x402: arhitektura in praktična uporaba strojnih plačil z uporabo HTTP 402

Martin Ferenec, Adnan Bratanović, Teo Lah, Muhamed Turkanović

Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Maribor, Slovenija
martin.ferenec@um.si, adnan.bratanovic@student.um.si, teo.lah@student.um.si, muhamed.turkanovic@um.si

Članek predstavi protokol x402, odprt standard za strojno izvajanje plačil na spletu, ki po treh desetletjih neuporabe statusno kodo HTTP 402 prvič uveljavlja kot dejansko delujoč plačilni mehanizem. Opisano je trenutno stanje zaračunavanja dostopa do podatkov ter ovire, kot so odvisnost od posrednikov, visoki stroški mikrotransakcij in odsotnost vgrajenega plačilnega mehanizma, ki so pripeljale do nastanka protokola. Podrobneje je predstavljena arhitektura x402 (odjemalec, strežnik vira, denarnica, plačilne sheme, plačilni posrednik), delovanje in primerjava različic v1 in v2. Sledi primerjava z alternativnim protokolom Machine Payments Protocol (MPP) s priporočilom, kdaj izbrati katerega. Prispevek se konča s predstavitvijo uporabe x402 v referenčnih implementacijah agentov umetne inteligence in v podatkovnih prostorih, na primeru povezovalnika DSX Engine v podatkovnem prostoru DADS (DIH AGRIFOOD Data Space).

Ključne besede:
HTTP 402
spletna plačila
strojna plačila
plačila za AI agente
pametne pogodbe

Univerzitetna založba Univerze v Mariboru

ISBN 978-961-299-188-3
DOI https://doi.org/10.18690/um.feri.7.2026.3

1 Uvod

Digitalno gospodarstvo v vse večji meri temelji na avtomatizirani izmenjavi podatkov med sistemi. Pri spletnih storitvah se je kot prevladujoč pristop uveljavil arhitekturni slog REST, formaliziran leta 2000 [1]. Kljub tehnični zrelosti tovrstnih spletnih vmesnikov pa področje neposredne strojne monetizacije podatkov dolgo ni imelo odprtega in standardiziranega plačilnega mehanizma. Standard HTTP je sicer že od svojih začetkov predvideval statusno kodo 402 (Payment Required), namenjeno digitalnim plačilom, vendar trenutno veljavna specifikacija to kodo označuje kot rezervirano za prihodnjo uporabo, zaradi česar v praksi ni bila nikoli splošno uveljavljena [2]. Posledično ponudniki podatkov večinoma uporabljajo lastne modele naročnin ali posredniške plačilne sisteme, ki povečujejo stroške, uvajajo odvisnost od centraliziranih ponudnikov ter zmanjšujejo interoperabilnost med sistemi [3] [4].

To vprašanje je v zadnjih letih dobilo novo težo z uveljavljanjem avtonomnih agentov umetne inteligence, ki podatkovne vire in programske vmesnike kličejo brez sprotnega človekovega posredovanja. Postopki, zasnovani okoli človeka kot plačnika, torej registracija računa, sklenitev naročnine in ročna potrditev plačila, v takem okolju postanejo ozko grlo, saj jih agent ne more izvesti samostojno. Prispevek zato obravnava vprašanje, kako plačilo vgraditi neposredno v protokol HTTP tako, da je izvedljivo strojno in brez predhodno vzpostavljenega poslovnega razmerja med odjemalcem in ponudnikom.

Da je tak mehanizem sploh smiseln, morajo biti izpolnjeni tudi ekonomski pogoji. Analize mikroplačil dosledno kažejo, da ta niso spodletela zaradi tehnoloških omejitev, temveč zaradi stroška na transakcijo in kognitivnega bremena, ki ju človeški plačnik ni bil pripravljen sprejeti [5] [6]. Oba dejavnika se v strojnem okolju bistveno spremenita. Agent nima miselnih transakcijskih stroškov, poravnava s stabilnimi kriptožetoni v nizkocenovnih omrežjih pa strošek posamezne transakcije zniža na raven, pri kateri je zaračunavanje posameznega klica programskega vmesnika ekonomsko vzdržno [7].

Prispevek prinaša štiri sklope. Prvi je sistematičen pregled ovir, ki jih ima obstoječa plačilna infrastruktura pri zaračunavanju strojnega dostopa do podatkov. Drugi je celovit opis arhitekture in poteka protokola x402, vključno s primerjavo različic v1 in v2 ter pregledom obstoječih implementacij plačilnih posrednikov. Tretji je primerjalna analiza z alternativnim protokolom Machine Payments Protocol (MPP) s priporočili, kdaj je smiselna izbira posameznega protokola. Četrti je predstavitev dveh referenčnih implementacij, ki x402 uporabita v praksi.

Prispevek je organiziran tako, da naslednje poglavje opiše prevladujoče načine izmenjave in zaračunavanja dostopa do podatkov ter ovire, ki so pripeljale do nastanka protokola. Sledi podroben opis arhitekture in delovanja x402, primerjava različic, pregled primerov uporabe in implementacij plačilnih posrednikov ter primerjava z MPP. Zadnji vsebinski del predstavi uporabo x402 pri agentih umetne inteligence in v podatkovnih prostorih na primeru povezovalnika DSX Engine v podatkovnem prostoru DADS, sklep pa povzame ugotovitve in odprta vprašanja.

2 Trenutno stanje

2.1. Prevladujoč način izmenjave podatkov

Izmenjava podatkov med sistemi dandanes v veliki meri poteka prek spletnih vmesnikov, zasnovanih po arhitekturnem slogu REST, ki ga je leta 2000 v svoji doktorski disertaciji formaliziral Roy T. Fielding in ki od takrat vodi zasnovo arhitekture sodobnega spleta. Ponudnik podatkov izpostavi množico končnih točk, odjemalec pa nanje pošilja zahtevke HTTP in v odgovoru prejme podatke [1]. Dostop do teh vmesnikov je pri komercialni rabi skoraj vedno vezan na avtentikacijo ter avtorizacijo, saj morajo ponudniki vsak zahtevek pripisati določenemu potrošniku, še posebej kadar želijo za svoje storitve zaračunavati [8].

2.2. Zaračunavanje dostopa

Ponudniki podatkov in vmesnikov (API-jev) zaradi odsotnosti standardiziranega plačilnega mehanizma, zgrajenega na podlagi protokola HTTP, uporabljajo vrsto lastnih poslovnih modelov monetizacije. Empirična analiza 268 dejanskih vmesnikov API in 54 podrobneje modeliranih cenovnih shem je pokazala, da so te sheme sestavljene iz naročniških paketov z vnaprej določenimi omejitvami in stroškom, ki je lahko fiksen (npr. mesečni znesek) ali dinamičen (npr. odvisen od porabe), pri čemer se dodaten strošek za preseganje kvote pojavi v približno 18,2 % analiziranih cenovnih shem [3]. V širšem smislu API-je vse pogosteje obravnavajo kot samostojne poslovne izdelke, katerih odpiranje navzven ustvarja nove poslovne priložnosti in organizacijske strukture (t. i. platformni posli) [8]. Nastali so tudi trgi podatkov (angl. data marketplaces), ki delujejo kot dvo- ali večstranske platforme za prodajo in nakup podatkov med uporabniki, ki so razpršeni.

Ne glede na izbrano cenovno shemo je skupna značilnost vseh naštetih pristopov, da je **plačilo strukturno ločeno od samega zahtevka** za podatke, pri čemer se mora uporabnik najprej registrirati, izbrati naročniški paket in šele nato prejeti poverilnico, s katero lahko dostopa do podatkov [3]. Del ponudnikov namesto lastne infrastrukture uporablja centralizirane trge podatkov oz. API-je, ki v zameno za poenostavljeno odkrivanje, naročanje in obračunavanje podatkov prevzamejo del neposrednega sporazumevanja s končnim uporabnikom. Raziskave kažejo, da tak model, čeprav zmanjšuje vstopne stroške za manjše ponudnike, hkrati osredotoča izmenjavo podatkov in denarja pri operaterju platforme ter uvaja dodatne tehnične in organizacijske ovire pri zagotavljanju varnosti, suverenosti, preglednosti, učinkovitosti in skladnosti z zakonodajo [4].

2.3. Izvedba plačila

Plačilo se v opisanih modelih izvede kot ločen, praviloma enkraten ali periodičen poslovni dogodek, ne kot sestavni del posameznega zahtevka za podatke. Uporabnik plača vnaprej ali pa se mu obračuna naknadno, na podlagi agregirane porabe [3]. Pri čezmejnih plačilih, ki so za mednarodne podatkovne prostore in trge pogosta, se v transakcijo vključi dodatna veriga finančnih posrednikov. Po analizah Banke za mednarodne poravnave (BIS) in pobude G20 za izboljšanje čezmejnih plačil povprečni stroški čezmejnih plačil še vedno znatno presegajo ciljno vrednost 1 %, ki si jo je skupina G20 zastavila za leto 2027, pri čemer so stroški v posameznih regijah, na primer v podsaharski Afriki, občutno višji od globalnega povprečja, napredek pri hitrosti, dostopnosti in preglednosti teh plačil pa je počasnejši od načrtovanega [9].

2.4. Ovire

Analiza obstoječih načinov izmenjave in zaračunavanja dostopa do podatkov razkriva pet ponavljajočih se ovir. Prve tri so ekonomske in vedenjske narave, preostali dve pa izvirata iz razdrobljenosti plačilne infrastrukture in tehnične zahtevnosti na strani ponudnika.

