Doctoral Thesis Summary

Babeș-Bolyai University

Faculty of Economics and Business Administration

Economics and Business Administration Doctoral School

Ph.D. Student: Andrei Chiș

Supervisor: Prof. Dr. Robert Andrei Buchmann

Cluj-Napoca, 2026

Babeș-Bolyai University

Faculty of Economics and Business Administration

Economics and Business Administration Doctoral School

Business Information Systems

Modeling Methods as Enablers of Model-Driven GraphRAG for Enterprise Knowledge Streamlining

Ph.D. Student: Andrei Chiș

Supervisor: Prof. Dr. Robert Andrei Buchmann

Cluj-Napoca, 2026

Acknowledgements

This thesis would not have been possible without the unconditional support of my family and mentors, who guided my academic path from the very beginning, when I wrote my first scientific article seven years ago.

I would like to thank my parents for their support, love, and guidance. Their encouragement has meant the world to me every day, throughout all the academic and business activities I have carried out over the course of my career.

I am especially grateful to my supervisor, Prof. Dr. Robert Buchmann, who has played an essential role in shaping my academic career over the past seven years. He was always available whenever I needed advice or support in my research, and he gave me the freedom to develop my own research identity by pursuing the topics that interested me most within the field of Information Systems.

I would like to thank Prof. Dr. Gheorghe Silaghi, who was part of my Ph.D. advisory committee, for his support and for trusting me with seminar teaching responsibilities in his courses.

I am thankful to Assoc. Prof. Dr. Virginia Niculescu and Assoc. Prof. Dr. Adrian Sterca for being part of my Ph.D. advisory committee and for their valuable advice throughout my doctoral studies.

Finally, I extend my sincere thanks to all the staff of the Faculty of Economics and Business Administration at Babeș-Bolyai University, who supported me with various matters during my Ph.D. years.

Abstract

This Ph. D. thesis proposes Model-Driven GraphRAG - a methodological contribution to enterprise knowledge management, based on a blueprint of modeling methods that can be applied for enterprise knowledge capture and streamlining - between traditional diagrammatic modeling environments and modern GraphRAG architectures. The blueprint has been successfully instantiated in two domain-specific modeling methods and one generic knowledge modeling method. From a usage procedure perspective, this method class aims to support a hybrid operating process – (a) on one hand supporting less technical stakeholders with a diagrammatic design space that is aligned with selected domains-specific areas of expertise; (b) on the other hand supporting AI systems engineering with the integration between human-oriented visual knowledge representation and machine-oriented GraphRAG mechanisms informed by the visual knowledge capture.

The application domains for the three demonstrative instance methods based on the proposed blueprint are: (a) work systems management based on the Work Systems Framework - a practice-specific conceptualization established in management science; (b) a cyberthreat mapping approach based on BPMN, DFD and threat-mitigation taxonomies from the cybersecurity domain; (c) generic knowledge graphs management based on the technology-specific conceptualization of the Resource Description Framework. All these are areas are hereby supported with modeling methods that enable the creation of knowledge graphs tailored to act as contextual enterprise knowledge in GraphRAG architectures, thus enabling new types of knowledge flows under the Hybrid Intelligence paradigm: diagrammatic designs already familiar to enterprise modelers or knowledge engineers thus contribute to the knowledge ingestion mechanisms emerging in the enterprise AI era.

Considering the artifact-centric nature of the work, the research was conducted according to the Design Science process, while the method and tool engineering followed the framework of Agile Modeling Method Engineering. As evaluation strategies, the work reports for the different instance methods a balance of user-centric assessments, tool comparisons and assessments of internal design decisions involved in the GraphRAG interoperability.

Doctoral Thesis Abbreviations List

AI Artificial Intelligence

AKM Active Knowledge Modeling

AMME Agile Modeling Method Engineering

BPMN Business Process Model and Notation

DFD Data Flow Diagram

DSL Domain-Specific Language

DSML Domain-Specific Modeling Language

DSR Design Science Research

EA Enterprise Architecture

GPML General Purpose Modeling Language

ISDM Information Systems Development Methodology

KG Knowledge Graph

LLM Large Language Model

ML Machine Learning

RAG Retrieval Augmented Generation

RDF Resource Description Framework

RDFS Resource Description Framework Schema

SECI Socialization, Externalization, Combination, Internalization

SME Situational Method Engineering

TOGAF The Open Group Architecture Framework

UML Unified Modeling Language

URI Uniform Resource Identifier

W3C World Wide Web Consortium

WSF Work System Framework

WSM Work System Method

WST Work System Theory

XML eXtensible Markup Language

Table of Contents of the Doctoral Thesis

Acknowledgements ..... **iii**

Abstract ..... **iv**

Rezumat ..... **v**

Abbreviations List ..... **vi**

List of Figures ..... **x**

List of Tables ..... **xii**

1 Introduction **1**

2 Preamble: Foundations, Working Definitions and Conceptual Framing of the Research **6**

2.1 Introducing Model-Driven GraphRAG 6

2.2 Scientific Areas Converging to the Contribution 9

2.2.1 Method Engineering 9

2.2.2 Situational Method Engineering 10

2.2.3 Modeling Method Engineering 10

2.2.4 Metamodeling 14

2.2.5 Domain Specific Modeling Languages & Language Engineering 15

2.2.6 Knowledge Modeling 17

2.3 Application Areas for the Contribution 19

2.3.1 Knowledge Representation using the RDF Standard 19

2.3.2 The Work System Theory 24

2.3.3 Process-centric Cyberthreat-mitigation Mapping 27

2.3.4 The SECI Knowledge Conversion Modes 30

3 Problem Statement & Research Methodology **32**

3.1 Problem Context: Knowledge Management Systems in the Age of Hybrid Intelligence & Research Questions 32

3.2 Research Methodology: Design Science Research 37

3.3 The Hybrid Design Science – Agile Modeling Method Engineering Process 44

3.4 Conceptual Framing of the Contribution 52

4 Contributions Overview **55**

4.1 A Blueprint for Modeling Methods That Enable Model-Driven GraphRAG 55

4.2 Repurposing SECI with a Hybrid Intelligence Model for Knowledge Modeling and Management 57

5 Literature Review **64**

5.1 Literature Related to System Design through the Work System Framework 64

5.2 Literature Related to Process-centric “Secure by design” Methods 66

5.3 Literature Related to Knowledge Graph Building Environments 72

5.4 Literature Related to Model-Driven GraphRAG 74
6 The Developed Instance Modeling Methods 78
6.1 Work Systems Design Instance Method 78
6.1.1 Requirements 81
6.1.2 Instance Method Design 83
6.1.3 Running Examples 98
6.1.4 Evaluative Experiments 102
6.2 Process-centric Cyberthreat-mitigation Mapping Method 118
6.2.1 Requirements 119
6.2.2 Instance Method Design 121
6.2.3 Running Examples 133
6.2.4 Evaluative Experiments 138
6.3 Visual Knowledge Graph Building Instance Method 158
6.3.1 Requirements 159
6.3.2 Instance Method Design 162
6.3.3 Running Examples 172
6.3.4 Evaluative Experiments 175
7 Discussions and Implications 189
7.1 Summary 189
7.2 Discussion of the Research Questions 193
7.2.1 RQ1: How can tacit enterprise knowledge be captured by means of metamodels acting as domain-specific mediators in the knowledge transfers between enterprise stakeholders and retrieval-augmented AI systems? 193
7.2.2 RQ2: How can enterprise KG development be facilitated for enterprise modelers familiar with domain-specific conceptualizations and enterprise modeling, but lacking technical KG skills? 194
7.2.3 How do traditional unstructured descriptions of enterprise content compare with DSML-based diagrammatic descriptions, in terms of LLM answering quality, using different retrieval strategies for model-based content? 196
7.3 Discussion of the Main Artifact Contribution 197
7.4 Discussion of Evaluation Results 198
7.5 Theoretical Implications 199
7.5.1 Implications for Knowledge Management and SECI 199
7.5.2 Implications for Hybrid Intelligence 200
7.5.3 Implications for Modeling Method Engineering 201
7.6 Practical Implications 201
7.6.1 Implications for Enterprise Knowledge Management 201

