumu sistēmas, paturpinot *SEPA* apskatu, nākamajā sadaļā tiks apskatīta Eiropas maksājumu sistēma *TARGET2*.

### *2.1.1. Maksājumu sistēma TARGET2*

*TARGET2* ir vadošā Eiropas platforma liela apjoma maksājumu apstrādei, kas ļauj Eiropas Savienības bankām veikt savstarpējus eiro maksājumus reālajā laikā. Šo sistēmu izmanto gan komercbankas, gan centrālās bankas. *TARGET2* maksājumu sistēmu procesu dēvē par reālā laika bruto norēķiniem jeb *RTGS*. Tās īpašnieks un operators ir Eurosistēma.

*TARGET2* mērķi ir atbalstīt Eurosistēmas monetārās politikas īstenošanu un eiro naudas tirgus darbību, samazināt sistēmisko risku maksājumu tirgū, t.i. iespēju, ka viens dalībnieks izraisīs visa tirgus sabrukumu. Tāpat viens no svarīgākajiem *TARGET2* mērķiem ir nodrošināt efektīvu pārrobežu maksājumu apstrādi eiro valūtā.

#### *Kā darbojas TARGET2?*

Maksājuma rīkojumi tiek iesniegti apstrādei un tiek īstenoti nekavējoties pa vienam centrālās bankas naudā. Centrālās bankas un komercbankas izmanto *TARGET2* monetārās politikas transakcijām, starpbanku maksājumiem un komerciāliem maksājumiem. Likviditātes pieejamība un izmaksas ir divi būtiski jautājumi, kas saistīti ar maksājumu vienmērīgu apstrādi *RTGS* sistēmās. *TARGET2* paredz virkni funkciju, kas ļauj efektīvi pārvaldīt likviditāti, ieskaitot maksājumu prioritātes, laika ierobežojumus, likviditātes rezervēšanas iespējas, limitus, likviditātes apvienošanu un optimizācijas procedūras.

*TARGET2* darbību vienkāršā veidā var skaidrot šādi: [4]

- gan bankai A, gan bankai B ir konts centrālajā bankā;
- no bankas A uz banku B tiek veikts maksājums eiro valūtā;
- banka A iesniedz platformai *TARGET2* maksājuma rīkojumu;
- maksājums tiek izpildīts - bankas A konts tiek debetēts, bet bankas B konts tiek kreditēts;
- banka B saņem maksājuma informāciju no platformas *TARGET2*.

Svarīgi ir tas, ka *ES* dalībvalstu centrālajām bankām, kuras vēl nav ieviesušas eiro, arī ir iespēja piedalīties *TARGET2* un norēķināties ar eiro, izmantojot platformu. Arī citas finanšu iestādes var pieslēgties *TARGET2* caur iesaistīto centrālo banku.

*TARGET2* darbība ir svarīga, jo Eiropas Centrālā Banka ir ieinteresēta nodrošināt efektīvas maksājumu sistēmas un tirgus infrastruktūru, saglabājot finanšu stabilitāti eirozonā. Līdz ar to *TARGET2* veido svarīgu elementu Eiropas Savienības finanšu integrācijā un nodrošina brīvu naudas plūsmu pāri robežām.

### *2.1.2. Ziņojumapmaiņas sistēma SWIFT*

Pasaules Starpbanku finanšu telekomunikāciju biedrība *SWIFT* ir starptautiski izmantots ziņojumapmaiņas pakalpojums, kas nodrošina standartizētu ziņojumapmaiņas sistēmu starp bankām visā pasaulē.

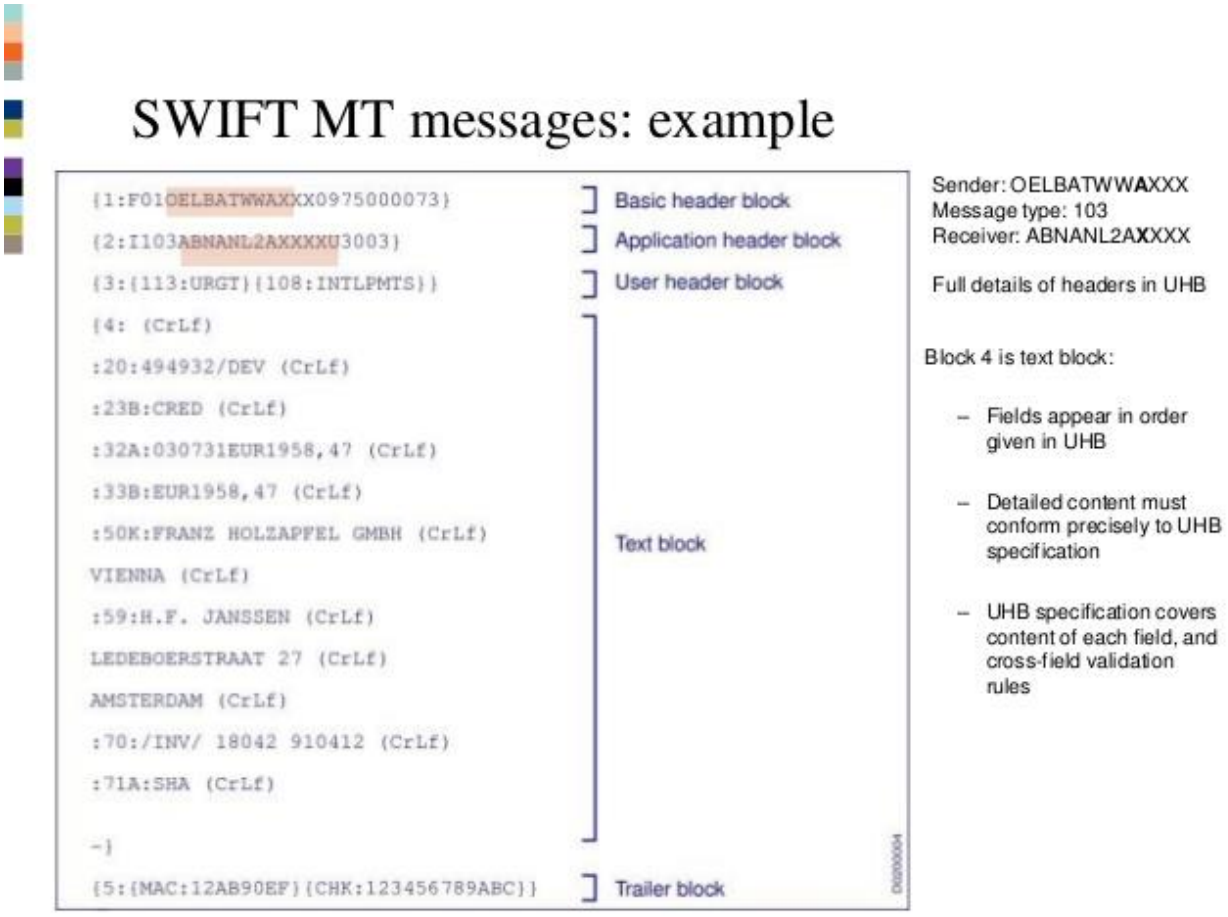
Starptautiski standartizētas ziņojumapmaiņas nozīmē, ka katra transakcija starp katru finanšu iestādi tiek ierakstīta tieši tādā pašā veidā, sniedzot visu informāciju skaidrā un caurspīdīgā veidā.

Katrai finanšu iestādei ir savs unikāls kods, kas sniedz informāciju par bankas nosaukumu un atrašanās vietu, un katra transakcija satur unikālu atsauces numuru, bankas operāciju kodu un informāciju par transakcijas laikā veiktajiem maksājumiem.

#### *SWIFT ziņojuma struktūra*

*SWIFT MT* ziņojums sastāv no šādiem blokiem vai segmentiem (sk. 2.1. att):

- {1:} Basic Header Block
- {2:} Application Header Block
- {3:} User Header Block
- {4:} Text Block
- {5:} Trailer Block




*2.1. att*. **SWIFT MT sūtījuma piemērs [5]**

*SWIFT* ziņu struktūra tiks attēlota šādi:
{1:}{2:}{3:}{4:}{5:}

Header piemērs:
{1:F01YOURCODEZABC1234567890}{2:I101YOURBANKXJKLU3003}{3:{113:SEPA}{108:ILOVESEPA}}{4:

***Basic Header bloks***

*Basic Header* bloka saturs – bits, kas sākas {1: Parasti tas sastāv no {1:F01YOURCODEZABC1234567890}, kur:

- {1: – Identificē bloku
- F – Norāda lietojumprogrammas ID - šajā gadījumā FIN
- 01 – Norāda pakalpojuma ID
  - 01 = FIN
  - 21 = Apstiprinājums jeb ACK or Negatīvs apstiprinājums jeb NAK
- YOURCODEZABC – Loģiskā termināļa adrese + Loģiskā termināļa kods (Z) + Filiāles kods
- 1234 – Sesijas numurs
- 567890 – Sērijas numurs
- } – Norādītas Basic Header bloka beigas

***Application Header bloks***

*Application Header* bloks vienmēr sāksies {2: Parasti tas sastāv no: {2:I101YOURBANKXJKLU3003}, kur:

- {2: – Norāda *Application Header* bloka sākumu
- I – Informē, ka atrodaties ievades režīmā (t.i., Sūtītājs), O norāda izejas režīmu - lai jūs būtu ziņojuma saņēmējs
  - 101 – Ziņojuma veids - šajā gadījumā MT101
- YOURBANKXJKL – Saņēmēji BIC, kas sastāv no viņu BIC (YOURBANK) + Saņēmēju loģiskā termināļa kods (X) + Saņēmēju filiāles kods(JKL)
- U – Ziņojuma prioritāte:
  - U – Steidzams
  - N – Normāls
  - S – Sistēma
- 3 – Piegādes uzraudzība
- 003 – Paziņošanas periods bez piegādes
- } – Norādītas Application Header bloka beigas

***User Header bloks***

*User Header* bloks vienmēr sāksies {3: Parasti tas sastāv no: {3:{113:SEPA}{108:ILOVESEPA}}, kur:

- {3: – Norāda *User Header* bloka sākumu

- {113:SEPA} Šis ir izvēles 4 burtu un ciparu bankas prioritātes kods
- {108:ILOVESEPA} – Norāda ziņu lietotāja paziņojumu vērtību, kas var būt līdz 16 rakstzīmēm un tiks atgriezti
- } – Norādītas *User Header* bloka beigas

***Text bloks***

Text bloks vienmēr sāksies{4:
Seko ziņas par sūtīto ziņu. Šajā gadījumā tas ir *MT101* – kā norādīts Application Header bloka ziņojuma veidā. Attiecīgā *SWIFT* ziņojumu specifikācija – *SWIFT MT101* F*ormat Specifications* Beidzas ar -}

***Trailer bloks***

*Trailer* bloks vienmēr sāksies {5: To var pievienot jūs vai sistēma. Un beidzas ar }

***SWIFT GPI***

Lai ievērojami uzlabotu starptautisko maksājumu procesu, *SWIFT* ir uzsācis savu globālo maksājumu iniciatīvu *Global Payments Initiative* jeb *GPI*. Izmantojot pašreizējos *SWIFT* ziņojumapmaiņas pakalpojumus un korespondentbankas, kas ir veco pārrobežu maksājumu pamats, *GPI* pamatā ir noteikumu kopums, kas paredz, ka bankas apņemas rīkoties saprātīgāk pārrobežu maksājumos, ko atbalsta maksājumu uzskaite un dati, lai uzraudzītu šo jauno noteikumu ievērošanu. *SWIFT GPI* nodrošina, ka pārrobežu maksājumi ir ātri, izsekojami vairākās bankās reālajā laikā, sniedzot apstiprinājumu, kad saņēmēja konts ir kreditēts.

Tehnoloģiju progress, pārrobežu tirdzniecības attīstība un mainīgās investīciju un uzņēmējdarbības prasības pieprasa jaunu risinājumu ieviešanu. Tā kā pārrobežu maksājumiem ir jābūt ātriem, caurspīdīgiem un izsekojamiem, *SWIFT GPI* tika izstrādāts, lai atbilstu šīm prasībām un risinātu esošās problēmas. Līdz ar to *SWIFT GPI* ietver tādas priekšrocības kā: [6]

- starptautiski maksājumi sekundēs vai minūtēs;
- maksājumu izsekošana reālajā laikā;
- iespēja redzēt maksu par bankas pakalpojumiem un valūtas konvertācijas tarifus;
- pārliecība, ka pārskaitījuma dati nav mainīti, kad tiek saņemts maksājums;

- samazinātas pieprasījuma izmaksas, pateicoties spējai izsekot maksājumus u.c.

Konkrēti bankām *SWIFT GPI* sniedz tādas priekšrocības kā spēja nodrošināt labāku, ātrāku un caurspīdīgāku pakalpojumu, vienkārša pielietošana jau esošajā *SWIFT* infrastruktūrā, spēja piedāvāt lielāku uzticību un jaunus, inovatīvus pakalpojumus klientiem, kā arī darījumu partneru viegla identificēšana.

Lai nodrošinātu iespēju uzlabot uzņēmējdarbības praksi un sadarbību starp iesaistītajām bankām, *SWIFT* ir izveidojis servisa līmeņa līgumu jeb *SLA*. Tas ietver noteikumus, kas bankām ir jāparaksta, lai pievienotos *GPI*.

Lai atbalstītu *SLA*, *SWIFT* ir izveidojusi *Observer* sistēmu, lai partnerbankas varētu pārraudzīt *SLA* atbilstību saviem partneriem visā sistēmā. Tāpat *SWIFT* ir izstrādājis maksājumu izsekošanas rīku, kurā maksājumu progresu var apskatīt gandrīz reālajā laikā, ko sauc par *Tracker*. Tāpat katrs *GPI* biedrs ir iekļauts sarakstā, ko sauc par *Directory*. Tas norāda, kādas bankas var sūtīt un saņemt *GPI* maksājumus, kādās valūtās un caur kādiem kanāliem. Kopumā *SWIFT GPI* vienkāršo pārrobežu maksājumus un nodrošina globālu savienojamību. [6]

Apskatot *SLA*, var secināt, ka tas būtībā ir komerciāls izaicinājums bankām. Dažu banku iekšējie procesi var būt tik vāji, ka tām patiešām ir grūtības ievērot *SLA*, bet lielākā daļa korporatīvo banku, kas pārvalda naudas līdzekļus, var veikt pārrobežu maksājumus par saprātīgām cenām tajā pašā dienā, kad tas ir nepieciešams.

Savukārt izsekošanas rīks ir vairāk tehniska problēma. Pirmkārt, dažām bankām var būt grūti izsekot maksājumus, izmantojot savas sistēmas. Otrkārt, tas prasa radošu izmantošanu *MT199* brīvā formāta ziņojumos (ar formatētiem datiem tajos), lai sasniegtu nepieciešamos atjauninājumus.

Pretstatā *SWIFT GPI* darba turpinājumā tika apskatīta cita pieeja pārrobežu maksājumos - *Ripple* tehnoloģija.

### *2.1.2. Globālā maksājumu sistēma RippleNet*

*RippleNet* nodrošina vienotu pieredzi globālajiem maksājumiem. Tas ir vienots, globāls banku tīkls, kas sūta un saņem maksājumus, izmantojot *Ripple* izplatīto finanšu tehnoloģiju - nodrošinot reālā laika ziņojumapmaiņu, klīringu un transakciju norēķinus. *Ripple* ir *DLT* risinājums pārrobežu maksājumiem, kas ir pierādījis savu efektivitāti. Tam ir savs digitālais aktīvs - *XRP*, ko var izmantot norēķiniem, lai samazinātu likviditātes izmaksas. [7]

*Ripple* decentralizētais tīkls ir balstīts uz vienošanos starp *Ripple* un tīkla dalībniekiem - visi izmanto to pašu tehnoloģiju un ievēro konsekventu maksājumu noteikumu un standartu kopumu. *Ripple* izplatītā finanšu tehnoloģija pārspēj šodienas infrastruktūru, samazinot izmaksas un palielinot apstrādes ātrumu.

Mūsdienu korporatīvo un mazumtirdzniecības klientu vajadzības ir būtiski attīstījušās. Papildus liela apjoma maksājumu nosūtīšanai ir nepieciešama iespēja nosūtīt starptautiskus zemas vērtības maksājumus reālajā laikā - ne tikai starp banku tīkliem, bet arī starp jauniem finanšu tīkliem. [7] Mūsdienu infrastruktūras ierobežojumu dēļ bankām, apstrādājot maksājumus, rodas augstas apstrādes izmaksas, ilgs norēķinu laiks un slikta klientu pieredze. Šīs neefektivitātes rezultātā rodas ne tikai milzīgas izmaksas, bet arī neatbilstība mūsdienu bankas klienta vajadzībām.

Kopumā *Ripple* rada jaunas iespējas ieņēmumu palielināšanai, nodrošinot piekļuvi jauniem tirgiem un jauniem produktiem, piemēram, mikromaksājumiem.

### *Ripple tehniskais apraksts*

*Ripple* programmatūra savieno tīklus ar atvērtu neitrālu protokolu - *Interledger* protokolu jeb *ILP*, kas nodrošina reālā laika norēķinus, transakciju drošību, novēršot norēķinu risku. Šī programmatūra ietver arī datu ziņojumapmaiņu starp visām transakcijas pusēm.

*Ripple* programmatūra *xCurrent* ļauj bankām diferencēt sevi, piedāvājot jaunus pārrobežu maksājumu pakalpojumus, vienlaikus samazinot to kopējās norēķinu izmaksas. Risinājums ir izstrādāts, lai apmierinātu banku vajadzības, pielāgojot to esošajām riska, atbilstības un informācijas drošības sistēmām.

Visi *RippleNet* dalībnieki ir savienoti ar *Ripple* standartizēto tehnoloģiju *xCurrent*. Tā ir pirmā globālā reālā laika bruto norēķinu sistēma, kas ļauj bankām ziņot, dzēst un veikt savas transakcijas ar lielāku ātrumu, pārredzamību un efektivitāti. Risinājums ir veidots, pamatojoties uz *ILP*, atklātu, neitrālu protokolu, kas ļauj sadarboties starp dažādām virsgrāmatām un maksājumu tīkliem. [7]

Ar *xCurrent* divvirzienu ziņojumapmaiņu bankas var efektīvāk apmainīties ar informāciju par sūtītāju, saņēmēju, komisiju, tarifiem, piegādes aplēsēm un maksājumu statusu, samazinot darbības izmaksas, kas saistītas ar starptautisko maksājumu apstrādi. Maksājumu apstrādes izmaksas ievērojami samazinās, jo *xCurrent* spēj novērst *SWIFT* komisijas maksas. Tāpat tiek

samazinātas saskaņošanas izmaksas, jo *xCurrent* spēj nodrošināt tūlītēju apstiprinājumu un reālā laika likviditātes uzraudzību.

### *Ripple process*

*Ripple* process ir visaptverošs, tas paredz bagātīgu informācijas apmaiņu, kā arī likviditātes nodrošināšanu un valūtas konvertāciju. *Ripple* īsteno automatizētu, tūlītēju izsoli likviditātes nodrošināšanai un valūtas konvertācijai, tādējādi nodrošinot izdevīgākas cenas. Bankas var ierobežot savus piedāvājuma pieprasījumus darījuma partneriem, kuri atbilst īpašām prasībām, piemēram, reitingam un reglamentējošam stāvoklim. *KYC* un *AML* atbilstība ir ietverta.

No korporatīvā viedokļa tas ir analogs tam, kā izmantot valūtas konvertācijas platformu, lai iegūtu piedāvājumus no vairākām bankām. [8] Tiek izdalīti četri galvenie posmi:

1. **Piedāvājuma saņemšana**: Sākotnējā banka nosūta piedāvājuma pieprasījumu *Ripple* tīklā par attiecīgo maksājumu. Saņemtie piedāvājumi ir valūtas konvertācijas tarifi un pakalpojumu maksas, kā arī atbilstības prasības.
2. **Piedāvājuma apstiprināšana**: Sākotnējā banka pieņem labāko piedāvājumu, par kuru tā var izpildīt atbilstības prasības. Saņēmēja banka tad var bloķēt piedāvājumu. Šajā brīdī *Ripple* bloķē līdzekļus divās banku virsgrāmatās - līdzīgi kā sekundārajam darījuma līgumam.
3. **Sūtāmā maksājuma iesniegšana**: Sākotnējā banka pārskaita naudas līdzekļus no maksātāja konta un caur *ILP* saņēmēja bankai.
4. **Saņemtā maksājuma iesniegšana**: Saņēmēja banka apstiprina, ka līdzekļi ir ieskaitīti saņēmēja kontā.

Saņemtā maksājuma iesniegšana nozīmē, ka līdzekļi ir ieskaitīti saņēmēja kontā. Tas viss notiek vienas vai divu sekunžu laikā. Kā minēts iepriekš, *Ripple* process ir visaptverošāks, mazāk sarežģīts, ātrāks un lētāks.

Veicot *SWIFT GPI* un *Ripple* tehnoloģiju izpēti, tika atklāts, ka bankas un finanšu tehnoloģiju uzņēmumi jau šobrīd izmanto tehnoloģijas, kas balstītas uz *DLT*. Līdz ar to nākamajā sadaļā tiks sīkāk apskatīta sadalītās virsgrāmatas tehnoloģija.

## 2.2. Sadalītās virsgrāmatas tehnoloģija

Sadalītās virsgrāmatas tehnoloģija balstās uz virkni datubāžu tīklu, kas ļauj dalībniekiem izveidot, izplatīt un uzglabāt informāciju efektīvā un drošā veidā. Šie datubāžu tīkli ir droši, tie neprasa centrālās puses vai centrālā administratora iesaisti. Tajā pašā laikā šie tīkli ir pieejami pilnīgai informācijas vēstures izsekošanas pārbaudei. Informācija var tikt izsekota līdz brīdim, kad tā pirmo reizi tika izveidota. Turklāt ir ļoti grūti vai pat neiespējami veikt neatļautas izmaiņas informācijā un tās vēsturē. Citiem vārdiem sakot, sadalītās virsgrāmatas tehnoloģijas operācijas ir veidotas tā, lai tīklos uzglabātajai un paziņotajai informācijai būtu augsts uzticamības līmenis, un katrs tīkla dalībnieks varētu vienlaicīgi piekļūt kopējam informācijas skatījumam.