Strukturna odvisnost od posrednikov. Centralizirani trgi podatkov in API-jev v menjavo vnašajo dodatno plast, ki nadzoruje razmerje s stranko in zase zadrži del vrednosti transakcije, zaradi česar so ponudniki odvisni od infrastrukture, nad katero nimajo neposrednega nadzora [4].

Visoki stroški mikrotransakcij. Tradicionalna plačilna infrastruktura ima stroške na transakcijo, ki so pri zneskih pod nekaj centov ali dolarjev nesorazmerno visoki glede na sam znesek. Andrew Odlyzko je že leta 2003 na konferenci Financial Cryptography utemeljeval, da ovire za uporabo mikroplačil niso predvsem tehnološke, temveč ekonomske, sociološke in psihološke narave. Potrošniki dosledno raje izberejo pavšalno ceno kot veliko število drobnih plačil, kar sistemsko spodkopava poslovni model mikroplačil [5].

Kognitivni napor. Evropska centralna banka v svoji analizi mikroplačil opozarja, da so t. i. miselni transakcijski stroški, kognitivni napor pri vsaki posamezni odločitvi o plačilu, relevantni predvsem takrat, ko je vsaj ena od strani v transakciji človek, medtem ko lahko postanejo nepomembni, ko transakcije med seboj izvajajo avtomatizirane naprave [6]. To neposredno kaže na omejitev vseh danes prevladujočih plačilnih načinov, ki so bili zasnovani okoli človeka kot plačnika (izpolnjevanje obrazcev, vnos kartice, potrditev nakupa), ne pa okoli strojnega plačevanja z uporabo na primer avtonomnih programskih agentov.

Geografska omejenost in nizka interoperabilnost. Razpršenost plačilnih shem, različne regulativne zahteve in visoki stroški pri čezmejnem poslovanju povzročajo, da ostaja dostop do plačljivih podatkovnih virov omejen, sistemi med različnimi ponudniki podatkov pa med seboj niso interoperabilni [9].

Tehnično in organizacijsko breme za ponudnike. Vzpostavitev merjenja porabe, uveljavljanja omejitev in zanesljivega obračunavanja je za ponudnika zahtevna tako s tehničnega kot z organizacijskega vidika. Raziskave s področja trgovanja s podatki ugotavljajo, da prav ti izzivi in ne le pomanjkanje povpraševanja pojasnjujejo, zakaj le maloštevilna podjetja uspejo ustvariti oprijemljivo vrednost neposredno iz svojih podatkov [4].

Naštete ovire, torej strukturna odvisnost od posrednikov, nesorazmerni stroški mikrotransakcij, kognitivni napor človeškega plačnika, geografska razdrobljenost plačilnih shem in tehnično-organizacijsko breme za ponudnike, so skupaj z odsotnostjo v protokol vgrajenega plačilnega mehanizma ozadje, na katerem je nastal protokol x402, ki kodo HTTP 402 prvič uveljavlja kot dejansko delujoč, standardiziran način zahtevanja in izvedbe plačila neposredno v obstoječem zahtevku HTTP.

3 Protokol x402

x402 je odprt plačilni protokol, ki omogoča neposredno zaračunavanje dostopa do spletnih virov na podlagi protokola HTTP. Temelji na statusni kodi 402 Payment Required, ki je bila v standardu HTTP predvidena za primere, ko je za dostop do vira potrebno plačilo, vendar v običajnih spletnih arhitekturah doslej ni bila splošno uporabljena. x402 tej statusni kodi dodaja standardiziran postopek izmenjave podatkov o plačilu, preverjanja avtorizacije in izvedbe poravnave [7].

Glavni namen protokola je omogočiti strojna plačila brez tradicionalnega postopka registracije uporabniškega računa, vzpostavljanja naročnine ali pridobivanja ključa API. Plačljivi spletni vir lahko zato uporablja človek, običajna programska aplikacija ali avtonomni programski agent. Plačilo je vključeno neposredno v cikel zahtevka in odgovora HTTP, zaradi česar je x402 primeren predvsem za plačljive vmesnike API, digitalne vsebine, mikrostoritve in storitve, ki jih uporabljajo agenti umetne inteligence [7] [10].

Protokol ne določa enega samega plačilnega omrežja ali ponudnika infrastrukture. Plačilne zahteve so ločene od konkretnega načina poravnave, zato je mogoče podpirati različne verige blokov, kriptovalute, plačilne sheme in ponudnike storitev za preverjanje ter poravnavo plačila. Referenčna implementacija trenutno največ pozornosti namenja plačilom s stabilnimi kriptožetoni, kot je USDC, vendar je protokol zasnovan razširljivo in neodvisno od posamezne kriptovalute in verige blokov, na kateri je kriptovaluta [7].

3.1. Delovanje

Komunikacija x402 praviloma poteka v dveh zaporednih zahtevkih HTTP. Pri prvem zahtevku odjemalec še nima podatkov o plačilu. Strežnik zato dostop zavrne s HTTP kodo 402 Payment Required in posreduje zahteve za izvedbo plačila [7] [10].

V različici x402 v2 so podatki o zahtevanem plačilu vključeni v glavo PAYMENT-REQUIRED. Ti podatki vsebujejo [11]:

- Uporabljeno različico protokola x402
- Podatkovni vir (angl. resource):

  spletni naslov URL do tega podatkovnega vira
  Opis vira (neobvezno)
  format MIME dejanskih podatkov, ki bodo poslani po uspešnem plačilu (neobvezno)

- Seznam posameznih plačilnih informacij za izvedbo plačila, ki jih ponuja ponudnik:

  Plačilno shemo
  Omrežje - verigo blokov v CAIP-2 formatu
  Znesek
  Naslov pametne pogodbe na zgoraj opredeljeni verigi blokov (kriptožeton, kot je na primer USDC)
  Naslov kripto denarnice, ki prejme plačilo (kripto denarnica ponudnika podatkov)
  Največja dovoljena časovna omejitev v sekundah za izvedbo plačila po prejemnikovi avtorizaciji plačila
  Dodatne informacije, potrebne za uporabo določenih kriptožetonov (neobvezno)

- Opis napake (neobvezno)

Odjemalec na podlagi prejetega seznama plačilnih informacij izbere eno podprto kombinacijo omrežja in plačilne sheme. V svoji denarnici nato pripravi in kriptografsko podpiše plačilni objekt. Prvotni zahtevek ponovi, podpisano avtorizacijo pa posreduje v glavi PAYMENT-SIGNATURE.

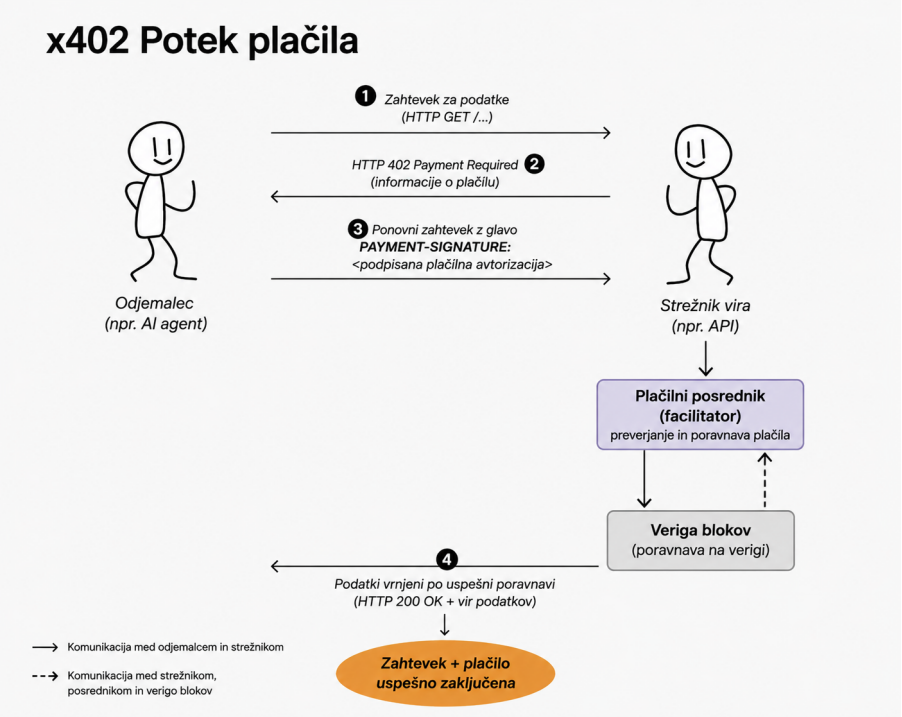
Strežnik ali izbrani plačilni posrednik preveri veljavnost podpisa, skladnost z zahtevanim zneskom, naslov kripto denarnice, časovno omejitev in podprte verige blokov. Če je preverjanje uspešno, se plačilo poravna v skladu z uporabljeno plačilno shemo. Strežnik nato vrne rezultat plačila v glavi PAYMENT-RESPONSE in v primeru uspešnega plačila zahtevani vir [7] [10].

Celoten postopek je shematsko prikazan na sliki 1 [7] [11].

Takšen način delovanja odpravlja potrebo po preusmeritvi na ločeno plačilno stran. Plačilna logika ostane v okviru istega programskega vmesnika, zaradi česar jo je mogoče vključiti v knjižnice, strežniško vmesno programsko opremo in avtonomne agente.

Za preprečevanje večkratnega zaračunavanja pri ponovitvah zahtevkov je na voljo razširitev za identifikator plačila (angl. Payment-Identifier). Odjemalec zahtevku doda enolični identifikator, na podlagi katerega lahko strežnik prepozna že obdelano plačilo. To je pomembno pri prekinitvah omrežne povezave, samodejnih ponovitvah in arhitekturah z več uravnoteženimi strežniki [7].

