Moregraph: Metadata-Driven Enterprise Architecture Using Conceptual Structures

POLOVINA, Simon <http://orcid.org/0000-0003-2961-6207>, FALLON, Richard <http://orcid.org/0000-0001-5747-8839> and SALEEM, Mahpara <http://orcid.org/0009-0003-1671-4796>

Moregraph: Metadata-driven Enterprise Architecture using Conceptual Structures

Simon Polovina[1,2][0000−0003−2961−6207], Richard Fallon[3][0000−0001−5747−8839], and Mahpara Saleem[4][0009−0003−1671−4796]

[1] Department of Computing and Digital Technologies, Sheffield Hallam University, Sheffield, UK
[2] Simile Logics Limited, Sheffield, UK
S.Polovina@shu.ac.uk, simon@similelogics.ltd
[3] Red Rooster IT, Vienna, Austria
richard@fallon.at
[4] Mahaz Tech Enterprise Limited (MTE), Manchester, UK
mahpara@mahaztechenterprise.com

Abstract. Enterprise Architecture (EA) enables organisations to articulate their business purpose, from strategy to execution, without incurring technical debt. Data infrastructures underpin their EA, enabling organisations to derive their purpose from their data landscape. Metadata, a crucial aspect of this infrastructure, is a human construct that gives meaning to the data and shapes the organisation's EA. The data objects referenced by metadata, however, are usually encapsulated inside applications. The Moregraph application visualises the data objects and their relations, unrestricted by application boundaries, thus allowing the data objects to express their semantics directly at a more precise level of granularity. Conceptual Graphs are used to maintain the requirements for Moregraph's conceptual structure, which is then implemented using SAP Graph to access the data objects in SAP Business Accelerator Hub. As these data objects are clean core, they can be extended or compared by organisations to address their technical debt at the same level of granularity, with further opportunities for application and Application Programming Interface (API) rationalisation. From these experiences alongside SAP as the representative case study, Moregraph advances metadata-driven enterprise architecture and informs the direction of future work.

Keywords: Enterprise Architecture · Technical Debt · Metadata · Data Object · Clean Core · Conceptual Graphs · Business Data Graphs · SAP Graph

1 Introduction

Enterprise Architecture (EA) recognises that enterprises arise from and are sustained by human endeavours that recognise market or societal gaps. To meet the challenges of these gaps, enterprises become organisations with defined business

roles and processes. EA functions enable organisations to articulate their humanly defined business purpose and align their strategy, structure, and systems to fulfil that purpose, ensuring that all these assets optimally contribute [9]. These assets include the IT (Information Technology) landscape, notorious for its technical debt. This debt is the present-day, light-speed, digital-era organisational debt classically known as bureaucracy [6,2]. Computers bring productivity to human creativity that characterises the enterprise, hence IT's success, but at the potential cost of technical debt. Solution Architecture aligns IT with human creativity, as articulated in the organisation's Business Architecture, under EA's overall control, to address this issue [1]. In this overall landscape, data is pivotal.

Furthermore, metadata sets the data context. For example, the data in Electronic Medical Records (EMR) in healthcare is complex. Therefore, EMR uses metadata to make healthcare data intelligible [20]. Indeed, the metamodel that underpins EA itself is based on metadata, as it is itself an entity-relationship diagram that illuminates the pathways between strategy and technology and is elegantly depicted by LeanIX's metamodel diagram [11]. Metadata describes a business's understanding of enterprise data through data objects [7,14]. Metadata is ultimately a human construct, capturing the mindset of business stakeholders and employees in their roles, which collectively define the purpose of an enterprise as a human endeavour; hence, metadata's value to EA.

The paper is thus organised as follows. The research question is identified and justified, supported by a single-case study research method, which is exemplified by the selection of SAP as the case study. Moregraph and its ability to address technical debt through clean core metadata integration in Solution Architecture are introduced. Moregraph's conceptual structure leverages Conceptual Graphs, which capture Moregraph's requirements, specifications, and updates. This conceptual structure motivated the development of the Moregraph application, which included the selection of SAP Graph, SAP's Business Hub Accelerator, and consideration of alternatives outside SAP, to demonstrate metadata-driven Enterprise Architecture (EA). Related research, current limitations and future work are discussed, culminating in the paper's research contributions in its concluding remarks, followed by the references.

2 Research Method

2.1 Research Question

Our research question is "How can Moregraph better drive Clean Core Strategy through Metadata-driven Enterprise Architecture?". This question focuses

[1] For more background information about EA, there is a helpful resource at `https://www.leanix.net/en/blog/enterprise-architecture-terms-to-know-glossary`. Although this content is from a vendor, it provides a lucid glossary of EA and its terminology across all its dimensions without being overly product-biased. For example, we refer to the Data Object derived from that glossary alongside many others.

on the role of Moregraph in achieving clean core, which is a modern approach to maintaining system reliability while enabling agility and digital transformation. Clean core strategies are therefore increasingly important for organisations looking to minimise customisations within their core systems while leveraging cloud-based solutions. The likely or intended impacts include Moregraph driving clean core, through metadata-driven enterprise architecture, providing valuable insights into improving system efficiency, reducing technical debt, and enhancing scalability.

2.2 Case Study

In support of our research question, we selected a single-case study research method, as it enabled us to advance the theory and its practical application within a single case, rather than managing across multiple cases. Related research, such as the value co-creation of AI in industrial markets through a single case study in healthcare, underscores these benefits [12]. SAP's Business Technology Platform (BTP) was the single case study in research that examined the role of knowledge boundary resources in platform ecosystems [8].