### *2.2.1. Tradicionālā datubāze un sadalītā virsgrāmata*

Tradicionālā datubāze ir centralizēta informācijas sistēma, kas ir pieejama caur noteiktiem dalībniekiem, un to uzrauga viens vai vairāki sistēmas administratori. Tie regulē piekļuvi datubāzē saglabātajiem datiem un kontrolē to integritāti. Tradicionālās datubāzes pārvaldības sistēmas, ko parasti izmanto finanšu iestādes, nodrošina piekļuvi tās informācijai, izmantojot uzticamus un labi zināmus lietotājus. Tradicionālajām datubāzēm ir jāizmanto augsts drošības līmenis, lai aizsargātu savu klientu datus.

Savukārt sadalītā virsgrāmata ir decentralizēta datubāze, kas ir pieejama un ko kontrolē liels skaits dalībnieku. Šādus dalībniekus sauc par decentralizētās datubāzes tīkla mezgliem. “Pilntiesīgajiem mezgliem” ir sistēmiskas tiesības attiecībā uz sadalīto datubāzi. [9] Savukārt „vieglie mezgli” ir tās pasīvie dalībnieki. Jebkuru datu atjauninājumu apstiprina visi mezgli, kas, izmantojot īpašu konsensa mehānismu, vienojas par datubāzes pašreizējo stāvokli.

Pastāv plašs sadalītās virsgrāmatas tehnoloģiju klāsts. Saskaņā ar pieeju, ko izmanto finanšu pakalpojumu nozarē, uz sadalītās virsgrāmatas tehnoloģiju attiecas blokķēdes tehnoloģija.

### *2.2.2. Blokķēdes tehnoloģija*

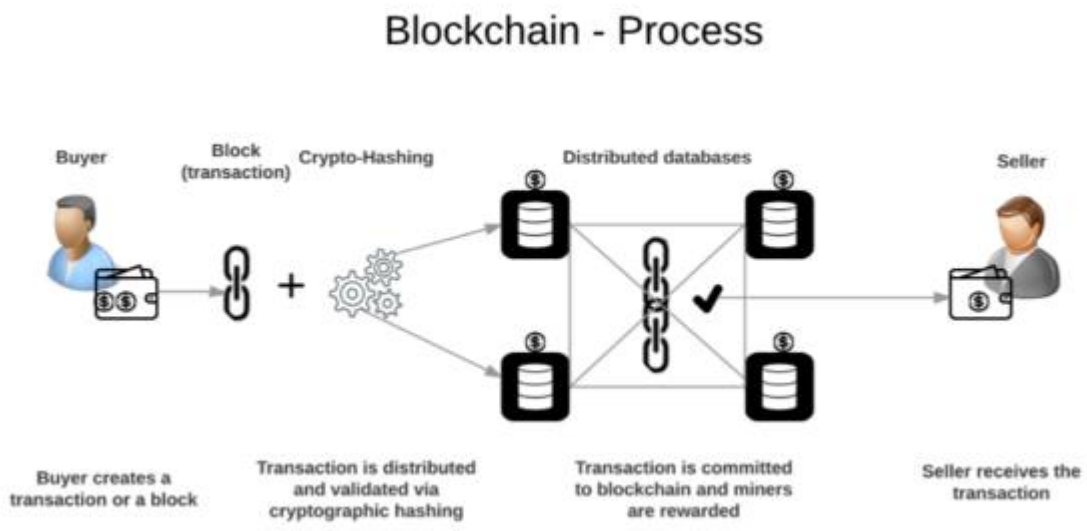
Blokķēdes tehnoloģija izmanto elektronisko parakstu saturošus laika rindu datus vai ierakstus, kas apvienoti blokos, kurus arī savstarpēji saista digitālais paraksts. Tādā veidā

samazinās iespēja veikt izmaiņas datos. *Bitcoin* projektā izmantotā blokķēdes tehnoloģija ir pirmā, visizplatītākā un visplašāk pētītā sadalītās virsgrāmatas tehnoloģija. Tā izmanto ļoti sarežģītu konsensa mehānismu, lai validētu un atļautu ievadīt jaunu informāciju datubāzē. [10] Blokķēdes sistēmas tehnoloģijas izkliedētais raksturs tiek panākts, izmantojot blokus un jaucējkoda jeb *hash* funkcijas (kas novērš nepieciešamību pēc centrālā darījuma partnera vai centrālās datubāzes) kopā ar sarežģītu konsensa mehānismu.

*Goldman Sachs* ziņojumā ir iekļauti īsi secinājumi, kas izskaidro konsensa mehānisma funkcionēšanas pamatkoncepcijas blokķēdē. Šis process tiek parādīts 2.2. att: [11]

1. Tā ir datubāze, kas satur informāciju par transakciju starp divām vai vairākām tās pusēm, kuras rezerves kopijas tiek glabātas daudzos punktos attiecīgajos datoros, kas ir informācijas sistēmas mezgli.
2. Šāda datubāze sastāv no blokķēdēm, kur katrs bloks satur datus, piemēram, transakcijas informāciju - pārdevējs, pircējs, cena, transakcijas noteikumi un citas būtiskas detaļas.
3. Transakcijas detaļas, kas ietvertas katrā blokā, validē visi mezgli, izmantojot algoritmu, ko sauc par *hashing*. Transakcija tiek apstiprināta, ja *hashing* rezultātu apstiprina visi mezgli.
4. Jauns bloks tiek pievienots esošajai transakcijas ķēdei tikai tad, ja tas veiksmīgi iztur validāciju.

Būtībā blokķēde ir sadalītās virsgrāmatas un finanšu darījumu verifikācijas sistēma. Šī tehnoloģija izmanto publiski skatāmu virsgrāmatu, lai ierakstītu un uzskaitītu transakcijas. Katrai transakcijas pusei tiek piešķirta kriptogrāfiskā atslēga. Katra transakcija ir jāapstiprina un jāvalidē tīkla dalībniekiem. Kad tīkls pārbauda akreditācijas datus, transakciju var pabeigt, un tiek izveidots šifrēts bloks.




*2.2. att.* **aprakstīts shematisks process, kas notiek transakcijas apstrādes laikā. [12]**

Bloks tiek pievienots publiskajai virsgrāmatai, tomēr transakcijas informācija blokā paliek privāta, jo katrai pusei ir piešķirtas kriptogrāfiskās atslēgas. Blokķēdes tīkls ļauj pusēm savstarpēji veikt transakcijas bez finanšu uzraudzības vai valdības aģentūru iesaistīšanas. Tā kā virsgrāmatu izplata iesaistītajām pusēm, vienādranga transakcijas bieži vien var tikt pabeigtas dažu minūšu laikā.

Uz virsgrāmatas balstīta tehnoloģija var tikt izmantota kā publiska blokķēde, kā arī var tikt izveidota personiskai lietošanai ar iepriekš noteiktu lietotāju kopumu, kā tas ir finanšu iestāžu gadījumā. Privātajā blokķēdē dalībnieki var veikt transakcijas vai piekļūt datiem tīklā. [11]

Sadalītās virsgrāmatas tehnoloģijas, kas nepieprasa īpašu piekļuvi tām, piemēram, *Bitcoin* un *Ethereum* ir atvērtas sistēmas, kurām nav dalības ierobežojumu. Šādu datubāžu dalībnieki veic mezglu funkcijas tīklā, tiem ir piekļuves tiesības datubāzei, kā arī tiesības pievienot informāciju datubāzēm un piedalīties validācijas procesā. Šādām datubāzēm nav nepieciešams centrālais darījumu partneris vai uzticami dalībnieki. Šeit uzticamie dalībnieki tiek aizstāti ar sadalītās virsgrāmatas tehnoloģijās iestrādātu matemātisku konsensa algoritmu.

Sadalītās virsgrāmatas tehnoloģijas, kurām nepieciešama īpaša atļauja, ietver daudzas iespējamās lietojumprogrammas un integrācijas. Integrācijas veic finanšu pakalpojumu nozare, un tās ir sistēmas, kuras kopīgi izmanto uzticami dalībnieki, kuriem ir autorizēta piekļuve sistēmai. Sadalītās virsgrāmatas kontroles mezgli (ieskaitot kopīgi izmantotās datubāzes) veic katra jauna dalībnieka autorizāciju saskaņā ar noteiktajiem kritērijiem. Sadalītās virsgrāmatas

tehnoloģijas, kurām nepieciešama īpaša atļauja, nav pilnībā decentralizētas. Tās būtiski atšķiras no *Bitcoin* blokķēdes, kas ir pilnīgi decentralizēta datubāze ar anonīmiem dalībniekiem.

### *Byzantine bojājumpiecietība*

*Byzantine* bojājumpiecietība ir decentralizētas sistēmas raksturojums, kas norāda, ka tā var pieciest *Byzantine* kļūdas. Sabrukuma kļūda jeb *crash failure* notiek tad, kad mezgli apstājas, lai kaut ko darītu. *Byzantine* kļūda notiek tad, kad mezgli vienkārši nedara neko vai demonstrē patvaļīgu uzvedību. Būtībā *Byzantine* kļūdas ietver sabrukuma kļūdas. [16]

Jebkurā decentralizētā skaitļošanas vidē, kurā tiek izmantota blokķēdes datu struktūra, pastāv risks, ka viens vai vairāki negodīgi vai neuzticami dalībnieki varētu būt iemesls vides izjaukšanai. Servera klasteris nedarbosies labi, ja daži tajā esošie serveri zaudēs konsekventu datu pārsūtīšanu uz citiem serveriem. Lai decentralizētā skaitļošanas vide būtu uzticama, to ir jāizstrādā tā, lai tai būtu risinājumi šāda veida *Byzantine* kļūdām.

Tā kā decentralizētajām blokķēdes lietojumprogrammām pēc definīcijas nav centrālās iestādes, tāpēc, lai sasniegtu *Byzantine* bojājumpiecietību, tiek izmantots īpašs protokola veids, ko sauc par konsensa protokolu.

### *Konsensa algoritmi*

Konsensa algoritmi novērš vajadzību pēc uzticamiem sadalītās virsgrāmatas dalībniekiem. Visbiežāk izmantotie konsensa algoritmi ir *proof-of-work*, *proof-of-stake*, *proof-of-authority* un *PBFT*. [16]

Algoritms *proof-of-work* ir konsensa algoritms, ko parasti izmanto sadalītās virsgrāmatas tehnoloģijās, kurām nav nepieciešamas īpašas atļaujas. Norādītais algoritms tīklā izmanto kādu skaitu "pilntiesīgo mezglu", kas brīvprātīgi veic datu validāciju. Noteiktas privilēģijas, parasti digitālo aktīvu veidā, tiek piešķirtas mezglam, kas pabeidza validāciju ātrāk nekā cits mezgls, pabeidzot jaucējkoda vērtības aprēķinu. P*roof-of-work* algoritmam ir savas stiprās puses - tas nodrošina pietiekamu aizsardzību pret izmaiņām. Tomēr tam ir arī trūkumi. Šis algoritms prasa ievērojamu skaitļošanas jaudu un elektrības patēriņu. Jo plašāka ir blokķēde, kurai nav nepieciešamas īpašas piekļuves atļaujas, jo centralizētāks kļūst tīkls, tāpēc ka arvien mazākam skaitām mezglu ir pietiekamas skaitļošanas jaudas transakciju verifikācijai. Turklāt apstrādes laiks palielinās kopā ar transakciju skaitu katrā blokā.

Dažās sadalītās virsgrāmatas tehnoloģijās, kurām nepieciešama īpaša piekļuves atļauja, tiek izmantots konsensa algoritms *proof-of-stake*. Šim algoritmam ir nepieciešams vairāku digitālo aktīvu savienojums, lai validētu un pievienotu blokus blokķēdē. Jo vairāk digitālo aktīvu ir savienoti, jo lielāka iespējamība ātrākai bloku validācijai.

Uzņēmumu blokķēdes parasti neizmanto *proof-of-work* vai *proof-of-stake* algoritmu. Uzņēmumiem nepatīk transakcijas, kas var tikt atceltas. Transakcijas, kas var pāriet no "apstiprinātām" uz "neapstiprinātām", var izraisīt pilnīgu haosu uzņēmējdarbībā. Piemēram, blokķēde *Corda* izmanto divu veidu konsensu - transakcijas derīgumu un transakcijas unikalitāti. Pirmajā gadījumā pusēm ir jāpanāk skaidrība, vispirms pārbaudot visus saistītos līguma kodus un pievienojot visus vajadzīgos parakstus. Otrajā procesā tiek pārbaudīts, vai neviena cita transakcija neizmanto nekādus konsekventus stāvokļus. [16]

### *Aktīvu tokenizācija*

Tokenizācija ir aktīva vai aktīva īpašnieka digitālās pārstāvības process. Tokens jeb *token* ir aktīvs vai tā īpašnieks. Šādi aktīvi var būt nauda, preces, vērtspapīri vai īpašuma tiesības. Lai plaši izmantotu sadalītās virsgrāmatas tehnoloģijas tirdzniecības un vērtspapīru uzskaites jomā, vērtspapīriem ir jābūt "tokenizētiem". Līdz ar to tokeni ir jānosaka ar likumdošanu, lai nodrošinātu likumīgu aktīvu īpašumtiesību apstiprinājumu. Tāpat arī naudai ir jābūt "tokenizētai", lai tā varētu izpildīt transakciju līdzekļu norēķinu funkcijas, kas tiek apstrādātas ar sadalītās virsgrāmatas tehnoloģijām. Šobrīd kā alternatīvie risinājumi tiek izskatītas "norēķinu monētas", lai veiktu transakciju norēķinus sadalītās virsgrāmatas ietvaros, kas pieprasa īpašas piekļuves atļaujas.

### *Viedo līgumu nozīme un decentralizētas lietojumprogrammas*

Viedais līgums ir blokķēdes tehnoloģijas sastāvdaļa, kas tiek glabāta blokķēdēs. Tas ir paredzēts, lai digitāli atvieglotu, pārbaudītu vai īstenotu līguma izpildi. Viedie līgumi ļauj veikt uzticamas transakcijas bez trešajām personām. Sistēmas decentralizētā rakstura dēļ neviens šos līgumus nekontrolē, tāpēc katra iesaistītā puse var uzticēties to derīgumam.

Viedo līgumu mērķis ir sniegt drošību, kas ir pārāka par tradicionālajām līgumtiesībām, un samazināt citas ar līgumu slēgšanu saistītas transakciju izmaksas.

Viedie līgumi ir datorprogrammas, kas darbojas sadalītā tīklā. Tie atspoguļo iepriekš rakstītu loģiku operāciju veikšanai, ko glabā un izpilda sadalītās virsgrāmatas mezgli. Pēc darbību izpildes un verifikācijas, ko izpilda viedais līgums, pēdējais informācijas stāvoklis , kas ir darbības rezultāts, saistīts ar darījuma operāciju, tiks ierakstīts un saglabāts blokā.

Šobrīd pētāmās iespējas ir viedo līgumu izmantošana vērtspapīru tirgū: tirdzniecība ar vērtspapīriem, norēķini un klīrings, korporatīvās darbības un peļņas pozīciju un nodrošinājuma pārvaldīšana. Lai izmantotu viedos līgumus, tos ir jānosaka ar likumdošanu.

Grūtības rada tas, ka viedie līgumi ir deterministiski, tie neparedz elastību un iespējas, kas raksturīga līgumsaistībām. Tādējādi ir vajadzīgi mehānismi, kas dažos gadījumos ļautu apturēt vai atcelt līgumus.

Decentralizētas lietojumprogrammas jeb *dApps* būtībā ir lietojumprogrammas, kuras nekontrolē viena vienība un parasti tās darbojas vienādranga tīklā. Šāda veida decentralizētas lietojumprogrammas, kas nav atkarīgas no blokķēdes, tiek uzskatītas par tradicionālajām *dApps* [16]. Katrai *dApp* vajadzētu:

1. **Atvērtu avotu** - šī prasība ir nepieciešama, lai iegūtu uzticību lietotāju vidū. Lai lietotāji varētu uzskatīt, ka lietojumprogramma ir patiešām decentralizēta, viņiem vajadzētu būt iespējai redzēt pirmkodu.
2. **Iekšējo valūtu** - šis punkts galvenokārt attiecas uz *dApp* rentabilitātes aspektu. Tā kā pirmkods ir publiski pieejams, tas var tikt kopēts, tādējādi padarot tradicionālās monetizācijas iespējas bezvērtīgas. Kad tiek palaista iekšējā valūta, bieži vien *ICO* (sākotnējā monētu piedāvājuma) veidā, lietotāji un atbalstītāji var ieguldīt lietojumprogrammā un iegādāties daļu no ierobežota vai ierobežota monētu kopuma. Lietojumprogrammas īpašniekiem un izstrādātājiem tiek izmaksāta tāda pati valūta.
3. **Decentralizētu konsensu** - to aptver blokķēdes konsensa algoritms un viedie līgumi, kas tiek ieviesti lietojumprogrammā.

### *2.2.3. Sadalītās virsgrāmatas tehnoloģijas iespējas*

Sadalītās virsgrāmatas tehnoloģijas priekšrocības izpaužas tās koncepcijas integrācijas rezultātā. Tālāk tiks apskatīti daži no visbiežāk minētajiem iespējamiem ieguvumiem saistībā ar finanšu pakalpojumiem. [13]

1. Izmaksu samazināšana norēķinos. Dažādos pētījumos tika secināts, ka izmaksas var samazināt, likvidējot neefektīvus norēķinus. Tāpat var samazināt cilvēka iejaukšanos un regulatora noteiktos limitus, kas savukārt samazinās darbības un norēķinu riskus.
2. Ātrāki norēķini. Viena no sadalītās virsgrāmatas tehnoloģijas priekšrocībām ir tāda, ka to var izmantot, lai nodrošinātu reālā laika norēķinus. Tomēr lēmumi, kas attiecas uz norēķinu laiku, var atšķirties atkarībā no aktīva veida, transakciju apjoma, likviditātes prasībām, ietekmes uz tirgus veidotājiem un vērtspapīru tirgus konkrētā segmenta efektivitāti. Tādējādi sadalītās virsgrāmatas tehnoloģiju ieviešana ne vienmēr paredz norēķinu īstenošanu reālajā laikā, taču tās var nodrošināt norēķinu laiku atbilstoši tirgus vajadzībām.
3. Uzticamība un iespēja izsekot ierakstus, jo īpaši, izmantojot datubāzi, kurai nav nepieciešamas īpašas piekļuves atļaujas. Jebkurš mēģinājums mainīt iepriekš veiktu ierakstu, piemēram, blokķēdes vēstures bloku, prasa visu bloku jaucējkodu pārrēķināšanu, kas ievadīta secīgi norādītajā blokā. Ja ieraksts ir jāmaina vai jāizdzēš, tad ir nepieciešama atcelšanas darbība, kas ir pilnībā izsekojama. Ja tiek izmantota sadalītās virsgrāmatas tehnoloģija, kas prasa īpašas atļaujas, lai tai piekļūtu, tad katru datu bloku paraksta dalībnieks, kas pievieno blokus. Lai veiktu izmaiņas ierakstā par izmaiņu vēsturi, tās ir jāapstiprina ierobežotam dalībnieku sarakstam. Tādas izmaiņas tiek izsekotas.
4. Automatizētas atskaites reālajā laikā. Daudzi sadalītās virsgrāmatas tehnoloģiju atbalstītāji norāda, ka viena no šīs tehnoloģijas priekšrocībām ir tā, ka regulators var piedalīties kā viens no mezgliem, tādējādi automātiski piekļūstot visiem datiem. Tas savukārt ļaus regulatoriem sekot izmaiņām informācijā reālajā laikā.
5. Jaunu aktīvu veidu iekļaušana. Viena no šīs tehnoloģijas priekšrocībām ir tā, ka aktīvus ar ievērojamām ražošanas izmaksām, transakcijām un piegādēm, var "tokenizēt", lai saglabātu īpašumtiesības uz tiem. Savukārt tokenizācija ļauj izmantot aktīvus kā nodrošinājumu.
6. Efektivitātes uzlabošana. *DLT* var aizstāt vairākas centralizētas datu bāzes, lai uzlabotu informācijas un datu apmaiņu. Datu validācijai vajadzīgais laiks ir atkarīgs no tīkla struktūras un validācijas mehānisma. Izmantojot *DLT*, norēķini ar vērtspapīriem tiek samazināti no dienas līdz minūtēm. Naudas pārskaitīšana, izmantojot *Bitcoin* blokķēdi, tiek veikta dažu sekunžu vai minūšu laikā, salīdzinot ar pašreizējo banku pakalpojumu praksi, kad šis process aizņem 2 vai 3 dienas.

7. Drošības uzlabošana. Drošība blokķēdē tiek nodrošināta, šifrējot blokus un to savienojumus. Turklāt veikt uzbrukumus katram mezglam blokā šobrīd ir daudz sarežģītāk nekā centralizētai datubāzei.

### *2.2.4. Sadalītās virsgrāmatas tehnoloģijas draudi un riski*

Liels skaits integrāciju šobrīd atrodas testēšanas stadijā. Pat, ja testēšana noritēs veiksmīgi, *DLT* ieviešana vērtspapīru tirgū, visticamāk, radīs dažādus tehnoloģiskus un darbības jautājumus.

Daudzi eksperti uzskata, ka *DLT* joprojām ir ļoti agrīnā attīstības stadijā, kas nozīmē, ka to plaša pielietošana vēl nenotiks tik drīz. Turklāt jebkura šo tehnoloģiju izmantošana, kas saistīta ar viedajiem līgumiem, rada risku, jo tā ir jauna metode, un šādu līgumu juridiskais statuss vēl nav noteikts. [13]

Atkarībā no izmantotās *DLT* veida, tostarp konsensa mehānisma, ir jāņem vērā mērogojamības problēma. Piemēram, blokķēde *Bitcoin* kā *DLT*, kurai nav nepieciešamas īpašas atļaujas piekļuvei, saskaras ar mērogojamības problēmām. Transakciju skaits, ko sistēma var apstrādāt sekundē, nav pietiekams, lai veiktu vērtspapīru reālā laika norēķinus. Savukārt *DLT*, kurai nepieciešamas īpašas atļaujas piekļuvei, mērogojamība nav tik būtiska problēma.