Pri paketni poravnavi se avtorizacija in končni prenos sredstev časovno ločita. Odjemalec za posamezne zahtevke podpisuje zunajverižna dokazila, strežnik pa več takšnih avtorizacij pozneje poravna v eni transakciji. S tem se zmanjšajo stroški in zakasnitev posameznega klica, kar je posebej pomembno pri storitvah z velikim številom nizko vrednostnih transakcij [7].



Slika 1: Potek plačila po protokolu x402 (prirejeno po [12])

3.2. Arhitektura

Arhitekturo x402 sestavlja pet osnovnih elementov: odjemalec, strežnik vira, denarnica, plačilna shema in plačilni posrednik za preverjanje ter poravnavo plačil (angl. facilitator) [7].

Odjemalec je aplikacija, spletni brskalnik, programska knjižnica ali avtonomni agent, ki želi dostopati do plačljivega vira. Odjemalec mora znati obdelati odgovor s statusom 402, pripraviti ustrezen plačilni objekt, ga podpisati in prvotni zahtevek ponovno poslati skupaj s plačilno avtorizacijo.

Strežnik vira zagotavlja plačljivi vmesnik API, vsebino ali drugo digitalno storitev. Za zaščiteni vir določi ceno, podprta omrežja, sprejeta sredstva, naslov prejemnika in uporabljeno plačilno shemo. Ob prejemu zahtevka brez ustreznega plačila vrne podatke, na podlagi katerih lahko odjemalec pripravi plačilno avtorizacijo.

Denarnica ima dvojno vlogo. Predstavlja plačilni mehanizem, s katerim odjemalec avtorizira prenos sredstev, hkrati pa lahko deluje kot kriptografska identiteta uporabnika ali programskega agenta. Zasebni ključ praviloma ostane pri odjemalcu, podpisovanje pa se izvede lokalno. Strežnik zato ne potrebuje dostopa do zasebnega ključa ali skrbništva nad sredstvi uporabnika [7].

Plačilna shema določa semantiko obračuna in poravnave. Ponudnik določi plačilno shemo glede na naravo podatkov oziroma storitve, ki jo izpostavlja v vmesniku API. Na voljo so tri plačilne sheme [7] [11]:

- exact, pri kateri odjemalec avtorizira plačilo natančnega zneska, ki ga določi ponudnik;
- upto, pri kateri odjemalec avtorizira najvišji dovoljeni znesek, strežnik pa poravna samo dejansko uporabo;
- batch-settlement, pri kateri se posamezne avtorizacije zbirajo in pozneje poravnajo v skupni transakciji.

Shema exact je primerna za vire s fiksno ceno, na primer za dostop do posameznega dokumenta ali enkratni klic vmesnika API. Shema upto je namenjena storitvam, katerih cena je odvisna od dejanske porabe, kot so število obdelanih žetonov jezikovnega modela, čas izvajanja, prenesena količina podatkov ali poraba računalniških virov.

Paketna poravnava je namenjena visokofrekvenčnim interakcijam, pri katerih bi bila izvedba ločene verižne transakcije za vsak zahtevek prepočasna ali ekonomsko neučinkovita [7].

Samostojen plačilni posrednik je neobvezna, vendar priporočena infrastrukturna komponenta. Strežniku omogoča, da preverjanje plačil in izvedbo poravnave prenese na zunanjo storitev. Strežniku zato ni treba neposredno upravljati povezave z verigami blokov, preverjati vseh podpisov ali izvajati transakcij. Protokol je zasnovan brez dovoljenj, zato lahko plačilnega posrednika upravlja sam ponudnik podatkov, ponudnik javne infrastrukture, podjetje ali upravljavec zasebnega sistema [7].

Plačilni posrednik po preverjanju in poravnavi plačila rezultat vrne strežniku, ki odjemalcu posreduje zahtevani vir. Ker je komunikacija izvedena prek običajnih zahtevkov HTTP in ne zahteva trajne uporabniške seje, je osnovni postopek brez stanja (angl. stateless). Takšna zasnova olajša horizontalno skaliranje strežnikov, uporabo posredniških strežnikov in vključitev x402 v obstoječe spletne arhitekture.

Protokol dopolnjujejo izbirne razširitve. Med pomembnejšimi so prijava z denarnico Sign-In-With-X, identifikator plačila za zagotavljanje idempotentnosti, podpisane ponudbe in potrdila ter mehanizmi za sponzoriranje stroškov transakcij. Sign-In-With-X temelji na standardu CAIP-122 in omogoča, da uporabnik dokaže svojo identiteto s podpisom sporočila z isto denarnico, s katero je predhodno že plačal, kar omogoča seje za večkratno uporabo brez ponavljanja celotnega plačilnega postopka. Identifikator plačila strežniku omogoča, da posamezno plačilno zahtevo enolično poveže z izdanim odgovorom in tako prepozna podvojene poskuse (npr. zaradi ponovnih zahtev ob omrežnih napakah), ne da bi prišlo do dvojnega zaračunavanja ali zamenjave zahtevane storitve s cenejšo. Podpisane ponudbe in potrdila osnovno dokazilo o plačilu razširjajo v kriptografsko preverljiv zapis, ki poleg izvedbe plačila dokazuje tudi pogoje nakupa ter to, kdo in za kakšen namen ga je odobril, kar je pomembno za revizijsko sledljivost, zlasti pri plačilih, ki jih v imenu uporabnika izvajajo avtonomni agenti. Mehanizmi za sponzoriranje stroškov transakcij (npr. prek standarda EIP-2612 ali sponzoriranja odobritve ERC-20) pa omogočajo, da namesto kupca strošek odobritve plačilnega žetona krije plačilni posrednik, zato kupec za celoten postopek plačila ne potrebuje lastnega goriva (angl. gas) [7] [11]. Razširitve se vključijo v življenjski cikel preverjanja in poravnave, ne da bi bilo treba spremeniti osnovni potek komunikacije.

3.3. Primerjava zmožnosti različice 2 z različico 1

Različica x402 v2 ni zgolj sprememba poimenovanja ali strukture paketov. Uvaja standardizirane identifikatorje, modularno programsko arhitekturo, formalizirane razširitve in boljšo podporo zahtevnejšim ekonomskim modelom [7] [10]. Ključne razlike med različicama povzema tabela 1.

Prehod na identifikatorje CAIP-2 odpravlja dvoumnosti pri poimenovanju omrežij ter izboljšuje združljivost med različnimi verižnimi ekosistemi. Omrežje Base se na primer ne označuje več z lastnim besedilnim imenom, temveč s standardiziranim identifikatorjem, ki vključuje prostor imen in referenco omrežja [7] [10].

Druga pomembna sprememba je modularizacija knjižnic. Funkcionalnosti za jedro protokola, posamezna omrežja, odjemalce HTTP in strežniška ogrodja so razdeljene v ločene pakete. Razvijalec lahko zato vključi samo komponente, ki jih njegova aplikacija dejansko potrebuje.

Različica v2 formalizira tudi sistem razširitev. Funkcije, kot so avtentikacija z denarnico, idempotentnost, podpisana potrdila in odkrivanje storitev, se lahko vključijo kot sestavljivi dodatki. Protokol se tako iz mehanizma za enkratno plačilo na zahtevek razvija v splošnejšo ekonomsko plast za programske storitve [7].

Migracija vključuje zamenjavo glav HTTP, prehod na identifikatorje CAIP-2, posodobitev paketov in spremembo konfiguracije strežniških poti. Referenčna programska oprema omogoča tudi začasno podporo obema različicama, kar olajša postopno nadgradnjo obstoječih sistemov [10].

Tabela 1: Primerjava x402 v1 in x402 v2

Vidik	*x402 v1*	*x402 v2*
Glava s plačilnim podpisom	X-PAYMENT	PAYMENT-SIGNATURE
Rezultat poravnave plačila	X-PAYMENT-RESPONSE	PAYMENT-RESPONSE
Posredovanje plačilnih zahtev	V telesu odgovora	HTTP glava PAYMENT-REQUIRED
Identifikacija omrežja	Lastna imena, npr. base-sepolia	Standard CAIP-2, npr. eip155:84532
Programska arhitektura	Manj modularni paketi	Modularni paketi @x402/*
Konfiguracija strežnika	Enostavnejša konfiguracija posamezne poti	Seznam accepts z več shemami in omrežji
Razširitve	Omejeno formalizirane	Standardiziran sistem razširitev
Odkrivanje storitev	Ni del standardiziranega toka	Podpora razširitvi za odkrivanje in sistemu Bazaar
Identiteta denarnice	Predvsem plačilni mehanizem	Podlaga za avtentikacijo in ponovni dostop
Ekonomski modeli	Predvsem enkratna plačila	Fiksna, porabniška, predplačniška in večstopenjska plačila
Združljivost	Osnovna različica	Referenčni SDK-ji lahko podpirajo tudi v1

3.4. Primeri uporabe

Protokol x402 se v praksi uporablja na številnih področjih. Primeri uporabe so:

- Dostop agentov umetne inteligence do API-jev, kjer agent plača za posamezen klic (npr. vremenski podatki, finančne kotacije, lokacijske storitve);
- Monetizacija vsebin in podatkov, na primer plačljiv dostop do posameznih člankov, raziskovalnih objav ali podatkovnih tokov v realnem času, kjer stranka plača le za dejansko uporabljeno vsebino namesto za celoten paket;
- Trgi podatkov (angl. data marketplaces), kjer ponudniki podatkov in API-jev omogočajo avtomatizirano trgovanje z nabori podatkov ali tokovi podatkov;
- Storitve, obračunane po porabi, kjer se aplikacije in računalniške storitve (npr. inferenca jezikovnih modelov, obdelava žetonov, računski viri) zaračunavajo sproti, glede na dejansko porabljene vire;
- Igre in virtualna sredstva, kjer se x402 uporablja za nakupe v igrah in prenose virtualnih sredstev;
- Plačila med agenti (agent-to-agent), kjer avtonomni agenti med seboj najemajo storitve ali si izmenjujejo vire in plačila izvedejo neposredno, brez posredovanja človeka;
- Naprave interneta stvari (IoT) in druga strojna plačila, kjer nizkocenovne naprave samostojno plačujejo za dostop do storitev ali podatkov.

