A TUDOMÁNY, AZ OKTATÁS ÉS A KÖZGYŰJTEMÉNYI KISZOLGÁLÁS ÚJ INFORMATIKAI SZINERGIÁI

NETWORKSHOP 2026
35. Országos Informatikai Konferencia

2026. március 31–április 2.
Debreceni Egyetem, Debrecen

Szerkesztette: Tick József, Kokas Károly, Holl András

HUNGARNET Egyesület
Budapest, 2026

HUN-REN
Magyar Kutatási Hálózat

NETWORKSHOP

Szerkesztette: Tick József, Kokas Károly, Holl András

Tipográfia és tördelés: Vas Viktória
Korrektúra: Danyi Melinda
Angol nyelvi lektor: Lukács Katalin

Networkshop 2026 konferencia előadásainak közleményei
Debreceni Egyetem, Debrecen
2026. március 31–április 2.

ISBN 978-615-6792-29-7
DOI: https://doi.org/10.31915/NWS.2026

Kiadja a HUNGARNET Egyesület
az MTA Könyvtár és Információs Központ közreműködésével
Budapest
2026

Borítókép: freepik.com

DOI: 10.31915/NWS.2026.8

VibeARP: ADATGAZDÁSZ ASSZISZTENS AZ ARP-BAN

Pataki Balázs
HUN-REN Számítástechnikai és Automatizálási Kutatóintézet
pataki@sztaki.hu

Absztrakt

A HUN-REN Adatrepozitórium Platform (ARP) használata egyszerre jelent kihívást a kezdő és a tapasztalt felhasználók számára: előbbieket a rendszer komplexitása, utóbbiakat az ismétlődő, időigényes adatkezelési feladatok terhelik. A VibeARP egy mesterséges intelligenciára épülő adatgazdász asszisztens, amely ezt a kettős problémát célozza meg azzal, hogy automatizálja a rutin feladatokat, miközben jelentősen lerövidíti a tanulási görbét az új felhasználók számára.

Bemutatjuk a VibeARP "klasszikus" megvalósítását, ami saját ágens ciklussal, eszközökkel és felhasználói felülettel készült, valamit a VibeARP "CAOS" (Coding Agent as Operating System) verzióját, ami a felhasználók gépére telepített meglévő kódoló ágensek, mint Codex vagy Claude Code teszi lehetővé ugyanazt a működést, de lokális módon.

Kulcsszavak: HUN-REN Adatrepozitórium Platform, ARP, Kutatástámogatás, Adatgazdászat, MI-ágens, Szoftver automatizáció

Abstract

Using the HUN-REN Data Repository Platform (ARP) poses challenges for both novice and experienced users: newcomers are burdened by the system's complexity, while advanced users are hindered by repetitive, time-consuming data management tasks. VibeARP is an AI-based data steward assistant that addresses this dual problem by automating routine tasks while significantly shortening the learning curve for new users.

We present the "classic" implementation of VibeARP, built with its own agent loop, tools, and user interface, as well as the VibeARP CAOS (Coding Agent as Operating System) version, which enables the same functionality locally by using existing coding agents installed on users' machines, such as Codex or Claude Code.

Keywords: HUN-REN Data Repository Platform, ARP, Research Support, Data Stewardship, AI Agent, Software Automation

Bevezetés

A HUN-REN Adatrepozitórium Platform (ARP) [1] egységes kutatási adatkezelési infrastruktúrát biztosít a Magyar Kutatási Hálózat (HUN-REN) kutatói számára. A hálózat sokszínűsége – 42 kutatóintézet, több mint száz egyetemi kutatócsoport és a tudományterületek teljes spektruma – összetett, több komponensből álló rendszer kialakítását tette szükségessé. Az ARP központi eleme egy Dataverse-alapú adatrepozitórium, amelyet CEDAR-alapú metaadatséma-regiszter, az RO-Crate [2] szabvány teljes körű támogatása, az AROMA metaadatszerkesztő eszköz, valamint egy országos tudásgráfra épülő egységes keresőrendszer egészít ki.