#### *Saderība*

Finanšu iestādes neplāno aizstāt esošo infrastruktūru straujā veidā, bet drīzāk cenšas pakāpeniski ieviest izmaiņas vienlaikus ar tiesību sistēmu precizēšanu. Tieši tāpēc ir būtiski, lai starp *DLT* un tiesību sistēmām tiktu izveidoti savstarpējās mijiedarbības kanāli. Ja šādas saderības nav, līdzāspastāvēšana radīs papildu izmaksas, samazinot priekšrocības, ko sniedz pāreja uz *DLT*. Piemēram, iespējamajā tehnoloģiju piemērošanas jomā pēc tirdzniecības norēķiniem ir jānodrošina visu tirgus dalībnieku (brokeru, emitentu, investoru, tirdzniecības platformu, finanšu tirgus infrastruktūras operatoru) sistēmu operatīva saderība. Turklāt ir nepieciešama dažādu tīklu sadarbība, kas izmanto *DLT*. [13]

Pirms tehnoloģiju standartizēšanas, iespējams, ka daudzi tīkli un lietojumprogrammas risinās problēmas paralēli, jo mijiedarbības protokoli vēl nav izstrādāti. Tiek apsvērta iespēja

izmantot, piemēram, viedos līgumus, lai nodrošinātu mijiedarbību datu vai digitālo aktīvu nodošanā starp dažādiem tīkliem, kas izmanto tehnoloģijas.

### *Kiberdrošība*

Šifrēšana nodrošina daļēju aizsardzību pret kibernoziedzības risku. Piemēram, ņemot vērā pieredzi, kas gūta, izmantojot *Bitcoin* blokķēdē *proof-of-work* jeb *PoW* metodi, ļaunprātīgajam mezglam jāpiemīt vairāk nekā 50% no tīkla skaitļošanas jaudas, lai kontrolētu blokķēdi un validācijas procesu [13].

Blokķēdes *Bitcoin* pieredze rāda, ka šādas jaudas iegūšana ir dārgs process. Validācijas process saskaņā ar *proof-of-stake* jeb *PoS* metodi sadala validācijas tiesības saskaņā ar dalībnieka daļu tīklā. Šāds validācijas process ir daudz lētāks nekā *proof-of-work* metode. Izmantojot šo metodi, skaitļošanas jaudas izmaksas, ko ņem vērā *proof-of-work,* tiek pārveidotas par reputācijas izmaksām vai nodrošinājuma zudumu, ja mezgli, kas veic validāciju, centīsies falsificēt datus *DLT*.

Visbiežāk sastopamie draudi nav uzbrukumi tīkliem, bet personisko atslēgu zādzība vai zudums. Personiskās atslēgas ļauj īpašniekiem kontrolēt savus digitālos aktīvus, un nozaudējuma gadījumā tiek zaudēta arī kontrole pār aktīviem. [16]

### *Vadība*

*DLT* var samazināt operacionālos riskus, novēršot informācijas plūsmu dublēšanos un uzturot vienotu, nemainīgu datu avotu, kas reģistrēts hronoloģiskā secībā. Tomēr, ja rodas kļūda, to ir grūti izsekot vai labot. Turklāt, kā norādīts sadaļā par mērogojamību, operacionālie riski, kas ir unikāli *DLT,* kam nepieciešamas īpašas atļaujas, ietver tīkla pārvaldību un stabilitāti. Mezgli, kas īsteno validāciju, var atstāt tīklu, ja transakcijas validācijas priekšrocības nav pietiekamas vai ja vajadzīgā skaitļošanas jauda kļūst pārāk dārga. Šis risks ir mazāks sadalītajā virsgrāmatā, kurai nepieciešamas īpašas atļaujas, jo vadības struktūrai ir kontrole pār operācijām un tīkla pārvaldību. [11]

Sadalītajā virsgrāmatā, kurai nav nepieciešamas īpašas atļaujas, lai samazinātu operacionālos riskus, ko rada kāds no mezgliem, vadības struktūrai ir jānosaka vispārīgi

noteikumi un pārvaldības principi, tostarp vadības noteikumi, līdzdalības kritēriji un rīcības noteikumi.

### *Viedie līgumi*

Teorētiski viedie līgumi samazina cilvēku kļūdas, izmantojot automatizāciju. Tomēr, ja rodas kļūda, to ir grūtāk labot, jo operācijas ir savstarpēji saistītas un iekļautas blokķēdē, kuras turklāt pašas izpildās saskaņā ar programmas kodu, kas norādīts viedajā līgumā. Turklāt viedajos līgumos atklājās cits cilvēka kļūdas veids: programmēšanas kļūdas. Viedā līguma programmatūras kodam nav precīzi jāatspoguļo cilvēku nodoms parakstīt līgumu, kas var būt operacionālā riska avots. [14]

### *Anulēšanas mehānisms*

Viena no svarīgākajām *DLT* iezīmēm ir transakciju neatceļamība: pēc validācijas un lejupielādes blokķēdē transakciju nevar mainīt, atcelt vai atsaukt. Tā kā resursu mehānisms nav pieejams, tad darījuma partneris, kas veica kļūdainu transakciju, var to mainīt tikai veicot reverso transakciju. Tāpēc anulēšanas mehānismam ir nepieciešama turpmākā izpēte.

### *Netings*

Sadalītās virsgrāmatas tehnoloģija ieraksta un nosūta informāciju par katru transakciju kopējai dalībnieku grupai bez netinga. [15] Šis mehānisms ir pretrunā standarta praksei vērtspapīru tirgū attiecībā uz noteiktiem produktiem, piemēram, atvasinātajiem finanšu instrumentiem, drošības prasībām, kas attiecas uz netingu. Tā trūkums noved pie drošības un ekspluatācijas kapitāla prasību pieauguma. Tomēr šobrīd finanšu iestādes mēģina ieviest netingu sadalītās virsgrāmatas tehnoloģijās.

### *Caurspīdīgums*

*DLT* ļauj atklāt dažus transakcijas datus (piemēram, darījuma partnera identitāti, līdzekļu un aktīvu atlikumu un aktīvu veidu) validācijas nolūkos. Tas neatbilst standarta tirgus praksei, jo šādus datus uzskata par konfidenciāliem. Neskatoties uz to, ka tiek veikti mēģinājumi, lai

atrisinātu šo problēmu, konfidenciālas informācijas aizsardzības pievienošana blokķēdē var negatīvi ietekmēt citas tās priekšrocības, jo īpaši caurspīdīgumu. [11]

*DLT*, kurai nepieciešamas īpašas piekļuves atļaujas, ir vismaz viens regulators, kas glabā visus ierakstus un informāciju par visiem iesaistītajiem mezgliem. Tāpēc regulatoriem ir salīdzinoši viegli izsekot dalībnieku darbības šādā tīklā. *DLT*, kurai nav nepieciešamas īpašas atļaujas, parasti nav iespējams noskaidrot, kas veic šīs vai citas darbības, ja netiek uzsākta atbilstošā procedūra. Ir arī grūti noteikt, kas var būt persona, kas pārrauga šādu datubāzi, jo tā sastāv no mezgliem, kas var atrasties dažādās jurisdikcijās.

### *2.2.5. Aktualitāte un regulatoru darbības*

Viena no *DLT* priekšrocībām ir tāda, ka regulatori var piedalīties kā viens no *DLT* mezgliem un tādējādi piekļūt visiem datiem. Tas savukārt ļauj regulatoriem iegūt pilnīgākus, izsekojamus ierakstus, kas tiek veikti reālajā laikā. Tomēr pašiem regulatoriem ir jānovērtē, vai viņi vēlas piekļūt paplašinātam datu apjomam, kas tiek atjaunināts reālajā laikā, vai arī ir viņiem pietiek ar atskaitēm. Ja regulatori vēlas kļūt par *DLT* mezglu, ir jāizstrādā automatizēta pārraudzības funkcija un jāpieņem attiecīgās tehnoloģijas eksperti. [17]

Blokķēdes virsgrāmatā datus nevar viegli mainīt, un tos datus, kas mainīti blokā, var izsekot un uzraudzīt, novēršot krāpšanu un ļaunprātīgu izmantošanu. Sadalītā virsgrāmata apvieno visus datus vienā platformā.

Blokķēdes tehnoloģija spēj nodrošināt decentralizētu risinājumu, vienlaikus nodrošinot drošības protokolu un normatīvo prasību izpildi. Blokķēdes šifrēšanas iespējas aizsargā sensitīvos datus un novērš atbilstības pārkāpumus. Tas ir uzticams tīkls, kurā datiem var piekļūt tikai uzticami avoti.

### *2.2.6. Blokķēdes izmaksu ietaupījums*

Balstoties uz pasaules līmeņa konsultāciju uzņēmuma "Accenture" un kompensācijas datu un konsultāciju uzņēmuma "McLagan" ziņojumiem, blokķēdes tehnoloģija var samazināt infrastruktūras izmaksas. [18]

Blokķēdes tehnoloģijas izmanto sasniegumus programmatūras, sakaru un šifrēšanas jomās, kas ļaus finanšu sektoram pāriet no atsevišķas, sadrumstalotas datubāzes struktūras uz kopīgu, sadalītu datubāzi, kas aptver organizācijas.

Aizstājot tradicionāli fragmentētas datubāzes sistēmas, kas atbalsta transakciju apstrādi ar sadalītās virsgrāmatas sistēmu, finanšu iestādes var samazināt vai novērst saskaņošanas izmaksas, vienlaikus uzlabojot datu kvalitāti. Saskaņā ar ziņojumiem tas radītu ievērojamus ietaupījumus daudzos finanšu iestāžu procesos. Piemēram, finanšu pārskatu izmaksas varētu samazināties par 70%, pateicoties optimizētai datu kvalitātei, caurspīdīgumam un iekšējai kontrolei. Arī atbilstības izmaksas varētu samazināties par 30% - 50%, jo uzlabojas transakciju caurspīdīgums. Tāpat centralizētas operācijas, kas atbalsta tādas funkcijas kā *KYC*, var radīt 50% ietaupījumu, izveidojot efektīvākus procesus, lai pārvaldītu digitālās identitātes un savstarpēji saskaņotu vai kopīgotu klientu datu avotu vairākās finanšu iestādēs.

## 2.3. Mašīnmācīšanās priekš AML

Mašīnmācīšanos izmanto, lai uzlabotu un pārdomātu esošos sistēmas elementus, piemēram, transakciju pārraudzību, riska novērtējumus un *KYC*.

Lietošanas gadījumi variē starp uzlabotas analīzes kā papildu filtra pielietošanas līdz esošajām pārraudzības sistēmām, lai samazinātu viltus pozitīvus skaitļus un pārraudzītu mašīnmācīšanos. Visbiežāk eksperimentālie lietošanas gadījumi aplūko iespējamo paradigmu maiņu uz holistisku pieeju pārraudzībai jeb klientu uzvedības pārraudzībai. [19]

Šobrīd liela uzmanība tiek veltīta transakcijām, izmantojot informāciju no *KYC* failiem un citām darbībām, kas veiktas ar iestādi, un informāciju no ārējiem avotiem, lai noteiktu iespējami aizdomīgu darbību.

Šiem lietošanas gadījumiem kopīgs ir tas, ka tiek saglabāts cilvēka elements *AML*. Pastāvīgais mērķis ir atbrīvot uzņēmuma analītiķu resursus, lai koncentrētos uz augstāka riska gadījumiem. Finanšu iestādes šobrīd nevēlas automatizēt lēmumu pieņemšanas procesu, bet atbalsta savus analītiķus ar arvien spēcīgākām tehnoloģijām, automatizējot procesu soļus, kas savukārt analītiķiem neļauj koncentrēties uz attiecīgajiem riskiem.

Lai padziļinātāk izpētītu mašīnmācīšanās ietekmi un iespējas *AML* ietvaros, tālāk tiks apskatīta *Azure Machine Learning* pakalpojuma būtība, par pamatu ņemot autora bakalaura darbā "Finanšu informācijas drošības pieejas un risinājumi" iekļauto pētījumu par *Azure Machine Learning* iespējām krāpšanas apkarošanai. [20]

Lai samazinātu izstrādes izdevumus un palielinātu efektīvo transakciju caurlaidību, var izmantot *Azure Machine Learning* - mākoņa pakalpojumu, prognožu analīzes uzdevumu risināšanai (*predictive analytics*), ko izstrādājusi kompānija *Microsoft*. Šī pakalpojuma šablonos ir pieejams transakciju pārraudzības atklāšanas risinājums.

Projektus *Azure ML Studio* sauc par eksperimentiem, kas ietver rīku komplektu, kas paredzēts speciālistam darbā ar datiem, un sniedz iespēju apstrādāt datus pēc sava modeļa. Datu saņemšana - kontroles elements Reader ļauj augšupielādēt gan strukturētas gan daļēji strukturētas datu kopas. Par datu kopu krāpniecisku maksājumu atpazīšanas modeli paredz transakciju žurnāls, kas sastāv no divām tabulām ar *NoSQL*-glabātavā: faktu tabula par transakcijām un tabulas ar iepriekš aprēķinātām statistikām metrikām.

**Datu saņemšana** - kontroles elements *Reader* ļauj augšupielādēt gan strukturētas, gan daļēji strukturētas datu kopas. Datu kopu modeļa atzīšanu krāpniecisku maksājumu paredz

transakciju žurnāls, kas sastāv no 2-galdiem ar *NoSQL* veikalā: tabula faktu par transakciju un tabulas ar iepriekš aprēķinātiem pēc statistikas datiem.

**Datu sagatavošana un pētījums** - parasti darbs notiek ar nepilnīgiem datiem - apmācības izlasē ir vai nu tukšas vai dublētas datu vērtības. Uzraudzības sistēmā, piemēram, valsts vai maksātāja *IP* adrese. Ar datu kontroles palīdzību būtu jāizdzēš rindas, kuras nesatur vērtības, kā arī rindas, kas satur acīmredzami nepareizus datus, kas ievieš neprecizitātes modelī. Ir nepieciešams arī atbrīvoties no neizmantotajiem laukiem modelī: adreses un darba informācijas, kas saņemti no *Azure* tabulas.

**Datu dalīšana** - lietojot apmācības algoritmus, vismaz reizi eksperimenta laikā nāksies datu kopu dalīt divās apakšgrupās: apmācības un testa izlase, papildus lietojot datu sajaukšanu, kas palīdz novērst neprecizitātes risku apmācības komplektā. Piemēram, krāpšanas roboti var ieviest neprecizitātes vispārējā modelī, kas būtu jāņem vērā pie sadalīšanas.

**Modeļa būvniecība** - solis, kuram ir milzīga ietekme modeļa precizitātei. Lai identificētu visus, modeļa ietvaros, būtiskus prediktorus un tajā pat laikā, lai norobežotu to skaitu, pētniekam nepieciešamas zināšanas kā matemātiskās statistikas tā arī priekšmeta pētījumu jomā. Daži no mašīnas apmācības algoritmiem nedarbojas korekti bez prediktoru vērtības normalizēšanas. Turklāt esošā modelī mainīgo / prediktoru skaita samazināšana ļaus uzlabot resursu utilizāciju, veicot apmācības algoritmu un izvairīties no atkārtotas modeļa apmācības. Modeļa novērtēšana - *Azure ML* ļauj pievienot vienā eksperimentā neierobežotu mašīnas apmācības algoritma skaitu. Tas ļauj pētījuma laikā salīdzināt vairākus algoritmus, lai noteiktu, kurš no tiem vislabāk piemērots krāpniecisko darījumu problēmas risināšanai.

Vēl viena iespēja, lai iegūtu labāku modeļa veiktspēju - iestatīt mašīnas apmācības algoritmu, izmantojot lielu algoritma konfigurācijai pieejamo parametru skaitu. Modeļa publicēšana - izbūvētos un aprēķinātos *Azure ML Studio* modeļus var izvērst mērogotā, atteikumnoturīgā tīmekļa servisa veidā.

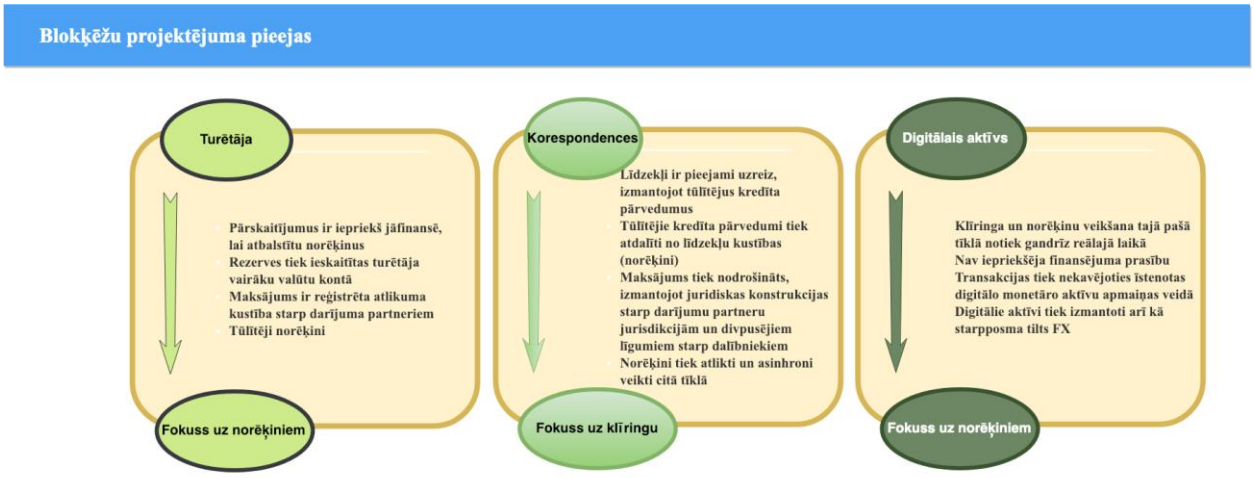
Serviss darbojas divos režīmos: pakešu režīmā (servisa asinhronas atbildes) un pieprasījuma / atbildes režīmā ar zemu aizturi (sinhronā atbilde). Serviss saņem un nosūta ziņojumus *JSON* formātā. Lai piekļūtu servisam, tiek izsniegta *API Key* piekļuves atslēga, kura ir iekļauta pieprasījuma galvenē. Ir iespējams pievienot neierobežotu skaitu galapunktu, caur kuriem var iegūt piekļuvi servisam.

# 3. DLT RISINĀJUMI

Šajā nodaļā tiks apskatīti esošie *B2B* un *B2C* risinājumi kā *Corda*, *Ethereum* un *Hyperledger Fabric*, kas ir balstīti uz *DLT*, kurās tiek izmantota ekonomisko transakciju digitālā virsgrāmata. Blokķēdes izmantošana identifikācijas un uzskaites datu glabāšanai var nodrošināt tādu datu nemainīgu glabāšanu, kas var nodrošināt precīzu to pārbaudi, kā arī novērst šādu datu falsifikāciju. Tāpat tiks apskatīti dizaina modeļi un blokķēdes risinājumi, lai apkopotu rezultātus no analīzes un radītu optimālo risinājumu, kas balstīts uz šo tehnoloģiju.

## 3.1. Blokķēdes dizaina modeļi

Pamatojoties uz "IBM" ziņojumu "Programmējama nauda" [21], blokķēde ir kļuvusi par ideālu tehnoloģiju maksājumiem, jo tā ir ietver transakciju un norēķinu drošību, datu integritāti, uzskaites un efektivitātes iezīmes. Maksājumu nozare piedāvā daudzsološas iespējas. Starptautiskā e-komercija ik gadu pieaug par vairāk nekā 20%, kas ir vairāk nekā divas reizes lielāks pieauguma temps salīdzinājumā ar lokālās e-komercijas pieauguma tempu. Lai gan digitālajā laikmetā nav praktiska iemesla, kāpēc starptautiskajām transakcijām būtu jāmaksā vairāk nekā iekšzemes transakcijām, tomēr starptautiskajiem maksājumiem ir lielāka sarežģītība un valūtas maiņas izmaksas salīdzinājumā ar iekšzemes transakcijām.




*3.1. att.* **Blokķēžu projektējuma pieejas [21]**

Pēdējos gados ir veikti daudzi blokķēdes eksperimenti saistībā ar starptautiskajiem maksājumiem. Gandrīz visi no tiem iedalās trīs dizaina modeļos: turētāja modelis, korespondences modelis un digitālā aktīva modelis (sk. 3.1. att).

Turētāja un korespondences modelis ir plaši izmantoti mūsdienu maksājumu tīklos. Šos modeļus izmanto lielie ārvalstu valūtas norēķinu uzņēmumi, kas veic lielāko banku ikdienas norēķinus visā pasaulē. Blokķēdes tehnoloģiju pielietošana abos šajos modeļos var sniegt papildu inovāciju esošajām funkcijām.

Atšķirībā no turētāja un korespondences modeļa digitālā aktīva modelis veic klīringu un norēķinus vienā tīklā. Saskaņā ar šo modeli maksājumu instrukciju vēsture kopā ar nemainīgu transakciju virsgrāmatu un norēķinu līdzekļiem tiek apvienota vienā tīklā. Digitālā aktīva modelis nodrošina integrētu tīklu, lai uzsāktu pārskaitījuma instrukcijas un veiktu transakcijas reālajā laikā. Visi soļi, sākot ar maksājuma instrukcijām līdz klīringam, var tikt iekļauti vienā transakcijā, kas tiek veikta gandrīz reālajā laikā.

## 3.2. B2B risinājums Corda

*Corda* ir platforma, uz kuras var veidot uz *DLT* balstītas lietojumprogrammas. *Corda* ir *R3* produkts. *R3* ir finanšu un tehnoloģisko pētījumu uzņēmums, kas strādā ar vairāk nekā 100 bankām, finanšu iestādēm, regulatoriem, tirdzniecības asociācijām, profesionāliem pakalpojumu un tehnoloģiju uzņēmumiem, lai attīstītu *Corda*.

Jaunākās *Corda* versijas mērķis ir aizstāt programmatūras, kas tiek izmantotas finanšu transakcijām, ļaujot organizācijām digitalizēt dažādus biznesa procesus, kas bija apgrūtinoši, izmantojot vecās programmatūras sistēmas.

*Bitcoin* un *Ethereum* platformas, kas bija pirms *Corda*, pat, ja tās nav tiešā veidā piemērotas finanšu iestādēm, palīdzēja saprast jaunu veidu, kā veidot sadalītas sistēmas. *Corda* platforma atbalsta viedos līgumus, kas ir automatizējami un var strādāt ar cilvēka veikto ievadi un kontroli. Svarīgākais ir tas, ka *Corda* viedie līgumi var pārstāvēt tiesības un pienākumus, kas izteikti juridiskajos pantos, un tie varētu būt juridiski izpildāmi.