Podobno kot pri drugih odprtih protokolih se je tudi okoli x402 razvila širša razvijalska skupnost, ki jo spodbujajo številna tekmovanja, ki jih organizirajo podjetja, kot so Coinbase, Solana in Cronos. Tovrstna tekmovanja so pospešila nastanek velikega števila konkretnih aplikacij, ki uporabljajo protokol x402.

Med prijavljenimi projekti tovrstnih tekmovanj velja izpostaviti naslednje primere:

- Avtonomne agentske ekonomije, kjer agenti drug drugega najamejo in plačujejo za opravljene storitve z mikroplačili v kriptožetonih;
- Orodja za samodejno investiranje, ki uporabljajo x402 za izvajanje strategij dolarskega povprečenja stroškov (angl. dollar-cost averaging) za avtonomne portfelje;
- Infrastrukturo za varna agentska plačila, ki agentom omogoča premikanje sredstev z omejenimi, programsko določenimi pooblastili;
- Rešitve za samostojno plačevanje naprav IoT, kjer nizkocenovne naprave samostojno izvajajo plačila na verigi blokov;
- Avtomatizacijo naročil pri spletni trgovini, kjer agenti umetne inteligence samostojno izvedejo nakup v imenu uporabnika [10];
- Sisteme za vzpostavljanje bonitete in ocenjevanja agentov, ki omogočajo vzpostavitev zaupanja med avtonomnimi storitvami na podlagi zgodovine plačil;
- Poenostavljeno integracijo x402 prek konfiguracijskih datotek, namenjeno hitrejšemu vključevanju plačil v obstoječe aplikacije.

Raznolikost prijavljenih projektov kaže, da se protokol x402 ne uporablja zgolj kot teoretičen koncept, temveč že v številnih produkcijsko usmerjenih rešitvah, hkrati pa nakazuje smer nadaljnjega razvoja ekosistema, ki se osredotoča predvsem na avtonomno delovanje agentov umetne inteligence, mikroplačila in medstrojna plačila.

3.5. Implementacije plačilnih posrednikov

Ker je plačilni posrednik zamenljiva infrastrukturna komponenta, je nastalo več neodvisnih implementacij, ki se med seboj razlikujejo po podprtih omrežjih in kriptožetonih, zanesljivosti, uporabnosti in ceni.

Primeri uveljavljenih implementacij [7]:

- CDP Facilitator (Coinbase): brezplačna poravnava USDC na omrežju Base Mainnet z vgrajenim preverjanjem skladnosti transakcij (angl. Know Your Transaction, KYT) ter preverjanjem sankcijskih seznamov urada OFAC (angl. Office of Foreign Assets Control) pri vsaki transakciji. Zahteva avtentikacijo s ključem CDP API in uporablja omejitve hitrosti (angl. rate limits) glede na raven računa.
- PayAI Facilitator: pokriva več verig blokov (Base, Ethereum, Polygon, Avalanche, Sei, IoTeX, Peaq, Solana) in podpira vse žetone EIP-3009 ter SPL. Avtentikacija ni potrebna; zasnovan je za podporo širokemu naboru verig blokov.
- Mogami Facilitator: podpira Base in Base Sepolia, avtentikacija ni potrebna. Posebej je optimiziran za odjemalske in strežniške implementacije v programskem jeziku Java prek lastnega Java SDK-ja.
- Polygon Facilitator: produkcijsko zrel plačilni posrednik, namenjen izključno omrežju Polygon. Podpira preverjanje in poravnavo USDC prek standarda EIP-3009 na omrežju Polygon Mainnet in testnem omrežju Polygon Amoy.

3.6. Alternative in primerjava s protokolom Machine Payments Protocol (MPP)

Ena od alternativ protokolu x402 je MPP, odprt protokol za plačila med programskimi sistemi, ki sta ga zasnovala *Tempo* in *Stripe*. MPP je namenjen zaračunavanju zahtevkov API, klicev programskih orodij in dostopa do digitalnih vsebin. Podobno kot x402 omogoča izvedbo plačila neposredno v okviru komunikacije HTTP, ne da bi bila potrebna preusmeritev uporabnika na ločeno plačilno stran [13].

Tako x402 kot MPP uporabljata kodo HTTP 402 Payment Required, vendar se razlikujeta v načinu vključitve plačilnih podatkov v protokol HTTP. x402 uporablja namenske glave PAYMENT-REQUIRED, PAYMENT-SIGNATURE in PAYMENT-RESPONSE. MPP pa plačilo obravnava kot posebno shemo avtentikacije HTTP, imenovano Payment. Strežnik plačilni izziv posreduje v standardni glavi WWW-Authenticate, odjemalec dokazilo o plačilu pošlje v glavi Authorization, strežnik pa lahko rezultat poravnave vrne v glavi Payment-Receipt [7] [10] [14].

Plačilni izziv MPP vsebuje enolični identifikator, področje veljavnosti, plačilno metodo, namen plačila in podatke, potrebne za njegovo izvedbo. Vključuje lahko tudi čas poteka, opis, zgoščeno vrednost telesa zahtevka in podatke za povezovanje plačila s strežniškim stanjem. Enolični identifikator izziva in možnost vezave izziva na vsebino zahtevka sta namenjena preprečevanju sprememb zahtevka, ponovnega predvajanja plačilnih dokazil in uporabe iste avtorizacije za drug vir [14].

MPP je neodvisen od posamezne valute ali plačilnega omrežja. Osnovna specifikacija opredeljuje splošno ogrodje, posamezne plačilne metode pa so opisane v ločenih specifikacijah. Uradni seznam vključuje metode za plačilne kartice (fiat denar), omrežja EVM, Hedera, Lightning, Solano, Stellar, Tempo in USDC ter integracije Stripe in Near Intents [14].

Protokol razlikuje med različnimi nameni plačila. Namen ***charge*** je enkratno plačilo določenega zneska. Namen ***session*** omogoča postopno obračunavanje storitev, pri katerih se poraba spreminja med izvajanjem. Pri seji lahko odjemalec odpira plačilni kanal in z zaporednimi podpisanimi dokazili povečuje skupni odobreni znesek. Takšen model je primeren za pretočno izvajanje jezikovnih modelov, obračunavanje posameznih žetonov, dolgotrajne računske postopke in druge merjene storitve [14].

Specifikacije MPP vključujejo tudi mehanizem za odkrivanje plačljivih storitev in preslikavo plačilne sheme na komunikacijo JSON-RPC ter Model Context Protocol (MCP). MPP zato poleg običajnih virov HTTP neposredno obravnava tudi plačljiva orodja in storitve, ki jih uporabljajo agenti umetne inteligence [14].

Glavne razlike med protokoloma so prikazane v tabeli 2 [7] [10] [14]. Prednost x402 je razmeroma preprosta integracija, jasna vloga plačilnega posrednika in obstoječ nabor referenčnih knjižnic za priljubljena strežniška ogrodja. Plačilni posrednik lahko ponudnika storitve razbremeni neposrednega preverjanja transakcij in izvedbe poravnave plačil na verigah blokov.

MPP se tesneje opira na obstoječo semantiko avtentikacije HTTP. Uporaba glav WWW-Authenticate in Authorization omogoča vključitev plačil v mehanizem, ki ga spletni strežniki, posredniki in knjižnice HTTP že poznajo. Njegova dodatna prednost je formalna ločitev med abstraktnim protokolom, namenom plačila in konkretno plačilno metodo.

MPP z namenom session neposredno podpira postopna plačila za pretočne in merjene storitve [14]. Protokol x402 podobne primere uporabe obravnava s shemama upto in batch-settlement, pri katerih se plačilo prilagodi dejanski porabi oziroma se več avtorizacij pozneje združi v skupno poravnavo [7].

Pri oceni zrelosti je treba upoštevati, da so bile osnovne specifikacije MPP julija 2026 objavljene kot internetni osnutki. Internetni osnutek predstavlja dokument v razvoju, ki še nima statusa dokončnega standarda RFC. Specifikacije se lahko zato pred dokončno standardizacijo spremenijo, nadomestijo ali umaknejo [14].

