Transformacija BPMN zapisa v Java programsko kodo

Janko Hriberšek,[1] Robert Leskovar,[2]
Alenka Baggia[2]

[1] NLB Skladi, d.o.o., Ljubljana, Slovenija
janko.hribersek@nlbskladi.si
[2] Univerza v Mariboru, Fakulteta za organizacijske vede, Kranj, Slovenija
robert.leskovar@um.si, alenka.baggia@um.si

Uvedba tehnologije veriženja blokov prinaša nove priložnosti za avtomatizacijo in izboljšanje učinkovitosti poslovanja, predvsem z uporabo pametnih pogodb, ki avtomatizirajo izvajanje poslovnih pravil. Kot osnovo za razvoj pametnih pogodb lahko uporabimo BPMN, uveljavljen standard za modeliranje poslovnih procesov. V prispevku predstavljamo transformacijo modelov BPMN v ogrodje Javanske kode, ki jo lahko uporabimo v okolju tehnologije veriženja blokov, specifično na platformi HyperLedger Fabric. Grafična predstavitev procesov v BPMN je zapisana v XML obliki, na osnovi katere lahko osnovne BPMN gradnike implementiramo kot kontrolne strukture v Javanski programski kodi. Predlagana rešitev omogoča hitro, točno in prilagodljivo pretvorbo modelov BPMN, s čimer optimiziramo razvoj pametnih pogodb. Prispevek odpira možnosti za nadaljnje raziskave in širitev metodologije z vključitvijo kompleksnejših gradnikov BPMN ter prilagoditev za različna okolja tehnologije veriženja blokov.

DOI
https://doi.org/10.18690/um.fov.2.2025.25

ISBN
978-961-286-963-2

Ključne besede:
tehnologija veriženja blokov,
pametna pogodba,
BPMN,
programski jezik java,
transformacija

Univerzitetna založba Univerze v Mariboru

DOI
https://doi.org/
10.18690/um.fov.2.2025.25

ISBN
978-961-286-963-2

TRANSFORMING BPMN NOTATION INTO JAVA CODE

JANKO HRIBERŠEK,[1] ROBERT LESKOVAR,[2]
ALENKA BAGGIA[2]

[1] NLB Skladi, d.o.o., Ljubljana, Slovenia
janko.hribersek@nlbskladi.si
[2] University of Maribor, Faculty of Organizational Sciences, Kranj, Slovenia
robert.leskovar@um.si, alenka.baggia@um.si

Keywords:
blockchain technology
smart contract,
BPMN,
Java programming
language,
transformation

Introducing blockchain technology brings new opportunities to automate and improve operational efficiency, particularly using smart contracts that automate the implementation of business rules. BPMN, an established standard for business process modelling, can be used to develop smart contracts. This paper presents the transformation of BPMN models into Java code framework that can be used in a blockchain technology environment, specifically on the HyperLedger Fabric platform. The graphical representation of BPMN processes is written in XML format, from which the basic BPMN building blocks can be implemented as control structures in Java program code. The proposed solution enables fast, accurate and scalable conversion of BPMN models, thereby optimising the development of smart contracts. The paper opens opportunities for further research and extension of the methodology by incorporating more complex BPMN building blocks and adaptations for different blockchain technology environments.

University of Maribor Press

1 Uvod

V današnjem hitro razvijajočem se poslovnem okolju organizacije potrebujejo agilne in učinkovite procese, ki se lahko hitro prilagajajo spreminjajočim se zahtevam. Za upravljanje poslovnih procesov, pri modeliranju in dokumentiranju procesov, organizacije pogosto uporabljajo standardni zapis Business Process Management Notation (BPMN). BPMN je postal standard za modeliranje poslovnih procesov, saj je grafična predstavitev procesa razumljiva deležnikom v poslovnem procesu in tudi IT strokovnjakom (Object Management Group, 2016). Na osnovi modelov BPMN se oblikujejo tudi programske rešitve, ki podpirajo opisane poslovne procese. Vendar je preslikava modelov BPMN v učinkovite računalniške aplikacije pogosto ročna, dolgotrajna in nagnjena k napakam. Vedno večja kompleksnost poslovnih procesov zahteva avtomatizirane rešitve za prevajanje modelov procesov v izvršljivo kodo.