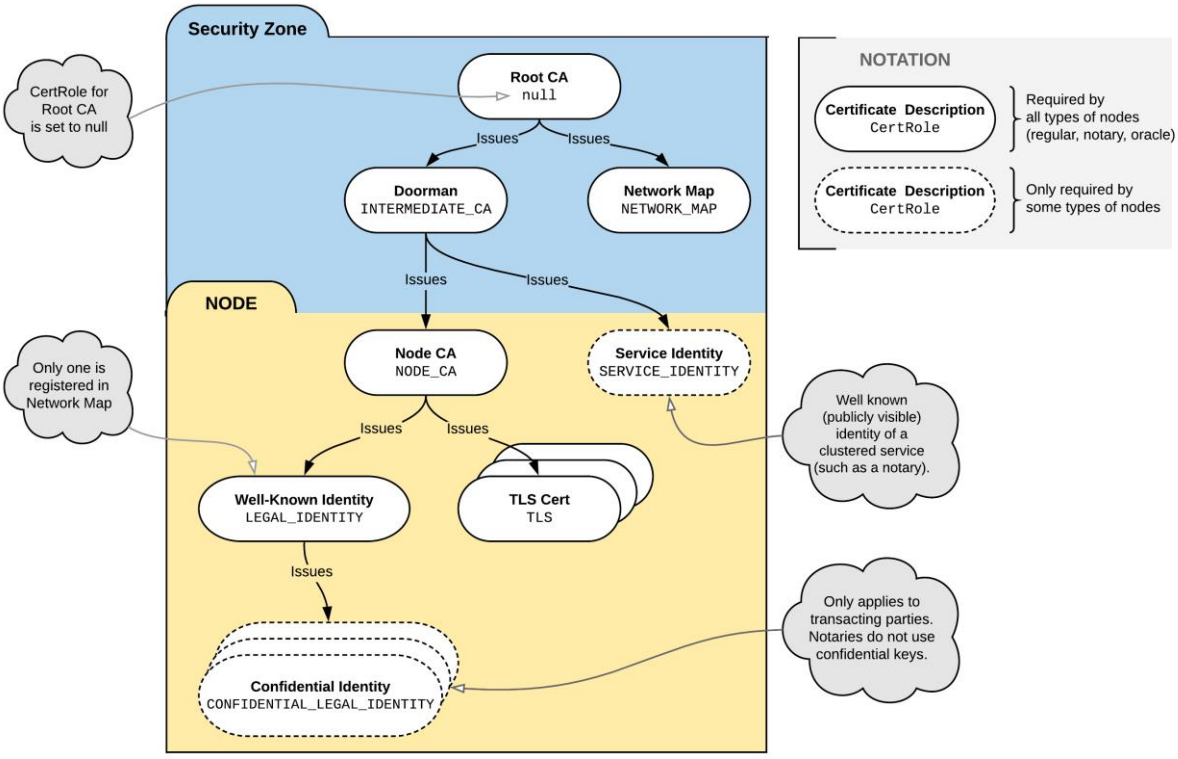
### *3.2.2. Arhitektūra*

*Corda* mezgls ir *JVM* darbības laika vide ar unikālu identitāti tīklā, kurā ir*Corda* pakalpojumi un *CorDapps*.

Corda tīkls sastāv no piecām galvenajām sastāvdaļām: [22]

1. Mezgli, kas sazinās ar *AMQP* (*Advanced Message Queuing Protocol*) ar *TLS*. Mezgliem ir datu glabāšanas relāciju datu bāze.
2. *TLS* (*Transport Layer Security*) sertifikātu izsniegšanas pakalpojums.
3. Tīkla kartēšanas pakalpojums, ko izmanto mezglu informācijas publicēšanai tīklā.
4. Viens vai vairāki notāra pakalpojumi, kas var tikt izplatīti vairākos mezglos.
5. Nulles pakalpojumi, kas palīdz virsgrāmatai pieslēgties reālajai pasaulei, parakstot transakcijas, ja tajās norādītie fakti tiek uzskatīti par patiesiem.

*Corda* ir blokķēdes tīkls, kuram, ir nepieciešama piekļuves atļauja. Katras puses identitāte ir zināma, sazinoties ar citiem. Piekļuvi tīkliem kontrolē arī *doorman*, kā tas tiek parādīts 3.2. att.




*3.2. att.* **Doorman darbība [23]**

*Corda* izmanto punkta-punkta sakarus jeb *point-to-point communication*, nevis globālo apraidi, kā daudzas citas blokķēdes platformas. Šie punkta-punkta sakari tiek saukti par plūsmām jeb *flows*. Plūsmas ļauj noteikt, kas ir attiecīgās transakcijas puses, un kāda veida

informācija ir nepieciešama. Tās var darboties vienlaicīgi, kas ļauj mezgliem darboties daudzās plūsmās, jo dažas plūsmas var ilgt vairākas dienas, saglabājot mezgla restartēšanu vai jauninājumu. [23]

Corda izmanto neizmantotās transakcijas modeli jeb *UTXO*, kas nozīmē, ka transakcijas veikšanai ir jāizmanto kāds no esošajiem stāvokļiem un jāizdod jauns stāvoklis. Ar to ir saistīts konsensa stāvoklis.

*Corda* izmanto divas galvenās metodes, lai atrastu konsensu starp virsgrāmatas jauninājumiem. Pirmkārt, validitātes konsenss pieprasa, lai transakcijas būtu apstiprinātas ar ievades un izvades stāvokļa līgumiem. Otrkārt, tā pārbauda, vai transakcijai ir visi nepieciešamie paraksti. Tālāk transakciju var uzskatīt par derīgu, kas padara to par katras transakcijas priekšnoteikumu, bet ar to nepietiek, lai pārbaudītu pašu transakciju. Tas prasa, lai neviena no piedāvātajām transakcijas ievadēm iepriekš nebūtu veikta ar citu parakstītu transakciju. Unikalitātes konsenss novērš dubulto izdevumu problēmu *Corda* tīklā. [16]

### *3.2.3. Transakcija*

Izveidojot jaunu transakciju, izvades stāvoklis norāda, ka transakcija vēl neeksistē, un tāpēc tā ir jāizveido transakcijas iesniedzējiem. Tomēr ievades stāvokļi jau pastāv kā iepriekšējo transakciju izvades stāvokļi. Tādēļ tie tiek iekļauti piedāvātajā transakcijā pēc saites. Šīs ievades stāvokļu saites sastāv no: [22]

- transakcijas jaucējkoda jeb *hash*, ko izveido ievades stāvoklis;
- ievades indeksa iepriekšējās transakcijas izvades stāvokļos.

Šīs ievades stāvokļa saites savieno transakcijas laika gaitā, veidojot to, kas ir pazīstams kā transakcijas ķēde. Sākotnēji transakcija ir tikai priekšlikums, lai atjauninātu virsgrāmatu. Tas atspoguļo virsgrāmatas turpmāko stāvokli, ko vēlas veikt transakcijas veidotājs.

Lai transakcija tiktu īstenota, ir jāsaņem paraksti no visiem nepieciešamajiem parakstītājiem. Katrs nepieciešamais parakstītājs pievieno transakciju parakstam, lai norādītu, ka viņi apstiprina priekšlikumu. Ja tiek apkopoti visi nepieciešamie paraksti, tiek veikta transakcija. Nepieciešamajiem parakstītājiem transakcija jāparaksta tikai tad, ja ir divi nosacījumi: [24]

- **Transakcijas derīgums**: gan piedāvātajai transakcijai, gan katrai transakcijai ķēdē, kas izveidojusi piedāvātās transakcijas ievades:

- transakciju digitāli paraksta visas nepieciešamās puses;
- transakcija ir derīga pēc līguma.

- **Transakcijas unikalitāte**: nepastāv neviena cita veikta transakcija, kas ir patērējusi kādu no ievades stāvokļiem piedāvātajās transakcijās.

Ja transakcija savāc visus nepieciešamos parakstus, bet šie nosacījumi nav spēkā, transakcijas izvades stāvokļi nebūs derīgi un netiks pieņemti kā ievades stāvokļi turpmākajās transakcijās. Transakcijas caur *Corda* var notikt tikai starp iepriekš noteiktu pušu skaitu, un transakciju rezultāti nav publiski.

### *3.2.4. Tehniskais pētījums*

Šajā sadaļā tika aprakstīts veiktais izvēlēto tehnoloģiju tehniskais pētījums, ņemot par pamatu piemērus, kas atrodas šo tehnoloģiju dokumentācijās. Tehniskā pētījuma gaitā tika apskatītas doto platformu tehniskās iezīmes, kas palīdzēs izvēlēties optimālāko risinājumu *DLT* tehnoloģiju integrācijā esošajās finanšu tehnoloģiju uzņēmumu sistēmās.

#### *Piemērs*

Tika veikts tehnisks pētījums, balstoties uz piemēru, kas atrodas *Corda* dokumentācijā. Dotajā piemērā pilnā mērā ir apskatītas visas dotās platformas tehniskās iezīmes.

Balstoties uz *IOU* mācību materiāliem https://github.com/corda/corda-training-template un *Corda wiki* [25], tiek izskatīts piemērs „IOU" (*I owe you*) process, kur kāds pieprasa naudu, kas tiks atmaksāta vēlāk.

Apraksts un instalācijas soļi ir pieejami *GitHub* repozitorijā zem *README.md* teksta datnes. Tos ir iespējams apskatīt *GitHub* repozitorijā pēc saites: https://github.com/wonderbeak/master-corda-example

#### *Notārs*

Notāra klasteris ir tīkla pakalpojums, kas sniedz unikalitātes konsensu, apliecinot, ka dotajai transakcijai tas vēl nav parakstījis citas transakcijas. Saņemot pieprasījumu transakcijas notariālai apstiprināšanai, notāra klasteris vai nu: [24]

- pieraksta transakciju, ja tas vēl nav parakstījis citas transakcijas, kas patērē kādu no ierosinātajiem transakcijas ievades stāvokļiem;
- noraida transakciju un atzīmē, ka ir noticis *double-spend* mēģinājums.

To darot, notāra klasteris nodrošina gala punktu sistēmā. Līdz notāra klastera paraksta iegūšanai puses nevar būt pārliecinātas, ka vienlīdz derīga, bet pretrunīga transakcija netiks uzskatīta par "derīgu" mēģinājumu tērēt noteiktu ievades stāvokli.

Tātad pēc notāra klastera paraksta iegūšanas tiek iegūta pārliecība, ka piedāvātās transakcijas ievades stāvokļi vēl nav patērējuši iepriekšējo transakciju. Tādējādi notariālais apstiprinājums ir sistēmas galējais punkts.

### *Stāvoklis*

Stāvoklis atspoguļo dalītos faktus virsgrāmatā, tāpēc tālāk tiks definētas stāvokļa shēmas. Stāvoklis nesatur *setters*, jo *Corda* stāvokļa objekti nav maināmi, kas nozīmē, ka viena stāvokļa gadījumu vērtības nav iespējams mainīt. Tā vietā tiek izmantota stāvokļa secība, kas nozīmē, ka, ja stāvoklis tiek izlietots, vecais stāvoklis tiek atzīmēts kā vēsturisks, un jauns stāvoklis ar jaunām vērtībām tiek ievietots virknē. Visi vēsturiskie stāvokļi un pašreizējie nepatērētie stāvokļi tiek glabāti *Vault* - mezgla specifiskajā relāciju datubāzē.

### *Līgums*

Līgumu izmanto, lai noteiktu, kā saistītais stāvoklis attīstās laika gaitā. *Corda* līgumi atšķiras no citiem *DLT* platformas viedajiem līgumiem attiecībā uz izmantošanu. *Corda* līgumos ir noteikts, kādas transakcijas ir atļauts veikt. Ja ievades vai izvades stāvoklis neatbilst līgumam, tad transakcija tiek uzskatīta par nederīgu. Tas ir nepieciešams, lai nodrošinātu, ka līgums ir izveidots vienreiz, lai izvairītos no dubultiem ierakstiem.

Runājot par transakcijas validitāti, tiek saprasta ne tikai nepieciešamo parakstu esamība, bet arī validitāte, kas noteikta līgumā. Katra transakcija ir saistīta ar līgumu, kas to pieņem un apstiprina ievades un izvades stāvokļus. Transakcija tiek uzskatīta par validētu tikai tad, ja visi tās stāvokļi ir validēti. Līgumi *Corda* ir rakstīti jebkurā *JVM* valodā (piemēram, *Java*, *Kotlin*).

Transakcijas validitātei ir jābūt deterministiskai, t.i. līgumam vienmēr ir jāpieņem vai jānoraida transakcija. Šajā gadījumā transakcijas validitāte nevar būt atkarīga no laika, izlases

numuriem, tīkla mezglu failiem utt. *Corda* līgumi tiek izpildīti tā sauktajā smilšu kastē - nedaudz pārveidotā *JVM*, kas garantē deterministisku līgumu izpildi. [16]

***Plūsma***

Šī daļa ļauj izstrādātajam noteikt soļu secību, ko *Corda* mezgls var veikt, lai atjauninātu virsgrāmatu noteiktā veidā. Plūsma ir soļu secība, kas norāda mezglam, kā veikt konkrētu virsgrāmatas atjauninājumu, un kādā gadījumā ir nepieciešams parakstīt un validēt transakciju.

Dažreiz ir nepieciešamas stundas, dienas, līdz transakcija tiek parakstīta no visām pusēm un iekrīt virsgrāmatā. Kas notiek, ja atspējot mezglu, kas iesaistīts transakcijā? Plūsmām ir kontrolpunkti, kuros plūsmas stāvoklis tiek ierakstīts mezglu datubāzē. Atjaunojot mezglu tīklā, tas turpinās no tās vietas, kur apstājās.

## 3.3. B2C risinājums Ethereum

*Ethereum* ir vispopulārākā decentralizētā blokķēdes platforma pēc *Bitcoin*, kas ļauj veidot citas uz blokķēdi balstītas decentralizētas lietojumprogrammas jeb *dApps*. Tās tiek veidotas, izmantojot *Ethereum* viedos līgumus.

Tas tiek paveikts, izmantojot iebūvēto programmēšanas valodu *Solidity*. Tā ļauj ikvienam viegli izstrādāt decentralizētas lietojumprogrammas un viedos līgumus, kuros ir iespējams definēt īpašumtiesības, transakciju formātus un pārejas funkcijas [26]. Visi kodi ir apkopoti *Ethereum* virtuālajā mašīnā un ievietoti rindā blokķēdē to izpildei. Neskaitot *Solidity*, ir dažas citas iespējas, kuras *Ethereum* komanda ir izveidojusi, piemēram:

- *Serpent* - valoda, kas līdzīga *Python*;
- *LLL* - valoda, kas līdzīga *Lisp*.

Tomēr *Solidity* paliek vispopulārākā un vislabāk atbalstītā iespēja no iepriekš minētajām valodām.

*Ethereum* atbalsta divus konsensa protokolus: *PoW* un *PoA*. Galvenais publiskais *Ethereum* tīkls konsensam izmanto *PoW*. Savukārt sava privātā *Ethereum* tīkla izvietošanai ir jāizmanto *PoA*. Lai blokķēde būtu droša, *PoW* pieprasa lielu skaitļošanas jaudu, tāpēc tas ir labs publiskai blokķēdes izmantošanai, savukārt *PoA* nav vajadzīgas šādas skaitļošanas jaudas - lai panāktu konsensu, ir nepieciešami daži autoritātes mezgli tīklā.

### *3.3.1. Arhitektūra*

Vienkāršība, universālums un modularitāte ir daži galvenie dizaina principi, kas izriet no *Ethereum* arhitektūras [26]. Tas nozīmē, ka *Ethereum* ir izveidojis pēc iespējas vienkāršāku platformu, cerot, ka tā sasniegs plašu izplatību starp izstrādātājiem. Turklāt platforma nav koncentrēta uz iekļautajām funkcijām, bet nodrošina rīkus ar iekšējo skriptu valodu, lai izveidotu jebkāda veida *dApp*.

Vienu no *Ethereum* svarīgākajiem konceptiem, ko izmanto, lai definētu stāvokli, sauc par kontu. Katram kontam ir adrese, un visas stāvokļu pārejas nozīmē vērtības un informācijas pārsūtīšanu starp kontiem. Katrs konts sastāv no 4 laukiem [26]:

1. Vienreizējais kods;
2. Pašreizējais *ether* līdzsvars;
3. Līguma kods (tikai tad, ja tas ir līguma konts);
4. Krātuve (tukša pēc noklusējuma).

Tiek izdalīti 2 kontu veidi: līguma konts un ārējais konts. Būtībā ārējos kontus pārvalda cilvēki. Minētos kontus kontrolē privātās atslēgas un tajos nav nekādu kodu. Viņi var sūtīt ziņojumus, vienkārši veidojot un parakstot transakciju. Savukārt, lai *dApp* varētu darboties, ir nepieciešami līgumu konti. Tie darbojas ar to iekšējā līguma kodu palīdzību. Katru reizi, kad līguma konts saņem ziņojumu, kods tiek aktivizēts. Tas ļauj lasīt un rakstīt iekšējo atmiņu, sūtīt ziņas vai izveidot jaunus līgumus.

### *3.3.2. Transakcija*

Termins „transakcija” tiek izmantots *Ethereum*, lai atsauktos uz parakstītu datu paku, kas glabā ziņojumu, kas ir jānosūta no ārējā konta uz citu kontu blokķēdē. Lai gan transakcijas izmanto dažādiem mērķiem, to struktūra ir vienāda: [26]

- ***From*:** Transakcijas sūtītājs. Tā ir 20 baitu adrese, kas pārstāv kontu, kas uzsāk šo transakciju.
- ***To*:** Transakcijas saņēmējs. Tā arī ir 20 baitu adrese. Atkarībā no izmantošanas, tas var būt cits ārējais konts, līguma konts vai vienkārši tukšs konts.
- ***Value*:** Līdzekļu daudzums *wei* (1 ether = $10^{18}$ *weis*), kas jānosūta no “From” uz “To”. Ja “To” ir ārējais konts, tas ir vienkārši līdzekļu pārskaitījums. Ja “To” ir līguma adrese,

tas ir līdzekļu apjoms, kas nodots izveidotajam līgumam. Līgums ir kodēts tā, ka tas var apstiprināt līdzekļus.

- ***Data/Input***: Šis datu lauks galvenokārt ir saistīts ar līgumiem. Jauna līguma izveidošana ir baitu kods un kodētie argumenti. Līgumfunkcijas izpildei tajā ir funkcijas paraksts un kodētie argumenti. Šis datu lauks paliek tukšs, ja tiek veikti līdzekļu pārskaitījumi.
- ***Gas Price*** un ***Gas Limit***: Abi ir saistīti ar šīs transakcijas izmaksu apstrādi. Katram transakcijas apstrādes posmam ir iepriekš definēta gāzes vienība (*gas unit*). Gāzes cena (*Gas Price*) ir summa (*wei*) vienai gāzes vienībai. Gāzes limits (*Gas Limit*) ir maksimāls gāzes vienību skaits, kas iztērēts šai transakcijai. Maksimālā gāzes vienība, ko iztērē transakcija, nepārsniegs gāzes limitu kā aizsardzība gadījumā, ja notiek kāda neatbilstība transakcijas apstrādes gaitā.

***Transakcijas parakstīšana***

Transakcijas "parakstīšana" ir process, ar kuru tiek ģenerēts paraksts, izmantojot transakcijas sūtītāja privāto atslēgu. Pēc tam parakstītā transakcija tiek apstrādāta ar visām sekojošajām darbībām, līdz tā tiek iekļauta jaunizveidotā blokā.

***Transakcijas izmaksu novērtēšana***

Transakcijas kopējās *ether* izmaksas ir balstītas uz diviem faktoriem (*Total cost = gasUsed * gasPrice*):

- izmantotā gāze (gasUsed) ir kopējais gāzes daudzums, ko patērē transakcija;
- gāzes cena (gasPrice) ir cena (*ether*) par vienu gāzes vienību, kas noteikta transakcijā.

***gasUsed***

Katrai darbībai *Ethereum* virtuālajā mašīnā tiek piešķirts patērētās gāzes daudzums. *gasUsed* ir visu gāzu summa visām veiktajām darbībām. Lai novērtētu gāzi, tiek izmantots *Gas API*, ko var izmantot, bet tam ir daži brīdinājumi.

***gasPrice***

Lietotājs konstruē un paraksta transakciju, un katrs lietotājs var norādīt, kādas gāzes cenas viņš vēlas, kas var būt nulle. *Ethereum* klientiem *Frontier* noklusējuma gāzes cena bija 0,05e12 *wei*. Tā kā "maineri" (*miners*) optimizē savus ieņēmumus un ja lielākā daļa transakciju tiktu iesniegtas ar gāzes cenu 0,05e12 wei, būtu grūti pārliecināt "maineri" apstiprināt transakciju, kas ir zemāka vai nulles gāzes cena.

### *3.3.3. Tehniskais pētījums*

Šajā sadaļā tika aprakstīts veiktais izvēlēto tehnoloģiju tehniskais pētījums, ņemot par pamatu piemērus, kas atrodas šo tehnoloģiju dokumentācijās. Tehniskā pētījuma gaitā tika apskatītas doto platformu tehniskās iezīmes, kas palīdzēs izvēlēties optimālāko risinājumu *DLT* tehnoloģiju integrācijā esošajās finanšu tehnoloģiju uzņēmumu sistēmās.

***Piemērs***

Dotās platformas piemēru izstrādāja autors, balstoties uz *Ethereum* tehnoloģijas dokumentāciju [26]. Dotajā piemērā pilnā mērā ir apskatītas visas dotās platformas tehniskās iezīmes. Apraksts un instalācijas soļi ir pieejami *GitHub* repozitorijā zem *README.md* teksta datnes. Tos ir iespējams apskatīt *GitHub* repozitorijā pēc saites: https://github.com/wonderbeak/master-ethereum-example

***Tīkla komunikācija***

*Ethereum* ir publiska, uz blokķēdi balstīta izplatīta skaitļošanas platforma. To var uzskatīt par vienu lielu datoru, kas sastāv no maziem datoriem visā pasaulē. Ir iespējams veidot lietojumprogrammas un palaist tās uz šī lielā datora. Platforma garantē, ka jūsu lietojumprogramma vienmēr darbosies bez dīkstāves, cenzūras, krāpšanas vai trešo personu iejaukšanās.