We also chose SAP as the case study for our topic due to its market-leading significance and leadership in the enterprise software industry. SAP's comprehensive suite of applications and services provides a robust platform for business and technological innovations, as well as best practices that extend beyond SAP to the broader business computing ecosystem, such as via its Business Technology Platform (BTP). SAP's 2024 Annual Report[2] showed a total revenue of €32.5 billion, marking a 9% increase from the previous year. The cloud segment achieved €14.2 billion in revenue, up by 18% year-over-year. SAP serves over 440,000 customers in more than 180 countries. The company invests €4.5 billion in research and development. Additionally, SAP's sustainability initiatives, including a 30% reduction in greenhouse gas emissions compared to 2020, highlight its dedication to responsible business practices.

Our evaluation covered all the data objects that are publicly available in the SAP Business Accelerator Hub's Graph Navigator[3]. The number of data objects in that resource comprises SAP SuccessFactors (sap.hcm, 999), SAP S/4HANA ERP (sap.s4, 1479), and SAP Cloud for Customer (sap.c4c, 2027)[4]. The sum of these data objects is 999 + 1479 + 2027 = 4505. These objects encompass most of SAP's central systems, thereby providing a representative, enterprise-scale metadataset of data objects on which to base our study. As they are documented in SAP Business Accelerator Hub, they are all clean core. SAP Graph (sap.graph) adds a further 55 data objects that interconnect the

[2] `https://www.sap.com/investors/`

[3] `https://api.sap.com/graph`

[4] An overview of these systems can be found at `https://www.sap.com/products/hcm.html`, `https://www.sap.com/uk/products/erp.html`, and `https://support.sap.com/en/alm/solution-manager/expert-portal/public-cloud-operations/sap-hybris-cloud-for-customer.html`, respectively.

4505 clean core data objects, bringing the total to 4560. We intentionally added another small number of custom data objects (6) as a simple test of how the 4560 data objects can be accessed with custom-coded data objects outside of the clean core without incurring technical debt.

Given all the above, SAP is an exemplary subject for a detailed and informative case study.

3 Moregraph

The Moregraph application identifies data objects across and within applications, thereby revealing technical debt at a more precise level of granularity, as it is unrestricted by application boundaries. To convey Moregraph's essence and how it fulfils this purpose, Fig. 1 depicts Moregraph's conceptual structure as a Conceptual Graph.

3.1 Moregraph's Conceptual Structure

Arising from Sowa's original work and as part of the wider endeavour in Conceptual Structures [18], Conceptual Graphs enable humans to draw concepts intuitively while developing the underlying complex relationships at both logical and mathematical levels. Their graphical nature makes them accessible to both humans and machines, combining human creativity with computer productivity for knowledge representation, reasoning, and artificial intelligence. They have multiple modern-day interpretations [3,15,13]. The Conceptual Graph shown by Fig. 1 was produced to capture the requirements for Moregraph.

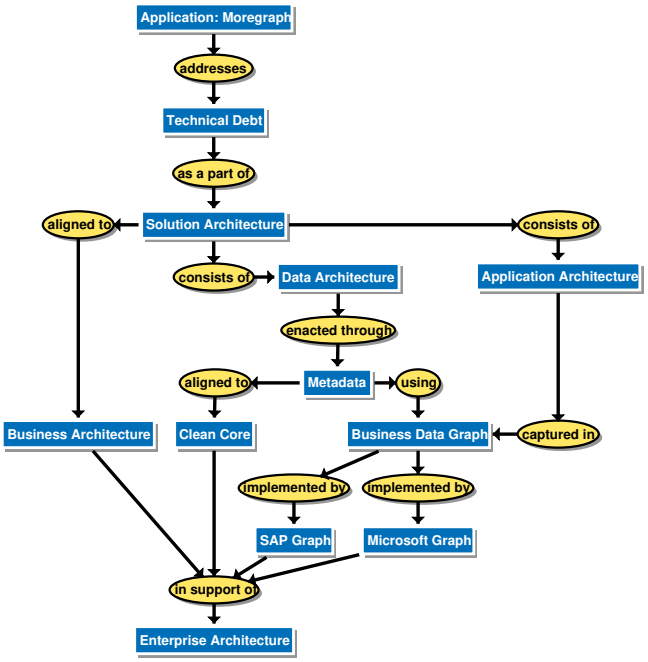

Fig. 1. Moregraph as a Conceptual Graph

This graph enables the key concepts that comprise Moregraph to be captured expressively, with each concept in Fig. 1 documented as follows. Taken together, Moregraph's formal purpose is revealed.

Technical Debt As stated earlier, Moregraph's modus operandi investigates technical debt by interconnecting data objects from multiple applications that do not hinder the investigation, adopting **Clean Core** as described shortly.

Solution Architecture, Business Architecture, and Enterprise Architecture (EA) Moregraph aligns Solution to Business Architecture concepts in support of EA, the culminating concept in Fig. 1. **Solution Architecture** consists of **Data Architecture** and **Application Architecture**, with a pathway from **Data Architecture** to **Metadata**, described earlier and **Clean Core**, which is described next.