V zadnjem času se v različnih sektorjih pojavljajo aplikacije, ki temeljijo na tehnologiji veriženja blokov (ang. »*blockchain*«). Tehnologija veriženja blokov preoblikuje razširljive informacijske sisteme in aplikacije z integracijo vse bolj priljubljene umetne inteligence, interneta stvari, računalništva v oblaku in masovnih podatkov (Lu, 2019). Omogoča razvoj decentraliziranih, varnih in transparentnih sistemov, ki jih lahko učinkovito uporabimo v različnih poslovnih procesih, od zavarovalništva (Alamsyah & Setiawan, 2025), zdravstva (Faheem Ahmad Reegu, 2021), gradbeništva (Scott et al., 2021), do celotne oskrbovalne verige (Dutta et al., 2020). Pomemben element tehnologije veriženja blokov so tudi pametne pogodbe, ki predstavljajo implementacijo dejanskih pogodb v izvršljivo programsko kodo in omogočajo avtomatizirano izvajanje dogovorov brez potrebe po posrednikih. Tako lahko pametne pogodbe uporabljamo za varne finančne transakcije ali pa za avtomatizacijo pravnih postopkov. Integracija poslovnih procesov, modeliranih z BPMN, s pametnimi pogodbami odpira nove možnosti za avtomatizacijo kompleksnih poslovnih pravil in zagotavljanje preglednosti in zaupanja v decentraliziranih aplikacijah.

V prispevku prikazujemo transformacijo BPMN modela v Java programsko kodo, ki omogoča integracijo modela BPMN v pametno pogodbo v okolju tehnologije veriženja blokov. Postopek transformacije prikažemo na enostavnem primeru

vzporednega prehoda. V razpravi opisujemo izzive, s katerimi smo se srečevali pri razvoju vmesnika ter predloge za nadaljnje delo.

2 Pregled literature

Za celovit prikaz obravnavane problematike smo v pregled literature vključili modeliranje poslovnih procesov, tehnologijo veriženja blokov in pametne pogodbe. V nadaljevanju pa predstavljamo različne poskuse transformacije zapisa poslovnega procesa v BPMN v pametno pogodbo.

2.1 Business Process Modeling and Notation (BPMN)

Glavni cilj konzorcija OMG je bil izdelati zapis poslovnega procesa, ki ga bodo različni uporabniki zlahka razumeli (Object Management Group, 2016). Predstavili so BPMN, standardizirano povezavo med oblikovanjem in implementacijo poslovnega procesa. Izkušnje in raziskave uporabe BPMN za modeliranje poslovnih procesov kažejo, da je BPMN zapis razumljiv tako izkušenim kot tudi novim uporabnikom (Gabryelczyk & Jurczuk, 2017). BPMN združuje najboljše prakse poslovnega modeliranja, s katerimi lahko opredelimo zapis in semantiko diagramov sodelovanja (ang. »*Collaboration diagrams*«), diagramov procesov (ang. »*Process diagrams*«) in diagramov koreografije (ang. »*Choreography diagrams*«). Predstavitev poslovnega procesa z BPMN (v nadaljevanju: model BPMN) je sestavljena iz različnih grafičnih elementov, kot so pasovi (ang, »*lines*«), zaporedna opravila (ang. »*sequenced tasks*«), dogodki (ang. »*events*«) in prehodi (ang. »*gateways*«). Tok zaporedja je usmerjena povezava med elementi toka, ki se lahko razcepijo in združijo s prehodi, kar omogoča modeliranje vzporednih nalog in odločitev. Specifikacija notacije poleg grafičnih simbolov definira tudi preslikavo diagrama procesa v izvršljiv jezik XPDL ali BPEL (ang. »*Business Process Execution Language*«), ki temeljita na XML.

2.2 Tehnologija veriženja blokov

Tehnologija veriženja blokov (ang. »*blockchain*«) je postala zanimiva z izdajo prve elektronske valute Bitcoin. Elektronski plačilni sistem temelji na kriptoloških dokazilih in ne na zaupanju, kar omogoča dvema udeležencema neposredne transakcije brez vključevanja tretje, zaupanja vredne entitete (Nakamoto, 2009). Predlagana rešitev temelji na neposredni komunikaciji med dvema strežnikoma (ang.