Visi mezgli ir savienoti viens ar otru, un tiem ir koda un datu kopija. Kad tiek izvietots kods uz *Ethereum* blokķēdes, tas tiek kopēts visos tīkla mezglos. Kad lietojumprogramma saglabā jebkādus datus, pat šādi dati tiek kopēti visos mezglos. Tīklā ir tūkstošiem mezglu, un

ir pilnīgi neiespējami kādam apturēt visus šos mezglus. Tas nodrošina, ka lietojumprogramma ir vienmēr pieejama.

### *Lietojumprogrammu komunikācija*

Katrs klients sazinās ar savu lietojumprogrammas instanci. Nepastāv centrālais serveris, kuram visi klienti varētu pieslēgties. Tas nozīmē, ka ideālā decentralizētā pasaulē katrai personai, kas vēlas mijiedarboties ar *dApp*, būs nepieciešama pilnīga blokķēdes kopija, kas darbojas viņu datorā utt. Tas savukārt nozīmē, ka pirms lietojumprogrammas izmantošanas, ir jāielādē visa blokķēde un pēc tam jāsāk lietot lietojumprogrammu.

Decentralizācijas ideja nav paļauties uz vienu centralizētu serveri. Līdz ar to kopiena ir izstrādājusi risinājumus, kur nav jātērē daudz cietā diska un operatīvas atmiņas, lejupielādējot un palaižot pilnīgu blokķēdes kopiju.

## *Datubāze*

Katra transakcija tīklā tiek saglabāta blokķēdē. Kad tiek izvietota lietojumprogramma, šis process tiek uzskatīts par transakciju. Visas šīs transakcijas ir publiskas un jebkurš tās var redzēt un validēt. Šie dati nekad nevar tikt mainīti. Lai pārliecinātos, ka visiem tīkla mezgliem ir tāda pati datu kopija un lai šajā datubāzē netiktu ierakstīti nederīgi dati, *Ethereum* izmanto algoritmu, ko sauc par *proof-of-work*, lai nodrošinātu tīkla drošību.

### *dApp* kods

Blokķēdes datubāzes tikai glabā transakcijas. *Ethereum* tiek rakstīts loģikas / lietojumprogrammas kods valodā, ko sauc par *Solidity*. Pēc tam tiek izmantots *Solidity* kompilators, lai apkopotu kodu *Ethereum* baitu kodam, lai izvietotu šo baitu kodu blokķēdē. Tātad *Ethereum* blokķēde ne tikai glabā transakcijas, bet arī saglabā un izpilda līguma kodu. Būtībā blokķēde glabā datus, glabā kodu un arī vada kodu *Ethereum* virtuālajā mašīnā. Savukārt, lai izveidotu tīmekļa *dApps*, *Ethereum* piedāvā ērtu *javascript* bibliotēku, ko sauc par *web3.js*, kas pieslēdzas blokķēdes mezglam.

## 3.4. B2B risinājums Hyperledger Fabric

*Hyperledger Fabric* jeb *HLF* ir platforma, kas ļauj veidot uz blokķēdi balstītas lietojumprogrammas, kurām nepieciešama piekļuves atļauja. To var uzskatīt par atvērtu pirmkodu, kas paātrinātu starpnozaru blokķēdes tehnoloģijas attīstību. *Hyperledger* ir arī pasaules mērogā koordinēts darbs, kas paredz naudas ietaupīšanu, ražošanas tīklu paplašināšanu un inovācijas. To izmanto, lai izveidotu sadalāmus ieraksta pielikumus no blokķēdes. Tāpat kā citas blokķēdes tehnoloģijas, tas tiek piegādāts ar virsgrāmatu un izmanto viedos līgumus, kas ļauj tam darboties kā sistēmai, kurā cilvēki var pārvaldīt transakcijas.

### *3.4.1. Arhitektūra*

*Hyperledger Fabric* atbalsta tikai sadalītu arhitektūru, un bloku izveidošana ir atkarīga no centrālā uzticamā mezgla, ko sauc par pasūtītāju. Tas atbalsta viedos līgumus, tīkla atļauju, privātumu un citas funkcijas.

*Hyperledger Fabric* ir iezīme, ko sauc par transakcijas apstiprinājumu, kas nodrošina mehānismu apstiprinājumu saņemšanai no atsevišķām pusēm pirms transakcijas nosūtīšanas. Kad dalībnieks tīklā ir apstiprinājis transakciju, tiek saprasts, ka dalībnieks ir pārbaudījis transakciju. Katram ķēdes kodam ir apstiprināšanas politika, kas nosaka kādiem dalībniekiem jāapstiprina ar šo ķēdes kodu saistītās transakcijas. Noklusējuma politika nosaka, ka katram kanāla dalībniekam ir jāparaksta transakcija.

*HLF* ir īpaša veida mezgls, ko sauc par *OSN*, ko uztur uzticama puse. *OSN* veido blokus un izplata tos vienādranga tīklos. Tā kā šis ir uzticams mezgls, konsenss nav nepieciešams. Šobrīd *HLF* atbalsta *CouchDB* un *LevelDB*, lai saglabātu blokķēdes stāvokli. Vienrangi (*peers*) tīklā pēc noklusējuma saglabā blokķēdes stāvokli *LevelDB* datubāzē. [27]

Tāpat viena kanāla vienrangi pārraida blokus viens otram neatkarīgi no *OSN* esamības vai neesamības, taču *OSN* neesamības gadījumā nav iespējams izveidot jaunus blokus kanālam. Vienrangi pārraida blokus, izmantojot speciālu protokolu, ko sauc par *"gossip data dissemination protocol". [27]*

*HLF* ietver arī kanālu konceptu privātuma nodrošināšanai. Kanāls atrodas zem blokķēdes tīklā un ļauj noteiktām pusēm kļūt par tā daļu. Faktiski katrai transakcijai ir jābūt piesaistītai kanālam, un, kad tiek izvietots *HLF* tīkls, tiek izveidots noklusējuma kanāls. *OSN* var redzēt

visus datus visos kanālos, tādēļ tā ir uzticama puse. Tehniski ir iespējams konfigurēt tīklu, lai tajā būtu vairāki *OSN*, kas mitina dažādus kanālus, ja nevar uzticēties vienai pusei visos kanālos.

Ja datplūsma ir milzīga vai *OSN* pieejamība ir kritiska, ir iespējams pievienot *Kafka OSN*, lai uzlabotu veiktspēju un palielinātu stabilitāti. Ir iespējams izmantot vairākus *OSN* vienā kanālā, kas savienots ar *Kafka*.

*Hyperledger Fabric* var rakstīt ķēdes kodus *Java* vai *Go* programmēšanas valodās. Nākotnē *HLF* ietvers *Simple Byzantine Fault Tolerance jeb* SBFT konsensa protokolu un dažas citas funkcijas, kas ļaus veidot d*Apps*. Tāpat tiek izstrādātas dažādas jaunas iezīmes, kas nākotnē tiks izlaistas kā produkta apakšversija. [16]

### *3.4.2 Transakcija*

Transakcija *Hyperledger Fabric* ir blokķēdes stāvokļa atjaunināšana, ķēdes koda izpildes rezultāts. Transakcija sastāv no ķēdes koda izsaukuma pieprasījuma ar dažiem argumentiem, ko paraksta izsaucošais mezgls, un atbilžu kopas no mezgliem, uz kuriem tika veikts transakcijas "apstiprinājums". Savukārt atbildes satur informāciju par bloku *Read-Write-Set* ķēdes atslēgas vērtības statusa maiņu un pakalpojuma informāciju. Tā kā atsevišķu kanālu bloku ķēdes ir fiziski atdalītas, transakciju var veikt tikai viena kanāla kontekstā.

*Hyperledger Fabric* izmanto transakcijas izpildes un izplatīšanas arhitektūru, kurā ir 3 pamatdarbības: [27]

- izpilde - ko izveido viedais līgums, kas darbojas vienā vai vairākos tīkla mezglos, transakcijas - sadalītās virsgrāmatas izmaiņu stāvoklis;
- pasūtīšana - transakciju secības izveidošana un grupēšana blokos, izmantojot specializētu pasūtītāja pakalpojumu sniedzēju, kas izmanto pieslēdzamu konsensa algoritmu;
- apstiprināšana - pasūtītāja transakciju pārbaude, ko veic tīkla mezgli pirms informācijas izvietošanas savā sadalītās virsgrāmatas kopijā.

Šī pieeja ļauj veikt transakcijas izpildi pirms tā nonāk blokķēdes tīklā, kā arī horizontāli mērogot tīkla mezglu darbību.

### *3.4.3. Tehniskais pētījums*

Šajā sadaļā tika aprakstīts veiktais izvēlēto tehnoloģiju tehniskais pētījums, ņemot par pamatu piemērus, kas atrodas šo tehnoloģiju dokumentācijās. Tehniskā pētījuma gaitā tika apskatītas doto platformu tehniskās iezīmes, kas palīdzēs izvēlēties optimālāko risinājumu *DLT* tehnoloģiju integrācijā esošajās finanšu tehnoloģiju uzņēmumu sistēmās.

***Piemērs***

Tika veikts tehnisks pētījums, balstoties uz piemēru, kas atrodas *Hyperledger Fabric* dokumentācijā. Dotajā piemērā pilnā mērā ir apskatītas visas dotās platformas tehniskās iezīmes.

Balstoties uz tīkla dokumentāciju [34]: tiek izskatīts dotās platformas piemērs. Apraksts un instalācijas soļi ir pieejami *GitHub* repozitorijā zem README.md teksta datnes. Tos ir iespējams apskatīt *GitHub* repozitorijā pēc saites: https://github.com/wonderbeak/master-hyperledger-example

***Blokķēdes lietojumprogrammas***

No izstrādātāja viedokļa blokķēdes lietojumprogramma sastāv no divām galvenajām daļām: [16]

- **On-chain -** viedie līgumi, kas darbojas blokķēdes tīkla izolētajā vidē, definējot transakciju atribūtu izveidošanas un sastāva noteikumus. Viedajā līgumā galvenās darbības ir datu nolasīšana, atjaunināšana un dzēšana no blokķēdes tīkla stāvokļa. Jāuzsver, ka datu dzēšana no stāvokļa atstāj informāciju par to, ka šie dati bija pieejami.
- **Off-chain** - lietojumprogramma, kas ar *SDK* starpniecību mijiedarbojas ar blokķēdes vidi. Mijiedarbība nozīmē viedo līgumu funkciju noskaidrošanu un viedā līguma notikumu novērošanu - ārējie notikumi var izraisīt datu maiņu viedajā līgumā, bet viedā līguma notikumi var izraisīt darbības ārējās sistēmās.

Datus parasti nolasa, izmantojot blokķēdes tīkla "mājas" mezglu. Lai ierakstītu datus, lietojumprogramma nosūta pieprasījumus organizāciju mezgliem, kas piedalās konkrēta viedā līguma apstiprināšanas politikā.

Lai izstrādātu *off-chain* kodu, tiek izmantots specializēts *SDK*, kas iekļauj mijiedarbību ar blokķēdes mezgliem, apkopo atbildes utt.

### *Ķēdes kods*

Ķēdes kods, ko var saukt arī par viedo līgumu, ir programma, kas nosaka noteikumus transakciju veidošanai, kas maina blokķēdes stāvokli. Programma tiek izpildīta vienlaicīgi uz vairākiem neatkarīgiem mezgliem sadalītā tīkla blokķēdē, kas veido neitrālu vidi viedo līgumu izpildei, pārbaudot programmas izpildes rezultātus visos mezglos, kas nepieciešami transakciju apstiprināšanai. Ķēdes kodam ir jāievieš saskarne, kas sastāv no metodēm:

- *Init* metode tiek pielietota ķēdes koda instancēšanas vai jauninašanas laikā. Šajā metodē tiek veikta nepieciešamā ķēdes koda stāvokļa inicializācija. Metodes kodā ir svarīgi atšķirt, vai izsaukums ir instancēšana vai jauninašana, lai kļūdas dēļ nevajadzētu inicializēt (anulēt) datus, kas ķēdes koda darba laikā ir saņēmuši nulles stāvokli.
- *Invoke* metode tiek izmantota, pielietojot jebkuru ķēdes koda funkciju. Šajā metodē tiek strādāts ar viedo līgumu stāvokli.

Ķēdes kods tiek instalēts blokķēdes tīkla mezglos. Sistēmas līmenī katrs ķēdes koda eksemplārs atbilst atsevišķam *Docker* konteineram, kas saistīts ar konkrētu tīkla mezglu, kurš nosūta izsaukumus, lai izpildītu ķēdes kodu.

### *Apstiprināšanas politika*

Apstiprināšanas politika definē konsensa noteikumus transakciju līmenī, ko izveido noteikts ķēdes kods. Politika paredz noteikumus, kas nosaka, kādiem kanāla mezgliem ir jāizveido transakcija. Lai to izdarītu, katram no apstiprināšanas politikā norādītajiem mezgliem ir jādarbojas ar ķēdes koda metodi (izpildes solis), veicot "simulāciju". Tālāk parakstītie rezultāti tiek apkopoti un pārbaudīti ar *SDK*, kas ierosināja transakciju (visiem simulācijas rezultātiem jābūt identiskiem, visiem politikā noteikto mezglu parakstiem jābūt klāt). Pēc tam *SDK* nosūta transakciju pasūtītājam, tālāk visi mezgli, kuriem ir piekļuve kanālam, izmantojot pasūtītāju, saņems transakciju un veikts validācijas soli.

Ir svarīgi uzsvērt, ka ne visiem kanāla mezgliem ir jāiesaistās izpildes solī. Apstiprināšanas politika tiek noteikta ķēdes koda instancēšanas vai jauninašanas laikā.

### *Tīkla mezgls*

Tīkla mezgls tiek pieslēgts nejaušam skaitam kanālu, kuriem mezglam ir piekļuves tiesības. Tīkla mezgls uztur savu bloku ķēdes versiju un bloku ķēdes stāvokli, kā arī nodrošina vidi ķēdes kodu sākšanai.

Ja tīkla mezgls nav iekļauts apstiprināšanas politikā, tad uz tā var nebūt uzinstalēti ķēdes kodi. Tīkla mezgla programmatūras līmenī esošais bloku ķēdes stāvoklis (*world state*) var tikt glabāts uz *LevelDB* vai *CouchDB*. *CouchDB* priekšrocība ir atbalsts paplašinātajiem vaicājumiem, kas izmanto *MongoDB* sintaksi.

### *Pasūtītājs*

Transakciju secības pakalpojums pieņem parakstītas transakcijas un nodrošina, ka tās tiek sadalītas tīkla mezglos pareizā secībā. Pasūtītājs nepalaiž viedos līgumus un neuztur bloku ķēdi un bloku ķēdes stāvokli.

### *Identifikācijas pakalpojums*

Tīklā visiem dalībniekiem ir zināmi citu dalībnieku rekvizīti. Identifikācijas procesa laikā tiek izmantota publiskās atslēgas infrastruktūra (*PKI*), ko izmanto, lai izveidotu *X.509* sertifikātus organizācijām, infrastruktūras elementiem, lietojumprogrammām un gala lietotājiem. Tā rezultātā piekļuvi datiem var kontrolēt, izmantojot piekļuves noteikumus tīkla līmenī, atsevišķā kanālā vai viedā līguma loģikā. Vienā blokķēdes tīklā vienlaicīgi var darboties vairāki dažāda veida identifikācijas pakalpojumi.

# 4. DLT RISINĀJUMU SALĪDZINĀJUMS

Šajā nodaļā tiks salīdzinātas trīs iepriekš minētās platformas. Platformas tiks salīdzinātas piecos aspektos: galvenais izmantošanas gadījums, arhitektūra, konsenss, valodas atbalsts un piemēru ieviešana. Šiem punktiem būtu jāattiecas uz bieži uzdotajiem jautājumiem, kas izstrādātājiem varētu rasties, izvēloties platformu *dApp* izstrādei. Galvenais lietošanas gadījums norādīs izstrādātājam, kāda veida lietojumprogrammu platforma ir paredzēta.

Arhitektūras punktā tiks izcelti ietekmīgākie attīstības cikla aspekti un tas, kā to sasniegt. Pēc tam tiks apskatīti konsensa algoritmi. Valodas atbalsts palīdz izstrādātājiem pieņemt gala lēmumu, vai platforma ir piemērota komandai, vai ir jāizvēlas jaunas valodas. Visbeidzot, ieviešanas salīdzinājums atklāj dažas priekšrocības un trūkumus.

## 4.1. Galvenais lietošanas gadījums

Izvēlēto platformu galvenais izmantošanas gadījums ir ļoti atšķirīgs. *Ethereum* sākotnēji tika uzskatīts par jaunu alternatīvu protokolu decentralizētu lietojumu veidošanai. Tātad tā ir veidota kā vispārējas nozīmes platforma dažādu veidu lietojumiem. Dotā platforma ir vienkārša un universāla. Tas padara to par ideālu platformu uzņēmumiem uzsākt sākotnējo monētu piedāvājumu jeb *ICO*. Pierādījums tam ir *Ethereum* ķēdē uzsākto pieteikumu skaits.

*Corda* ir īpašs mērķis apvienot un vienkāršot transakcijas starp finanšu iestādēm, gadījumos, kas ietver vienkāršu naudas pārskaitījumu uz sarežģītākiem obligāciju, akciju un aizdevumu pārvedumiem. Katru transakciju var veikt ar reālās dzīves līgumu, kas nepieciešams finanšu iestādēm, kas nodarbojas ar lielām naudas summām. Tas padara platformu par nišas produktu, kas norāda uz mazāku lietotāju skaitu, kuriem tas ir paredzēts. No otras puses, tas, ka lielākās bankas atbalsta *Corda*, liecina par ieinteresētību šajā produktā. Nākotnē ir iespējama jaunu darbavietu izveide, kas saistīta ar *Corda*, gan tradicionālajos finanšu uzņēmumos, gan finanšu tehnoloģiju jaunuzņēmumos.

*Hyperledger* piedāvā „lietussarga" stratēģiju, inkubējot un popularizējot daudzas uzņēmējdarbības blokķēdes tehnoloģijas, sistēmas, bibliotēkas, saskarnes un lietojumprogrammas. Dažas no galvenajām priekšrocībām, izvēloties *Hyperledger* kā uzņēmuma blokķēdes risinājumu, ietver piešķirtas atļaujas dalību (*Hyperledger* ir piešķirtas atļaujas tīkls, kur visiem dalībniekiem ir zināmas identitātes), veiktspēju, mērogojamību un

uzticības līmeņus, kā arī digitālo atslēgu un jutīgu datu aizsardzību. Finanšu iestādes īpaši novērtē šo funkciju, lai aizsargātu klientu informāciju un citus sensitīvus dokumentus.

*Ethereum* platforma ir izsniegusi savu kriptovalūtu. Tas nav platformas galvenais izmantošanas gadījums, tomēr tam ir dažādi izmantošanas veidi, piemēram, kā ieguldīšanas līdzeklis vai valūta tiešsaistes tirdzniecībai. No otras puses, fakts, ka *Ethereum* tiek publiski tirgots, varētu ietekmēt platformu izstrādes virzienu, lai paaugstinātu valūtas cenas, jo tas ir galvenais finansējums attīstības komandai, kā arī platforma var tikt uzskatīta par mirušu, kad cena krasi krītas. *Corda* kā platformai nav nekāda sakara ar savu valūtu. To finansē konsorcijs, kas galvenokārt sastāv no bankām. Tā kā platformas pirmkods ir publiski pieejams, tajā netiek saskatīti trūkumi. Pretēji tam tajā tiek saskatītas priekšrocības, jo tas palīdz attīstības komandai pievērsties īpašām vajadzībām un piedāvāt instrumentu, no kura uzņēmumi varētu gūt labumu.

## 4.2. Arhitektūra

*Ethereum* ir būvēts kā publiska blokķēde bez piekļuves atļaujas nepieciešamības, kas nozīmē, ka ikviens var būt tīkla daļa, palīdzot atrast garāko ķēdi, kas nepieciešama datu validācijai. Visi virsgrāmatas atjauninājumi tiek pārraidīti visiem tīkla mezgliem. Daži drošības līdzekļi tiek nodrošināti ar mezglu anonimitāti. Viedie līgumi *Ethereum* ir tā sauktie autonomie aģenti tīklā, kas automātiski pārbauda un atvieglo transakcijas starp mezgliem. Kad transakcijas ir pabeigtas, rezultāta stāvoklis ir saistīts ar atbilstošajiem kontiem. Konti *Ethereum* var pārstāvēt gan klienta mezglus, gan līgumus. Izvēloties platformu, jāņem vērā transakcijas caurlaidība. Tā kā *Ethereum* ir publisks, un ķēdē esošo lietojumprogrammu skaits nepārtraukti pieaug, tas varētu kļūt par problēmu. Datu izmaiņas ir ļoti saistītas ar ieguves aspektu. Tas liecina, ka jaunu bloku pievienošana var kļūt daudz lēnāka. No otras puses, vispārējo arhitektūru var uzskatīt par vieglāku nekā *Corda*, jo pastāv tikai viedais līgums, kas uztur kontus.

Savukārt *Hyperledger* paredz iepriekš noteiktu dalībnieku kopienu un ļauj piekļūt tīklam. Tas nozīmē, ka atļaujai piekļūt datiem tīklā nepieciešama atļauja, kas var būt šifrēšanas atslēgu veidā. *Hyperledger* ieceļ privātumu citā līmenī, jo tikai tīklā iesaistītie lietotāji var piekļūt datiem. Arī pašu valūtas trūkums ļauj izmantot mērogojamu konsensa algoritmu, pateicoties kam, tīkls var apstrādāt transakcijas ar lielu ātrumu. *Hyperledger* ir ideāls projekts organizācijām un uzņēmumiem, kas vēlas izvairīties no problēmām, kas saistītas ar mērogojamību un

konfidencialitāti blokķēdes ietvaros. Atļautais darbības veids ievērojami palielina konfidencialitātes līmeni, jo tiek nodrošināta neliela piekļuves kontrole.