7.6.2 Implications for Managers and Work-System Practitioners 202
7.6.3 Implications for Cybersecurity Analysts 202
7.6.4 Implications for Knowledge Engineers 203
7.7 Technical and Architectural Implications 203
7.8 Methodological Implications for Design Science Research 204
7.9 Limitations 205
8 Conclusions and Future Work 208
Doctoral Dissemination 211
Journal Papers 211
Conference Papers 212
References 216
Appendix 240

List of Figures of the Doctoral Thesis

Fig. 1. RAG flow example 7
Fig. 2. GraphRAG flow example 7
Fig. 3. The main components of modeling methods (Karagiannis & Kühn, 2002) 12
Fig. 4. The Agile Modeling Method Engineering framework (Karagiannis, 2015) 13
Fig. 5. Metamodel example 15
Fig. 6. Key differences between DSMLs and GPMLs (Frank, 2011) 16
Fig. 7. Data to knowledge spaces representation (Lillehagen & Krogstie, 2008) 18
Fig. 8. Knowledge Graph example 21
Fig. 9. The Work System Framework (Alter, 2006) 26
Fig. 10. Hybrid Intelligence in the context of artificial and human intelligence (reproduced from Jarrahi et al. (2022)) 34
Fig. 11. SECI knowledge conversion modes in the context of Hybrid Intelligence and Conceptual Modeling (inspired by (Nonaka, 1994)) 35
Fig. 12. Research scope delimitation in the context of SECI and Hybrid Intelligence 36
Fig. 13. The typical Design Science Research process model (Peffers et al., 2006) 40
Fig. 14. The three cycle view for Design Science Research (Hevner, 2007) 41
Fig. 15. The revised SEQUAL framework (Krogstie et al., 2006) 43
Fig. 16. The AMME-DSR hybrid cycle (Dolha et al., 2025) 46
Fig. 17. Domain positioning of the research contribution 53
Fig. 18. The modeling method blueprint repurposed for model-driven GraphRAG (adapted from Karagiannis & Kühn (2002)) 55
Fig. 19. The modeling method blueprint pipeline 56
Fig. 20. The modeling method blueprint architectural view 57
Fig. 21. The repurposed SECI knowledge conversion modes for Hybrid Intelligence (inspired by Nonaka & Takeuchi (1995)) 63
Fig. 22. ADOxx metamodel of the DSML of WSF4ADO 84
Fig. 23. Model-level attributes of a Work System Overview Diagram for WSF4ADO 85
Fig. 24. Instance-level attributes of a WorkSystemSnapshot element (on the left) and AI agent specific attributes of a SimpleWSTElement (on the right) 86
Fig. 25. Core ontology fragment supporting knowledge capture in Work System Framework diagrams 87
Fig. 26. Architecture for a model-driven WSF knowledge system 91
Fig. 27. Mechanisms and functionalities implemented in WSF4ADO 93
Fig. 28. Extract from the script converting WSF diagrams into RDF graphs 96
Fig. 29. GraphRAG flows architectures in WSF4ADO 98
Fig. 30. Examples of WSF4ADO diagrams 99
Fig. 31. Work Subsystems as specific kind of BPMN subprocess to facilitate drill-down 100
Fig. 32. RDF patterns derived from diagrammatic WS designs 101
Fig. 33. Single diagram knowledge base for a banking industry use-case 102
Fig. 34. Quantitative evaluation results in terms of creation and serialization times for WSF4ADO 104
Fig. 35. Metamodel of the hybrid DSML implemented in CTM4ADO (BPMN not included, except for the docking subprocess construct) (Chiș et al., 2026) 122
Fig. 36. Model-level attributes (up) and element-level attributes (down) in CTM4ADO (Chiș et al., 2026) 124
Fig. 37. Architectural proposition for DSML-driven knowledge flows in CTM4ADO (Chiș et al., 2026) 129

Fig. 38. Extract from the diagram-to-RDF knowledge graph serialization for CTM4ADO (Chiș et al., 2026) 129
Fig. 39. Extract from the guiding ontology for CTM4ADO (Chiș et al., 2026) 130
Fig. 40. CTM4ADO security audit scoring (Chiș et al., 2026) 131
Fig. 41. Running example for CTM4ADO (Chiș et al., 2026) 135
Fig. 42. Dynamic hops retrieval architecture for CTM4ADO (Chiș et al., 2026) 147
Fig. 43. OWASP threat modeling exemplar (Chiș et al., 2026) 153
Fig. 44. Metamodel of the DSML of GRAPHxx (Chiș et al., 2025a) 163
Fig. 45. Model level attributes in GRAPHxx (Chiș et al., 2025a) 164
Fig. 46. Attributes of ThingIdentifier (left) and attributes of RelatedTo (right) (Chiș et al., 2025a) . 165
Fig. 47. Dual view of how relationships can be handled either visually or in a tabular format (upper side) and an AdoScript extract for class node generation (bottom) (Chiș et al., 2025a) 167
Fig. 48. Architectural view on integration with RDF-reliant external systems (Chiș et al., 2025a) ... 169
Fig. 49. Extract from the script converting diagram elements into RDF serialization (Chiș et al., 2025a) 170
Fig. 50. Excerpt from the Python script for LLM integration via LangChain (Chiș et al., 2025a) 172
Fig. 51. KG exemplar split across two linked diagrams in GRAPHxx (Chiș et al., 2025a) 173
Fig. 52. Visual design of a graph snippet for SEO (left) and an excerpt from its JSON-LD serialization (right) (Chiș et al., 2025a) 175
Fig. 53. PowerPoint representation of a graph example used in evaluation tasks (top), its narrative equivalent (bottom-left) and Turtle equivalent (bottom-right) (Chiș et al., 2025a) 177
Fig. 54. The AQL query (on the left) & its results (on the right) (Chiș et al., 2025a) 185
Fig. 55. Quantitative evaluation results in terms of creation and export times (Chiș et al., 2025a) ... 186
Fig. 56. Instance methods in the context of Hybrid Intelligence 193

List of Tables of the Doctoral Thesis

Table 1. GraphRAG types categorization 8
Table 2. Mapping between SECI spiral and thesis contributions 61
Table 3. Modeling concepts of WSF4ADO and visual notations 88
Table 4. Consistency rules and checks 92
Table 5. Quantitative measures for scalability evaluation for WSF4ADO 103
Table 6. Quantitative measures for user centric evaluation experiment 1 for WSF4ADO 105
Table 7. Quantitative measures for user centric evaluation experiment 2 for WSF4ADO 106
Table 8. Semantic roundtrip experiment results for WSF4ADO 107
Table 9. Question categories for the IT company scenario for WSF4ADO 113
Table 10. Average elapsed times and costs aggregated per retrieval strategy for WSF4ADO 114
Table 11. Average percentage of responses in each category aggregated per retrieval strategy using an LLM-as-Judge approach for WSF4ADO 114
Table 12. Overall per retrieval strategy aggregated RAGAS metrics for WSF4ADO 115
Table 13. Overall per retrieval strategy aggregated cosine similarity for WSF4ADO 117
Table 14. CTM4ADO modeling concepts and their attributes (Chiș et al., 2026) 125
Table 15. The core (DFD) elements in the CTM4ADO running example (Chiș et al., 2026) 136
Table 16. Question categories for the e-commerce scenario in experiment 1 for CTM4ADO (Chiș et al., 2026) 148
Table 17. Average elapsed times and costs aggregated per retrieval strategy for experiment 1 for CTM4ADO (Chiș et al., 2026) 149
Table 18. Average percentage of correct, partially correct and incorrect responses aggregated per retrieval strategy for experiment 1 using an LLM-as-Judge approach for CTM4ADO (Chiș et al., 2026) 149
Table 19. Overall per retrieval strategy aggregated RAGAS metrics for experiment 1 for CTM4ADO (Chiș et al., 2026) 150
Table 20. Question categories for the medical test set of experiment 2 for CTM4ADO (Chiș et al., 2026) 154
Table 21. Average elapsed times and costs aggregated per retrieval strategy for experiment 2 for CTM4ADO (Chiș et al., 2026) 154
Table 22. Average percentage of correct, partially correct and incorrect responses aggregated per retrieval strategy for experiment 2 using an LLM-as-Judge approach for CTM4ADO (Chiș et al., 2026) 154
Table 23. Overall per retrieval strategy aggregated RAGAS metrics for Experiment 2 for CTM4ADO (Chiș et al., 2026) 155
Table 24. Overall per retrieval strategy aggregated cosine similarity for Experiment 1 157
Table 25. Overall per retrieval strategy aggregated cosine similarity for Experiment 2 157
Table 26. Modeling conepts of GRAPHxx (Chiș et al., 2025a) 164
Table 27. Evaluation findings summary for GRAPHxx (Chiș et al., 2025a) 179
Table 28. Relation between instance methods and AKM 190
Table 29. Coverage of HI characteristics by modeling artifacts 191
Table 30. Aggregated metrics for 1-4 hops for experiment 1 in CTM4ADO case 206

1 Introduction

This doctoral thesis makes a methodological contribution, supported by demonstrative artifacts, to **Enteprise Knowledge** capture and streamlining between enterprise modeling environments and GraphRAG architectures. Concretely, the thesis proposal enhances the role of conceptual and enterprise modeling in artificial intelligence systems which leverage structured enterprise knowledge. Its central premise is that enterprise diagrams should not be treated solely as aids to communication, documentation, or analysis. When created using explicit modeling languages and governed by appropriate metamodels, they can also become structured sources of knowledge for knowledge graph-based retrieval and large language models (LLMs). The thesis consequently proposes **Model-Driven GraphRAG** as a methodological bridge between human-oriented diagrammatic knowledge capture and machine-oriented knowledge retrieval.