»*peer-to-peer*«) za časovno žigosanje in za generiranje izračunljivega dokazila o shranjeni transakciji.

S prihodom okolja Ethereum in z decentraliziranimi replikacijami programskih rešitev je tehnologija veriženja blokov, ki se je prvotno uporabljala na področju kriptovalut, dosegla tudi druga poslovna področja. Novost okolja so pametne pogodbe (ang. »*smart contract*«), ki so zapisane v jezikih Solidity ali Vyper. Tehnologija se je nato razvijala v smeri zasebnih okolij, kjer se uporabniki avtenticirajo (zasebno okolje ali okolje z odobritvami (ang. »*permissioned environment*«)). Eden od predstavnikov zadnje generacije okolij veriženja blokov je Hyperledger Fabric. To je odprtokodni sistem za veriženje blokov, zasnovan za nadgradnjo medpanožnih tehnologij. Kot prvi omogoča izvajanje porazdeljenih aplikacij v standardnih programskih jezikih, brez odvisnosti od fiat ali kripto valut. To ga razlikuje od drugih platform, ki zahtevajo specifične jezike za pametne pogodbe ali temeljijo na kripto valutah. Vključuje tudi integriran sistem upravljanja identitet (Androulaki et al., 2018).

2.3 Pametna pogodba

Pametna pogodba je poslovna logika v obliki izvedljive kode, ki deluje na verigi blokov (Hyperledger, 2024). Z logiko pametne pogodbe običajno ustvarjamo nove transakcije, ki se dodajajo v evidenco (ang. »*Ledger*«). Čeprav v pogovoru o verigah blokov izraza pametna pogodba (ang. »*Smart contract*«) in verižna koda (ang. »*Chaincode*«) pogosto zamenjujemo, gre za različna termina in verižna koda predstavlja paket pametnih pogodb v verigi blokov. V evidenci obstajata dve vrsti stanj, in sicer veriga blokov, ki beleži zgodovino vseh transakcij, ter svetovno stanje (ang. »*World state*«), ki vsebuje trenutno vrednost stanj. Primarno pametne pogodbe izvajajo operacije shranjevanja, pridobivanja in brisanja stanj v svetovnem stanju. Veriga blokov pa vsebuje zapis vseh sprememb. Razvijalci pametnih pogodb tako obstoječa poslovna pravila in procese zapišejo kot pametno pogodbo, v določenem programskem jeziku. V okolju Hyperledger Fabric je to običajno JavaScript, Go ali Java (Hyperledger, 2024).

Izvajanje pametnih pogodb vključuje več korakov, kot so razvoj, določitev potrditvenih politik, preverjanje veljavnosti transakcij, uporaba kanalov in komunikacija med pametnimi pogodbami. Za razliko od ostalih sistemov verig

blokov, se v okolju Hyperledger Fabric veljavnost transakcije določi na osnovi politike potrditve (ang. »*Endorsement*«), ki določa, kateri deležniki morajo podpisati transakcijo, da postane veljavna. Transakcija je veljavna, ko jo podpišejo vsi deležniki in se trenutno stanje svetovnega stanja ujema s stanjem transakcij po zaključenem zbiranju podpisov. Pametne pogodbe najpogosteje uporabljamo za določanje poslovnih pravil, razvijalci pa z njimi definirajo tudi nizkonivojsko programsko kodo, ki omogoča delovanje verige blokov.