Tabela 2: Primerjava x402 in MPP

Vidik	*x402*	*MPP*
Osnovni transport	HTTP in koda 402	HTTP in koda 402
Posredovanje plačilnega izziva	PAYMENT-REQUIRED	WWW-Authenticate: Payment
Posredovanje dokazila	PAYMENT-SIGNATURE	Authorization: Payment
Potrdilo o plačilu	PAYMENT-RESPONSE	Payment-Receipt
Protokolni model	Namenske glave in podatkovni objekti x402	Standardna shema avtentikacije HTTP
Abstrakcija plačila	Omrežje in plačilna shema	Plačilna metoda in namen plačila
Osnovni plačilni modeli	exact, upto in paketna poravnava	charge in session
Preverjanje in poravnava	Izvede ju strežnik ali plačilni posrednik	Določena sta v specifikaciji posamezne plačilne metode
Vloga plačilnega posrednika	Plačilni posrednik je izrecno definiran element arhitekture	Protokol ne določa enotne vloge plačilnega posrednika
Odkrivanje storitev	Bazaar in razširitve za odkrivanje	Ločena specifikacija za odkrivanje storitev
Podpora MCP	Vodič in integracija strežnika MCP	Posebna specifikacija transporta JSON-RPC in MCP
Stanje standardizacije	Različica x402 v2 in referenčni SDK-ji	Specifikacije so objavljene kot osnutki IETF

Izbiro med protokoloma za razvijalce, ki načrtujejo integracijo plačilnega mehanizma na svojih API-jih, je smiselno utemeljiti glede na naravo storitve in profil uporabnikov (agentov), ki jo bodo klicali. x402 je priporočljiva izbira za tiste, ki ponujajo ozko usmerjene (npr. funkcije z enim specializiranim namenom), visokofrekvenčne storitve z nizko vrednostjo posamezne transakcije, kot so dostop do podatkovnih virov, klici modelov AI, cenovni tokovi ali računski viri, kjer je ciljna publika že seznanjena s kripto plačili oziroma tehnično zmožna upravljati denarnice in podpisovati transakcije. Prednost je hitra in enostavna integracija brez potrebe po vzpostavitvi računov ali plačilne infrastrukture, dodaten strošek na transakcijo pa je zanemarljiv (le gorivo na verigah blokov). MPP je smiselna izbira za razvijalce, ki gradijo storitve za t. i. poslovne stranke, ponujajo fizično blago ali storitve, ki zahtevajo skladnost s predpisi, nadzor tveganj ali možnost plačila s poslovnimi karticami in kjer agent deluje v imenu uporabnika ali organizacije, ki nima kripto infrastrukture. Model session je primeren za storitve z neprekinjenim, visoko pretočnim značajem, kjer bi plačevanje za vsako posamezno zahtevo posebej (kot pri x402) povzročalo nepotrebne obremenitve. Razvijalcem, ki jim viri in čas to dopuščajo, velja priporočiti podporo obema protokoloma hkrati, saj lahko strežnik v istem odgovoru 402 oglašuje oba plačilna mehanizma in prepusti odjemalcu (agentu), da izbere tistega, ki ga podpira, kar razvijalcu zagotavlja najširšo možno dostopnost njegovih storitev.

4 Uporaba x402 v praksi

4.1. Agenti umetne inteligence

Agenti umetne inteligence lahko za izvedbo uporabnikovih nalog uporabljajo zunanje podatkovne vire, programske vmesnike in računalniške storitve. Pri tem klasični modeli dostopa, ki zahtevajo predhodno registracijo uporabniškega računa, sklenitev naročnine ali ročno izvedbo plačila, omejujejo njihovo avtonomno delovanje.

Protokol x402 jim omogoča, da informacije o ceni in pogojih plačila prejmejo neposredno kot del komunikacije HTTP, pripravijo ustrezno plačilno avtorizacijo ter po uspešni poravnavi nadaljujejo z uporabo zahtevanega vira. Agent tako ne potrebuje ločene interakcije uporabnika s plačilno stranjo, temveč lahko celoten postopek izvede programsko v okviru izvajanja naloge.

Primer takšne uporabe je agent, ki mora za pripravo odgovora zbrati spletne podatke, do katerih nima brezplačnega dostopa (npr. zbiranje objav z družbenih omrežij, pridobivanje podatkov o poslovnih subjektih ali odčitavanje cenovnih tokov), uporabnik pa za nalogo določi proračun.

Kot referenčno okolje uporabljamo platformo Apify, ki od junija 2026 omogoča, da agenti poženejo in plačajo njene programske komponente s stabilnim kriptožetonom USDC na omrežju Base prek protokola x402, in sicer brez uporabniškega računa, vzpostavljenega obračunavanja ali ključa API [15].

Agent v katalogu ponudnika Apify poišče orodje, ki ga potrebuje za izvedbo uporabnikove naloge, in zahtevek za njegov klic prek protokola MCP preda plačilni komponenti, ki je agentu izpostavljena kot strežnik MCP z orodjem, ki sprejme opis zahtevka HTTP (naslov, metoda, telo) in vrne odgovor ponudnika. Komponenta hrani namensko kriptodenarnico s sredstvi, dodeljenimi za pridobivanje plačljivih virov, in plačilno politiko – strojno berljiv zapis omejitev, ki določa dovoljene plačilne sheme, omrežja in kriptožetone, najvišji dovoljeni znesek posameznega plačila ter razpoložljivi proračun naloge. Takšna razmejitev ima dve posledici. Zasebni ključ, s katerim se podpisujejo plačila, ostane zunaj procesa agenta, hkrati pa jezikovni model o izvedbi plačila ne odloča, saj lahko plačilo kvečjemu predlaga s klicem orodja, avtorizirati pa ga ne more. Vrednotenje politike je namreč deterministično postopek nad podpisanimi plačilnimi pogoji in ne nad besedilom, ki ga model prejme od ponudnika ali iz vsebine pridobljenih podatkov. Primer takšne politike prikazuje izpis 1.

```
{
    "task": {
        "id": "task-8f21c4",
        "budgetUsd": "2.50",
        "spentUsd": "0.87"
    },
    "maxAmountPerRequestUsd": "0.25",
    "allowedNetworks": [
        "eip155:8453"
    ],
    "allowedAssets": [
        {
            "network": "eip155:8453",
            "address": "0x833589fCD6eDb6E08f4c7C32D4f71b54bdA02913",
            "symbol": "USDC"
        }
    ],
    "allowedSchemes": [
        "exact",
        "upto"
    ],
    "maxTimeoutSeconds": 600
}
```

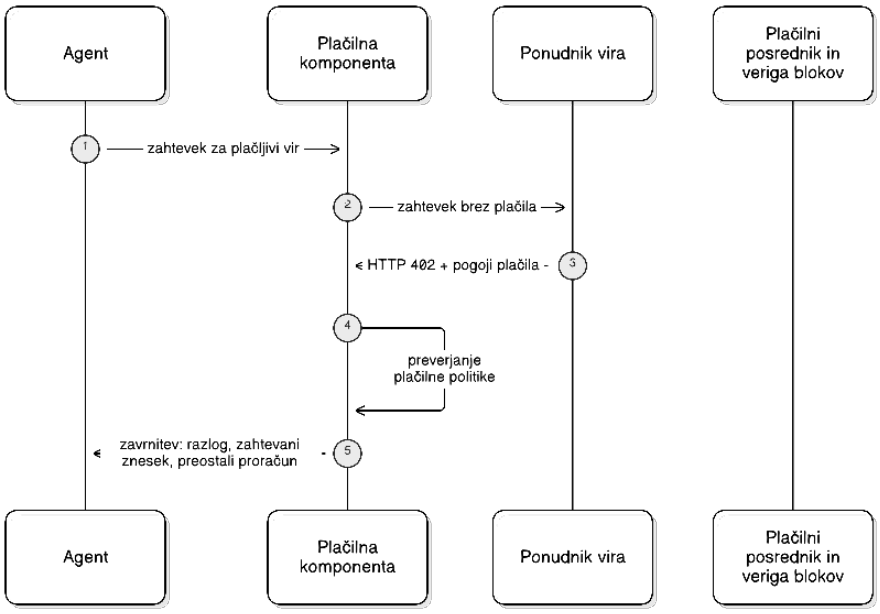
Izpis 1: Primer plačilne politike agenta

Potek. Postopek je enak osnovnemu poteku plačila po protokolu x402, prikazanemu na sliki 1, le da vlogo odjemalca prevzame plačilna komponenta, ki deluje v imenu agenta, vlogo strežnika vira pa ponudnik orodja.

Poteka v šestih korakih:

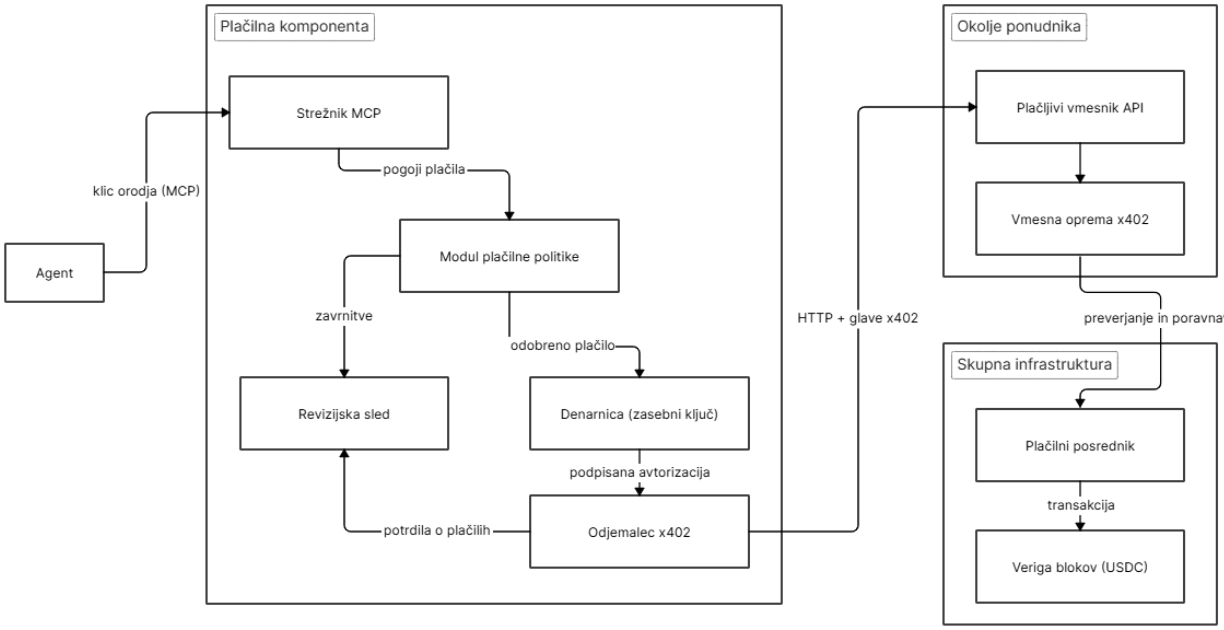
1. agent preda klic plačljive končne točke plačilni komponenti, ta pa zahtevek posreduje ponudniku;
2. ponudnik dostop zavrne z odgovorom HTTP 402 Payment Required in v glavi PAYMENT-REQUIRED posreduje seznam sprejetih plačilnih pogojev;
3. plačilna komponenta vsako ponujeno kombinacijo plačilne sheme, omrežja, kriptožetona in zneska ovrednoti glede na plačilno politiko in razpoložljivi proračun naloge;
4. če je katera od kombinacij odobrena, komponenta v denarnici lokalno pripravi in podpiše plačilno avtorizacijo ter zahtevek ponovi z glavo PAYMENT-SIGNATURE, ki ji doda enolični identifikator plačila;
5. ponudnik prek plačilnega posrednika preveri podpis in skladnost z zahtevanimi pogoji, plačilo poravna na verigi blokov ter vrne zahtevani vir skupaj s potrdilom o poravnavi v glavi PAYMENT-RESPONSE;
6. plačilna komponenta potrdilo zabeleži v revizijsko sled, zmanjša razpoložljivi proračun naloge in odgovor vrne agentu, ki ga uporabi pri oblikovanju odgovora uporabniku.