Savukārt *Corda* ir privāta blokķēdes platforma, kurai nepieciešama piekļuves atļauja. Tas nozīmē, ka "ieeja tīklā" ir bloķēta ar *doorman*, kur katram mezglam ir jāidentificē sevi. *Corda* izmanto punkta-punkta sakarus, kas pārsūta datus tikai pilnvarotām personām. Tas nodrošina datu drošību un zināmu transakciju anonimitāti. Nav arī centrālā datu glabāšanas punkta. Katram mezglam ir sava kopīgo faktu apakškopa virsgrāmatā. Līgumi *Corda* paredz transakcijas derīguma pārbaudi un tiek veikti automātiski. Līgumi *Corda* var būt juridiski saistoši, kas atvieglo konfliktu risināšanu starp pusēm, jo tās var paļauties uz parasto tiesību sistēmu.

Plūsmas ir transakcijas procesa atspoguļojums. Tās nosaka, kuras ir asociētās puses, kāda veida transakcijām ir jāparedz līgumi un kam tie ir jāparaksta. Savukārt apakšplūsmas tiek veidotas, lai pieprasītu līgumslēdzēju pušu specifisku pārbaudi. Tas liecina, ka arhitektūrai ir vairāk komponentu, kas no vienas puses apgrūtina izpratni vai plānošanu. No otras puses, tas nodrošina visus instrumentus, kas nepieciešami lietojumprogrammām, kas atbilst galvenajam lietošanas gadījumam, kas aplūkots iepriekšējā nodaļā. Parasti tas ir vajadzīgs sarežģītāku lietojumprogrammu veidošanai, neparakstot pārāk daudz tekstveidnes kodu.

## 4.3. Konsenss

*Ethereum* pašlaik izmanto *proof-of-work* algoritmu. Tas pieprasa, lai mezgli kopīgi pārbaudītu transakciju un pārraidītu jauno ķēdes stāvokli tīklā. Tas var kavēt platformas darbību. Iespējams, ka nākotnē *Ethereum* ieviesīs *proof-of-stake* algoritmu. Tas padarītu *Ethereum* mazāk atkarīgu no tīkla kolektīvās skaitļošanas jaudas un izmantotu dažādus algoritmus, lai ķēdei pievienotu jaunu transakciju bloku. Izmaiņas var izraisīt milzīgas lietotāju bāzes vai platformu popularitātes svārstības.

*Hyperledger* paredz cita veida konsensu, saskaņā ar kuru mezgliem ir atļauts izvēlēties starp konsensa un vienošanās protokolu. Šajā gadījumā divas vai vairākas puses var piekrist nolīgumam un būtiski ietekmēt rezultātu. Piemēram, *Hyperledger Fabric* izmanto *PBFT*.

*Corda* ir ieviesusi notāru konceptu, lai panāktu konsensu. Notāri pārbauda, vai konkrētā transakcija nepatērē kādu no ievades stāvokļiem, ko patērēja kāda no iepriekšējām jau parakstītām transakcijām. Konsensa otrā daļa tiek panākta, izmantojot *Corda* līgumus, kas pārbauda transakcijas ievades un izejas derīgumu un pieprasa parakstus no saistītām pusēm.

Attiecīgās puses var darboties ar tādu pašu līguma kodu neatkarīgi, kas turklāt palielina uzticību. Rezultātā transakcijas laikā tiek panākts nemainīgs ātrums. To neietekmē tīkla lietotāju skaits, un ātrums lielākoties ir atkarīgs no transakcijas pusēm. Tā kā galvenais lietošanas gadījums prasa, lai līgumi būtu juridiski saistoši, situācija, ka pretējās puses var patstāvīgi vadīt savu līguma kodu, ir ļoti izdevīga.

## 4.4. Programmēšanas valoda

*Ethereum* ir neatkarīga programmēšanas valoda *Solidity*. To ir viegli izmantot un iemācīties, jo to spēcīgi ietekmē citas skriptu valodas, piemēram, *Python* un *JavaScript*. *Solidity* galvenais mērķis ir padarīt *Ethereum* ķēdes līgumus vienkāršus un pieejamus. Visi kodi ir paredzēti *Ethereum* virtuālajai mašīnai. Tā kā *Solidity* ir vislielākā lietotāju bāze un atbalsts, ir ļoti ieteicams to pieņemt izstrādātājiem, kas plāno izmantot *Ethereum*. Tas nozīmē papildu piepūli un laiku, lai sasniegtu efektivitāti.

*Hyperledger* strādā pie ķēdes koda, kas ir sinonīms viedajam līgumam un apstrādā uzņēmējdarbības loģiku, kas tiek saskaņota starp tīkla dalībniekiem. Ķēdes kodi ir uzrakstīti programmēšanas valodā, ko izstrādājusi *Google* ar nosaukumu *Go*.

*Corda* kodola kods galvenokārt tiek izstrādāts *Kotlin* un darbojas ar *Java* virtuālo mašīnu. Tas nozīmē, ka var izmantot jebkuru *JVM* valodu, bet *Java* un *Kotlin* tiek atbalstītas oficiāli. Tām ir paraugi un pamācības par *Corda* dokumentāciju. Tā kā *Java* dominē uzņēmumu lietojumprogrammu izstrādē un tiek uzskatīta par visizplatītāko programmēšanas valodu, šobrīd finanšu uzņēmumiem būtu vieglāk sākt lietot *Corda*.

## 4.5. Lietošanas gadījuma pielietošana

Platformu pielietošana ir ļoti atšķirīga. To lielā mērā ietekmē platformu galvenais mērķis un no tā izrietošā arhitektūra. Viedā līguma rakstīšana uz *Ethereum* ar *Solidity* ir ātra un vienkārša, bet to savienojot ar ķēdi, ir vajadzīgi dažādi rīki un sistēma. *Ethereum* priekšrocības šeit ir lielāka lietotāju bāze un ilgāks laiks tirgū. [28]

*Hyperledger Fabric* ķēdes kodu izstrāde un testēšana, pateicoties nepieciešamībai izvietot ievērojamu skaitu blokķēdes tīkla komponentu, varbūt diezgan ilgs process, kuram ir lielas laika pārbaudes izmaksas.

Tādas pašas lietojumprogrammas pielietošana *Corda* aizņem daudz vairāk laika, veicot izmaiņas, lai to pielāgotu *Corda* arhitektūrai. Lielāko problēmu rada tas, ka *Corda* ir privāts tīkls, kam ir vajadzīga piekļuves atļauja. Lai izmantotu platformas piedāvātās koncepcijas, tiek izveidots mazāks tīkls ar diviem dalībniekiem. Rezultāti varētu būt ļoti atšķirīgi ar sarežģītākām un lielākām lietojumprogrammām no citām jomām.

# 5. CORDA INTEGRĀCIJAS DIZAINS

Šajā nodaļā tika aprakstīta reālā finanšu tehnoloģiju uzņēmuma problēma attiecībā uz pārrobežu maksājumu tehnoloģiju izmantošanu, kā arī izstrādātās praktiskās daļas procesa apraksts un risinājuma pārskats.

Transakciju caurspīdīguma palielināšanai maksājumos var izmantot *DLT* risinājumu, it īpaši viedos līgumus. Balstoties uz salīdzinājumu, kas tika veikts iepriekšējā nodaļā, tika secināts, ka finanšu tehnoloģiju uzņēmumiem atbilstošākais ir *Corda* risinājums, ko atbalsta finanšu iestāžu konsorcijs *R3*. Līdz ar to darba praktiskajā daļā tika izstrādāts *Corda* integrācijas dizains, kas ļaus finanšu tehnoloģiju uzņēmumiem integrēt jaunu risinājumu jau esošajās sistēmās. Risinājums tika izstrādāts, lai uzlabotu transakciju caurspīdīgumu, kā arī samazinātu izmaksas un laiku, kas nepieciešams pārrobežu maksājumu veikšanai.

Parasti transakcija sastāv no saņēmēja un sūtītāja informācijas, kas konstatē tikai pašu darījumu. Ar viedā līguma palīdzību ir iespējams sadalīt transakcijas mazos darījumos, kuru pamatā ir viedie līgumi. Tādā veidā ir iespējams nogādāt detalizētāku informāciju *AML* sistēmai un savlaicīgi veikt darbības, ja tiek konstatētas potenciāli aizdomīgas transakcijas pirms tika veikts darījums.

Regulatori pieprasa finanšu iestādēm veikt transakciju pārraudzību, kā arī klientu pārbaudi, un dinamiskais *Corda* risinājums ļauj to izdarīt.

## 5.1. Problēmas apraksts

Uzņēmums X ir finanšu tehnoloģiju uzņēmums, kas saviem klientiem piedāvā dažādus finanšu produktus. Tam ir plaša klientu bāze, kurai ir piekļuve saviem kontiem uzņēmumā. Starp klientiem notiek gan iekšējie, gan starptautiskie maksājumi *SWIFT* sistēmā. Augstas transakciju komisijas maksas sistēmā un ilgs izpildes laiks liek uzņēmumam meklēt jaunus risinājumus. Pāriešana uz jaunu kanālu *SWIFT GPI* vēl ir īstenošanas procesā.

Uzņēmums X pievienojās *R3* konsorcijam, lai attīstītu tehnoloģiju un sadarbotos ar citiem uzņēmumiem *Corda* tīklā. Saviem klientiem tiek izstrādāts jauns produkts, kas ļaus veikt darījumus un padarīs maksājumus caurspīdīgākus, sadalot tos, lai īstenotu noteiktus līgumus starp pusēm. Tā kā jau pastāv iekšējās sistēmas, kas klientam nodrošina gatavus produktus, jaunā tehnoloģija ir jāintegrē ar uzņēmuma eksistējošām sistēmām.

Jaunās regulatoru *AML* un *KYC* prasības, lai pārraudzītu transakcijas un pārbaudītu klientus, prasa lielāku caurspīdīgumu jebkurām uzņēmuma sistēmā veiktajām darbībām. Nākotnē uzņēmums plāno pārcelt savus risinājumus uz jaunu arhitektūru, tāpēc tam ir nepieciešami dinamiski risinājumi, kurus ir iespējams viegli mērogot.

Uzņēmums X vēlas izveidot caurspīdīgākas transakcijas, pateicoties transakciju detalizētākai sadalīšanai uz loģiskiem soļiem, kur pie transakcijas tiek pievienotas citas sistēmas atskaites vai dokumentu ievadīšana. Tāpat uzņēmums vēlas paplašināt produktu, lai klienti varētu reģistrēties *Corda* tīklā, un uzņēmums varētu piedāvāt savus resursus, lai piedāvātu viedo līgumu izstrādi.

## 5.2. Integrācijas mērķi

Šajā sadaļā tiks aprakstītas integrācijas sistēmas prasības un vēlamā uzvedība, balstoties uz reālas finanšu tehnoloģiju uzņēmuma problēmas apraksta. Lai veiksmīgi veiktu integrācijas dizaina izstrādi, vispirms ir nepieciešams nodefinēt un aplūkot prasības, kuras nepieciešamas risinājuma implementēšanai. Turpinājumā tiks apskatītas klientu un kontu attiecības *Corda* tīklā.

### *5.2.1. Prasības*

Jaunās sistēma paredz zemāk norādītās prasības, kuras nepieciešamas risinājuma implementēšanai. Tās ir:

- sistēmai būtu jāspēj pieņemt darījumus, izmantojot operatora specifisko *REST API;*
- *API* ir jāatbalsta operācijas - *CREATE, MODIFY, REJECT;*
- esošajām sistēmām ir jābūt spējai mijiedarboties ar jauno sistēmu, vienlaikus saglabājot kanāla drošību;
- decentralizēta lietojumprogramma dinamiskai darījumu izveidei ar dažādiem uzdevumu līmeņiem;
- darījuma formātam jābūt ar vairākiem dažādu uzdevumu veidu līmeņiem;
- uzdevumi varētu sastāvēt no dažāda veida apakšuzdevumiem;
- spēja integrēties *Ripple* tīklā un *Swift GPI;*
- atbalsts garantētam darba risinājumam ar iepriekš noteiktu atkārtošanas mehānismu;
- vienkārša pārslēgšanās uz dažādiem pakalpojumu sniedzējiem, jo pašreizējās uzņēmuma sistēmas varētu novecot nākotnē;

- visiem ziņojumiem risinājumā jābūt izsekojamiem. Saskaņā ar uzņēmuma standartiem reģistrācijai un uzraudzībai ir jāaptver katru risinājuma loģikas daļu;
- jāievēro *GDPR* - nodrošinot datu integritāti un konfidencialitāti;
- infrastruktūrai jābūt mērogojamai, jo klienti var izveidot jaunus mezglus (kā atsevišķs uzņēmuma X produkts);
- eksistējošu *AML* rīku integrācija;
- *Corda* mezgls *h2* atmiņas datubāzes replikācija;
- ātras piegādes iespējas.

### *5.2.2. Vēlamā uzvedība*

Klientam ir nepieciešams autentificēties esošajā uzņēmuma X sistēmā, droši pārbaudīt savu jau eksistējošu kontu. Viņam ir jābūt pieejai *Corda* viedajiem līgumiem. Tāpat klients, kas ir juridiska persona, var pievienot savus dalībniekus, kas var būt gan izstrādātāji, gan darbinieki. Kontam ir iespēja veidot darījumus ar citiem *Corda* tīkla klientiem, kas var būt gan uzņēmuma iekšējie klienti, gan ārējie klienti, kuri izmanto *Corda* tīklu.

Darījumus var sadalīt mazos uzdevumos, kurus iestata pats konts. Lai veiktu šos uzdevumus un tos pabeigtu, ir nepieciešams darba pierādījums, kas var izpausties galīgā dokumentā vai citas sistēmas *API* pieprasījumā. *AML* ir jāakceptē katra darbība sistēmā ar iespēju noraidīt darbību, lai savlaicīgi reaģētu uz aizdomīgu darījumu.

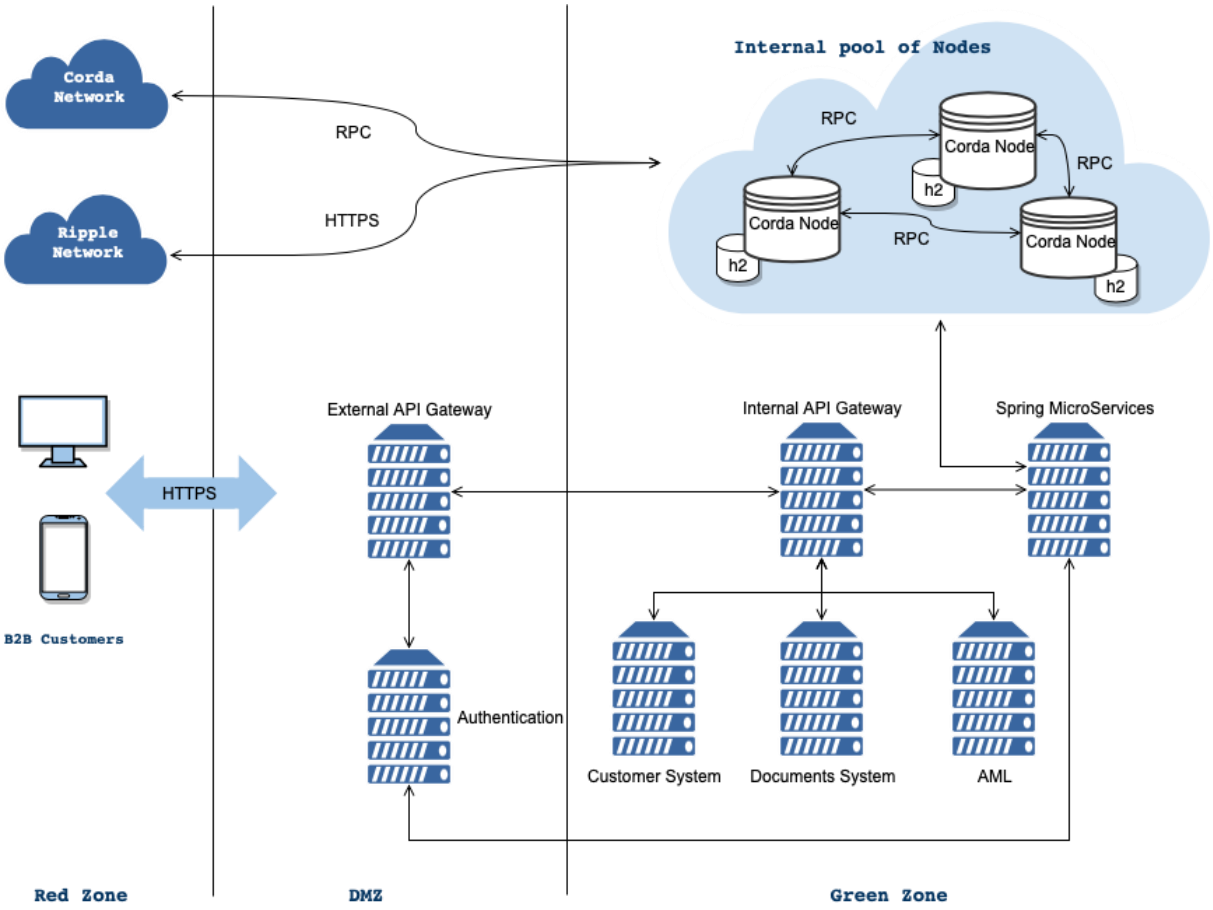
## 5.3. Arhitektūras pārskats

Šis risinājums integrē *Corda DLT* uzņēmuma X infrastruktūrā, tādējādi ļaujot esošajiem klientiem veikt starptautiskus darījumus tīklā, izmantojot gan esošos starptautiskos maksājumus, gan *Ripple*, lai samazinātu darījumu izmaksas.

### *5.3.1. Arhitektūras diagramma*

Arhitektūras diagramma ir grafisks konceptu kopuma attēlojums, kas ir daļa no arhitektūras, ieskaitot principus, elementus un komponentus. Sistēmu izstrādātājiem ir

nepieciešamas sistēmas arhitektūras diagrammas, lai saprastu, izskaidrotu un paziņotu sistēmas struktūru un lietotāju prasības.




*5.1. att.* **Arhitektūras diagramma**

Balstoties uz arhitektūras risinājumiem, kuri aprakstīti grāmatā "*Blockchain for Business*" [29], tika izveidota arhitektūras diagramma (sk. 5.1. att.), kurā tiek parādīta risinājuma dizains - kādā veidā ir iespējams uzņēmumā X ieviest šo risinājumu ar esošajām sistēmām. Par pamatu tiek ņemts *Corda* tīkls un uzņēmums X atbalsta šo *Corda* tīklu un nodrošina *back-end* pakalpojumus, kas tiek veidoti uz *Spring Framework*. Klienti var izmantot doto produktu no savām ierīcēm ar *HTTPS* protokolu, vispirms autorizējoties caur autentificēšanas sistēmu. Šāda veida arhitektūra palīdz aizsargāt datus no ārējiem uzbrukumiem, kā arī īsteno visas prasības, kas tika apskatītas iepriekš.

### *5.3.2. Arhitektūras stils*

Šajā risinājumā servisu izvietošanai, kas sazinās ar *Corda*, ir jāapsver mikroservisa arhitektūra, lai nodrošinātu sistēmas mērogošanas prasību izpildi. Mikroservisa arhitektūru ir lietojumprogrammu arhitektūra, kas tiek lietota, lai izveidotu uzņēmuma lietojumprogrammas. Klasiskajā arhitektūrā dažādas servera lietojumprogrammas funkcionālās sastāvdaļas.

Mikroservisa arhitektūrā servera lietojumprogramma ir sadalīta servisos. Mikroserviss ir mazs un neatkarīgs process, kas paredz servera lietojumprogrammas daļas noteiktu funkcionalitāti.

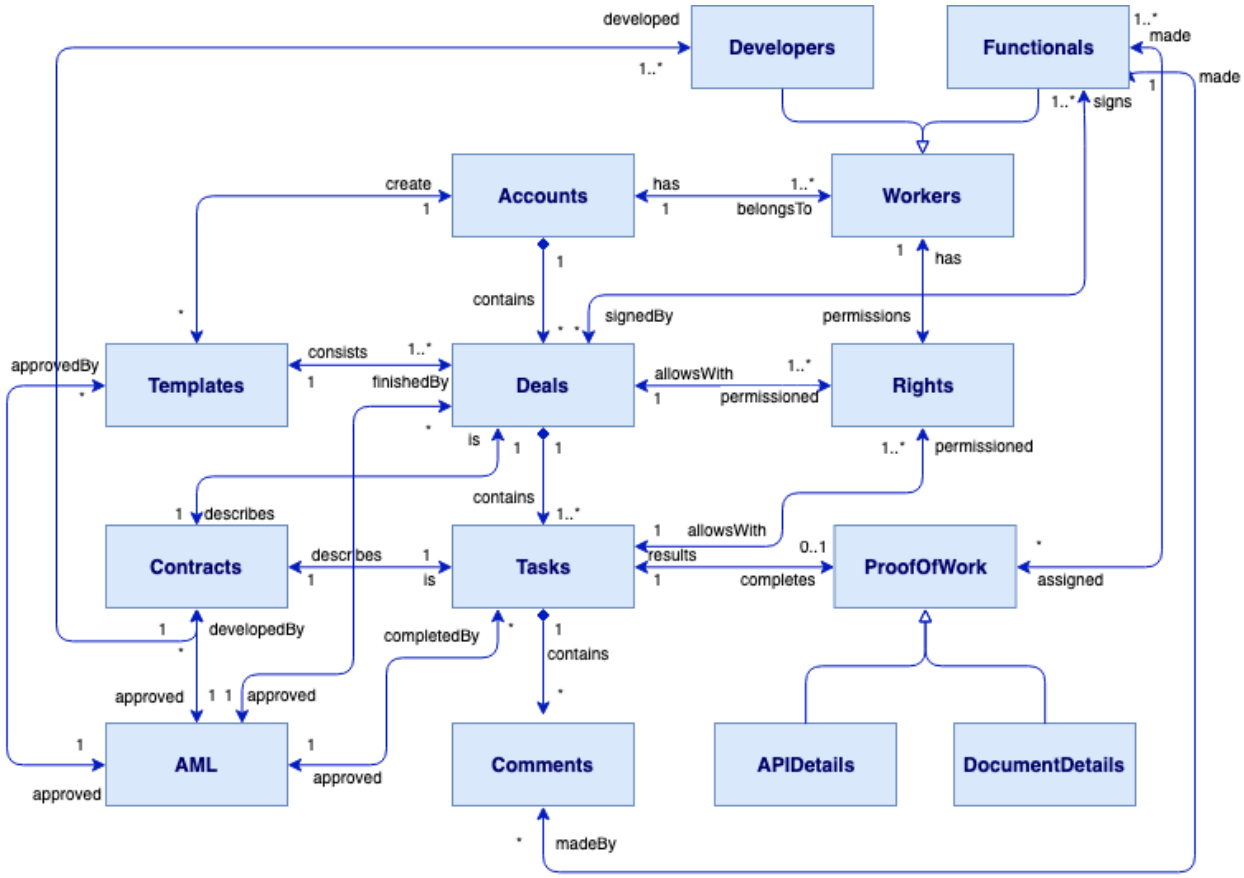
Mikroservisa arhitektūras izmantošanas priekšrocības ir tādas, ka servisi apmainās ar datiem, izmantojot tīklu, un lai mainītu servisu, ir nepieciešama tikai šī konkrētā servisa pārdale. Paliek daudz vieglāk izmērīt, cik daudz resursu katrs serviss patērē, jo tas darbojas atsevišķā procesā. Testēšana un atkļūdošana ir vieglāka, jo katru servisu var analizēt atsevišķi. Mikroservisi patērē mazāk resursu, un tos var viegli pārvērst atkārtotai izmantošanai citās lietojumprogrammās.