2.4 Transformacija BPMN v pametno pogodbo

V predstavljeni raziskavi se osredotočamo na transformacijo pravil, ki so zapisana v formalizirani obliki, v programsko kodo pametne pogodbe. V literaturi najdemo le nekaj zapisov o podobnih poskusih. V prispevkih López-Pintado, Dumas, et al. (2019) in López-Pintado, García-Bañuelos, et al. (2019) je predstavljen sistem Caterpillar, ki izvaja poslovne procese na verigi blokov. BPMN model prevedejo v pametno pogodbo za okolje Ethereum in ga tako izvajajo v decentraliziranem okolju. Sistem omogoča transformacijo večjega števila elementov BPMN, vključno s podprocesi. Lauster et al. (2020) podaja pregled uporabe BPMN za modeliranje poslovnih procesov na verigi blokov in ugotavlja, da raziskave preučujejo predvsem, kako BPMN diagrame prevesti v pametne pogodbe, noben pristop pa še ni standardiziran. Liu et al. (2022) v prikazanem primeru razširja BPMN s konceptom sodelovalnih transakcij, kar implementirajo v sistemu za avtomatizirano transformacijo modelov v pametne pogodbe. Sistem TABS (Bodorik et al., 2023) prav tako avtomatizira pretvorbo BPMN modelov v pametne pogodbe, tako za okolje Ethereum kot tudi za Hyperledger Fabric. Zaradi možnosti izbire delov poslovnega procesa (procesi na glavni verigi ali procesi na stranskih verigah), predlagana rešitev zmanjša stroške delovanja in omogoča večjo zasebnost.

3 Metodologija

Model BPMN poslovnega procesa je zapisan v XML formatu, ki služi kot osnova za transformacijo modela v pametno pogodbo. Vsebina XML datoteke se nato pretvori v strukturo, ki omogoča generiranje Java programske kode, v kateri se doda programska logika za nadzor poteka in izvajanje posameznih nalog skladno s BPMN modelom.

Zapis BPMN je sestavljen iz dveh delov:

- Zapisa gradnikov (objektov) in povezav med njimi - <*bpmn2:process*>
- Zapisa grafičnega prikaza posameznih gradnikov (objektov) v programu za risanje procesov BPMN (npr. ProcessMaker) - <*bpmn2:BPMNDiagram*>

Za pretvorbo BPMN je potrebno uporabiti samo zapis procesa (<*bpmn2:process*> ... </*bpmn2:process*>). V tej raziskavi so bili uporabljeni osnovni gradniki procesa BPMN:

- začetek: <bpmn2:startEvent>
- konec: <bpmn2:endEvent>
- naloga: <bpmn2:task>
- povezava med dvema gradnikoma: <bpmn2:sequenceFlow>
- nadzorni gradniki:
  - izključujoč prehod (XOR): <*bpmn2:exclusiveGateway*>
  - vzporedni prehod (AND): <*bpmn2:parallelGateway*>
  - vključujoč prehod (OR): <*bpmn2:inclusiveGateway*>

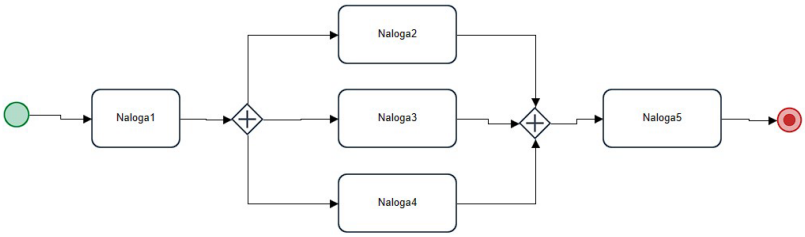
Postopek pretvorbe zapisa BPMN v ogrodje Javanske kode smo izvedli v več korakih. Najprej smo zapis BPMN prenesli v tabelo oziroma matriko povezav med objekti. Nato smo na osnovi obstoja in atributov povezave generirali ogrodje Javanske kode, ki predstavlja izhodišče za pripravo pametne pogodbe v verigi blokov.

4 Rezultati

Postopek pretvorbe zapisa BPMN v pametno pogodbo obsega več stopenj, ki jih v nadaljevanju prikazujemo na primeru modela BPMN, predstavljenega na sliki 1.

BPMN model preslikamo v tabelo, kjer vsaka vrstica predstavlja prehod med aktivnostmi in pogoj prehoda. Vsako polje v tabeli vsebuje informacijo o pogoju, ki mora biti izpolnjen, da se povezava izvede. Za medprocesne povezave pogoja za prehod ni treba opredeliti. Pogoje moramo obvezno opredeliti v primeru dveh vrst

kontrolnih stavkov, in sicer izključujoč (XOR) in vključujoč prehod (OR). Povezave v izključujočih ali vključujočih prehodih se izvedejo le, če je pogoj za povezavo izpolnjen. Če ni, se upošteva privzeta vrednost povezave. Pri vzporednih (AND) prehodih pogoji niso potrebni, saj se vse poti izvajajo sočasno.