Knowledge exists in 2 main forms: tacit and explicit. Tacit knowledge is regarded as knowledge internal to knowing subjects, lacking a format in which it can be stored in a knowledge management system or accessed by different users, while explicit knowledge is the form which is stored in different knowledge management systems, documents, repositories etc. According to Nonaka's knowledge conversion spiral (1994) there are 4 main modes in which knowledge is produced and consumed:

- Socialization: Knowledge remains in a tacit form through direct interactions between knowing subjects;
- Externalization: Tacit knowledge gets transformed into explicit knowledge; different knowledge management systems are used to capture and store knowledge;
- Combination: Knowledge management systems synthetize explicit knowledge from different sources and representations having the purpose of enhancing it;
- Internalization: Knowing subjects acquire new knowledge by working with different knowledge management systems, allowing them to broaden their internal tacit knowledge leveraging explicit forms.

The research addresses a persistent enterprise problem: relevant enterprise knowledge is dispersed across individuals (tacit knowledge), documents, information systems, process descriptions, databases, and modeling repositories (explicit knowledge). These sources differ substantially in structure, expressiveness, and intended audience. While enterprises have long relied on knowledge management systems to coordinate knowledge creation, preservation, discovery, and reuse, the emergence of generative AI places additional demands on how enterprise knowledge is structured and accessed. Text retrieval based on semantic similarity

can identify relevant passages, but it does not always preserve explicit relationships between business activities, resources, systems, actors, or decisions. An AI service may therefore produce plausible answers without recovering the precise domain relationships required by a question.

The work done in this thesis was aimed primarily at knowledge externalization and combination, as a means to capture tacit knowledge through diagrammatic representations and provide semantic knowledge graph structuring of knowledge for **Model-Driven GraphRAG** knowledge flows in modern AI-ready enterprises.

The thesis distinguishes **knowledge capture**, understood as collecting, documenting, and storing valuable enterprise information and expertise, from **knowledge streamlining**, understood as organizing and optimizing knowledge storage, sharing, and access. It also distinguishes explicit knowledge, which may be articulated and codified, from tacit knowledge, which remains embedded in individual experience and practice (Becerra-Fernandez & Sabherwal, 2014; Merali & Davies, 2001). The central methodological challenge is not to assume that all tacit knowledge can be stored, but to enable knowledgeable stakeholders to externalize useful parts of their understanding through meaningful, situated modeling activities.

The proposed solution is a reusable **blueprint for modeling methods** that supports a continuous transformation of knowledge. Domain experts express knowledge in diagrams using concepts familiar to their work. Modeling languages and metamodels constrain those descriptions, preserve distinctions between types of concepts and relations, and supply a basis for deterministic transformation. Dedicated mechanisms then serialize the captured structures into Resource Description Framework (RDF) knowledge graphs. These graphs can be stored in RDF repositories, accessed through SPARQL, and supplied as contextual evidence to LLM-based question-answering systems, following **GraphRAG** patterns. In this way, the same modeling effort supports both human interpretation and computational exploitation of enterprise knowledge.

The contribution is demonstrated in three different modeling settings. The **Work Systems Design Instance Method**, implemented in **WSF4ADO**, supports managers and other stakeholders who reason about enterprise work through Work System Theory and the Work System Framework (Alter, 2006; Alter, 2013). The **Process-centric Cyberthreat-mitigation Mapping Instance Method**, implemented in **CTM4ADO**, supports the analysis of threats, mitigations, business processes, information flows, and trust boundaries. The **Visual Knowledge Graph Building Instance Method**, implemented in **GRAPHxx**, provides a more generic, technology-specific environment for creating RDF-oriented graphs visually. These applications are deliberately different: the first takes a managerial conceptualization as its starting point, the second addresses a specialized cybersecurity domain, and the third targets

the knowledge-representation technology itself. Together, they offer evidence that the blueprint is not restricted to one narrowly specified notation or business scenario.

A distinctive aspect of the thesis is the position assigned to **Hybrid Intelligence** (Akata et al., 2020, Dellermann, Ebel et al. 2019, Jarrahi et al., 2022). The research does not advocate transferring responsibility for enterprise understanding to AI. Instead, human participants determine what is meaningful, establish relevant relationships, and judge whether a representation is adequate; AI systems assist with retrieving, combining, and communicating what has been represented. This orientation is linked to Nonaka's spiral model of knowledge conversion, especially its externalization and combination modes (Nonaka, 1994; Akata et al., 2020). The diagram becomes an accessible vehicle for articulation, the knowledge graph becomes a machine-readable representation of selected knowledge, and GraphRAG provides a means of activating that representation through natural-language interaction.

The thesis is situated within Design Science Research (Wieringa, 2014) because its principal contribution consists of designed and evaluated artifacts rather than only observational propositions. The method engineering of the three instance methods follows Agile Modeling Method Engineering (AMME), enabling iterative refinement of modeling concepts, notations, functionality, semantic mappings, and integrations (Karagiannis, 2015; Wieringa, 2014). Evaluation is correspondingly plural: the study examines usability and modeling effort, semantic preservation, technical scalability, ontology-based competency questions, and the quality of answers produced using different retrieval strategies. This combination of methodological and empirical work underpins the thesis's main argument: **modeling methods can be engineered as knowledge capture and streamlining means for AI-ready enterprises**, enhacing knowledge externalization and combination (Nonaka, 1994) rather than remaining isolated instruments of human-facing representation.

2 Preamble: Foundations, Working Definitions and Conceptual Framing of the Research

2.1 Introducing Model-Driven GraphRAG

The second chapter establishes the technical and conceptual vocabulary needed to explain the proposed contribution. Retrieval-Augmented Generation (RAG) supplements an LLM with relevant information retrieved from sources outside the model's training parameters. Its basic logic combines a retrieval stage, in which relevant content is selected, with a generation stage, in which that content is used to support an answer. This can improve the currency and contextual appropriateness of responses and reduce certain hallucination risks, although it does not guarantee factual correctness (Lewis et al., 2020). Conventional implementations frequently operate over text passages indexed through vector embeddings.

GraphRAG extends this pattern by using relationally organized information as an important part of the retrieval context. Instead of treating each retrieved document fragment as a largely independent piece of evidence, a GraphRAG system can exploit entities, properties, paths, neighborhoods, subgraphs, or graph-derived representations. As the literature reviewed in the thesis indicates, GraphRAG does not denote one uniform architecture. It encompasses graph traversal, formal graph queries, graph-derived text, embeddings, graph-learning approaches, and hybrid combinations of these mechanisms. The critical distinction is that graph structure participates meaningfully in locating and organizing evidence, rather than serving merely as a visualization of documents (Han et al., 2025; Pan et al., 2024).

GraphRAG flows can leverage RDF Knowledge Graphs and SPARQL based queries. RDF expresses knowledge as subject–predicate–object statements and allows statements from different sources to be combined through shared identifiers and vocabularies. SPARQL provides a formal mechanism for selecting, constructing, and inspecting graph content. Together, they permit retrieval that is constrained by entity types and relationships, an important capability when questions concern organizational or enterprise dependencies rather than merely thematic relevance (W3C, 2014a; W3C, 2008).