## 5.4. Risinājuma diagrammas

Šajā nodaļā tiks paskaidrota konceptuālā un lietošanas gadījuma diagramma, kas atbilst iepriekš aprakstītam risinājumam, balstoties uz augsta līmeņa arhitektūras diagrammu. Dotās diagrammas palīdz izstrādātajam izprast produktu, ko uzņēmums X vēlas realizēt. Tāpat, pamatojoties uz šo informāciju, piedāvāt funkcionālu dizainu.

### *5.4.1. Konceptuālā diagramma*

Konceptuālās diagrammas ir efektīvs rīks, lai vienkāršā un informatīvā veidā komunicētu sarežģītus ziņojumus. Balstoties uz prasībām un vēlamo uzvedību, tika izveidota konceptuālā diagramma, kas ir attēlota 5.2. att.



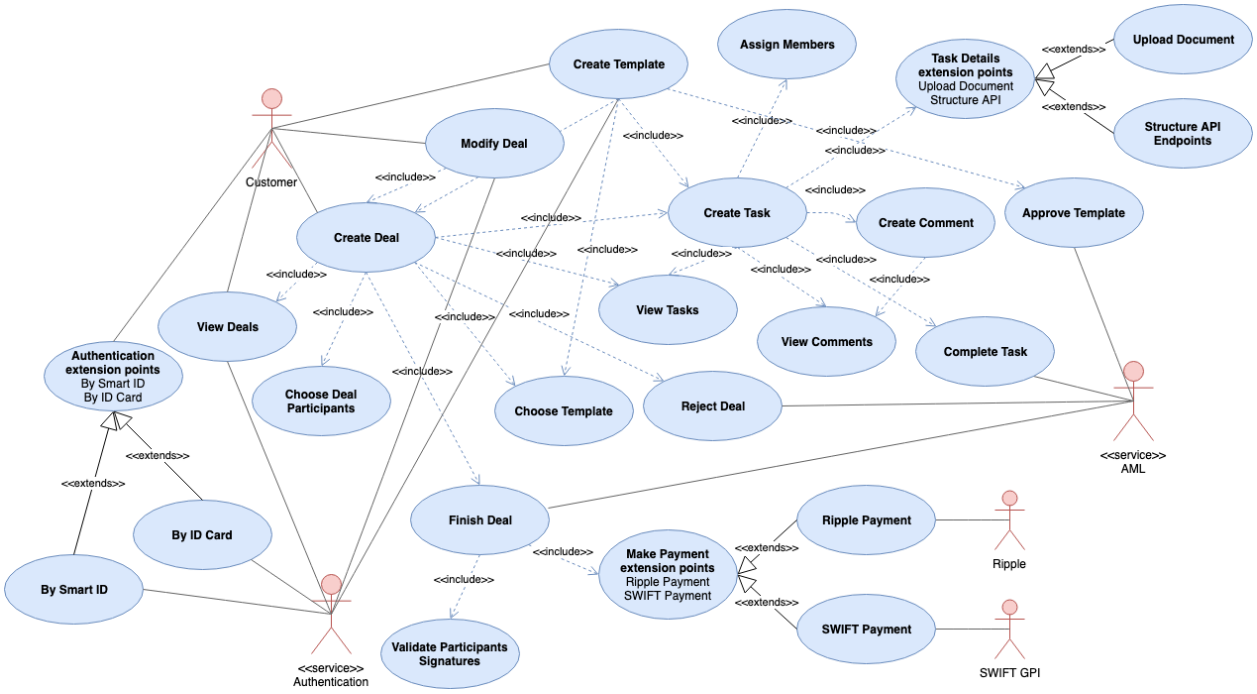

*5.2. att.* **Lietošanas gadījumu diagramma**

Pēc klienta autentificēšanās esošajā uzņēmuma X sistēmā, klients nokļūst savā kontā **Accounts**. Tam ir darbinieki **Workers**, kas iedalās 2 veidu darbiniekos: izstrādātājos **Developers** un funkcionālajos darbiniekos **Functionals**. Katram ir savs darbs šajā sistēmā. Izstrādātāji **Developers** var pievienot jaunus līgumus **Contracts**, bet funkcionālie darbinieki **Functionals** var pievienot darba pierādījumus **ProofOfWork**, kurus tāpat var iedalīt divos veidos: detalizētā citas sistēmas aprakstā **APIDetails** vai dokumentā **Documentdetails**. Katrs konts **Accounts** var izveidot veidni **Templates**, kuru apstiprina **AML**. Tikai pēc apstiprināšanas šī veidne **Templates** ir pieejama. Katram kontam **Accounts** ir darījumi **Deals**, kas ir pieejami tikai konta darbiniekiem **Workers** ar nepieciešamajām tiesībām **Rights**. Katrs darījums **Deals** ir līgums **Contracts**, kas sastāv no uzdevumiem **Tasks**, kas arī ir līgums **Contracts**. Uzdevumu **Tasks** ir atļauts veidot un pabeigt tikai, ja ir atļautās tiesības **Rights**. Katram uzdevumam **Tasks** ir iespēja pievienot komentārus **Comments**, kuri ir pieejami tikai funkcionālajiem darbiniekiem **Functionals**. Katram

uzdevumam **Tasks** rezultātā ir jābūt darba pierādījumam **ProofOfWork**, un tikai pēc tam **AML** var pabeigt pārraudzību un apstiprināt uzdevumu **Tasks**.

### *5.4.2. Lietošanas gadījumu diagramma*

Lietošanas gadījumu diagramma ir uzvedības diagramma, kas modelē sistēmas funkcionalitāti, izmantojot dalībniekus un lietošanas gadījumus. Lietošanas gadījumi ir darbību, pakalpojumu un funkciju kopums, kas ir jāveic sistēmai. Šajā kontekstā "sistēma" ir kaut kas, kas tiek izstrādāts vai darbināts, piemēram, *Corda* mezgls. "Dalībnieki" ir cilvēki vai organizācijas, kas sistēmā izpilda noteiktas funkcijas.




*5.3. att.* **Lietošanas gadījumu diagramma**

Balstoties uz konceptuālo diagrammu un integrācijas prasību aprakstu, tika izveidota lietošanas gadījumu diagramma. Tālāk tiks aprakstītas katra dalībnieka darbības lietošanas gadījumu diagrammā, kas norādīta 5.3. att.

**Customer** ir dalībnieks, kas ir uzņēmuma X klients un kam ir konts. To var vadīt dažādi klienta darbinieki, kuriem ir noteiktas tiesības uz noteiktām darbībām:

- **Authentication** - klients var autentificēties uzņēmuma X sistēmā, izvēloties divu veidu autentifikāciju - **By Smart ID** un **By ID Card**. Tikai pēc tam klients nokļūst savā kontā un veic darbības.
- **View Deals** - pēc tam, kad klients autentificējas, viņš var redzēt visus darījumus, kas attiecas uz viņa kontu. Atkarībā no tiesībām, klienta darbinieki var veikt dažādas darbības.
- **Create Deal** - ja pastāv tādas tiesības, klienta darbinieki var izveidot jaunu darījumu, iekļaujot tajā citus klientus, kas var būt gan uzņēmuma X iekšējie, gan ārējie klienti, izmantojot **Choose Deal Participants** lietošanas gadījumu. Lai detalizētāk aprakstītu darījumu, ir iespēja izveidot to pēc veidnes ar **Choose Template** gadījumu, ko iepriekš nodefinē **Customer** vai **AML**, vai izveidot jaunu darījumu manuāli, izveidojot katru uzdevumu ar **Create Task** gadījumu, kuram tiek piešķirti atbildīgie un aprakstītas visas detaļas ar **Task Details** gadījumu, kurā ir 2 varianta uzdevumi - **Upload Document** un **Structure API Endpoints**. Klienta darbinieki var izpildīt uzdevumu, tikai iekļaujot darba pierādījumu, kas ir aprakstīts **Task Details** ar **Complete Task** gadījumu. Katrā darījumā var skatīt visus uzdevumus ar **View Tasks** gadījumu. Katram uzdevumam ir iespējams izveidot komentārus klienta darbiniekiem, kuriem ir tiesības to darīt. **Create Comment** gadījums un **View Comments** gadījums.
- **Modify Deal** - ja kaut kāda iemesla dēļ, ir nepieciešams izmainīt darījuma prasības, tad tādā gadījumā vecais darījums tiek noraidīts ar **Reject Deal**, saglabāts sistēmā un uz tā bāzes tiek izveidots jauns darījums ar **Create Deal** gadījumu, kurā ir reference uz veco darījumu.
- **Create Template** - klientam ir iespēja izveidot veidni ar iepriekš noteiktiem darījumiem un uzdevumiem, kas ir saglabāti klienta kontā. Tāpat pastāv iespēja izmantot eksistējošas veidnes ar **Choose Template** un uz tā bāzes izveidot jaunu veidni.

**AML** ir sistēmas serviss, kas veic visu darbību pārraudzību un aktīvi piedāvā katra darījumu aizvēršanu un veidnes apstiprināšanu.

- **Finish Deal** - kad klients vēlas pabeigt darījumu, tad AML pieslēdzas un validē parakstus ar **Validate Participants Signatures**, apstrādā savā sistēmā visus darījumu dalībniekus, darba pierādījumus un tikai pēc veiksmīgas pārraudzības īstenošanas atļauj veikt **Make Payment** gadījumu. Negatīva rezultāta gadījumā, tad izmanto **Reject Deal** gadījumu**.**
- **Reject Deal** - jebkurā gadījumā **AML** var noraidīt jebkuru darījumu, ja pastāv tāda vajadzība.

- **Complete Task** - ja uzdevumu veica kāds no klienta darbiniekiem, visi šī uzdevuma dati tiek nosūtīti uz **AML**.
- **Approve Template** - lai veidne būtu pieejama klientam pēc **Create Template** gadījuma, **AML** sākumā ir nepieciešams pārbaudīt un apstiprināt šo veidni.

**Authentication** ir sistēmas pakalpojums, kas autentificē klienta kontu un katra klienta darbinieka atļauju sistēmā.

- **By Smart ID** - pastāv iespēja autentificēties, izmantojot Smart ID lietojumprogrammu.
- **By ID Card -** pastāv iespēja autentificēties, izmantojot speciālu ierīci, kas nolasa ID karti.
- **View Deals** - tikai pēc autentificēšanas klienta darbiniekam tiek atļauts veikt šo darbību.
- **Modify Deal** - tikai pēc autentificēšanas klienta darbiniekam tiek atļauts veikt šo darbību.
- **Create Template** - tikai pēc autentificēšanas klienta darbiniekam tiek atļauts veikt šo darbību.

**Ripple** ir globāli maksājumu pakalpojumu sniedzēji, kuri ļauj veikt pārrobežu maksājumus.

- **Ripple Payment** - ja tiek izvēlēts **Make Payment** pakalpojuma sniedzējs **Ripple** gadījums, tad maksājums notiek caur Ripple tīklu.

**SWIFT GPI** ir maksājumu pakalpojums un jauns standarts globālajos maksājumos.

- **Swift Payment** - ja tiek izvēlēts **Make Payment** pakalpojuma sniedzējs **SWIFT GPI** gadījums, tad maksājums notiek caur SWIFT tīklu.

## 5.5. Risinājuma pārskats

Šajā sadaļā ir aprakstītas integrācijas implementēšanas platformas, kā arī visu to *API* metožu apraksts, ar kurām tiks nosūtīti ziņojumi starp sistēmām. Tāpat, pamatojoties uz aprakstu, autors sniegs piemēru gataviem pieprasījuma / atbildes ziņojumiem, kurus sistēma spēj pārraidīt.

### *5.5.1. Izstrāde*

Šajā sadaļā tiks sīkāk paskaidrota izstrādes un integrācijas platforma, kā arī, ko paredz un ietver integrācijas testēšana. Sadaļā tiks sīkāk pārskatīts izstrādes process, kas tiks veikts integrācijas ietvaros.

### *Izstrādes platforma*

Šajā sadaļā ir noteikta integrācijas kopējā izstrādes platforma. Integrācija tiks izstrādāta ar *Kotlin*, izmantojot *Intellij IDEA*. Izstrādātāji izmantos programmatūras konfigurācijas pārvaldības sistēmu, lai veiktu darbu kopīgi. Programmatūras konfigurācijas pārvaldības sistēmas nodrošina līdzekļus sadalītajām komandām, lai kopīgi strādātu kopīgotajos dokumentos. Integrācijas ietvaros izstrāde tiks veikta, izmantojot *Git*.

Automatizētās *build* sistēmas, kas paredzētas, lai automatizētu programmu apkopošanas procesu, rūpējas par atkarībām un automatizētu apkopošanu. Integrācijai tiek izmantota automatizēta *Maven build* sistēma, kas ir projektu vadības un izpratnes rīks.

### *Integrācijas platforma*

Nepārtrauktajā integrācijā tiek izmantota nepārtrauktas integrācijas pieeja, lai apvienotu apakšsistēmas un nodrošinātu savietojamību. Nepārtraukta integrācija nozīmē nepārtrauktu integrācijas pasākumu piemērošanu, jo īpaši apvienojot visas apakšsistēmas un pārbaudot to sadarbību.

Integrācijas testēšana paredz, ka komponenti tiks integrēti lielākās apakšsistēmās un visbeidzot sistēmā kopumā. Tomēr joprojām ir iespējams, ka šīs sastāvdaļas satur defektus, jo testēšanas draiveri, ko parasti izmanto testos, nodrošina tikai aptuveno simulēto komponentu uzvedību.

## *5.5.2. API apraksts*

*REpresentational State Transfer* jeb *REST API* arhitektūras stils kļūst par kopīgu pieeju pakalpojumu sniegšanai globālajam tirgum. Tajā definēts, kā lietojumprogrammas sazinās ar hiperteksta pārsūtīšanas protokolu jeb HTTP, izmantojot *JSON* ziņojuma formātu. Lietojumprogrammas, kas izmanto *REST*, ir brīvi saistītas un ātri un efektīvi pārsūta informāciju.

API saites dizains ir sīkāk aprakstīts 5.1. tab. un domāts URI https://corda.example.com/api/v1

*5.1. tabula*

| Metode | Saites | Apraksts |
|---|---|---|
| POST | /deals | Izveidot jaunu darījumu |
| GET | /deals?limit=25&offset=50 | Atgriezt visus Klientam piešķirtos darījumus |
| GET | /deals/{dealId} | Atgriezt noteiktu informāciju par darījumu |
| GET | /deals/findDealsByParticipantId | Meklēt darījumus pēc Dalībnieka ID |
| GET | /deals/findDealsByStatus | Meklēt darījumus ar noteiktu statusu |
| GET | /deals/findDealsByDate | Meklēt darījumus pēc noteikta datuma |
| PUT | /deals/{dealId} | Pārveidot esošo darījumu, izveidojot jaunu darījumu |
| DELETE | /deals/{dealId} | Noraidīt noteiktu darījumu |
| POST | /deals/{dealId}/tasks | Izveidot jaunu uzdevumu noteiktam darījumam |
| GET | /deals/{dealId}/tasks | Atgriezt visus darījuma uzdevumus |
| GET | /deals/{dealId}/tasks/{taskId} | Atgriezt uzdevuma detaļas |
| PUT | /deals/{dealId}/tasks/{taskId} | Atjaunināt esošo uzdevumu |
| DELETE | /deals/{dealId}/tasks/{taskId} | Apturēt noteiktu uzdevumu darījuma ietvaros |
| POST | /templates | Izveidot jaunu veidni |
| GET | /templates?limit=25&offset=50 | Atgriezt visas pieejamās veidnes |
| GET | /templates/{templateId} | Atgriezt noteiktu veidni |
| PUT | /templates/{templateId} | Atjaunināt noteiktu veidni |
| DELETE | /templates/{templateId} | Dzēst noteiktu veidni |

Tālāk ir iespējams apskatīt piemērus pieprasījuma / atbildes ziņojuma formātam, kā arī ieteicamās sistēmas īstenošanas atbildes.

#### *Ziņojuma formāts*

Ziņojuma formāts ietver *API* pieprasījuma un atbilžu atspoguļojumu, kas ir darījumu pieprasījuma, darījumu atbildes un kļūdas atbildes ziņojums. Pilns ziņojuma formāts dots 1. pielikumā.

#### *API atbildes*

Zemāk ir iespējams apskatīt API atbildes datu kļūdām, autentificēšanas kļūdām un standarta statusiem.

#### *Datu kļūdām*

- **400** kad pieprasītā informācija ir nepilnīga vai kļūdaina;
- **422** kad pieprasītā informācija ir pareiza, bet nederīga;
- **404** kad viss ir pareizi, bet resurss nepastāv;
- **409** kad pastāv datu konflikts, pat ar derīgu informāciju.

#### *Autentificēšanas kļūdām*

- **401** kad piekļuve nav nodrošināta vai nav derīga;
- **403** kad piekļuve ir derīga, bet prasa vairāk privilēģiju.

#### *Standarta statusiem*

- **200** kad viss ir pareizi;
- **204** kad viss ir pareizi, bet nav satura, lai atgrieztos;
- **500** kad serveris izmet kļūdas pilnīgi negaidīti.

### *5.5.3. Lietojamās tehnoloģijas*

Šajā sadaļā tiek aprakstīti papildu moduļi *Corda* mezglam, kurus nepieciešams izmantot implementācijas procesā, lai visas prasības būtu izpildītas. *Corda* ir liels lietojumprogrammu izvēles klāsts, ko var piemērot izstrādē. Tālāk tiek apskatīti pamata moduļi, kā arī tiek aprakstītas tehnoloģijas, kas tiek izmantotas šajā integrācijā.

### *Corda Settler modulis*

Atvērtā koda *CorDapp* palīdz *Corda* lietotājiem veikt transakcijas. *Corda Settler* nodrošina tiltu jebkurai maksājumu platformai, kas var atgriezt norēķinu kriptogrāfisko pierādījumu. Tas ietver visas "tradicionālās" iekšzemes un pārrobežu maksājumu sistēmas, kas spēj atgriezt šādu pierādījumu, ietverot arī blokķēdes un uz kriptovalūtas balstītas platformas. [30]

### *Business Network Membership Service (BNMS) modulis*

*BNMS* uzņēmējdarbības tīkls ir neatkarīgu pušu grupa, kas veic transakcijas kopā. Tās mērķis ir ļaut saviem biedriem izveidot kopīgu informācijas vai faktu pārstāvniecību un pēc tam izmantot šo faktu kopīgu apstrādi, lai panāktu vienošanos vai konsensu par darbībām, kas saistītas ar tām.

Šī spēja nodrošināt gan kopīgu izpratni par faktiem, gan kopīgu izpratni par to, kā tos izmantot, ir kas unikāls *DLT* sistēmās. Agrākās sistēmas koncentrējās uz kopīgu informācijas atspoguļojumu, bet nevarēja konsekventi garantēt tās pareizību, nedz arī nodrošināt, ka visi dalībnieki apstrādāja lietas vienādi. *BNMS* mērķis ir atrisināt šādas problēmas: [31]

- jaunu dalībnieku iekļaušana biznesa tīklā;
- dalībnieku atcelšana no biznesa tīkla;
- dalībnieku saraksta izplatīšana biznesa tīkla dalībniekiem;
- pielāgotu metadatu apvienošana ar mezgla identitāti;
- dalība vairākos biznesa tīklos ar vienu mezglu.

Kopumā *BNMS* mērķis ir palīdzēt biznesa tīkla operatoriem darboties ar saviem tīkliem. Tas ir atvērts kods, kas ir pieejams kā ieteicamā īstenošanas programmatūra.

### *Drošība*

HTTP ziņojumu drošībai vajadzētu izmantot *OAuth 2.0*, piedāvājot drošības mērogojamību. Pakalpojumu sniedzēji šajā laikā var meklēt tikai autentifikāciju, bet sistēma, kas dabiski atbalsta spēcīgu autorizāciju papildus iekļautajām autentificēšanas metodēm, ir ļoti vērtīga un samazina implementēšanas izmaksas ilgtermiņā.

Lielākā daļa drošības aizsardzību tiek deleģētas *HTTPS / TLS*. *OAuth 2.0* ir elastīgs, tas ietver arī klientus, kas nav interneta lietotāji. Tas var atdalīt resursu pieprasījumu apstrādi un lietotāja autorizācijas apstrādi.