Bár ezek az eszközök - Dataverse, CEDAR, AROMA [3] - önmagukban jól használhatók, az ökoszisztéma egésze elsőre nehezen áttekinthető lehet. Emellett a tapasztalt felhasználóknak is jelentős manuális munkát igényel a megfelelő metaadatok összegyűjtése különféle forrásokból, például fájlokból, dokumentációból, jegyzetekből vagy webes tartalmakból.

VibeARP

A VibeARP [4] az ARP automatizált használatára kínál megoldást egy chat-alapú felhasználói felületen keresztül működő MI-ágens segítségével. A felhasználók természetes nyelven írhatják le a létrehozni kívánt adatkészletet, miközben az ágens különféle eszközöket használ: webes keresést végez, releváns oldalakról információt gyűjt, fájlokból metaadatokat nyer ki, majd ezeket a CEDAR-alapú sémákhoz illesztve automatikusan RO-Crate metaadatstruktúrává alakítja. A metaadatok folyamatosan megjelennek egy beágyazott, leegyszerűsített AROMA-szerkesztőben is, amely a manuális finomhangolást is lehetővé teszi.

A rendszer egy integrált böngészőágenst is tartalmaz, amely interaktív webes műveletek végrehajtására képes egy MI által vezérelt, tényleges Chrome böngésző használatával. Ennek segítségével például a felhasználó bejelentkezhet a Dataverse-be, majd a bejelentkezés után a böngészőágens hozzáférhet ahhoz az azonosítási tokenhez, amelynek segítségével az elkészült RO-Crate csomag feltölthető a Dataverse-be, létrehozva a kívánt adatcsomagot.

A VibeARP általános feladata, hogy az adatgazdásszal beszélgetve, az ő instrukciói alapján összegyűjtse a szükséges metaadatokat (direkt inputból, webes kereséssel, vagy fájlokból kinyerve), majd azokat az adott RO-Crate-hez megadott metaadat sémákhoz és profilokhoz illessze, végül ezek alapján a végleges RO-Crate JSON-t előállítsa.

A konkrét lépések, amiket az ágens végez, egy előre definiált munkafolyamatot alkotnak:

1. **Profil felfedezés.** A munkamenet első lépése az aktív profilok és a profil megkötések megismerése. Ez a lépés biztosítja, hogy az ágens ne spekuláljon a metaadatstruktúráról, hanem a tényleges profil megkötések alapján dolgozzon.