Clean Core In software engineering, "clean core" leverages clean code [10]. Clean core refers to a design philosophy where the central, essential part of a software system is kept simple, well-organised, and free from unnecessary complexity. Clean core refines clean code into reusable code that any enterprise can leverage, as all enterprises share a common structure and do not need to customise more than 5% of the code in their IT systems. The other 95% is not unique, as 80% is code that is generic to all enterprises, such as accounting or human resources and in the purchasing-to-sales cycle. Industry-specific software accounts for the other 15%, which covers functionalities tailored to the specific needs of an industry sector. For example, a healthcare organisation might need software for patient management, whereas a retail business might need inventory management systems. While these percentages are not hard and fast, they reflect a common understanding in contemporary enterprise software development [1]. Clean core, therefore, significantly reduces development time and costs. Customising only the necessary parts allows enterprises to focus their resources on areas that truly differentiate them. This technical debt risk is simultaneously reduced to at most 5% of the organisation's IT landscape. Clean core furthermore extends the core, rather than modifying it, thus not breaking the core itself. Equally, updates to the core do not break the customisation extensions, forming an implementable contract that further avoids technical debt, which becomes even less than 5%, especially if 'no' or 'low' code alternatives to full custom coding are adopted [19]. A clean core enables future updates and migrations (e.g., from on-premise solutions to the Cloud) to be completed with fewer technical issues. The success of clean core is exemplified in an industrial context by SAP [17]. Moregraph takes clean core to the finer-grained data object level of granularity, demonstrated in Section 4.

Business Data Graph, and SAP Graph SAP Graph is a Business Data Graph because it provides a unified API for accessing data objects across SAP

and non-SAP systems and applications, reflecting a connected graph of business data objects [4]. SAP Graph identifies opportunities for simplifying API management, as the other APIs can be encapsulated and replaced by SAP Graph's navigation with fewer APIs, thus making API rationalisation possible. SAP Graph's focus is on the **Data Architecture**-oriented data objects in place of the applications where those objects happen to be processed. **Application Architecture** is also captured as shown in Fig. 1 because SAP Graph adds the application name as a metadata attribute to a data object. These features are demonstrated in Section 4. **Microsoft Graph** is another example of Business Data Graphs, which are not SAP-specific.

3.2 Moregraph's Application Architecture

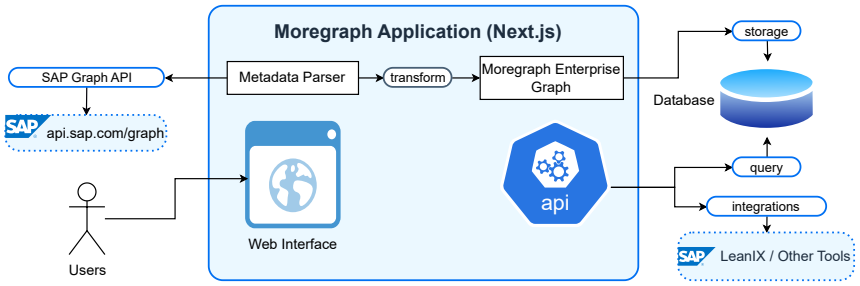

Fig. 2. Moregraph's Application Architecture

The implementation of the Moregraph application is scoped by the Conceptual Graph in Fig. 1 and the definitions provided in subsection 3.1. Fig 2 depicts Moregraph's application architecture. The Moregraph application is built using *Next.js*, a React framework for creating high-performance, feature-rich web applications, and *TypeScript*, a language designed for developing large-scale applications. These choices enable Moregraph to focus on integrating and visualising enterprise data objects by connecting to SAP Graph APIs. Additionally, to demonstrate connectivity with other applications, it integrates with SAP LeanIX, but is not limited to SAP applications. The system ingests metadata, transforms it into an enterprise graph, and stores it in a PostgreSQL database for querying and interaction through a web interface and API layer. It features the following components:

1. **SAP Graph API Integration**. The application connects to the SAP Graph API in the SAP Business Accelerator Hub's Graph Navigator, referred to earlier, and fetches enterprise metadata from the 4505 data objects across three core SAP applications interconnected by SAP Graph's 55 data objects. This metadata is ingested in raw form and passed to the internal

parsing system. Future planned work includes other APIs, such as Microsoft Graph[5].

2. **Metadata Parser**. This component parses the incoming metadata from the SAP Graph API. It leverages SAP Graph's interconnectivity and normalises the data objects and relationships into a canonical format suitable for graph construction. The metadata parser also handles mapping and cleansing logic to prepare for graph generation.
3. **Moregraph Enterprise Graph**. This component converts the parsed metadata into an internal graph structure that represents the data objects, applications, and interdependencies. It performs transformation logic to structure data as a directed, typed graph, and encapsulates core graph logic and domain relationships.
4. **Database**. The enterprise graphs are stored in a relational schema within a PostgreSQL database layer that supports the efficient querying of nodes, edges, and metadata properties.
5. **API**. The Application Programming Interface (API) layer exposes the enterprise graphs and related data via RESTful APIs. It handles authentication, business logic, and orchestrates database queries. It also mediates between the frontend and external integrations, such as LeanIX.
6. **LeanIX Integration**. The API interface with SAP LeanIX enables the creation of Data Objects and their associated applications in the LeanIX EA software tool[6]. Alternatively, Moregraph can export content as Excel spreadsheets in a format that can be imported into LeanIX or tailored for import to other tools. We have also explored and piloted an earlier integration with the Essential Project, an open-source non-SAP EA tool, with initial success that we plan to take forward [16].
7. **Web Interface**. Using Next.js frontend capabilities, enhanced with UI5 Web Components through a React wrapper[7], Moregraph produces dynamic web pages for end-users to query the enterprise graph that the application automatically visualises, explore metadata, and discover further insights, including clean core.

4 Demonstration

As shown by Fig. 2 and the narrative supplied in Section 3.2, the Moregraph application ingested the publicly available, clean core reference architecture documented in SAP's Business Hub Accelerator (*api.sap.com*). Together with SAP Graph, Moregraph interconnects the three pivotal SAP systems and all their data objects described in Section 2.2. These applications were: 1) sap.s4: SAP S/4HANA ERP (Enterprise Resource Planning), 2) sap.c4c: SAP Sales & Service Cloud CRM (Customer Relationship Management), and 3) sap.hcm: SAP SuccessFactors HCM (Human Capital Management).