***Corda tīkls***

Šobrīd dalība *Corda* tīklā ir paredzēta tikai juridiskām personām. Pašlaik *Corda* tīklā ir divi dalības pakas: Pamata un Pilna paka. Pilnas dalības paka ietver sava mezgla pārvaldību, vairāku mezglu lietojumprogrammas, pastāvīgu atbalstu, dalību vairākos uzņēmējdarbības tīklos, balsstiesības par fonda pārvaldību un citas priekšrocības. Šāda paka izmaksā $2500 gadā. [32]

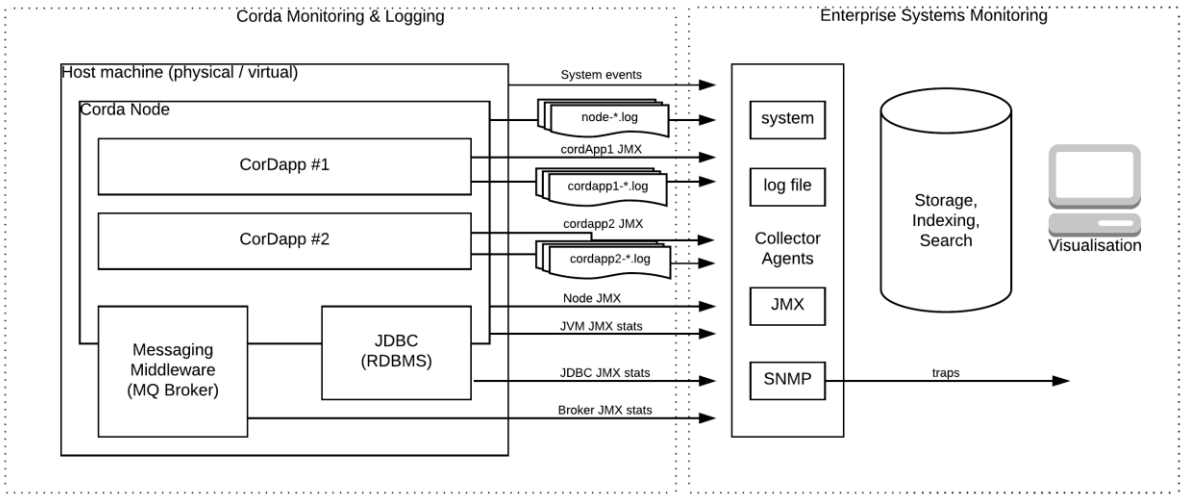
Savukārt otrs dalības līmenis paredz pakalpojuma saņemšanu caur uzņēmējdarbības tīkla / lietojumprogrammas sniedzēju. Šī paka neietver balsstiesības, pastāvīgu atbalstu un vairāku mezglu lietojumprogrammas. Ir iespējams kļūt par dalībnieku tikai vienam uzņēmējdarbības tīklam. Šajā gadījumā mezglu pārvaldīs cita puse. Šāda paka izmaksā $100 gadā. [32]

Transakciju maksas ir strukturētas divos dažādos modeļos. Pirmais modelis *Pay As You Go* tiek rēķināts retrospektīvi gada beigās. Tāpat kā dalības maksas, rēķinus Pamata pakas dalībniekiem pārvalda sponsori, kuri arī izsūta tos Pilnas pakas dalībniekiem pa tiešo. Savukārt otrais modelis *Up-front Package* ir paredzēts, lai sniegtu pārliecību par notariālās apstiprināšanas izmaksām. Rēķins tiek rēķināts perspektīvi gada sākumā. Tāpat kā dalības maksas, rēķinus Pamata pakas dalībniekiem pārvalda sponsori, kuri arī izsūta tos Pilnas pakas dalībniekiem pa tiešo.

### *5.5.4. Notikumu reģistrēšana un pārraudzība*

Veiksmīgai *Corda* izvietošanai un darbībai ir nepieciešama atbalstoša pārraudzība un vadība, kas nodrošina, ka gan *Corda* mezgls, gan izvietotie *CorDapps* darbojas funkcionāli pareizā un konsekventā veidā. Proaktīvs pārraudzības risinājums ļaus nekavējoties brīdināt par

negaidītu uzvedību, lai saistītie vadības instrumenti ļautu veikt ātru koriģējošu darbību (sk. 5.4. att.).




*5.4. att.* **Corda notikumu reģistrēšana un pārraudzība [33]**

Šobrīd *Corda* ietver vairākus pārraudzītā satura veidus:

- *Corda* izmanto *Apache Log4j 2* sistēmu, lai reģistrētu izeju uz konfigurētu reģistrētāju kopu. Pašlaik to pašu reģistrēšanas failu kopumu izmanto mezgls un *CorDapp;*
- nozares standarta *JMX* metrikas, gan standarta *JVM*, gan pielāgotās lietojumprogrammas rādītāji tiek parādītas tieši. Turklāt *Corda* izmanto arī *Jolokia* sistēmu, lai padarītu tos pieejamus *HTTP* beigu punktā.

*Corda* reģistrēšana un pārraudzība viegli integrējas ar jebkurā uzņēmumā esošajām sistēmām, kas atbild par šo funkcionalitāti.

## 5.6. Risinājuma rezultāts

Risinājuma rezultātā tika izstrādāts integrācijas dizains, ko var izmantot reālā izstrādē, kur detalizētāk nepieciešams aprakstīt esošās uzņēmuma sistēmas. Izvēlētā *Corda* sistēma ļauj mērogoties esošajās sistēmās ar mazākām izmaksām, ļaujot uzņēmumiem veikt tiešas transakcijas, saistītas ar uzņēmējdarbību. Turklāt klients var saņemt sistēmu, kur veikt konkrētus darījumus un izstrādāt savus produktus, izmantojot esošo uzņēmuma X infrastruktūru. Tāpat klientam ir lielāka brīvība attiecībā uz darījumiem ar citiem klientiem.

Darījumi, kas ir sadalīti mazos uzdevumos, ļauj *AML* sistēmām precīzāk analizēt katru darbību un savlaicīgi reaģēt uz aizdomīgu darījumu. Tādā veidā tiek izpildītas regulatoru prasības, un uzņēmums X samazina reputācijas risku saistībā ar aizdomīgām transakcijām, īstenojot transakciju caurspīdīgumu.

Risinājums tika izstrādāts, lai uzlabotu transakciju caurspīdīgumu, kā arī samazinātu izmaksas un laiku, kas nepieciešams pārrobežu maksājumu veikšanai. Tā kā *Corda* ir tehnoloģija, kas pieejama caurspīdīgu finanšu transakciju implementēšanai, tad dotais risinājums pilnā mērā atbilst izvirzītajam mērķim, īstenojot *Corda* integrācijas dizaina izstrādi, ko ir iespējams integrēt esošajās sistēmās.

# REZULTĀTI

Lai sasniegtu izvirzīto maģistra darba mērķi, vispirms tika apskatītas pārrobežu maksājumu tehnoloģiju *SWIFT GPI* un *Ripple* iezīmes un funkcijas, kas atklāja, ka bankas un finanšu tehnoloģiju uzņēmumi jau šobrīd izmanto tehnoloģijas, kas balstītas uz *DLT*. Tāpat tika noteikts, ka pastāv nepieciešamība pēc jaunu tehnoloģiju implementēšanas. Tas ir saistīts ar šī brīža pārrobežu maksājumu stāvokli, jo šobrīd tie ir pārāk lēni un dārgi, un veids, kā bankas nosaka cenu, ir nepārredzams.

Caurspīdīgu transakciju gadījumā liela loma ir blokķēdes tehnoloģijai, kas ir ir pirmā, visizplatītākā un visplašāk pētītā *DLT*. Tā sniedz augstu privātuma līmeni, nodrošinot, ka transakciju informācija tiek dalīta tikai starp transakcijās iesaistītajiem dalībniekiem. Līdzās privātuma līmenim blokķēdes tehnoloģijai ir arī augsts caurspīdīguma līmenis. Tā var paātrināt pārrobežu transakcijas - tās var veikt dažu sekunžu laikā, jo tās ir jāapstiprina tikai caur blokķēdes sistēmu.

Savukārt *DLT* paredz tādas priekšrocības kā izmaksu samazināšana norēķinos, ārāki norēķini, uzticamība un iespēja izsekot ierakstus, kā arī automatizētas atskaites reālajā laikā un drošības uzlabošana. Tas savukārt sniedz labākas iespējas *AML* regulatoriem, jo dotie risinājumi sniedz detalizētāku informāciju *AML* sistēmai, ļaujot savlaicīgi veikt darbības, ja tiek konstatētas potenciāli aizdomīgas transakcijas pirms tika veikts darījums.

Veicot *DLT* risinājumu - *Corda, Ethereum* un *Hyperledger Fabric* - salīdzinājumu, tika atklāts, ka optimālākais ir *B2B* risinājums *Corda,* kas ir platforma, kuras jaunākās versijas mērķis ir aizstāt programmatūras, kas tiek izmantotas finanšu transakcijām, ļaujot organizācijām digitalizēt dažādus biznesa procesus. Pētījuma gaitā tika atklāts, ka decentralizētā blokķēdes platforma *Ethereum,* lai gan paredz transakciju šifrēšanu, būtībā ir atklāts tīkls, kas paredzēts *B2C* darījumiem. Savukārt *Hyperledger Fabric* platforma, kas ļauj veidot uz blokķēdi balstītas lietojumprogrammas, ir pārāk dinamiska tehnoloģija ar daudzām izmantošanas gadījumu iespējām. Līdz ar to, tika atklāts, ka *Corda,* kas ir īpaši izstrādāta tieši finanšu darījumiem, ir optimālākais risinājums *B2B* nozarei.

# SECINĀJUMI

Uzņēmējdarbība caurspīdīgumu definē kā slēptās darba kārtības un nosacījumu trūkumu, kā arī pilnīgas informācijas pieejamību, kas nepieciešama sadarbībai un kolektīvai lēmumu pieņemšanai. Caurspīdīgums jāveido tā, lai nodrošinātu transakcijas procesa datu, tehnoloģiju un caurspīdīguma nodrošināšanu.

Transakciju caurspīdīguma, apstrādes ātrdarbības un izmaksu problēmas spēj novērst mūsdienu risinājumi, kas nodrošina elastību maksājumu opcijās, vieglu maksājumu saskaņošanu un caurspīdīgu reālā laika piekļuvi transakciju informācijai.

Apkopojot rezultātus, tika secināts, ka *DLT* risinājumi jau šobrīd tiek integrēti esošajās sistēmās, jo spēj gan risināt iepriekš minētās transakciju un pārrobežu maksājumu problēmas, gan atbilst *AML* un *KYC* prasībām.

Tā kā pētījuma gaitā tika atklāts, ka tieši *Corda* ir vispiemērotākā platforma *B2B* darījumiem, tika veikta *Corda* integrācijas dizaina izstrāde, ko ir iespējams integrēt esošajās sistēmās. Maģistra darba gaitā tika pilnībā atklāta un apskatīta transakciju caurspīdīguma problēma, pieejamās un jaunās tehnoloģijas, kā arī *DLT* risinājumi šīs problēmas risināšanai. Tas savukārt parāda to, ka maģistra darba mērķis tika pilnībā sasniegts un īstenots.

# IZMANTOTĀ LITERATŪRA UN AVOTI


1. Understanding Online Payment Services: The Complete Guide. - [atsauce 01.02.2019.]. Pieejams: https://www.wildapricot.com/articles/online-payment-services
2. System implementation considerations for payment transparency. - [atsauce 01.02.2019.]. Pieejams: https://www.infosys.com/industries/cards-and-payments/resources/Documents/payment-transparency.pdf
3. What faster payments means for anti-money laundering compliance. - [atsauce 01.02.2019.]. Pieejams: https://www.complianceweek.com/thought-leadership/white-paper/forresters-vendor-landscape-anti-money-laundering-solutions-2017
4. What is TARGET2? - [atsauce 06.05.2019.]. Pieejams: https://www.ecb.europa.eu/paym/target/target2/html/index.en.html
5. How the SWIFT System Works. - [atsauce 11.03.2019.]. Pieejams: https://www.investopedia.com/articles/personal-finance/050515/how-swift-system-works.asp
6. SWIFT dpi - Cross-border Payments transformed. - [atsauce 12.04.2019.]. Pieejams: https://www.swift.com/file/57196/download?token=9ssLjNJX
7. Ripple Solution Overview. - [atsauce 02.03.2019.]. Pieejams: https://ripple.com/files/ripple_solutions_guide.pdf
8. In the pursuit of transparency. - [atsauce 02.03.2019.]. Pieejams: https://ripple.com/insights/in-the-pursuit-of-transparency/
9. Real-time cross-border payments using DLT. - [atsauce 15.04.2019.]. Pieejams: https://www.accenture.com/us-en/insight-real-time-cross-border-payments
10. The difference between Blockchain & Distibuted Ledger Technology. - [atsauce 15.04.2019.]. Pieejams: https://tradeix.com/distributed-ledger-technology/
11. Blockchain. - The new technology of Trust - [atsauce 20.04.2019.]. Pieejams: https://www.goldmansachs.com/insights/pages/blockchain/
12. Understanding Blockchain. - [atsauce 20.04.2019.]. Pieejams: https://totalcareit.com/understanding-technology/understanding-blockchain

13. Distributed Ledger Technology: Implications of Blockchain for the Securities Industry. - [atsauce 20.04.2019.]. Pieejams: http://www.finra.org/sites/default/files/FINRA_Blockchain_Report.pdf
14. What is a Smart Contract? - [atsauce 20.04.2019.]. Pieejams: https://corporatefinanceinstitute.com/resources/knowledge/deals/smart-contract/
15. Netting in Finance. - [atsauce 01.05.2019.]. Pieejams: https://www.bellin.com/blog/netting-in-finance/
16. Blockchain for Enterprise. - Narayan Prusty - Packt Publishing, 2018
17. Blockchain to 'radically' transform anti-fraud, anti-money-laundering efforts. - [atsauce 01.05.2019.]. Pieejams: https://www.computerworld.com/article/3265111/blockchain/blockchain-to-radically-transform-anti-fraud-anti-money-laundering-efforts.html
18. Blockchain Technology Could Reduce Investment Banks Infrastructure Costs by 30 Percent. - [atsauce 12.04.2019.]. Pieejams: https://newsroom.accenture.com/news/blockchain-technology-could-reduce-investment-banks-infrastructure-costs-by-30-percent-according-to-accenture-report.htm
19. Leveraging Machine Learning for AML transaction monitoring. - [atsauce 12.04.2019.]. Pieejams: https://www.accenture.com/_acnmedia/PDF-61/Accenture-Leveraging-Machine-Learning-Anti-Money-Laundering-Transaction-Monitoring.pdf
20. Finanšu informācijas drošības pieejas un risinājumi. - [atsauce 15.05.2019.]. Pieejams: https://dspace.lu.lv/dspace/handle/7/33268
21. Programmable money. - [atsauce 01.02.2019.]. Pieejams: https://www.sepaforcorporates.com/wp-content/uploads/2018/10/IBM-programmable_money_world-wire.pdf
22. Key concepts. - [atsauce 20.04.2019.]. Pieejams: https://docs.corda.net/key-concepts.html
23. Permissioning. - [atsauce 20.04.2019.]. Pieejams: https://docs.corda.net/releases/release-V3.1/permissioning.html
24. Corda. - [atsauce 20.04.2019.]. Pieejams: https://buildmedia.readthedocs.org/media/pdf/corda/v3.1_zh-cn/corda.pdf
25. IOU template. - [atsauce 06.04.2019.]. Pieejams: https://github.com/corda/corda-training-template

26. Ethereum Wiki. - [atsauce 16.04.2019.]. Pieejams: https://github.com/ethereum/wiki
27. Hyperledger Architecture, Volume 1. - [atsauce 06.03.2019.]. Pieejams: https://www.hyperledger.org/wp-content/uploads/2017/08/Hyperledger_Arch_WG_Paper_1_Consensus.pdf
28. Comparison of DLT. - [atsauce 15.04.2019.]. Pieejams: http://explore-ip.com/2017_Comparison-of-Ethereum-Hyperledger-Corda.pdf
29. Blockchain for Business. - Nitin Gaur, Jerry Cuomo, Jai Singh Arun. - Addison-Wesley Professional, 2019
30. Membership Tiers. - [atsauce 06.03.2019.]. Pieejams: https://corda.network/participation/membership-tiers.html
31. Corda Settler. - [atsauce 02.05.2019.]. Pieejams: https://github.com/corda/corda-settler
32. Corda BNMS. - [atsauce 02.05.2019.]. Pieejams: https://solutions.corda.net/designs/business-networks-membership-service.html
33. Logging design. - [atsauce 02.05.2019.]. Pieejams: https://docs.corda.net/design/monitoring-management/design.html
34. Build Network - [atsauce 02.05.2019.]. Pieejams: https://hyperledger-fabric.readthedocs.io/en/latest/build_network.html

# PIELIKUMI

# 1. pielikums. API pieprasījums un atbildes

### *Darījumu Pieprasījums*

```
curl -v -X POST https://corda.example.com/api/v1/deals \
-H "Content-Type: application/json" \
-H "Authorization: Bearer Access-Token" \
-d '{
  "account": {
    "accountId": "string",
    "rights": [{
        "rightsId": "integer",
        "type": "string",
        "access": "string"
      }],
  },
  "deals": [{
      "dealAmount": {
        "total": "number",
        "currency": "string",
        "details": {
          "subtotal": "number",
          "tax": "number",
          "handlingFee": "number"
        }
      },
      "description": "string",
      "customId": "string",
      "invoiceId": "string",
      "templateId": "string",
      "contractId": "string",
      "notaryId": "string",
      "encumbrance": "boolean",
      "status": "string",
        "tasks": [
          {
            "name": "string1",
            "description": "string",
            "customId": "string",
            "invoiceId": "string",
            "status": "string",
            "members": [{
              "memberId": "integer",
              "accountId": "integer",
              "name": "string",
              "surname": "string",
              "personalCode": "string",
              "birthDate": "date",
              "rights": [{
                  "rightsId": "integer",
```

```
            "type": "string",
            "access": "string"
          }],
        "position": "string"
        }],
      "taskAmount": {
        "total": "number",
        "currency": "string",
        "details": {
          "subtotal": "number",
          "tax": "number",
          "handling_fee": "number"
        }
      },
      "comments": [{
        "commentId": "integer",
        "memberId": "integer",
        "text": "string",
        "rights": [{
            "rightsId": "integer",
            "type": "string",
            "access": "string"
          }],
        "proofOfWork": {
          "attachmentId": "string"
        }
        }],
      "description": "string",
      "price": "number",
      "tax": "number",
      "currency": "string",
      "proofOfWork": {
        "attachmentId": "string"
      },
      "contractId": "string",
      "notaryId": "string",
      "encumbrance": "boolean",
      "ref": {
        "txhash": "string",
        "index": "integer"
      }
    }
  ]
 }
],
"note": "string",
"redirect_urls": {
  "return_url": "https://corda.example.com/return",
  "cancel_url": "https://corda.example.com/cancel"
},
"ref": {
```

```
    "txhash": "string",
    "index": "integer"
  }
}'
```

### *Darījumu Atbilde*

```
{
  "dealId": "string",
  "create_time": "datetime",
  "modify_time": "datetime",
  "reject_time": "datetime",
  "state": "string",
  "templateId": "string",
  "account": {
    "accountId": "string",
    "rights": [{
        "rightsId": "integer",
        "type": "string",
        "access": "string"
      }],
  },
  "state": {
    "deals": [{
        "dealAmount": {
          "total": "number",
          "currency": "string",
          "details": {
            "subtotal": "number",
            "tax": "number",
            "handlingFee": "number"
          }
        },
        "description": "string",
        "customId": "string",
        "invoiceId": "string",
        "templateId": "string",
        "contractId": "string",
        "notaryId": "string",
        "encumbrance": "boolean",
        "status": "string",
        "state": {
          "tasks": [
            {
              "name": "string1",
              "description": "string",
              "customId": "string",
              "invoiceId": "string",
              "status": "string",
```

```
 "members": [{
  "memberId": "integer",
  "accountId": "integer",
  "name": "string",
  "surname": "string",
  "personalCode": "string",
  "birthDate": "date",
  "rights": [{
     "rightsId": "integer",
     "type": "string",
     "access": "string"
   }],
  "position": "string"
  }],
 "taskAmount": {
  "total": "number",
  "currency": "string",
  "details": {
   "subtotal": "number",
   "tax": "number",
   "handling_fee": "number"
  }
 },
 "comments": [{
  "commentId": "integer",
  "memberId": "integer",
  "text": "string",
  "rights": [{
     "rightsId": "integer",
     "type": "string",
     "access": "string"
   }],
  "proofOfWork": {
   "attachmentId": "string"
  }
  }],
 "description": "string",
 "price": "number",
 "tax": "number",
 "currency": "string",
 "proofOfWork": {
  "attachmentId": "string"
 },
 "contractId": "string",
 "notaryId": "string",
 "encumbrance": "boolean",
 "ref": {
  "txhash": "string",
  "index": "integer"
 }
}
```

```
        ]
       }
      }
    ],
  "links": [
    {
      "href": "https://corda.example.com/api/v1/deals/{dealId}",
      "rel": "self",
      "method": "GET"
    },
    {
      "href": "https://corda.example.com/api/v1/redirect",
      "rel": "approval_url",
      "method": "REDIRECT"
    },
    {
      "href": "https://corda.example.com/api/v1/deals/{dealId}",
      "rel": "execute",
      "method": "POST"
    }
  ],
  "ref": {
    "txhash": "string",
    "index": 0
  }
}
```


### *Kļūdas atbilde*


```
{
  "error": {
    "message": "string”,
    "type": "string",
    "code": “string”,
    "trace_id": "number"
  }
}
```

# Dokumentārā lapa

**Maģistra darbs "Caurspīdīgas finanšu transakciju implementēšanai pieejamās tehnoloģijas" izstrādāts LU Datorikas fakultātē.**

Darba teksta galīgā versija izgatavota 20.05.2019**.**
Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: _____

(Autora paraksts un datums)

Ar savu parakstu apliecinu, ka esmu lasījis augstāk minēto maģistra darbu un atzīstu to par **piemērotu / nepiemērotu** (nevajadzīgo svītrot) aizstāvēšanai Latvijas Universitātes datorzinātņu maģistrantūrā.

Darba vadītājs: _____

(Vadītāja paraksts un datums)

Darbs iesniegts **maģistratūras sekretariātā** 20.05.2019.
Ar šo es apliecinu, ka darba elektroniskā versija ir augšupielādēta LU informatīvajā sistēmā.

Studiju metodiķe: _____.

(Metodiķes paraksts)

Recenzents: _____

(Akad.amats, zin.grāds, vārds, uzvārds)

Darbs aizstāvēts maģistra gala pārbaudījuma komisijas sēdē

_____ prot. Nr. _____

(Darba aizstāvēšanas datums)

Komisijas sekretārs: _____

(Sekretāra paraksts)