Slika 1: BPMN diagram s primerom vzporednega prehoda
Vir: Lasten

4.1 Postopek prepisa procesa BPMN v tabelo

Pri prepisu procesa BPMN so objekt, identifikator in ime predstavljeni kot niz znakov, vhod in izhod pa kot tabela znakovnega niza. Sintaksa za posamezne objekte BPMN je definirana v dokumentaciji (Object Management Group, 2016). Prepis zapisa BPMN v tabelo se izvede v več korakih:

1. Branje podatkov o objektih iz zapisa procesa (<*bpmn2:process*>)
2. Priprava tabele, v kateri so viri povezav zapisani v prvi stolpec, povezava pa se zapiše v prvo vrstico
3. Identifikacija začetnega objekta (<*bpmn2:startEvent*>) in podatkov o začetni povezavi (<*bpmn2:sequenceFlow*>).
4. Branje in številčenje objektov povezav (<*bpmn2:sequenceFlow*>) v vrstnem redu. Oznaka izvornega objekta se zapiše v ustrezno polje tabele. Pri kontrolnih objektih se v polje vpiše pogoj, prebran iz podatkovnega polja 'ime'.
5. Razvrščanje po zaporedni številki zapisa v stolpcu in zaporedni številki zapisa v vrstici.

Na sliki 2 je predstavljen primer tabele, ki opisuje diagram, prikazan na sliki 1.

	A	B	C	D	E	F	G	H	I	J	K	L	M
3	Element				bpmn2:task	bpmn2:parallelGateway		bpmn2:task	bpmn2:task	bpmn2:task	bpmn2:parallelGateway	bpmn2:task	bpmn2:endEvent
4	el_61506180065636f222ca4e4022933058	1	bpmn2:startEvent		ST								
5	el_46143783265636f2226ae96094561825	2	bpmn2:task	Naloga1		TA							
6	el_19749221665636f72c19fd0041879667	3	bpmn2:parallelGateway				PG						
7	el_79705082765636f221fff04046807368	4	bpmn2:task	Naloga2				TA					
8	el_90166900465636f22221c46048174055	5	bpmn2:task	Naloga3					TA				
9	el_63007491165636f22246298040974308	6	bpmn2:task	Naloga4						TA			
10	el_82908638065636f9ad3ac55042788496	7	bpmn2:parallelGateway								PG		
11	el_32252431665636f221dd299036640846	8	bpmn2:task	Naloga5								TA	
12	el_21901879565636f222b99c2009278295	9	bpmn2:endEvent										EE

Slika 2: Prikaz tabele za primer vzporednega prehoda
Vir: Lasten

Vrstice v tabeli predstavljajo izvorne elemente, stolpci pa ponorne elemente. Vsako polje, presečišče v tabeli, vključuje informacije o nalogi, ki jo je potrebno vključiti v ogrodje kode v Javi, ter pogoje za prehode, ki določajo potek izvajanja. Posebnost predstavljajo kontrolni stavki, pri katerih je na začetku nastavljen kazalnik (stikalo), ki zagotavlja, da se vse nadaljnje aktivnosti pravilno pretvorijo v ogrodje kode Java.

4.2 Postopek pretvorbe tabele v ogrodje Javanske kode

Tabela 1: BPMN kontrolni stavki

Kontrolni stavek	Kratka oznaka	Opis	Pogoj
bpmn2:parallelGateway	AND	Vzporedni prehod, kjer se vzporedno izvedejo VSE poti brez preverjanja pogojev.	Ne
bpmn2:inclusiveGateway	OR	Vključujoči prehod, pri katerem lahko proces nadaljuje po ENI, VEČ ali VSEH možnih poteh hkrati, odvisno od izpolnjenih pogojev. Obstajati mora tudi privzeta pot, ki se izvede, če noben pogoj ni izpolnjen.	Da
bpmn2:exclusiveGateway	XOR	Pri izključujočem prehodu se izvede SAMO ENA izmed možnih poti, odvisno od izpolnjenega pogoja. Ko je pogoj izpolnjen, se naslednji ne preverjajo več. Obstajati mora tudi privzeta pot, ki se izvede, če noben pogoj ni izpolnjen.	Da

Prvi element v tabeli sproži generiranje datoteke Java. V tej fazi se dodajo tudi osnovni elementi programskega modula, kot sta »*public class Main*« in »*public static void*«.