Zavrnitev plačila. Če plačilna politika nobene od ponujenih kombinacij ne odobri, ker zahtevani znesek presega dovoljeno mejo, ker preostali proračun naloge ne zadošča ali ker ponudnik ne sprejema nobenega dovoljenega omrežja oziroma kriptožetona, plačilna avtorizacija ni podpisana. Komponenta v tem primeru agentu vrne strukturirano napako z razlogom zavrnitve, zahtevanim zneskom in preostalim proračunom, zavrnitev pa zabeleži v isto revizijsko sled kot izvedena plačila. Odločitev o nadaljevanju je nato na strani agenta, ki lahko poskusi z drugim, cenejšim virom, od uporabnika zahteva izrecno odobritev nakupa ali nalogo prekine in razlog vključi v odgovor. Ta postopek je prikazan na sliki 2.



Slika 2: Diagram plačila agenta z zavrnitvijo po plačilni politiki

Slika 3 prikazuje razporeditev komponent rešitve. Iz tega sta razvidni dve lastnosti zasnove. Prvič, zasebni ključ ostane znotraj plačilne komponente in ni dosegljiv procesu, v katerem se izvaja jezikovni model, zato agent plačila ne more podpisati mimo nje. Drugič, plačilna komponenta je edina pot, po kateri zahtevki za plačljive vire zapustijo okolje agenta, zato je plačilna politika uveljavljiva omejitev in ne le priporočilo. Zaradi obojega plačilna komponenta ni zgolj programska knjižnica znotraj agenta, temveč ločena storitev z lastnim varnostnim obsegom, ki jo je mogoče uporabiti tudi za več agentov hkrati pri čemer se revizijska sled zbira na enem mestu.



Slika 3: Arhitektura rešitve za plačevanje dostopa agenta do plačljivih virov po protokolu x402

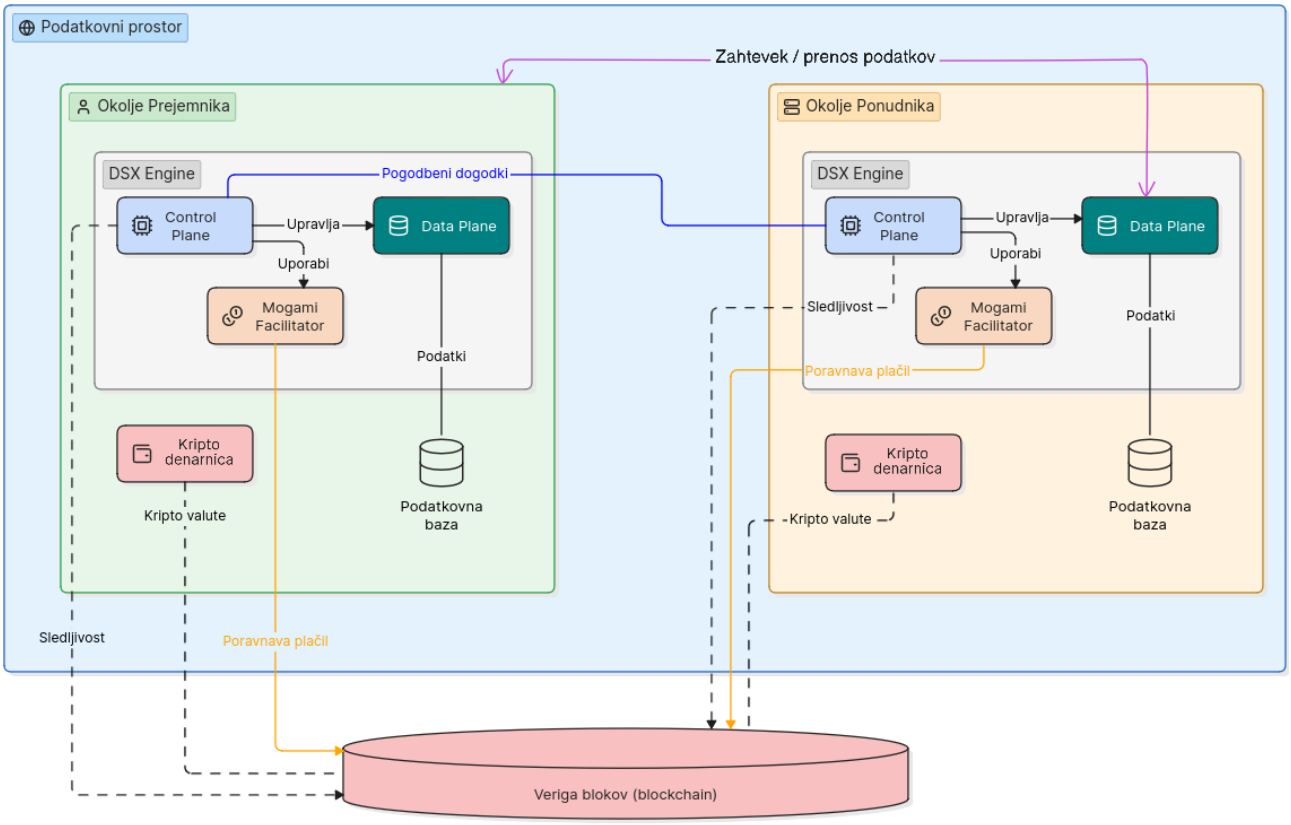
4.2. Podatkovni prostori

Zahteve, ki jih morajo izpolnjevati podatkovni prostori in ki temeljijo na evropski strategiji za podatke ter standardu združenja International Data Spaces Association (IDSA) [16], vključujejo suverenost udeležencev nad lastnimi podatki, preglednost postopkov in sledljivost izvedenih transakcij. Te zahteve se dobro ujemajo z lastnostmi protokola x402. Ker je x402 odprt standard, ki ni vezan na posrednike ali centralizirano infrastrukturo, ponudnikom podatkov omogoča, da sami določajo pogoje monetizacije svojih podatkov, kar je skladno z načelom suverenosti. Ker plačilne zahteve in odgovori potekajo prek standardiziranih glav HTTP (na primer PAYMENT-REQUIRED in PAYMENT-RESPONSE), je postopek plačila razumljiv tako za ponudnika kot za prejemnika. Ker se vsaka izvedena transakcija zabeleži v verigi blokov in jo je mogoče navesti v potrdilu v okviru glave PAYMENT-RESPONSE, x402 podpira tudi sledljivost transakcij v podatkovnem prostoru. Njegova neodvisnost od posamezne panoge, tehnologije ali posrednika ter enostavna integracija v obstoječo infrastrukturo HTTP omogočata, da se lahko x402 vključi v različne implementacije podatkovnih prostorov.

Decentraliziran povezovalnik podatkovnih prostorov DSX Engine [16] podpira monetizacijo podatkov v podatkovnih prostorih z uporabo protokola x402 [17], s čimer omogoča dinamično odkrivanje plačilnih informacij za monetizirane podatke in izvedbo pripadajočih plačil. DSX Engine z decentralizacijskimi mehanizmi in novimi funkcionalnostmi v obliki razširitev nadgrajuje referenčno implementacijo povezovalnika Eclipse Dataspace Components (EDC) podatkovnih prostorov, ki temelji na referenčni arhitekturi in standardih IDSA ter specifikacijah Gaia-X krovnega okvirja zaupanja. Na referenčni implementaciji EDC s svojimi razširitvami temeljijo tudi drugi podatkovni prostori, kot je Catena-X na področju avtomobilske industrije [16].