[5] `https://learn.microsoft.com/en-us/graph/overview`
[6] `https://docs-eam.leanix.net/reference/available-apis`
[7] `https://github.com/SAP/ui5-webcomponents-react`

4.1 Navigating the Clean Core

Fig. 3 shows an extract of the resulting visualisation instantiated with the interconnected 4560 data objects (sap.graph, sap.s4, sap.c4c, and sap.hcm) and the flow directions between each data object.

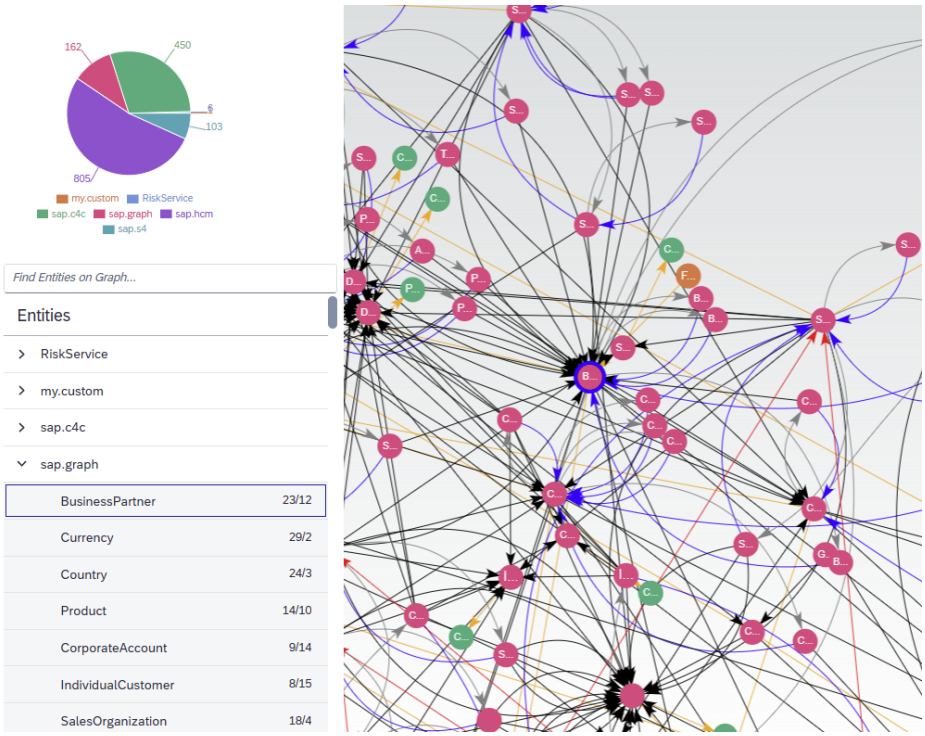

Fig. 3. Moregraph's visualisation from api.sap.com

Fig. 4 shows a *drill-down* into one of these data objects: BusinessPartner. The figure reveals that Moregraph, through SAP Graph, has interlinked the corresponding data object A_BusinessPartner in sap.s4: S/4HANA and BusinessPartnerCollection in sap.c4c: SAP Sales & Service Cloud.

When we select A_BusinessPartner in sap.s4 or BusinessPartnerCollection in sap:c4c, the navigation takes us into the heart of these systems, as shown by Fig. 5 and Fig. 6, respectively. While both figures primarily consist of the colour coding of those systems, the interconnections with sap.graph with its colour coding remain evident in both figures, including the pie chart on the left-hand side. A similar exercise from BusinessUser would point to the sap.hcm data objects, shown in Fig. 7.