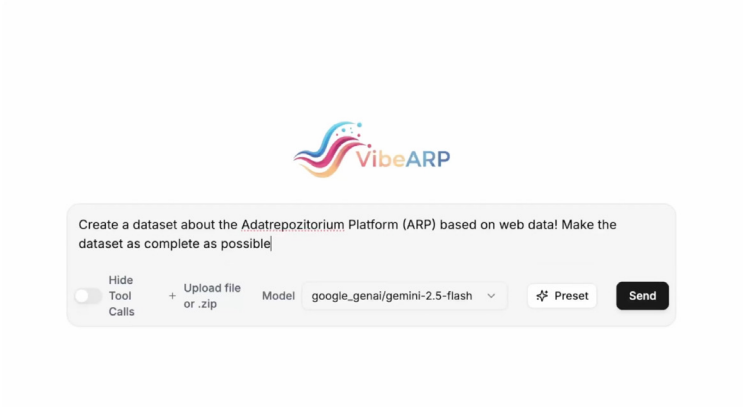
2. **Mezőtervezés.** A profil szabályok alapján az ágens osztályozza a mezőket: kötelező, ajánlott, opcionális, és nem engedélyezett mezőkre. Létrehoz egy explicit tervet, amely meghatározza, milyen értékek hiányoznak még, és milyen sorrendben lesznek kitöltve. A profil illesztési szabályzat előírja, hogy kötelező mezőket először, ajánlott opcionális mezőket másodszor, majd a többi engedélyezett mezőt utoljára kell kezelni.
3. **Adatgyűjtés.** Ha metaadat értékek ismeretlenek és nem vezethetők le lokálisan, az ágens bekérheti az adatokat az adatgazdásztól, vagy webes keresést végezhet, szükség esetén a talált oldalakat is letöltve és elemezve. Az ágens az elsődleges forrásokat részesíti előnyben, mint hivatalos intézményi oldalak, kanonikus dokumentáció, szabványok, stb.
4. **Entitás létrehozása**. Miután minden az aktuális profilnak megfelelő adatot összegyűjtött az ágens, ezekből létrehozza a profilnak megfelelő mezőadatokat és RO-Crate entitásokat.
5. **RO-Crate JSON előállítás.** Az LLM-ek önmagukban is képesek helyes RO-Crate JSON-t létrehozni, ha az adatok már rendelkezésre állnak, azonban ez meglehetősen token-igényes művelet, így ez jelentősen meg tudja drágítani az ágens működését. Ehelyett speciális RO-Crate szerkesztő eszközt adunk az ágens kezébe, amivel részletekben tudja a JSON-t szerkeszteni: hozzáadni, módosítani, törölni. Ezek a módosítási műveleteket egy speciális eszköz (tool) segítségével kerülnek végrehajtásra, és ezzel épül és változik az RO-Crate JSON.
6. **Ellenőrzés.** Mivel az LLM-ek működése nem determinisztikus, akármilyen pontosan adjuk is meg a szabályokat, az elvégzett műveletek után folyamatos ellenőrzésre és visszajelzésre van szükség. A validálás az RO-Crate profilok és az RO-Crate alapját képező JSON-LD szabványnak való megfelelés alapján történik. Ha a validálás során hibát tapasztalunk, arról tudatjuk az ágenst, amely azt egy következő körben képes javítani.

A VibeARP implementációjához a LangGraph [5] ágens keretrendszert használtuk. Ebben valósítottuk meg a saját ágens ciklusunkat, valamint a működéshez szükséges eszközöket (toolokat).

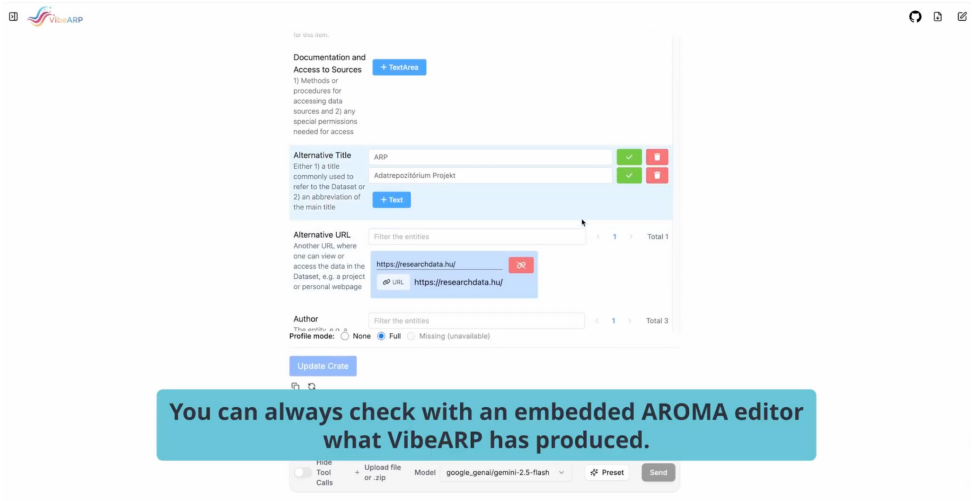
A felhasználói felületet a LangChain Agent chat UI" [6] megoldására alapoztuk. Ez egy React-alapú chatalkalmazás, amely saját komponensekkel bővíthető és a nyílt „Agent Protocol"-on [7] keresztül képes kommunikálni a LangGraphban futó ágenssel.