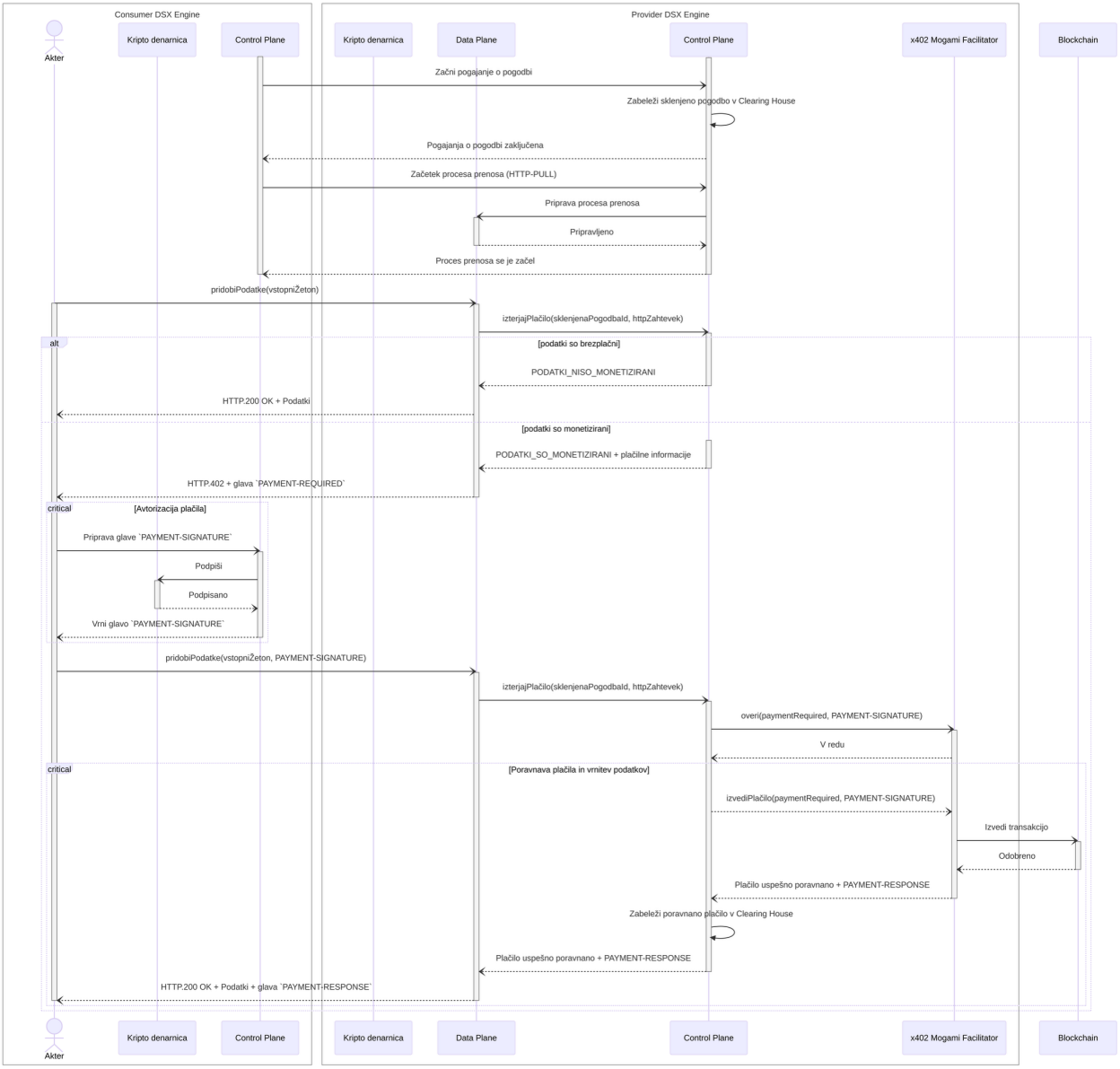
DSX Engine povezovalniki se uporabljajo v podatkovnem prostoru DADS (DIH AGRIFOOD Data Space), ki deluje v realnem okolju na področju kmetijstva. Koncept podatkovnih prostorov in zasnova povezovalnika DSX Engine sta bila podrobneje predstavljena že na lanski konferenci OTS 2025, zato ju v tem prispevku ne obravnavamo ponovno; podrobnosti so na voljo v [16], [17] in [18]. Ključna komponenta, ki omogoča izvedbo plačila x402 v DSX Engine, je Mogami Facilitator, ki je ena od implementacij plačilnih posrednikov x402. DSX Engine vsebuje svoj Mogami Facilitator, kot je razvidno na sliki 4, vendar je možna tudi uporaba zunanjega posrednika Mogami Facilitator. Funkcionalnost pridobivanja monetiziranih podatkov v podatkovnem prostoru omogoča razširitev monetizacije (angl. Monetization Extension), ki je del povezovalnika DSX Engine.

Komponenta, ki prejemniku ob zahtevku po monetiziranih podatkih vrne odgovor po protokolu x402, je ponudnikova komponenta Data Plane.



Slika 4: Komponentni diagram DSX Engine za pridobivanje monetiziranih podatkov

Član podatkovnega prostora, v katerem se uporabljajo DSX Engine povezovalniki, pridobi želene (monetizirane) podatke od ponudnika podatkov tako, da z njim najprej sklene pogodbo (glej sliko 5). Po sklenitvi pogodbe med prejemnikom in ponudnikom podatkov v podatkovnem prostoru prejemnik sproži postopek prenosa tipa HTTP-PULL in do podatkov dostopa tako, da zahtevo pošlje na javni vmesnik HTTP API ponudnikove komponente Data Plane [18]. Data Plane je komponenta na strani ponudnika, ki skrbi za dejanski prenos podatkov in komunikacijo s prejemnikom. Ponudnikov Data Plane prejeto zahtevo najprej overi, pri čemer med drugim preveri, ali so zahtevani podatki monetizirani. To preverjanje delegira Control Plane komponenti, ki v ozadju upravlja pravila, politike in odločitve o tem, ali in pod kakšnimi pogoji se podatki lahko prenesejo, ločeno od komponente, ki prenos dejansko izvede. Control Plane tako na podlagi sklenjene pogodbe dinamično izračuna ceno trenutnega dostopa do podatkov in določi, ali je prenos dovoljen, hkrati pa po potrebi generira objekt s plačilnim zahtevkom ali izvede plačilo s pomočjo lokalnega plačilnega posrednika x402 Mogami, ki vrne potrdilo transakcije, ki bo vključeno v končni odgovor komponente Data Plane prejemniku. Kadar je za dostop do podatkov treba plačati, Data Plane s pomočjo Control Plane komponente vrne odgovor HTTP 402 Payment Required, ki vsebuje informacije o zahtevanem plačilu v skladu s protokolom x402. Če so podatki sicer monetizirani, a je bilo plačilo zanje že izvedeno v preteklosti in za trenutno zahtevo ni več potrebno, Data Plane vrne zahtevane podatke skupaj s potrdilom o predhodno izvedeni transakciji, vključenim v HTTP glavo PAYMENT-RESPONSE. Če podatki niso monetizirani, pa Data Plane preprosto vrne zahtevane podatke, uporaba x402 pa v tem primeru ni potrebna.



Slika 5: Sekvenčni diagram DSX Engine pridobivanja podatkov z uporabo x402

Kot je določeno v protokolu x402, se ob pridobivanju monetiziranih podatkov pojavijo tri vrste glav HTTP, ki so PAYMENT-REQUIRED, PAYMENT-SIGNATURE in PAYMENT-RESPONSE. Ponudnikov DSX Engine Data Plane ob zahtevku po monetiziranih podatkih brez dodane glave HTTP PAYMENT-SIGNATURE kot odgovor vrne HTTP 402 z glavo PAYMENT-REQUIRED, ki jo prikazuje izpis 2 (po pretvorbi iz formata BASE64).

```
{
    "x402Version": 2,
    "error": "PAYMENT-SIGNATURE header is required",
    "resource": {
        "url": "http://127.0.0.1:29291/api/public",
        "description": "Data asset with id 'asset-1'",
        "mimeType": "application/octet-stream"
    },
    "accepts": [
        {
            "scheme": "exact",
            "network": "eip155:84532",
            "amount": "24335",
            "asset": "0x036CbD53842c5426634e7929541eC2318f3dCF7e",
            "payTo": "0x04ec5b689e74a259b6fb6874391156926c4143b7",
            "maxTimeoutSeconds": 600,
            "extra": {
                "name": "USDC","version": "2"
            }
        }
    ]
}
```


Izpis 2: Odgovor HTTP 402 z glavo PAYMENT-REQUIRED


```
{
    "x402Version": 2,
    "resource": {
        "url": "placeholder-url",
        "description": "Data asset with id 'test-asset-5'",
        "mimeType": "application/octet-stream"
    },
    "accepted": {
        "scheme": "exact",
        "network": "eip155:84532",
        "amount": "24335",
        "asset": "0x036CbD53842c5426634e7929541eC2318f3dCF7e",
        "payTo": "0x04ec5b689e74a259b6fb6874391156926c4143b7",
        "maxTimeoutSeconds": 600,
        "extra": {
            "name": "USDC","version": "2"
        }
    },
    "payload": {
        "signature":
"0x5bb7e1910b7229779fe0d77217631d64d33edbf74545319ce2a52db717debd1a1c6281772a62c310c25e117ec23e5e
f5e240b8e3012e6ae1b9665225985d3a5b1c",
        "authorization": {
            "from": "0x25bb103e321bbf0b6ce3c08cd12f92a3292c09a5",
            "to": "0x04ec5b689e74a259b6fb6874391156926c4143b7",
            "value": "24335",
            "validAfter": "1785327538",
            "validBefore": "1785328138",
            "nonce": "0xfd362eae021141a56bde660036644d1b26c6d76610ed99c280351f9a2bb87c2b"
        }
    },
    "extensions": {}
}
```


Izpis 3: Glava PAYMENT-SIGNATURE, ki jo prejemnik pošlje ponudniku

Izpis 3 prikazuje primer glave HTTP PAYMENT-SIGNATURE, ki jo prejemnik pošlje ponudniku (po pretvorbi iz formata BASE64):

Tehnične značilnosti implementacije podpore monetizaciji podatkov v povezovalniku DSX Engine povzema tabela 3. Pri implementaciji podpore monetizaciji podatkov v DSX Engine smo najprej želeli problem zahtevanja in izvedbe plačila rešiti z uporabo lastnih pametnih pogodb, ki bi znotraj podatkovnega prostora urejale preverjanje in izvedbo plačil. Izkazalo se je, da implementacija takšnih pametnih pogodb zahteva veliko časa in znanja ter prinaša tveganja zaradi nepravilnosti, ki bi se odkrile pozneje med uporabo, saj pametnih pogodb ni mogoče spreminjati po objavi. Uvesti bi bilo treba tudi mehanizem posodabljanja pametnih pogodb, za preverjanje in izvajanje monetizacije ter plačil, kar je kompleksno. Protokol x402 nam je omogočil enake funkcionalnosti na standardiziran in odprt način, kjer je še vedno zagotovljena sledljivost, preglednost in suverenost. Prav tako so komponente, kot so plačilni posredniki x402 že razvite in na voljo za uporabo, zaradi česar smo se lahko popolnoma posvetili integraciji x402 v DSX Engine namesto lastni celoviti implementaciji mehanizma plačil z lastnimi pametnimi pogodbami. Skladnost s standardom x402 nam je prav tako omogočila morebitni razvoj pridobivanja podatkov z DSX Engine v podatkovnih prostorih z uporabo agentov umetne inteligence, pri katerem bi agenti našli želene podatke v podatkovnem prostoru in jih samodejno pridobili oziroma kupili.