Po začetnem delu običajno sledijo posamezne naloge, ki se lahko izvajajo zaporedno ali pogojno, odvisno od kontrolnih stavkov. V Tabeli 1 so predstavljeni različni tipi kontrolnih stavkov. Različni tipi prehodov (XOR, OR, AND) imajo svoje posebnosti pri generaciji kode, saj lahko vsak prehod vodi v eno ali več izhodnih poti, kar vpliva na strukturo in pogoje v ustvarjeni kodi.

```
// bpmn2:parallelGateway Start
// Create an ExecutorService with a fixed pool size
int threadPool_1 = 3;
ExecutorService executorService_1 = Executors.newFixedThreadPool(threadPool_1);

// Create runnable task_1_1
Runnable task_1_1 = () -> {
    // bpmn2:task
    public static void Naloga2() {
        // Add your Naloga2 logic here
    }
};

// Create runnable task_1_2
Runnable task_1_2 = () -> {
    // bpmn2:task
    public static void Naloga3() {
        // Add your Naloga3 logic here
    }
};

// Create runnable task_1_3
Runnable task_1_3 = () -> {
    // bpmn2:task
    public static void Naloga4() {
        // Add your Naloga4 logic here
    }
};

// bpmn2:parallelTask
executorService_1.submit(task_1_1);
// bpmn2:parallelTask
executorService_1.submit(task_1_2);
// bpmn2:parallelTask
executorService_1.submit(task_1_3);
// bpmn2:parallelGateway End
// Shutdown the ExecutorService
executorService_1.shutdown();

try {
    // Wait for tasks to complete
    executorService_1.awaitTermination(Long.MAX_VALUE, TimeUnit.NANOSECONDS);
} catch (InterruptedException e) {
    e.printStackTrace();
}
```

Slika 3: Izsek programske kode Java za opis vzporednega prehoda
Vir: Lasten

Rezultat pretvorbe tabele je ogrodje Java programske kode, ki predstavlja osnovno strukturo. Koda še ne vsebuje natančnih pogojev za izvajanje nadzornih stavkov (if-else, while itd.). Ker model BPMN običajno ne vsebuje specifičnih pogojev za

odločanje, temveč le splošno strukturo procesa, pogoje dodajamo kasneje, ročno ali z drugim delom avtomatiziranega procesa. Na sliki 3 je prikazan izsek programske kode Java za opis vzporednega prehoda, prikazanega na sliki 1.

5 Zaključek

V raziskavi je predstavljen postopek pretvorbe BPMN modela v ogrodje Java kode prek dvodimenzionalne tabele, pri čemer se uporablja neobdelana BPMN koda v razširjenem XML formatu. Razvita rešitev omogoča hitro in preprosto uporabo modela BPMN za avtomatsko generiranje osnovne strukture kode, ki se nato dopolni s pogoji in programskimi ukazi. Trenutna implementacija podpira osnovne gradnike BPMN, kar omogoča učinkovito inicializacijo programske kode. Pametno pogodbo z Javansko programsko kodo lahko uporabimo v okolju Hyperledger Fabric.

Tehnologija veriženja blokov v poslovnem procesu izboljša sledljivost procesov, saj zapisi v verigi blokov ostanejo trajno shranjeni, kar odpravlja potrebo po dodatnih revizijskih sledeh in omogoča transparentno preverjanje zgodovine izvajanja procesov. Del te tehnologije so tudi pametne pogodbe, s katerimi implementiramo poslovno logiko. Uporaba prikazane metode bi lahko znatno pospešila razvoj pametnih pogodb ter omogočila njihovo neposredno integracijo v rešitve veriženja blokov. Uporaba avtomatizirane pretvorbe bi imela več koristi, vključno z zmanjšanjem stroškov razvoja, povečanjem varnosti poslovnih procesov ter zmanjšanjem napak pri pripravi kode.