A VibeARP böngésző-automatizációs megoldása a „VibeARP Browser AI". Ez egy Browser Use [8] -alapú ágens, amihez egy olyan webes megoldást készítettünk, amely a browser-use által irányított Chrome böngészőt integráltan jeleníti meg a chatfelületen. Az eredmény egy böngészőben futó, interaktív böngészőélmény. Ezt jellemzően a „VibeARP Browser AI"

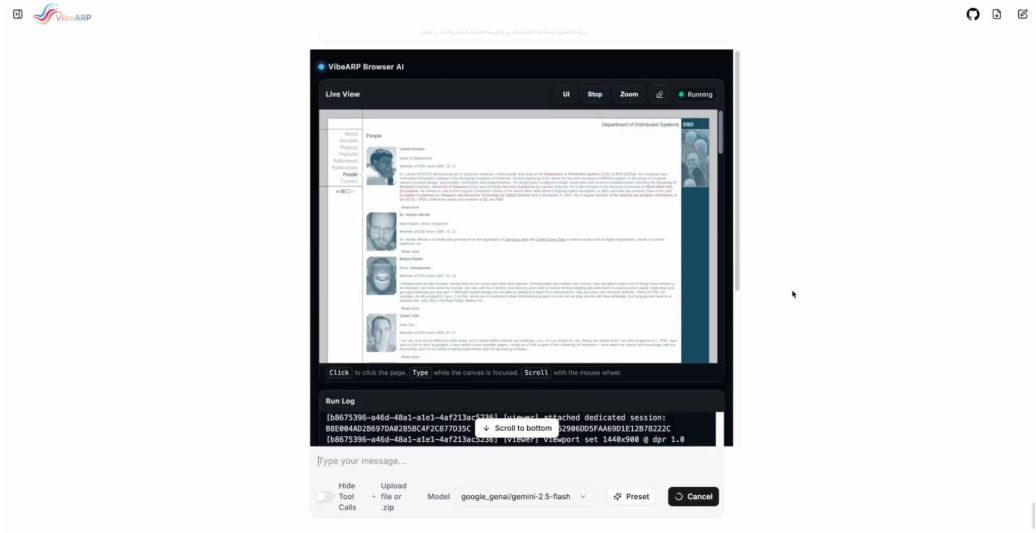
irányítja, szükség esetén azonban az irányítás átadható a felhasználónak is, például jelszó megadásához.



1. ábra: A VIbeARP chat felülete



2. ábra: A VIbeARP-ba integrált AROMA RO-Crate szerkesztő az RO-Crate metaadatok megjelenítésére és szerkesztésére



3. ábra: A VIbeARP Browser AI-ágens egy beépülő böngésző, amin az ágens interaktív weboldalakkal tud feladatokat elvégezni, pl. szerzői adatokat kigyűjteni weboldalból

VibeARP CAOS

A VibeARP CAOS VibeARP koncepció továbbfejlesztése, amely alapvetően megváltoztatja az implementációs megközelítést: ahelyett, hogy saját MI-ágens ciklust építenénk, a szabványos Model Context Protocolra (MCP) és a kereskedelmi forgalomban kapható kódoló ágensekre támaszkodunk. A névben a CAOS is erre utal: *Coding Agent as Operating System*. Az elnevezést az indokolja, hogy az AI kódoló ágensek, mint Codex, Calude code, Gemini, Kilo stb., valójában már nem csak kódolási feladatokra alkalmasak, hanem általános ágens keretrendszerként is értelmezhetők. Ebből a szempontból egy operációs rendszerre hasonlítanak, ami az alap intelligencia működést biztosítja (LLM-mel való kommunikáció, alap shell-alapú eszközhasználat stb.), és amelyekre különböző módokon saját applikációkat telepíthetünk.