Tabela 3: Tehnične značilnosti implementacije podpore monetizaciji podatkov v DSX Engine

Podprta različica x402	x402 v2
Podprta omrežja oz. verige blokov	Vse verige blokov, ki jih podpira uporabljeni plačilni posrednik x402. Mogami Facilitator podpira verigi blokov Base in Base Sepolia. Član podatkovnega prostora mora imeti kriptodenarnico v verigah blokov, v katerih želi poslovati.
Podprti kriptožetoni	Vsi ERC-20 žetoni in potencialno tudi drugi žetoni, kot so žetoni v verigi blokov Solana, če jih podpira tudi uporabljeni plačilni posrednik x402. Mogami Facilitator podpira vse žetone ERC-20 (na podprtih verigah blokov).
Podprti plačilni posredniki x402	Trenutno je podprt le Mogami Facilitator, vendar je mogoče implementirati razširitev za podporo tudi drugim plačilnim posrednikom [17].
Način namestitve	Podpora monetizaciji podatkov s protokolom x402 je del decentraliziranega povezovalnika DSX Engine podatkovnih prostorov. Namestitev je enaka osnovni namestitvi DSX Engine povezovalnika [16], le da mora član podatkovnega prostora imeti še kriptodenarnico v verigi blokov, v kateri želi prejemati oz. pošiljati kriptožetone, ki jih mora tudi pridobiti [17] [18].
Programski jezik in ogrodje	DSX Engine temelji na EDC. Implementiran je v programskem jeziku Java [18].

5 Zaključek

Protokol x402 ponuja odgovor na ovire, ki so bile identificirane pri obstoječih pristopih k zaračunavanju dostopa do podatkov in storitev: strukturno odvisnost od centraliziranih posrednikov, nesorazmerno visoke stroške tradicionalne plačilne infrastrukture pri mikrotransakcijah in plačilne poti, zasnovane okoli človeka kot plačnika. Glavna prednost protokola je vključitev plačilne logike neposredno v cikel zahtevka in odgovora HTTP, kar odpravlja potrebo po registraciji, naročnini ali ključu API in omogoča dostop tako ljudem kot avtonomnim agentom umetne inteligence. Enako pomembna je odprtost protokola, saj ni vezan na enega ponudnika, omrežje ali kriptovaluto, kar udeležencem omogoča prosto izbiro infrastrukture, plačilne sheme in plačilnega posrednika.

Različica v2 to prednost dodatno krepi s standardiziranimi identifikatorji CAIP-2, modularno arhitekturo in formaliziranim sistemom razširitev [7] [10] [11].

Kljub temu protokol ostaja v razmeroma zgodnji fazi zrelosti. Zanašanje na tehnologijo verige blokov pomeni, da so uporabniška izkušnja, regulativna obravnava in stabilnost vrednosti plačilnih sredstev odvisne od širšega kripto ekosistema, kar za del ponudnikov in uporabnikov predstavlja vstopno oviro. Vloga plačilnega posrednika uvaja novo točko trenja, saj mora ponudnik podatkov izbrani komponenti zaupati preverjanje in poravnavo plačil ali pa jo izvajati sam [7].

Primerjava s protokolom MPP pokaže dva sorodna, a različna pristopa k istemu problemu. Izbira med x402 in MPP je predvsem odvisna od infrastrukture in ciljne skupine uporabnikov. Za razvijalce, ki vstopajo v prostor strojnih (agentskih) plačil, je pragmatičen pristop naslednji: če gre za preprosto, visokofrekvenčno storitev z odjemalci, ki že poznajo kriptosredstva, je smiselno začeti z x402 zaradi hitre integracije, zrelega nabora SDK-jev in nizkih stroškov; če ciljate na podjetniške uporabnike, fizično blago ali potrebujete skladnost s predpisi in fiat plačila, izberite MPP. Kadar viri to dopuščajo, pa je dolgoročno najbolj smiselna podpora obema protokoloma hkrati, saj tako storitev ostane dostopna najširšemu krogu odjemalcev ne glede na to, kateri standard bo v prihodnje prevladal.

Literatura

[1] R. T. Fielding, "Architectural styles and the design of network-based software architectures", Ph.D. dissertation, Dept. Inf. and Comput. Sci., Univ. of California, Irvine, CA, USA, 2000. Dostopno: https://ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

[2] R. T. Fielding, M. Nottingham in J. Reschke, "RFC 9110: HTTP Semantics", junij 2022. Dostopano: 29. junij 2026. [Na spletu]. Dostopno: https://www.rfc-editor.org/rfc/rfc9110.html

[3] R. Fresno-Aranda, P. Fernandez, A. Gamez-Diaz, A. Duran, in A. Ruiz-Cortes, "Pricing4APIs: A rigorous model for RESTful API pricings", Computer Standards & Interfaces, let. 91, str. 103878, jan. 2025, doi: 10.1016/j.csi.2024.103878.

[4] Q. Ramadan, Z. Boukhers, M. AlShaikh, C. Lange, in J. Jürjens, "Data Trading and Monetization: Challenges and Open Research Directions", 17. januar 2024, arXiv: arXiv:2401.09199. doi: 10.48550/arXiv.2401.09199.

[5] A. Odlyzko, "The Case Against Micropayments", v Financial Cryptography, let. 2742, R. N. Wright, Ur., v Lecture Notes in Computer Science, vol. 2742, Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, str. 77–83. doi: 10.1007/978-3-540-45126-6_6.

[6] European Central Bank. A big future for small payments?: Micropayments and their impact on the payment ecosystem. LU: Publications Office, 2023. doi: 10.2866/07262.

[7] x402 Foundation, "x402 documentation", x402. Dostopano: 6. julij 2026. [Na spletu]. Dostopno: https://docs.x402.org

[8] M. Zachariadis in P. Ozcan, "The API Economy and Digital Transformation in Financial Services: The Case of Open Banking", SSRN Journal, 2016, doi: 10.2139/ssrn.2975199.

[9] T. Lammer, D. Rees, T. Rice, in T. Shirakami, "Enhancing cross-border payments: state of play and way forward", Bank for International Settlements, BIS Bulletins 119, dec. 2025. Dostopno: https://www.bis.org/publ/bisbull119.pdf

[10] Coinbase, "x402 documentation", *Coinbase*. Dostopano: 6. julij 2026. [Na spletu]. Dostopno: https://docs.cdp.coinbase.com/x402

[11] x402 Foundation, "x402", GitHub repozitorij. Dostopano: 6. julij 2026. [Na spletu]. Dostopno: https://github.com/x402-foundation/x402

[12] Tiger Research, "x402: Coinbase and the Beginning of the AI Agent Era", CoinGecko. Dostopano: 3. avgust 2026. [Na spletu]. Dostopno: https://www.coingecko.com/learn/x402-autonomous-ai-agent-payment-coinbase

[13] "Machine Payments Protocol", MPP. Dostopano: 8. julij 2026. [Na spletu]. Dostopno: https://mpp.dev/overview

[14] "Machine Payments Protocol Specifications," PaymentAuth.org. Dostopano: 8. julij 2026. [Na spletu]. Dostopno: https://paymentauth.org

[15] Apify, "Agentic payments with x402 protocol", Apify Documentation. Dostopano: 3. avgust 2026. [Na spletu]. Dostopno: https://docs.apify.com/platform/integrations/x402

[16] M. Ferenec, T. Lah, in M. Turkanović, "DSX Engine: decentraliziran povezovalnik podatkovnih prostorov", v OTS 2025: sodobne informacijske tehnologije in storitve: zbornik 28. konference, L. Pavlič, T. Beranič, in M. Heričko, Ur., 1. izd. Maribor: Univerza v Mariboru, Univerzitetna založba, 2025, str. 109–130. doi: 10.18690/um.feri.7.2025.11.

[17] M. Ferenec, T. Lah, M. Bunderla, D. Copot, in M. Turkanović, "Podpora ekonomiji podatkov v podatkovnih prostorih: implementacija monetizacije z DSX Engine", predstavljeno na 33. konferenci Dnevi slovenske informatike 2026: Digitalna družba na razpotju: ko se srečata regulacija in inovacija, Portorož, Slovenija, 15. april 2026.

[18] M. Ferenec in M. Turkanović, "DSX Engine CE", GitHub repozitorij, 2025. [Na spletu]. Dostopno: https://github.com/blockchain-lab-um/DSX-Engine-CE