The thesis gives particular attention to **Natural Language to SPARQL-based GraphRAG**, leveraging pipelines in which natural language inputs are transformed into SPARQL queries around entities of interest, leveraging RDF knowledge graphs, as well as to **SPARQL graph traversal-based retrieval,** leveraging pipelines in which entities of interest are selected from natural language inputs, followed by n-hop breadth-first neighborhood expansion for extracting relevant knowledge.

The thesis defines **Model-Driven GraphRAG** as a form of GraphRAG in which the underlying knowledge structures originate from a diagrammatic modeling effort. This definition shifts attention upstream, from how a knowledge graph is queried to how enterprise knowledge becomes a knowledge graph in the first place. Modeling environments act as entry points; domain-specific metamodels define the permissible concepts and relations; model transformation produces RDF structures; triplestores make them queryable; and retrieval mechanisms select relevant evidence for LLM-based interaction. The use of deterministic model-to-graph transformation is significant because important semantics may reside in element annotations, cross-model hyperlinks, and typed references that a visual interpretation of a diagram alone would not reliably recover.

2.2 Scientific Areas Converging to the Contribution

Method engineering provides the disciplinary foundation for constructing and adapting systematic ways of working rather than relying on ad hoc tools. Classical definitions regard a method as a coordinated arrangement of activities, rules, techniques, and development

products, while method engineering concerns the design, construction, and adaptation of those arrangements (Brinkkemper, 1996). **Situational Method Engineering** extends this approach by emphasizing that methods must be tailored to the characteristics of an application context rather than applied as universal prescriptions (Harmsen et al., 1994; Henderson-Sellers et al., 2014).

This orientation is especially important in enterprise knowledge work because the vocabulary of a manager describing a work system differs from that of a cybersecurity analyst analyzing threats and from that of a knowledge engineer creating RDF structures. A single generic editing interface would not necessarily make knowledge capture easier for all three groups. The thesis therefore treats methodological specificity as a design choice: the modeling language should fit users' existing conceptualizations, while the resulting knowledge should remain usable within a shared family of graph-oriented AI architectures.

The more specific discipline of **Modeling Method Engineering** concerns methods whose central artifacts are symbolic representations of selected aspects of reality. Following Karagiannis and Kühn (2002), a modeling method is composed of a **modeling language**, a **modeling procedure**, and **mechanisms or algorithms** operating on the resulting models. The modeling language includes syntax, semantics, and notation. The procedure guides the construction and use of models. Mechanisms and algorithms provide functions such as checking, transformation, querying, navigation, and other operations. The thesis builds on this established decomposition but extends its mechanisms toward RDF generation and AI-supported knowledge use.

The chapter positions **metamodeling** as a particularly important source of rigor. A metamodel defines which types of modeling elements, attributes, associations, and constraints are available. It therefore supports a controlled modeling vocabulary while preventing arbitrary or semantically inconsistent connections. This is useful for human modelers, who receive guidance through the modeling environment, and for downstream knowledge representation, where the type and meaning of an entity should not depend solely on a model's graphical appearance. Metamodels operate as mediators between domain conceptualization and formal knowledge structures.

The thesis draws a distinction between general-purpose modeling languages and **domain-specific modeling languages (DSMLs)**. General-purpose notations aim for broad applicability; DSMLs prioritize the concepts, operations, and constraints of a delimited domain or practice. Their advantage is not simply reduced complexity. Domain-specific symbols can improve communication with knowledgeable stakeholders, make modeling errors easier to detect, and retain relevant domain semantics that might otherwise be relegated to free-text annotations (Frank, 2011; Frank, 2013). Their limitation is that specificity must be engineered and maintained, and different domains require different conceptual choices.

Agile Modeling Method Engineering provides the basis for the iterative development cycles used throughout the thesis (Karagiannis, 2015). The AMME produce–use principle connects the design of a modeling method with its practical employment and evaluation. During conceptualization, relevant requirements are acquired, the metamodel and language are designed, notations are formalized, technical functionalities are developed, and the method is deployed. During use, modelers apply the resulting environment, generate models and derived artifacts, and provide evidence for further refinement. The **ADOxx** platform (BOC GmbH, n.d.) is employed as a metamodeling and implementation environment because it supports tailored visual languages, model repositories, scripting, and model-to-data transformations (Fill & Karagiannis, 2013).

Knowledge modeling is presented as a process of making relevant concepts, relationships, and contextual assumptions explicit. Here the thesis draws on the tradition of **Active Knowledge Modeling (AKM)**, which treats models as potentially operational artifacts used in situated enterprise activities rather than purely passive documentation (Lillehagen & Krogstie, 2008). The technical evolution of semantic-web standards and LLM-supported retrieval makes this aspiration more practicable: a model can be transformed into formal statements, queried during work, and used to inform generated explanations. This does not imply that the diagram, ontology, or graph captures every dimension of an enterprise; it makes a selected, governed representation available for reuse.

2.3 Application Areas for the Contribution

The first application domain is **RDF knowledge representation**. The chapter introduces the essential RDF approach to representing identifiable resources, their types, their properties, and their relations. Related standards, including RDFS and OWL, support schema-level descriptions and reasoning, while serializations such as Turtle, TriG, and JSON-LD provide machine-processable forms. The distinction between schema-oriented knowledge (often described as TBox) and instance-level assertions (ABox) is relevant to the design of GRAPHxx. A key requirement is to support knowledge graphs that need not begin with a complete ontology, thereby lowering the entry barrier for exploratory or educational applications.

The second application domain is **Work System Theory (WST)**. A work system is an arrangement in which human participants and/or machines perform processes and activities using information, technologies, and other resources to produce products or services for customers. The Work System Framework provides a managerial lens for understanding these arrangements, including their internal constituents and external context (Alter, 2006; Alter, 2013). Unlike highly technical architecture languages, WST offers terms that many managers and business stakeholders can understand. The research sees an opportunity to convert work

system descriptions from loosely structured templates into typed diagrams supporting decomposition, consistency checking, traceability, and semantic querying.

The third application domain concerns **process-centric cyberthreat–mitigation mapping**. The thesis combines business process notation, data-flow modeling, and threat-modeling taxonomies to describe not only how information systems operate but also where security concerns arise and which mitigations apply. BPMN represents the process context; Data Flow Diagrams (DFDs) represent system components, information exchanges, data stores, external entities, and trust boundaries; threat-modeling schemes such as STRIDE and LINDDUN classify relevant threats (Kohnfelder & Garg, 1999; Wuyts et al., 2020). The resulting combined perspective aims to connect security reasoning to business objectives and system designs, instead of leaving threats in disconnected lists.

The chapter also provides more details about **Nonaka's SECI knowledge conversion spiral** as a theoretical basis for understanding knowledge movement. In its original formulation, knowledge is developed through socialization, externalization, combination, and internalization, with ongoing interaction between tacit and explicit forms (Nonaka, 1994; Nonaka & Takeuchi, 1995). In the thesis, diagrammatic modeling offers a means of externalizing selected personal knowledge into shared conceptual representations, while model-to-RDF transformation and graph-based querying strengthen the combination of explicit knowledge. LLM-mediated access may help individuals interpret and internalize that knowledge in their work.

The distinction between these modes prevents an overly technological reading of knowledge management. Socialization depends on human interaction; externalization depends on what individuals can and choose to articulate; formal graph operations depend on the quality of this articulation; and internalization remains a human learning process. These foundations lead to the research's central positioning: **the value of Model-Driven GraphRAG rests on the cooperation of domain-sensitive human modeling, explicit semantic structures, and AI-based knowledge access**.

3 Problem Statement & Research Methodology

3.1 Problem Context: Knowledge Management Systems in the Age of Hybrid Intelligence & Research Questions

The third chapter defines the research problem in the context of enterprise knowledge management and Hybrid Intelligence. Following Maier (2007), a knowledge management system is understood as a socio-technical arrangement integrating knowledge resources, communication mechanisms, people, processes, and technologies. Its purpose extends beyond storing documents or data; it must preserve the contexts in which knowledge is meaningful and actionable. Enterprise knowledge is therefore heterogeneous by nature, and any attempt to exploit it through AI must address not only content acquisition but also semantic organization and provenance.