A kódoló ágenseket az alábbi módokon bővíthetjük, vagy szabhatjuk testre:

1. **Prompt-alapú bővítés (CLAUDE.md, AGENTS.md)**

   A legegyszerűbb bővítési forma a projektkönyvtárban elhelyezett Markdown-fájlok, amelyeket a kódoló ágens automatikusan beolvas és a kontextusába injektál. Claude Code a CLAUDE.md, az OpenAI Codex CLI az AGENTS.md stb. A tartalom tetszőleges szöveg lehet: kódolási konvenciók, architekturális döntések, projektspecifikus iránymutatások. Például egy CLAUDE.md tartalmazhatja, hogy „Mindig TypeScriptet használj, sosem JavaScriptet”, vagy „Az API-végpontok a /api/v2/ prefixszel kezdődnek”.

Előnyök: csak egy szövegfájl, a legegyszerűbb módja annak, hogy befolyásoljuk az ágens viselkedését.

Hátrányok: nem ad hozzá új képességeket az ágenshez, csak azt befolyásolja, hogyan viselkedjen, nem azt, mit tudjon. A metaadat-szerkesztés kontextusában például egy CLAUDE.md megmondhatja az ágensnek, hogy „ne szerkeszd közvetlenül az ro-crate-metadata.json fájlt", de nem ad neki eszközt arra, hogy validálja a csomagot vagy feltöltse a Dataverse-be. Emellett fogyasztja a kontextusablakot, és a modell figyelmen kívül is hagyhatja az utasításokat.

2. **Skillek (Skills / Slash Commands)**

   A skill egy újrafelhasználható, feladat-specifikus ágenscsomag, amely instrukciókat, munkafolyamatot, kontextust, referenciákat és opcionálisan futtatható segédprogramokat tartalmaz. Az instrukciókat egyszerű szöveges Markdown-fájlokként lehet megadni, és mellé megadott módon lehet csatolni azokat a segédprogramokat, amiket a skill instrukciói alapján használatba tud venni az ágens. Például, ha van egy képzeletbeli extract_metadata_from_pdf programunk, és ennek működését leírjuk egy skillben, onnantól, ha a felhasználó olyan feladatot akar végezni, amihez egy PDF-ből kell metaadatokat kinyerni, akkor az ágens tudni fogja, hogy ezt hogyan kell csinálnia, és miképpen kell paramétereznie és futtatnia az extract_metadata_from_pdf programot.

   Előnyök: az ágens működése az AGENT.md-nél sokkal jobban testreszabható, különösen a csatolt programok felhasználásával

   Hátrányok: bonyolultabb karbantartani, mert külön skill-struktúrát, leírást, verziózást és esetenként futtatható segédprogramot igényel

3. **Model Context Protocol (MCP)**

   A Model Context Protocol [9] egy 2024 vége felé az Anthropic által publikált nyílt szabvány, amely JSON-RPC 2.0-ra épül. Kliens-szerver architektúrát definiál, ahol egy MCP gazdaalkalmazás (host), például Claude Code, Codex stb. MCP szerverekhez csatlakozik. A szerverek három dolgot nyújthatnak:

   Eszközök (Tools) – hívható függvények, amelyeket a modell meghívhat. Például: „validáld ezt az RO-Crate-et", „keress rá erre a kifejezésre a weben".

   Erőforrások (Resources) – olvasható adatforrások, amelyeket a modell lekérhet. Például: „add vissza a jelenlegi séma-regiszter tartalmát".

   Promptok (Prompts) – újrafelhasználható prompt-sablonok, amelyeket a szerver kínál.

   Előnyök: nyílt szabvány, iparági szintű hordozhatóság. Strukturált, típusos eszköz-interfész (JSON Schema-alapú bemeneti validáció). Új, végrehajtható képességeket

ad az ágensnek. Minimális kontextusköltség (csak igény szerint, eszközhíváskor). A szerver állapotot tarthat fenn (profil-kontextus cache, séma-regiszter).

Hátrányok: szerverfolyamat futtatását igényli (operációs terhelés). A gazdaalkalmazásnak támogatnia kell az MCP-t.