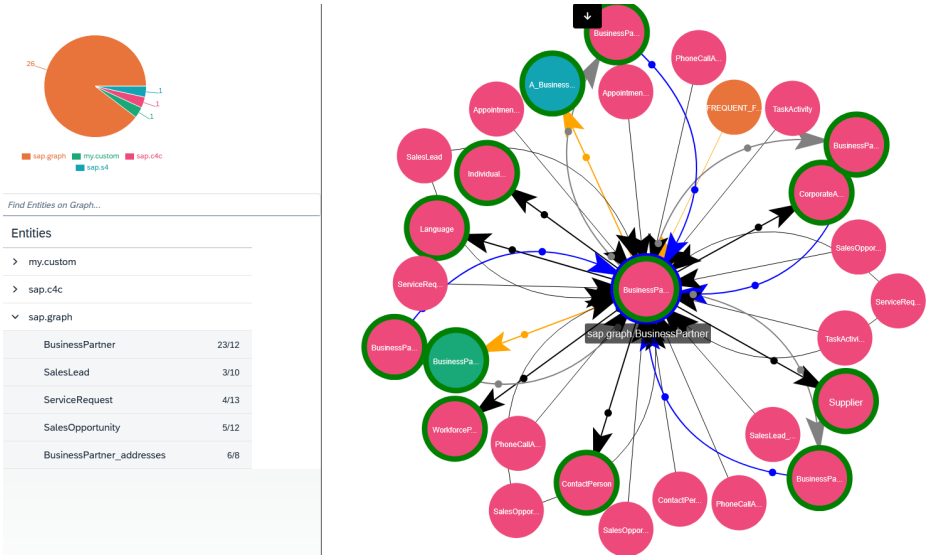

Fig. 4. Drill-down to the sap.graph BusinessPartner Data Object

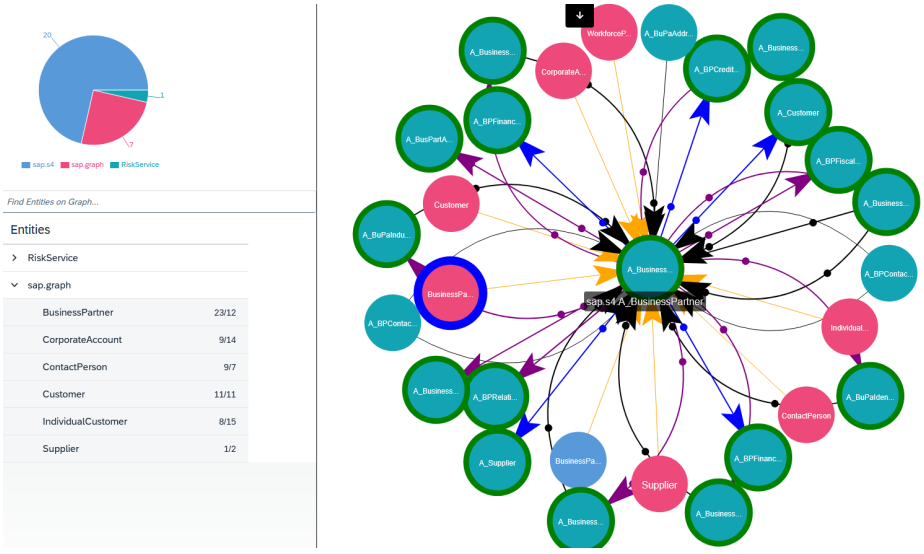

Fig. 5. Drill-down to the sap.s4 A_BusinessPartner Data Object

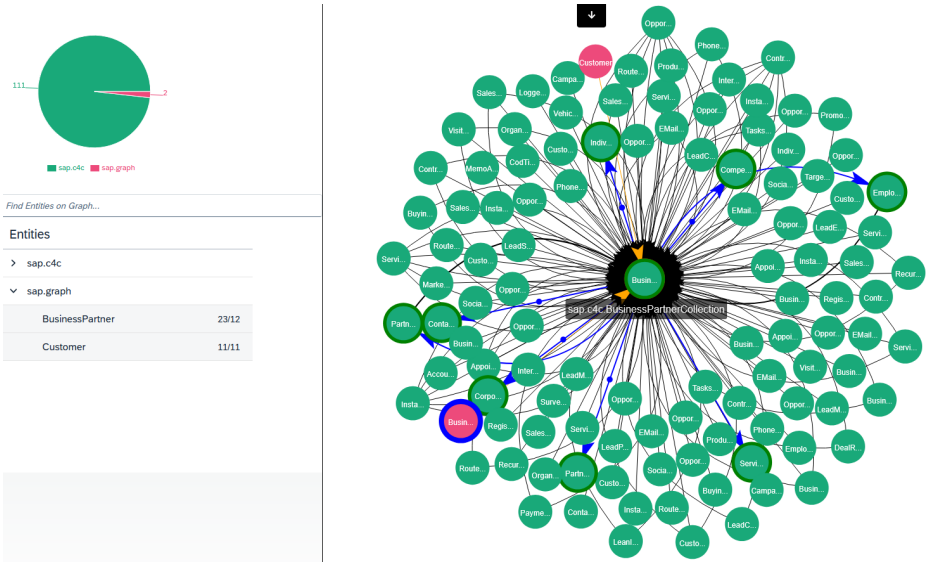

Fig. 6. Drill-down to the sap.c4c BusinessPartnerCollection Data Object

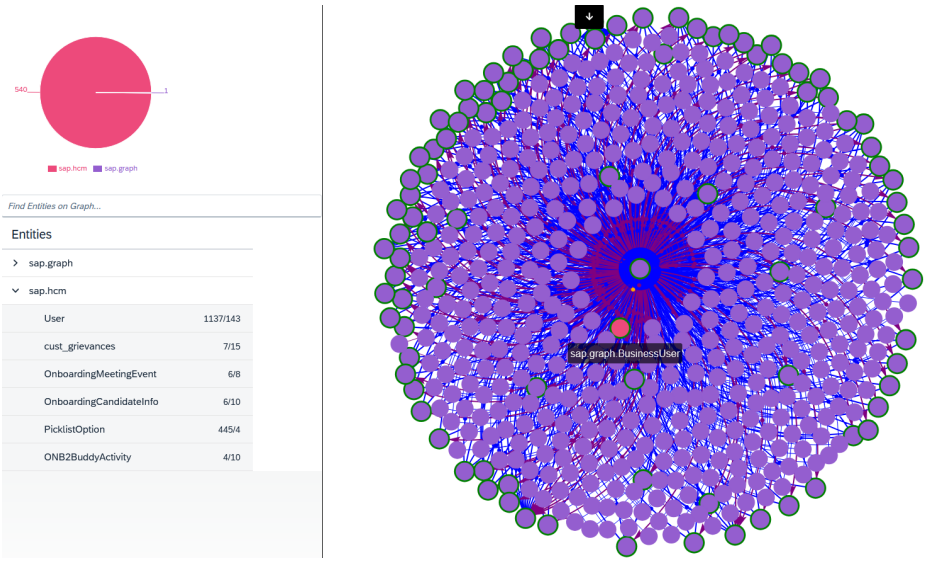

Fig. 7. Drill-down from the sap.graph BusinessUser Data Object to sap.hcm

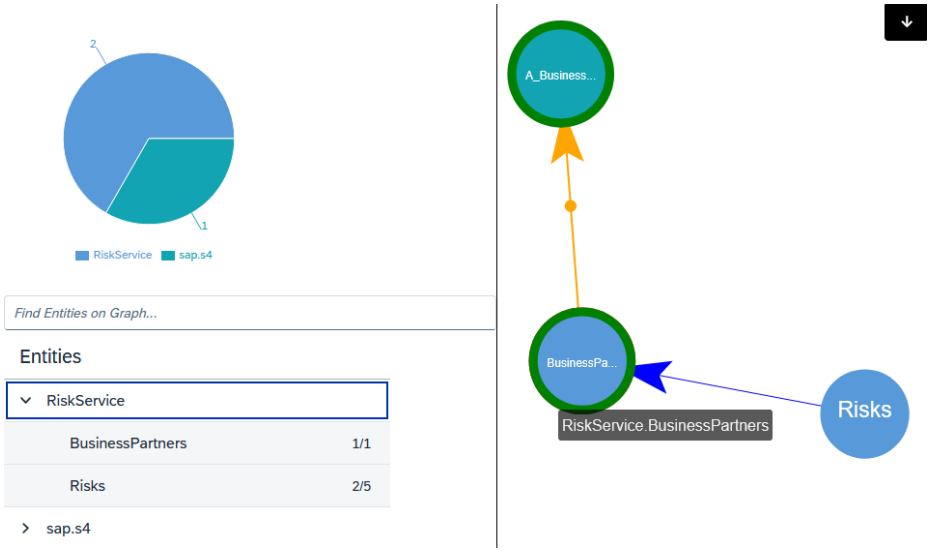

Fig. 8. sap.s4 RiskService extension

4.2 Extending the Clean Core

The link to a custom non-SAP FREQUENT_FLYER application shown towards the top-right and in the pie chart of Fig. 4 demonstrates a use case where SAP Graph, hence Moregraph, is not restricted to SAP systems. Notably, the link is clean core as this custom application has *not* modified the BusinessPartner data object. We similarly linked a custom extension to RiskService from A_BusinessPartner in S/4HANA [8]. Fig. 8 illustrates this extension, which is linked from Fig. 5 and is visible towards the bottom right of that diagram and the associated pie chart. Unlike FREQUENT_FLYER, which is an application in its own right, RiskService modifies A_BusinessPartner by extending it, thus following the clean core strategy articulated by SAP [17]. This extension has likewise not modified the core.

4.3 A Pathway to Clean Core

With the above sections 4.1 and 4.2 in mind, we considered data objects in typical organisational landscapes that were not clean core to identify a pathway to how they could become clean core. One such exemplar is shown in listing 1.1

Listing 1.1. Product example that's not Clean Core

```
<edmx:DataServices ...>
  <Schema Namespace="SEPMRA_PROD_MAN"... />
```

[8] This example is taken from `https://learning.sap.com/learning-journeys/build-side-by-side-extensions-on-sap-btp`.

```
    <EntityType Name="Product">
      <Key>
        <PropertyRef Name="ProductID"/>
      </Key>
      <Property Name="ProductID" ... />
      <Property Name="Name" ... />
      <Property Name="Price" ... />
      <!-- Custom Field added directly into standard object -->