Hybrid Intelligence supplies an important normative position for this effort. It emphasizes collaborative arrangements in which human and artificial intelligence complement one another and learn through interaction rather than allowing technology to replace human contextual judgment (Dellermann, Ebel et al., 2019; Akata et al., 2020). The thesis aligns its design rationale with the principles of collaboration, adaptability, responsibility, and explainability. Enterprise modelers remain responsible for identifying relevant domain concepts and maintaining accurate representations; automated mechanisms transform and query those representations; LLMs turn retrieved evidence into accessible answers. This division of responsibility is especially valuable when AI outputs may influence managerial decisions or cybersecurity analysis.

The thesis organizes its inquiry around three research questions:

RQ1: How can tacit enterprise knowledge be captured by means of metamodels acting as domain-specific mediators in the knowledge transfers between enterprise stakeholders and retrieval-augmented AI systems?

RQ2: How can enterprise KG development be facilitated for enterprise modelers familiar with domain-specific conceptualizations and enterprise modeling, but lacking technical KG skills?

RQ3: How do traditional unstructured descriptions of enterprise content compare with DSML-based diagrammatic descriptions, in terms of LLM answering quality, using different retrieval strategies for model-based content?

The first question concerns the **nature of mediation**. Modelers do not enter raw triples into a graph database; they work through a familiar language whose metamodel carries domain-level constraints. The second concerns the **practical conditions of adoption**: a method only helps knowledge management if appropriate users can create, inspect, and maintain content without

unreasonable technical effort. The third concerns the **effects of representation and retrieval**: the thesis does not presume that knowledge graphs outperform text in every use case, but asks where explicit typed relationships provide an advantage and at what cost. These three dimensions justify evaluating the artifacts from methodological, usability, and technical perspectives.

3.2 Research Methodology: Design Science Research

Design Science Research (DSR) is adopted because the thesis seeks to develop and evaluate a purposeful solution to an identified class of problems. In contrast to research directed exclusively toward explanation or prediction, DSR constructs artifacts and produces knowledge about their suitability, utility, and design characteristics (Hevner, 2007; Wieringa, 2014). The principal artifact is the reusable modeling method blueprint; the three developed instance methods and their corresponding tools serve as operationalized and evaluated manifestations of that blueprint.

The research follows the recognizable activities of problem identification, objective definition, design and development, demonstration, evaluation, and communication (Peffers et al., 2006). These activities are iterative rather than strictly linear. Domain requirements influence modeling language design; prototype use reveals gaps in concepts or interactions; experimental results prompt revisions to graph conversion and retrieval; and findings from different instance methods inform the general blueprint. This design orientation enables the thesis to connect generalizable methodological knowledge with concrete working implementations.

The distinction between **design problems** and **knowledge questions**, emphasized by Wieringa (2014; Wieringa, 2016), is central. A design problem asks how a situation can be improved through an intervention, such as providing a modeling environment that supports accessible creation of AI-ready knowledge graphs. Knowledge questions ask whether certain design decisions achieve their intended effects, such as whether RDF transformation preserves meaning or whether ontology-guided retrieval improves answer quality. The study therefore treats the tools as treatments to be justified by evidence, not as self-validating software demonstrations.

The research also draws on the three-cycle view of DSR. Its **relevance cycle** is represented by enterprise knowledge needs and stakeholder requirements; its **rigor cycle** draws from prior theory, methods, modeling languages, and research on knowledge graphs and LLMs; and its **design cycle** iteratively constructs and evaluates artifacts (Hevner, 2007). This framing situates the three tool-building efforts within a common research agenda rather than describing three unrelated software projects.

3.3 The Hybrid Design Science – Agile Modeling Method Engineering Process

At the engineering level, the study employs **AMME** because modeling method artifacts require systematic decisions about language grammar, semantic meaning, notation, modeling procedures, and model-processing mechanisms (Karagiannis, 2015). The hybrid DSR–AMME process allows an overarching scientific cycle to contain multiple development iterations. The three main instantiations represent major design-science iterations, while refinements of the metamodels, user interfaces, serialization routines, and GraphRAG mechanisms form AMME sub-iterations within them.

The approach integrates theoretical grounding with rapid prototyping. Requirements are derived from the expected modeling tasks and the domain theories or standards involved. ADOxx is used to implement executable modeling languages and accompanying mechanisms, including content validation, navigation, generation, and external integrations. Demonstration takes place through scenarios and example models; evaluation then addresses different aspects of artifact quality. For WSF4ADO (Chiș et al., 2025b), these aspects include semantic preservation, modeling efficiency, scalability, competency-question support, and comparative retrieval. For CTM4ADO (Chiș et al., 2026), the emphasis extends to security-specific traceability and the effects of alternative graph retrieval strategies. For GRAPHxx (Chiș et al., 2025a), usability, feature adequacy, and graphical graph-building effort become particularly important.

According to the DSR template (Wieringa, 2016) the principal problem statement is formulated below:

Enhance **the capabilities of streamlining knowledge transfers in AI-ready organizations having an enterprise modeling culture** (problem context)

… by treating them with a knowledge capture approach based on diagrammatic modeling methods **(instantiated here for selected application areas in the Work Systems Design Method, the Process-centric Cyberthreat-mitigation Method and the Visual Knowledge Graph Building Method)** (nature of the artifact/treatment)

… so that the knowledge is structured and retrieved according to domain-specific conceptualizations without retraining AI systems on them, considering the scarcity and highly customizable nature of such content (artifact characteristics and services)

… in order to enable model-based human-AI collaboration in a Hybrid Intelligence setting, with a particular focus on model-based question answering
(stakeholder goals)

The thesis incorporates quality considerations from **SEQUAL**, which distinguishes physical, empirical, syntactic, semantic, pragmatic, social, and organizational dimensions of modeling quality (Krogstie et al., 2006). Rather than claiming that every experiment measures every dimension, the study links artifact requirements and evaluation procedures to relevant quality concerns. Metamodel constraints support syntactic validity; domain-grounded concepts support semantic quality; accessible interfaces address practical usability; and querying and GraphRAG integration contribute to pragmatic utility by making modeled knowledge actionable.

3.4 Conceptual Framing of the Contribution

The conceptual framing clarifies that the thesis operates at the intersection of enterprise modeling, knowledge management, semantic-web technology, and retrieval-augmented AI. The contribution is not a new foundational RDF standard, a novel general-purpose LLM, or a universal enterprise ontology. It is a **modeling method engineering blueprint that bridges diagrammatic descriptions and GraphRAG**, complemented by software artifacts and evaluations demonstrating that bridge under different domain conditions.

The proposed blueprint instantiates a collection of modeling methods builds a bridge towards Semantics-Driven Systems Engineering, coming from past metamodeling frameworks such as AMME.

The **Work Systems Design Instance Method** deployed in the WSF4ADO tool is an instance of the Work Systems Modeling Methods spectrum, which gained popularity in the research literature over the course of last years. This method takes input from the Work Systems Theory, previously discussed in this thesis, on the side of mapping enterprise knowledge on specific WST constructs in the modeling tool, building Work Systems Framework based diagrams, which are then serialized following RDF formats.

The **Process-centric Cyberthreat-mitigation Mapping Instance Method** deployed in the CTM4ADO tool, is an instance of Cyber Threat Modeling Methods, being a domain-specific application of conceptual modeling in the context of cybersecurity. This modeling component takes input from the RDF specification, BPMN modeling fragments and established threat modeling frameworks & methodologies (Shostack, 2014).

The **Visual Knowledge Graph Building Instance Method** is an instance of a larger spectrum of General-purpose Knowledge Modeling Methods, which may not be limited to RDF, as other standards such as OWL (W3C, 2012) are also part of that area. This modeling method conceptualizes a modeling language based on the RDF specification, as all the serialization of the modeling effort is done in TriG or Turtle formats, which are part of the RDF spectrum.

As a whole, this collection of modeling methods outputs model-driven knowledge graphs, which, then provide input for Semantics-Driven Software Engineering. All the instance methods, due to the outputting of KGs also provide input for model-driven GraphRAG. In this case, the focus is on graphs as content stores and graphs as subject matter experts (Ontotext, n.d.-b), leaving the graph as database flavor for future work.

On the evaluation side of the instance methods, the selected empirical strategies reflect both the potential and the limits of DSR evidence. Quantitative results include response correctness, RAGAS-based measures, semantic similarity, model creation effort, and execution time and cost. Other evidence comes from competency questions, feature comparisons, and user-oriented experiments. Each measure illuminates only part of the relevant quality space: a fast method need not be accurate, a highly relevant response need not be correct, and a semantically preserved representation need not be readily usable by every stakeholder category. The chapter thus establishes a balanced basis for interpreting later results and for distinguishing technological feasibility from broader enterprise adoption.