Bár implementáció szempontjából az MCP igényli a legtöbb munkát, a VibeARP CAOS-hoz mégis ezt a megoldást alkalmaztuk, mert ez tűnik végső soron a legkompatibilisebbnek és legflexibilisebbnek. A VibeARP MCP a már előzőleg ismertetett munkafolyamatot megvalósító eszközöket implementálja és teszi az MCP protokollon keresztül elérhetővé a kódoló ágenseknek.

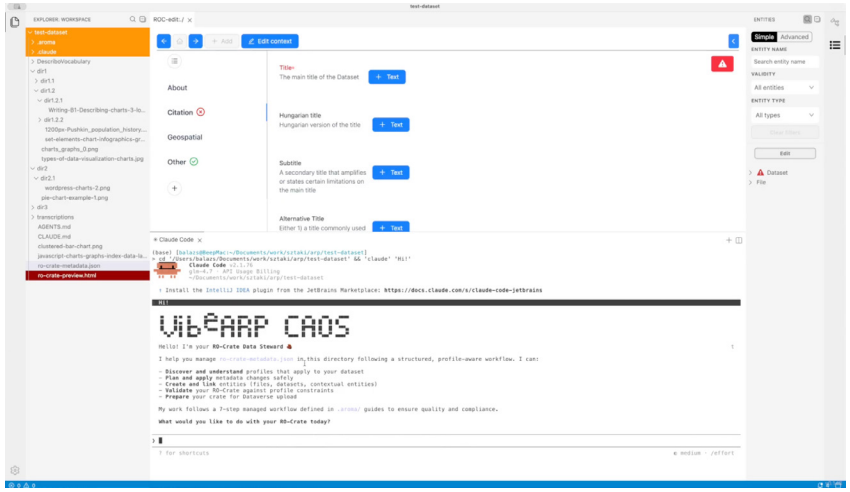
A VibeARP CAOS egy nagyobb projekt, az AROMA-2 keretében került implementálásra. Az AROMA első változata az ARP-ben a Dataverse-ben lévő adatcsomagok RO-Crate szerinti szerkesztését teszi lehetővé, szorosan integrálódva a Dataverse-szel.

Az AROMA-2 célja egy olyan RO-Crate szerkesztői "IDE" létrehozása, amivel a kutatók a saját fájlrendszerükben vagy egyéb forráshelyeken elérhető fájljaikat tudják az RO-Crate szerint metaadatolni, majd a végén a kész RO-Crate csomagot a kiválasztott repozitóriumba feltölteni. Ehhez egy Theia keretrendszeren alapuló új AROMA-megoldást hoztunk létre és a VibeARP CAOS ennek az egyik eleme, egyben ez adja azt az RO-Crate grafikus felületet, amit a klasszikus VibeARP-ban az AROMA beépülő verziója adott. Vagyis a VibeARP CAOS által végzett változtatásokat az AROMA-2-ben követhetik nyomon a felhasználók, és magát a VibeARP CAOS-t is onnan tudják indítani.

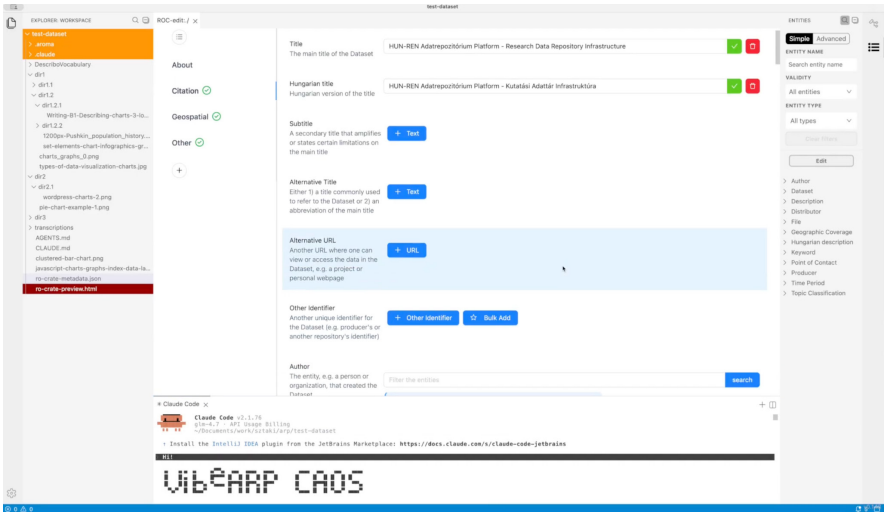
A VibeARP CAOS ágensek kétféleképpen futtathatók: vagy terminálban, a megszokott CLI eszközzel vagy beépülve az AROMA-2-be. Az első megoldás ugyanazt a felhasználói élményt nyújtja, mintha a felhasználó maga indítaná el terminálból a Claude-ot vagy Codexet: ugyanazt a terminál UI-t használhatja, amit már megszokott, és amit esetleg már testre is szabott. A AROMA-2-be beépülő megoldásnál az AROMA-2 adja a chat felületet, de a háttérben ugyanúgy a kiválasztott ágenssel (jelenleg Claude vagy Codex) kommunikál.