      <Property Name="LegacySystemFlag" ... />
    </EntityType>
    <EntityContainer Name="SEPMRA_PROD_MAN_Entities">
      <!-- Overwriting standard EntitySet -->
      <EntitySet Name="Products"
          EntityType="SEPMRA_PROD_MAN.Product" ... />
    </EntityContainer>
  </Schema>
</edmx:DataServices>
```

The XML extract depicted in listing 1.1 highlights a typical situation where a) a custom field has been directly added (`LegacySystemFlag` in this case), and b) an entity type has been overwritten (`SEPMRA_PROD_MAN.Product` in this case), causing mutable access to the standard data create/update/delete operations on SAP-delivered data objects. It is evident from these two commonplace issues in a data object where a clean core-compliant metadata extension could replace that data object if it is to be retained, leading to its reconnection with the standard Product data object, thus reunited with the clean core 4560 data objects and visualised in Moregraph in the same way as shown by Fig. 8 for the RiskService extension.

4.4 Evaluation

Moregraph navigated to the individual data objects and applied clean core extensions at the metadata level, integrated custom applications, and included a pathway for hitherto non-clean core custom data objects. Notably, Moregraph can navigate through all the data objects, rather than being constrained by the broader application interfaces. This navigation level is akin to the Solution Data Flow Diagram familiar to EA[9].

Moregraph also provides an alternative means of reusing reference architecture content using the Business Hub Accelerator (api.sap.com/graph) alongside the capability or process-oriented methods in which that reference content is presently structured[10].

Furthermore, the existing capability or process-oriented reference architecture content also points to the same reference solution architecture at

[9] More specifically, in the SAP Enterprise Architecture Methodology Guide (`https://help.sap.com/docs/SAP_ENTERPRISE_ARCHITECTURE_FRAMEWORK` and search for 'Solution Data Flow Diagram').

[10] Refer to the previous footnote's link, but this time search for 'Reference Architecture Content'.