4 Contributions Overview

4.1 A Blueprint for Modeling Methods That Enable Model-Driven GraphRAG

The fourth chapter states the thesis's central methodological contribution. It adapts the established modeling method architecture of Karagiannis and Kühn (2002) to a new purpose: connecting diagram-based enterprise knowledge capture to knowledge graph-based LLM retrieval. The conventional components of a modeling method remain necessary, but they are augmented with explicit semantic grounding, machine-readable serialization, repository interoperability, and mechanisms for retrieving evidence for LLM-based question answering.

The **modeling language** supplies the human-facing concepts, relations, constraints, and notations suitable for a specific domain or practice. The **modeling procedure** specifies how those concepts are used in capturing, refining, and connecting enterprise knowledge. The **mechanisms and algorithms** extend beyond ordinary diagram editing: they manage model content, perform consistency checks, transform diagrams into RDF graph fragments, publish or export these fragments, and connect them to graph queries and LLM-based retrieval. In this interpretation, a diagram is both a human-readable representation and a formally transformable knowledge asset. Diagrams serve as both means of transforming knowledge from a tacit state to an explicit state and they also provide the basis for reaching the machine-oriented knowledge space where LLMs leverage diagrammatic knowledge expressed as RDF semantic graph structures.

The blueprint distinguishes a **human-oriented design space** from a **machine-oriented knowledge space** without treating them as disconnected. In the design space, domain specialists describe business operations, dependencies, cybersecurity concerns, or knowledge graph statements through visual objects and relations. In the knowledge space, those choices become identifiable RDF resources and statements with suitable types and properties. The transformation should be deterministic and driven by the source model's explicit structure and annotations. The knowledge graph can then be placed in a triplestore and accessed directly through SPARQL or indirectly through GraphRAG. By locating the transformation mechanism within the modeling method, the blueprint makes integration an intended feature of the method rather than an unrelated downstream data-engineering step.

The thesis also emphasizes **separation of concerns**. Users who understand a work-system framework should not have to write RDF triples to represent participants or processes. Cybersecurity specialists should be able to reason in terms of data flows, threats, and mitigations without being required to manipulate underlying graph schemas. Conversely, technical knowledge engineers may wish to control RDF identifiers, schema relationships, and serializations directly. The common architectural principles can accommodate these distinct

levels of abstraction through different instance methods, each developed from a domain-relevant conceptualization.

The blueprint's technological realization makes use of ADOxx underlying concepts and AdoScript for modeling and serialization, RDF repositories such as GraphDB for storage and semantic querying, Python components for orchestration, and LLMs for interpreting retrieved evidence. The thesis does not claim that the research contribution depends on one proprietary AI provider; different LLM endpoints could be substituted without changing the underlying design rationale. Its more durable contribution is the specification of a reusable knowledge flow from **domain-specific modeling to structured graph representation to evidence-based AI interaction**.

4.2 Repurposing SECI with a Hybrid Intelligence Model for Knowledge Modeling and Management

The second contribution develops a **Hybrid Intelligence adaptation of the SECI knowledge-conversion cycle**. In the classical model, socialization concerns tacit-to-tacit knowledge exchange, externalization articulates tacit knowledge into explicit forms, combination reorganizes explicit knowledge, and internalization concerns the acquisition of tacit understanding through engagement with explicit representations (Nonaka, 1994; Nonaka & Takeuchi, 1995). The thesis positions diagrams, RDF knowledge graphs, and LLM-based retrieval within these knowledge movements rather than considering AI solely as a tool for question answering.

In the proposed interpretation, **socialization** remains predominantly a human and enterprise activity. Discussions, interviews, workshops, and shared working experiences allow stakeholders to expose issues and negotiate understandings. The modeling methods can support these interactions through shared visual representations, but they do not replace social knowledge creation. **Externalization** is the primary entry point for the research: knowledgeable stakeholders translate aspects of their situated understanding into typed elements, annotations, and connections. The modeling tool then serializes those articulated relationships into a machine-readable graph. This is a two-part movement from implicit understanding to diagrams and from diagrams to explicit semantic data.

Combination is strengthened by the possibility of merging, connecting, querying, and selectively retrieving formalized knowledge. RDF identifiers, shared vocabularies, and graph patterns enable relationships to be investigated across models and, where appropriate, across enterprise boundaries. GraphRAG uses selected knowledge fragments as context for an LLM, creating new accessible descriptions and explanations from already explicit content. **Internalization** occurs when users interpret such explanations and integrate them into their

understanding or decisions. The thesis treats this step as an implication and prospective area of study rather than as evidence that learning outcomes have been comprehensively measured.

This argument is closely related to **Active Knowledge Modeling**, which calls for knowledge representations to be operational during work rather than simply archived (Lillehagen & Krogstie, 2008). Model-Driven GraphRAG gives this objective a concrete contemporary mechanism: enterprise models become sources for queries and generated answers. At the same time, the framework is deliberately human-centered. The graph contains articulated and formalized aspects of knowledge, not an exhaustive representation of tacit judgment. Its quality depends on competent modelers, appropriate concepts, validation, and enterprise maintenance.

The chapter therefore contributes two complementary elements. The **blueprint** describes how the required modeling and technical components can be engineered. The **SECI–Hybrid Intelligence framing** explains why those components matter to knowledge management and how humans and AI participate in an ongoing flow of interpretation, formalization, retrieval, and use. The three instance methods developed later in the thesis test the practicality of this combined vision from three different entry points.

5 Literature Review

5.1 Literature Related to System Design through the Work System Framework

The fifth chapter situates the developed artifacts against relevant literature and identifies the research gap at the intersection of enterprise modeling, diagrammatic knowledge capture and AI-enhanced knowledge retrieval. The work-system literature reveals a long-standing imbalance between the conceptual accessibility of the **Work System Framework** and the comparatively weak technological support traditionally available for it. WST offers a practical way to describe business work through processes and activities, participants, information, technologies, products and services, customers, and contextual elements. However, many applications rely on textual snapshot templates or lightly structured documents rather than interoperable modeling repositories (Alter, 2006; Alter & Bork, 2019; Bork & Alter, 2020).

Previous studies have investigated visual representations, modeling notations, and WST-related formalizations, establishing that the work-system perspective can be integrated into the broader tradition of conceptual modeling. The literature nevertheless leaves opportunities for typed, decomposable work-system representations that can be reused outside the original diagramming environment. The thesis therefore identifies a specific methodological need: WST-based descriptions should remain legible to managers while supporting automated consistency checks, tracing relationships across work systems, and machine-readable export for semantic querying. WSF4ADO addresses this need by treating work-system concepts as an explicit language rather than simply reproducing the original document form within a drawing application.

5.2 Literature Related to Process-centric “Secure by design” Methods

The review of cybersecurity literature centers on **secure-by-design reasoning**, particularly the relationships between business processes, data flows, information-system components, threats, and mitigation measures. Established threat-modeling approaches, including STRIDE and the tools that operationalize it, provide systematic ways to identify potential vulnerabilities and document protective measures (Kohnfelder & Garg, 1999). DFDs reveal data movement, processing, trust boundaries, and stores; BPMN reveals business activities and workflows. However, these perspectives are often managed separately, making it difficult to follow a security concern from an enterprise process to the technical data flow through which a threat could materialize.

The thesis reviews approaches to integrating process models with security requirements and risk analysis. Their contributions establish the value of relating security concerns to operational

architecture, but the reviewed landscape does not fully address a uniform, diagram-driven path to graph-based traceability and natural-language querying. Security tools may provide structured exports without providing a general semantic representation suitable for reusable SPARQL paths across process and data-flow models. Conversely, diagramming tools may offer rich visual interaction without implementing AI-ready knowledge integration. The CTM4ADO contribution is positioned as a response to this fragmentation, emphasizing established notations, extensible threat taxonomies, and explicit semantic links rather than proposing a new threat classification standard.