A *CAOS*-módszer alkalmazása azért tűnik előremutatónak, mert jó pár problémát megold a VibeARP első megoldásához képest:

- nem kell az ágens ciklussal és az alacsony szintű LLM-kommunikációval és eszköz használattal nekünk foglalkozni,
- a kódoló ágensek önmaguktól is sok hasznos képességgel rendelkeznek, amiket nem nekünk kell megvalósítani,
- rendelkeznek olyan beépített korlátokkal, amiket a VibeARPban még meg sem valósítottunk,
- a felhasználó a saját előfizetésével tudja használni, így a költségeket is ő állja, nem pedig a VibeARP szolgáltatásnak kell központilag finanszíroznia.



4. ábra: A Claude Code, mint a VibeARP CAOS megvalósítása az AROMA-2-ben egy terminál ablakban nyílik meg az RO-Crate szerkesztőfelület alatt



5. ábra: A VibeARP CAOS eredményét az RO-Crate szerkesztőfelületén lehet megnézni és szerkeszteni

Összefoglalás

A VibeARP és VibeARP CAOS eszközök jelenleg kísérleti-fejlesztési fázisban vannak és azon dolgozunk adatgazdászokkal együttműködve, hogy olyan eszköz készüljön, ami az ARP-ben hatékonyabbá teheti az adatgazdászi munkát, levéve a repetitív és unalmas munkák terhét róluk. A következő lépésként elkezdjük az eszközt adatgazdászok közreműködésével tesztelni és a visszajelzéseik alapján elkészíteni a VibeARP végleges változatát, amit elérhetővé teszünk az APR felhasználói számára.

Bibliográfia

Kovács, L. Az ELKH Adatrepozitórium Platform projekt a kutatási adattárolás és megosztás országos föderált rendszere 2023 MAGYAR TUDOMÁNY, 184(7)., pp. 839-851. ISSN 0025-0325

Research Object Crate. Research Object Crate (RO-Crate). https://www.researchobject.org/ro-crate/

Pataki, B. Dataverse, CEDAR and RO-Crate: The building blocks of ARP. YouTube videó. https://www.youtube.com/watch?v=0_ENdITtIQg

Pataki, B. VibeARP: An experimental AI agent for the Adatrepozitórium Platform. YouTube videó. https://www.youtube.com/watch?v=ZMvsET-5S5Y

LangChain. LangGraph: Agent Orchestration Framework for Reliable AI Agents. https://www.langchain.com/langgraph

LangChain AI. agent-chat-ui: Web app for interacting with any LangGraph agent via a chat interface. GitHub-repozitórium.: https://github.com/langchain-ai/agent-chat-ui

LangChain AI. agent-protocol. GitHub-repozitórium. https://github.com/langchain-ai/agent-protocol

Browser Use. browser-use: Make websites accessible for AI agents. GitHub-repozitórium. https://github.com/browser-use/browser-use

Model Context Protocol. https://modelcontextprotocol.io/