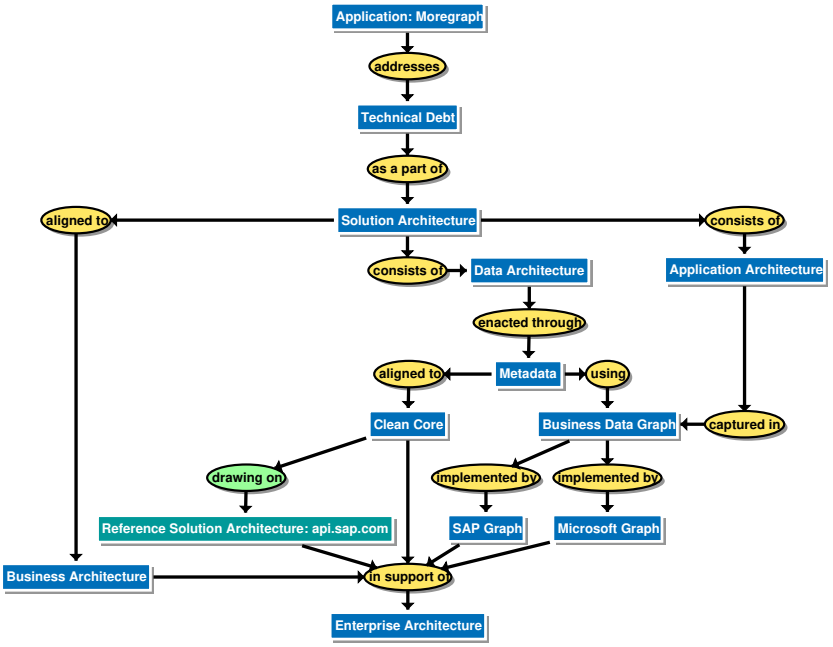

Fig. 9. Moregraph Conceptual Graph with the Reference Solution Architecture join

api.sap.com, thereby demonstrating alignment with it when accessed through the novel metadata route. This new knowledge can be joined to the original Conceptual Graph in Fig. 1, resulting in Fig. 9, which highlights the new join. In the pursuit of addressing **Technical Debt**, Moregraph has consequently advanced **Clean Core** to **Data Architecture**, hence **Enterprise Architecture**, and added **Reference Solution Architecture: api.sap.com**.

5 Discussion

5.1 Related Research

Knowledge and property graphs were also considered powerful alternatives to SAP Graph. Knowledge graphs emphasise semantic relationships and use triples (such as subject-predicate-object) to manage and infer knowledge, often employing ontologies for precise data description. Property graphs represent entities and their relationships using nodes and edges, with properties attached to both, making them suitable for dynamic schema changes and complex queries. Moregraph, however, focused on SAP Graph as it is based on business data graphs. Business data graphs provide a unified API for accessing data objects across heterogeneous systems, simplifying data integration and enabling Moregraph to interconnect applications through their data objects, rather than managing

complex data connections. Therefore, the convenience of SAP Graphs' generally available, wide-ranging implementation by SAP, together with the relative simplicity of business data graphs, presently meets Moregraph's needs.

5.2 Limitations and Future Work

While we have demonstrated Moregraph's research contribution, it has yet to be tested with other substantial use cases. We are in the early stages of two such industrial use cases. The first is an organisation that seeks the integration of SAP Graph with Microsoft Graph in its clean core journey, thus demonstrating Moregraph's broader appeal. The second allows another organisation to seamlessly interrogate their current data infrastructure at the data object level across the entire development, quality, and production (D/Q/P) software development life cycle and across multiple ERP systems resulting from the acquisition of other organisations. Moregraph would visually identify the data object differences (D/Q/P) from this use case and indicate where clean core principles have been adhered to, including the techniques shown in Section 4.

Like many others, both organisations desire to 'fit-to-standard', which means not revisiting what is already known. Instead, those risks are pushed back to the software vendor, e.g., SAP, who in turn has the opportunity to demonstrate that their standardised software does meet 95% of all their customers' needs through a rich, tested, and well-maintained common core. Customisations are then the exception rather than the norm, as organisations are freed to focus their IT budgets on their 5% core differentiating activities that distinguish their enterprise from the others in their industry sector.

Conceptual Graphs could potentially play a far bigger role in future work, as they emphasise logical structure in a human-digestible form, as lucidly depicted in Fig. 1. Similarly, Formal Concept Analysis with its mathematical underpinning can be used in conjunction with Conceptual Graphs [3], or from related work [5]. We will also explore how Artificial Intelligence (AI) agents can analyse the data objects of an organisation's landscape across all their SAP and non-SAP applications, thus aligning its non-clean core data objects with clean core opportunities as an automated service in Moregraph.

6 Concluding Remarks

The Moregraph application automatically visualises the data objects and their relations within and between applications. It offers the eradication of technical debt by enabling clean core to be applied to the detailed granularity of data objects within and across applications. While SAP Graph already identifies opportunities for rationalising APIs, Moregraph extends SAP Graph, and thus Business Data Graphs as a whole, to efficiently discover and display *all* the interconnected APIs and locate their data objects, as demonstrated in Section 4. Moregraph's navigations help to identify the deviances from standard data objects caused by an organisation's over-customised data objects outside of its

distinctive '5%'. As SAP Graph captures application information, and Moregraph can visualise all the interconnected data objects, it becomes possible to identify opportunities for both API and application rationalisation, where applications add too little or even negative value from their data objects, exposed by Moregraph.

These experiences facilitate the organisation's return to a fit-to-standard approach by establishing a clean core at the detailed Data Architecture level, as opposed to the Application Architecture level, where the application's boundaries encapsulate that detail, thus causing valuable insights to remain undiscovered. Moregraph's conceptual structure, implementation, and demonstration have begun the journey to advance Metadata-driven Enterprise Architecture (EA). It has set the direction for an organisation's journey toward eradicating technical debt and to advance it further through our future work.

Acknowledgments. The authors are grateful for the invaluable contributions towards this work made by Lorenzo Ramirez—additionally, Stephanie Lewellen and her colleagues who were on the SAP Graph beta program team at SAP. We also thank Truc Huynh and Zairul Jilani from the Department of Computing and Digital Technologies at Sheffield Hallam University through the local Digital Innovation for Growth (DIfG) programme.