5.3 Literature Related to Knowledge Graph Building Environments

The literature on knowledge graph construction indicates that **creating** RDF-oriented knowledge is not equivalent to **visualizing** an existing graph. Knowledge engineers have access to mature ontological and semantic-web tools, notably Protégé, yet many interfaces are organized around formal ontology editing and presume familiarity with RDF/OWL terminology. Graph visualizers, on the other hand, may display existing triples or ontology structures but do not necessarily support intuitive authoring of semantically meaningful graph content. General-purpose drawing applications support rapid visual expression but do not guarantee that the relationships drawn correspond to well-formed, machine-processable graph statements.

This distinction motivates the low-code orientation of GRAPHxx. Its intended role is neither to replace a triplestore nor to reproduce the full capabilities of an advanced ontology editor. Instead, it seeks a productive middle ground in which a user can start with **things, types, relations, and data attributes**, progressively encounter RDF-specific choices, and still obtain useful serializations. The literature also supports the value of separating or combining schema-level and instance-level representations according to the task, rather than requiring all graph building to begin with a complete formal ontology. Educational use, prototyping, knowledge engineering, and structured-data generation for web applications offer complementary application contexts for such a method (Buchmann & Ghiran, 2023; Bock & Frank, 2021).

5.4 Literature Related to Model-Driven GraphRAG

A fourth literature stream concerns the convergence of **conceptual modeling, knowledge graphs, and LLMs**. Earlier model-transformation work, including CM2KG and Enterprise Architecture Knowledge Graph approaches, shows that structured diagrams can be translated into knowledge graphs and used for computational analysis (Smajević & Bork, 2021; Glaser et al., 2022). Research on semantics-driven systems engineering further argues that formal models can serve as semantic repositories for reasoning and intelligent systems rather than only as inputs for code generation (Buchmann et al., 2024). Related work explores LLM-assisted semantic mapping, prompts aligned with modeling languages, natural-language-to-model

generation, and process knowledge acquisition (Reitemeyer & Fill, 2024; Sandkuhl et al., 2026; Schinckus et al., 2026).

The thesis's claimed gap is more specific than the absence of conceptual-model-to-graph transformations or the absence of GraphRAG. Those fields already contain relevant solutions. The gap lies in a **systematic modeling-method approach** that deliberately joins domain-specific diagrammatic knowledge capture, deterministic generation of graph structures, and LLM-mediated retrieval over those structures. The reviewed approaches tend to emphasize one portion of this chain: model production, ontology representation, graph extraction, reasoning, or AI interaction. The three developed methods seek to integrate the chain and examine its effectiveness across distinct stakeholder groups.

A further lesson from the literature is that **retrieval representation does not determine retrieval quality on its own**. GraphRAG approaches vary considerably, and a graph can still produce inadequate context through poor entity grounding, incorrectly generated queries, excessive neighborhood expansion, or inappropriate serialization. Evaluations must therefore distinguish the capability of a knowledge representation from the effectiveness of the algorithm that selects and verbalizes evidence. This critical insight informs the controlled retrieval comparisons reported in the subsequent chapter and supports the thesis's rejection of the idea that more graph context is invariably better.

6 The Developed Instance Modeling Methods

The sixth chapter constitutes the empirical and engineering core of the thesis. It presents three instantiations of the overarching modeling-method blueprint, each beginning from an identifiable knowledge-work setting and proceeding through requirements, modeling language design, implementation, demonstrations, and evaluation. Together, the implementations show the different ways in which the **same method-engineering rationale** can produce AI-ready graphs from knowledge articulated by different stakeholder communities. The methods differ in their conceptual foundation and intended users; they share metamodel-based modeling, deterministic semantic transformation, and the aspiration to make diagrams operational for graph queries and LLM-based interaction.

6.1 Work Systems Design Instance Method

The first instance method addresses the representation and exploitation of **work-system knowledge** and it is implemented in the WSF4ADO tool (Chiș et al., 2025b). In traditional WST practice, managers and researchers may document work systems using snapshots organized around a set of familiar elements. Such descriptions are attractive because they avoid the complexity of comprehensive enterprise-architecture frameworks, but they are comparatively weak in formal semantics, cross-model traceability, and automatic reuse. The method developed in the thesis seeks to preserve the familiarity of WST while adding rigorous visual structures and a machine-readable knowledge layer. Its potential users include managers, business analysts, educators, and students who can reason about work systems but do not necessarily possess RDF or knowledge engineering expertise (Alter, 2006; Alter & Bork, 2020).

The method is guided by several interconnected requirements. A work system should be represented through a structured design aligned with the Work System Framework. Its constituent elements and their relations should be captured using accessible drag-and-drop modeling. Users should be able to navigate between an overall work-system view and more detailed representations, including links with familiar languages such as BPMN. The method should facilitate validation, querying in WST terminology, and the traceability of strategies, processes, resources, products, customers, and other elements across work-system boundaries. Finally, the same captured knowledge should be converted deterministically to RDF and made available to both SPARQL queries and GraphRAG interactions.

The methodological significance of these requirements is their **dual orientation**. The method must support users who want to understand a work system visually, but it must also preserve enough formal detail to support computational questions that would be difficult to answer from separate descriptive documents. This requirement rules out approaches based exclusively on screenshots or vision-based LLM interpretation. Important domain semantics can be stored in

element attributes or inter-model references that are not readily visible in a static image, whereas a metamodel-driven transformation can use those values directly.

The WSF4ADO language is grounded in the Work System Framework and supported by an ontology-oriented formalization of relevant WST concepts (Chiș, Ghiran & Alter, 2024). It supports descriptions of **work-system snapshots** and their associated constituents, including processes and activities, participants, customers, information, technologies, products and services, strategies, infrastructure, and environmental elements. Distinct internal and external viewpoints can be represented without forcing all relevant concepts into one overloaded visual space. Model relationships and annotations provide semantic links that remain available in the resulting graph even if the visual layout is reorganized.

A particularly important design choice is to treat the metamodel as a **domain mediation layer**. A business user creates a typed relationship between a process and a technology or between a product and a customer; the modeling method determines the corresponding RDF representation. The user does not need to memorize predicate identifiers or manually assemble triples. At the same time, more sophisticated modelers can use established model relations, controlled identifiers, and detailed annotations to make knowledge more precise. The approach supports a gradual movement from informal descriptions toward stronger semantics without requiring an immediate transition to formal ontology engineering.

Work-system decomposition adds a second dimension of value. A higher-level work system can be related to subordinate systems or linked activity models. The resulting knowledge graph can preserve relationships across these levels, allowing queries to connect enterprise objectives and customer value to more detailed operational elements. This addresses a common weakness of document-based descriptions: even when several snapshots are individually understandable, their relations are often implicit or scattered across files. In WSF4ADO, diagrams can be part of a **semantic network**, and cross-model navigation can become graph-based traceability.

The implementation uses the ADOxx modeling environment for authoring and tool functions. AdoScript mechanisms inspect model objects and relevant attributes, then produce RDF serializations that exclude graphical attributes with no domain-level meaning. The created graph structures can be published to an RDF repository, specifically GraphDB in the implementation described by the thesis. The approach incorporates concept and relation validation as well as usability mechanisms intended to reduce errors and unnecessary user interactions. It follows the wider aim of making work-system knowledge reusable outside the original modeling tool.

The method offers an interface for asking natural-language questions about modeled work systems. The question is linked to an RDF repository and processed through Python orchestration and LLM services. The thesis implements two main direct strategies within the

modeling environment. **Automated LLM-based SPARQL generation** provides the user question and WST ontology to an LLM, which generates a formal graph query, executes it, and may attempt to repair failures. **LLM-powered template-based SPARQL generation** first classifies the question into an established query category and then uses a corresponding template to guide the generation of a query. Both approaches use retrieved results to ground a final natural-language response.

The template-based strategy is important for the thesis's design argument. It embeds **domain knowledge into the retrieval procedure**, not only into the knowledge base. Templates restrict the space of possible queries and offer structural patterns already aligned with the WST ontology. This can make querying more reliable when the domain is known and representative question families can be specified in advance. Automated query generation offers greater flexibility, but its quality depends more directly on how well the language model understands the ontology and translates an individual question into a correct graph pattern.

The experimental assessment extends beyond these two options. To evaluate the role of explicit ontology knowledge and the contrast between graphical and textual content, the thesis compares five strategies: automated SPARQL, template-guided SPARQL, schema-aware SPARQL without the full ontology, naïve text-based RAG, and Microsoft GraphRAG operating over corresponding textual source material. The comparison is relevant because the question is not simply whether one tool can answer WST questions, but **which combinations of representation and retrieval** are effective under defined conditions.