V prihodnje bi bilo smiselno razširiti metodo z dodatnimi gradniki BPMN ter nadgraditi prevajalnik BPMN-to-Java z naprednejšo podporo za kompleksne procese. S tem bi omogočili še bolj avtomatizirano in natančno generacijo kode, kar bi dodatno povečalo uporabnost rešitve v različnih domenah poslovnih procesov in aplikacij tehnologije veriženja blokov. Prav tako bi lahko BPMN pretvorili v druge programske jezike ter tako razširili uporabnosti tudi na druge verige blokov.

Literatura

Alamsyah, A., & Setiawan, I. P. S. (2025). Enhancing privacy and traceability of public health insurance claim system using blockchain technology. *Frontiers in Blockchain*, *8*, 1474434. https://doi.org/10.3389/fbloc.2025.1474434

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., … Yellick, J. (2018). Hyperledger fabric: A distributed operating system for permissioned blockchains. *Proceedings of the Thirteenth EuroSys Conference*, 1–15. https://doi.org/10.1145/3190508.3190538

Bodorik, P., Liu, C. G., & Jutla, D. (2023). TABS: Transforming automatically BPMN models into blockchain smart contracts. *Blockchain: Research and Applications*, *4*(1), 100115. https://doi.org/10.1016/j.bcra.2022.100115

Dutta, P., Choi, T.-M., Somani, S., & Butala, R. (2020). Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation Research. Part E, Logistics and Transportation Review*, *142*, 102067. https://doi.org/10.1016/j.tre.2020.102067

Faheem Ahmad Reegu, F. A. R. (2021). A systematic review of benefits and threats of blockchaintechnology in Healthcare. *International Journal of Engineering Technology and Management Sciences*, 33–36. https://doi.org/10.46647/ijetms.2021.v05i03.006

Gabryelczyk, R., & Jurczuk, A. (2017). Does Experience Matter? Factors Affecting the Understandability of the Business Process Modelling Notation. *Procedia Engineering*, *182*, 198–205. https://doi.org/10.1016/j.proeng.2017.03.164

Hyperledger. (2024). Smart Contracts and Chaincode. *A Blockchain Platform for the Enterprise, Hyperledger Fabric*. https://hyperledger-fabric.readthedocs.io/en/release-2.2/smartcontract/smartcontract.html

Lauster, C., Klinger, P., Schwab, N., & Bodendorf, F. (2020). Literature review linking blockchain and business process management. *Proceedings of the 15th International Conference on Business Information Systems 2020 'Developments, Opportunities and Challenges of Digitization', WIRTSCHAFTSINFORMATIK 2020*. https://doi.org/10.30844/wi_2020_r10

Liu, C. G., Bodorik, P., & Jutla, D. (2022). Supporting Long-term Transactions in Smart Contracts. *2022 Fourth International Conference on Blockchain Computing and Applications (BCCA)*, 11–19. https://doi.org/10.1109/BCCA55292.2022.9922193

López-Pintado, O., Dumas, M., García-Bañuelos, L., & Weber, I. (2019). *Interpreted Execution of Business Process Models on Blockchain*. 206–215. https://doi.org/10.1109/EDOC.2019.00033

López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I., & Ponomarev, A. (2019). Caterpillar: A business process execution engine on the Ethereum blockchain. *Software - Practice and Experience*, *49*(7), 1162–1193. https://doi.org/10.1002/spe.2702

Nakamoto, S. (2009). Bitcoin: A Peer-to-Peer Electronic Cash System. *Cryptography Mailing List at Https://Metzdowd.Com*. https://bitcoin.org/bitcoin.pdf

Object Management Group. (2016). *Business Process Model and Notation*. http://www.bpmn.org/

Scott, D. J., Broyd, T., & Ma, L. (2021). Exploratory literature review of blockchain in the construction industry. *Automation in Construction*, *132*, 103914. https://doi.org/10.1016/j.autcon.2021.103914