Disclosure of Interests. The authors have no competing interests to declare relevant to this article's content.

References

1. Abendroth, A., Bender, B., Gronau, N.: The Evolution of Original ERP Customization: A Systematic Literature Review of Technical Possibilities:. In: Proceedings of the 26th International Conference on Enterprise Information Systems. pp. 17–27. SCITEPRESS - Science and Technology Publications, Angers, France (2024). https://doi.org/10.5220/0012305500003690, https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0012305500003690
2. Ahmad, M.O., Al-Baik, O.: Beyond Technical Debt Unravelling Organisational Debt Concept. In: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing. pp. 802–809. SAC '24, Association for Computing Machinery, New York, NY, USA (May 2024). https://doi.org/10.1145/3605098.3635913, https://dl.acm.org/doi/10.1145/3605098.3635913
3. Andrews, S., Polovina, S.: Exploring, Reasoning with and Validating Directed Graphs by Applying Formal Concept Analysis to Conceptual Graphs. pp. 3–28. Springer (Mar 2018), http://www.springer.com/gb/book/9783319781013, issue: 10775 Num Pages: 139 Number: 10775
4. Bendelac, C.: Introduction to SAP Graph (Jun 2021), https://community.sap.com/t5/technology-blogs-by-sap/part-1-introduction-to-sap-graph/ba-p/13503946, section: Technology Blogs by SAP
5. Bendimerad, A., Mathonat, R., Remil, Y., Kaytoue, M.: Exploiting Formal Concept Analysis for Data Modeling in Data Lakes. In: Cabrera, I.P., Ferré, S., Obiedkov, S. (eds.) Conceptual Knowledge Structures. pp. 270–285. Springer Nature Switzerland, Cham (2024)

6. Grinyer, P.H., Yasai-Ardekani, M.: Strategy, Structure, Size and Bureaucracy. Academy of Management Journal **24**(3), 471–486 (Sep 1981). https://doi.org/10.5465/255569, https://journals.aom.org/doi/abs/10.5465/255569, publisher: Academy of Management
7. Inmon, W.H., O'Neil, B., Fryman, L.: Business metadata: Capturing enterprise knowledge. Morgan Kaufmann (2010)
8. Kauschinger, M., Schreieck, M., Krcmar, H.: The role of knowledge boundary resources for platform ecosystems – insights and design implications from SAP's Business Technology Platform. R&D Management **54**(3), 470–495 (2024). https://doi.org/10.1111/radm.12644, https://onlinelibrary.wiley.com/doi/abs/10.1111/radm.12644
9. Kotusev, S., Kurnia, S., Dilnutt, R., van de Wetering, R.: The Structuring of Enterprise Architecture Functions in Organizations. Business & Information Systems Engineering **66**(4), 465–488 (Aug 2024). https://doi.org/10.1007/s12599-023-00845-4, https://doi.org/10.1007/s12599-023-00845-4
10. Latte, B., Henning, S., Wojcieszak, M.: Clean Code: On the Use of Practices and Tools to Produce Maintainable Code for Long-Living. vol. Vol-2308, pp. 96–99. Stuttgart (Feb 2019), http://CEUR-WS.org
11. LeanIX: LeanIX Meta Model (2025), https://docs-eam.leanix.net/docs/meta-model
12. Leone, D., Schiavone, F., Appio, F.P., Chiao, B.: How does artificial intelligence enable and enhance value co-creation in industrial markets? An exploratory case study in the healthcare ecosystem. Journal of Business Research **129**, 849–859 (May 2021). https://doi.org/10.1016/j.jbusres.2020.11.008, https://www.sciencedirect.com/science/article/pii/S0148296320307517
13. Luger, G.F.: Association-Based Representations: Frames, Conceptual Graphs, WordNet, and FrameNet. In: Artificial Intelligence: Principles and Practice, pp. 321–339. Springer (2025)
14. Olesen-Bagneux, O.: The Enterprise Data Catalog. O'Reilly (2023)
15. Quintero, R., Torres-Ruiz, M., Saldaña-Pérez, M., Guzmán Sánchez-Mejorada, C., Mata-Rivera, F.: A Conceptual Graph-Based Method to Compute Information Content. Mathematics **11**(18), 3972 (Jan 2023). https://doi.org/10.3390/math11183972, https://www.mdpi.com/2227-7390/11/18/3972, number: 18 Publisher: Multidisciplinary Digital Publishing Institute
16. Saleem, M., Polovina, S.: Towards Metadata-Driven Enterprise Architecture by Integrating SAP Graph and Essential Project. In: Proceedings of MOVE2024: The Second International Measuring Ontologies for Value Enhancement Workshop. pp. In press – to be updated. CEUR-WS (2025), https://ceur-ws.org/(toappear)
17. SAP: RISE with SAP | ERP Clean Core Strategy (2025), https://www.sap.com/products/erp/rise/methodology/clean-core.html
18. Sowa, J.F.: Conceptual Graphs for Representing Conceptual Structures. In: Conceptual Structures in Practice. Chapman and Hall/CRC (2009), num Pages: 36
19. Sufi, F.: Algorithms in Low-Code-No-Code for Research Applications: A Practical Review. Algorithms **16**(2), 108 (Feb 2023). https://doi.org/10.3390/a16020108, https://www.mdpi.com/1999-4893/16/2/108, number: 2 Publisher: Multidisciplinary Digital Publishing Institute
20. Xu, B.: Healthcare data analytics: Using a metadata annotation approach. Journal of Management Analytics **3**(2), 136–151 (2016)