The evaluation program begins with the modeling environment itself. In a synthetic scalability experiment, diagrams were populated with growing numbers of instances representing nine types of work-system constituents, with different densities of links. The test reached **2700 instances** in the no-link condition. Results indicated that the creation of modeling objects in the chosen platform became progressively more expensive, with observed quadratic-like growth in the tested setup. By contrast, RDF serialization scaled approximately with the number of nodes and links, indicating a more favorable, near-linear export behavior. The distinction identifies an important architectural bottleneck: large-scale editing is constrained more by the underlying visual platform than by the semantic serialization algorithm.

User-oriented experiments investigated whether types and relations could be manipulated efficiently. One experiment assessed the effect of showing type information during link creation. In very small tasks, the additional interface information introduced some overhead. As task complexity increased, however, persistent type cues substantially reduced cognitive search and execution time, with reported speed improvements in the approximate range of **2.25 to 3.66 times**. A second experiment evaluated automation for creating appropriate connections between a process and other work-system constituents. It showed that model-aware actions could substantially reduce the repeated interaction effort of manually establishing many

relations. These findings demonstrate that usability is not separate from semantic rigor: **well-designed type-aware interactions** can make formal modeling more practical rather than simply adding constraints.

Another experiment evaluated **semantic preservation** during a transformation round trip. Ten work-system snapshots were modeled, serialized into knowledge graphs, and transformed back into textual descriptions using different LLMs. BERTScore was then used to compare the reconstructed descriptions with the originals. High-performing models achieved macro-average F1 results around **0.99**; the tested models' macro averages ranged from approximately **0.94 to 0.99**. These results suggest that the graph representation retained the essential meaning of the source snapshots under the tested conditions. The thesis also identifies difficult cases, such as dual-role entities, reused labels, and conditional branching, where semantic reconstruction becomes more ambiguous, especially for weaker language models.

Competency questions provided another form of evidence by examining whether the resulting graphs could support **relationship tracing, aggregation, inference, and governance analysis**. The examples included linking strategic intentions to operational activities and customer value, deriving classifications for critical infrastructure, assessing relations among agents participating in a shared system, and tracing responsibilities across work-system boundaries. These inquiries require information to be combined across elements and models rather than found in one isolated textual sentence. The experiments demonstrate the expressive potential of the modeled knowledge structures, although they do not by themselves establish universal industrial performance.

The most distinctive quantitative finding for WSF4ADO concerns the retrieval strategy used to answer questions about work-system relationships. Under an LLM-as-judge evaluation, **template-guided SPARQL generation achieved 86.97% fully correct answers**, followed by automated LLM-based SPARQL at **65.76%**, schema-aware SPARQL at **44.85%**, Microsoft GraphRAG at **32.73%**, and naïve text-based RAG at **28.48%**. The comparison indicates a substantial advantage for structurally guided retrieval when questions depend on known domain relationships. It also shows that an RDF graph alone is insufficient: the much weaker outcome of schema-aware generation illustrates the value of the **complete domain ontology and established query patterns**.

The runtime and cost results introduce a practical qualification. Naïve RAG was the fastest and least expensive strategy at approximately **1.64 seconds and USD 0.005 per question**. Template-guided SPARQL averaged approximately **5.90 seconds and USD 0.021**, while automated SPARQL averaged **4.83 seconds and USD 0.016**. Thus, stronger correctness comes with an operational overhead. The thesis interprets this overhead as potentially acceptable for enterprise questions where an incorrect relational answer would be more costly than modest

additional retrieval time or API usage. The appropriate trade-off would nevertheless depend on the frequency, criticality, and complexity of the intended questions.

RAGAS-based metrics refine the interpretation. Template-guided SPARQL achieved a particularly balanced profile, with reported **faithfulness of 0.850**, **context recall of 0.825**, and **retrieval F1 of 0.819**. Microsoft GraphRAG, despite lower full correctness, achieved high measured answer relevancy and context precision. This apparent discrepancy is instructive: retrieved material can be on-topic and an answer can sound appropriate without fully reconstructing the exact relationships required by the ground truth. The text-based approaches were especially vulnerable on questions requiring aggregation, multi-model joins, or traceability through explicit relationships.

Cosine-similarity evaluation showed a broadly consistent but less decisive pattern. Template-guided SPARQL yielded the highest reported score (**0.765**), followed by automated SPARQL (**0.710**), whereas naïve RAG scored **0.505**. The thesis explicitly treats this measure as supplementary because semantic similarity between two answers does not guarantee factual correctness. The evaluation therefore supports a nuanced conclusion: **model-derived RDF combined with domain-informed SPARQL retrieval is particularly effective for structurally demanding work-system questions**, while simpler, descriptive questions may not justify the same retrieval complexity.

WSF4ADO shows how a familiar managerial conceptualization can be transformed into a formal, interoperable knowledge resource without asking every stakeholder to become a semantic-web specialist. It extends the original WST practice through diagrams that preserve typed relationships, decomposition, and multi-model traceability; it makes those diagrams machine-readable through RDF; and it demonstrates how domain-grounded retrieval can improve LLM question answering. The evaluation also identifies limitations of the solution, including platform-level creation costs, incomplete capture of tacit understanding, and sensitivity to the retrieval strategy and language model. Its principal contribution is a **practice-specific modeling method whose knowledge products can participate directly in GraphRAG**.

6.2 Process-centric Cyberthreat-mitigation Mapping Method

The second instance method applies the same knowledge-streamlining principles to **cybersecurity analysis**, focusing on the relationships among business processes, technical data flows, potential cyberthreats, and mitigating controls. It is implemented in the CTM4ADO tool (Chiș et al., 2026). Cybersecurity decisions frequently require analysts to move across abstraction levels: a business activity relies on particular technical processes; those processes exchange information across trust boundaries; specific components and flows are exposed to threats; and mitigating actions need to be matched to the identified risks. The CTM4ADO

method is designed to preserve these dependencies in visual representations and in the knowledge graphs derived from them.

Its motivating problem is therefore not limited to drawing a threat diagram. It is to support security-by-design analysis and the audit of existing systems while keeping **traceability to operational business context**. The method is intended for cybersecurity specialists and other stakeholders who already work with BPMN, DFDs, and threat taxonomies. It aims to retain familiar modeling grammars, offer a low-friction modeling procedure, connect otherwise separate perspectives, provide knowledge-graph generation and querying, and expose those graphs to natural-language AI services. Additional requirements concern tracing threats and mitigations across process and data-flow elements, aggregating security information, supporting derived relationships, and calculating exposure-oriented scores.

The method takes a **domain-specific** rather than a technology-specific starting point. Security analysts should be able to express their knowledge in terms of processes, data stores, external entities, trust boundaries, threat categories, mitigations, and relevant relations. The RDF layer provides interoperability and semantics but is not the primary user-facing jargon. This mirrors the thesis's wider claim that enterprise knowledge should be represented using the conceptual structures familiar to the relevant knowing subjects, rather than requiring those users to adopt graph database terminology in order to articulate their knowledge.

The CTM4ADO language combines several complementary perspectives. **BPMN** provides the process model and permits activities to be related to lower-level technical descriptions. **Data Flow Diagrams** identify processing elements, data stores, external entities, information exchanges, and trust boundaries. Dedicated **threat–mitigation models** capture security concerns and the controls intended to address them, while additional categorizations enable the use of standard or customized taxonomies. The environment supports linked BPMN process diagrams, DFDs, threat–mitigation diagrams, and threat–mitigation categorization diagrams.

A significant feature is the connection between a BPMN subprocess and a corresponding DFD representation. This allows analysts to inspect the information-system implementation of a business activity without losing its organizational purpose. Conversely, when a threat is associated with a technical component or information flow, that relationship can be followed toward relevant business activities. The method therefore treats cross-model linking as **semantic traceability**, not merely a navigation convenience. When diagrams are serialized, the links become formal relationships available to graph queries.

The modeling language also allows users to create or reuse threat and mitigation taxonomies, including approaches such as STRIDE and LINDDUN (Kohnfelder & Garg, 1999; Wuyts et al., 2020). Typed data-flow elements can be annotated with susceptible threats, possible and applied mitigations, severity-related information, and relevant technical details. Grouping

constructs are used where they make the visual interaction manageable, but not every grouping mechanism needs to remain as a first-class entity in the RDF output. Some visual helper constructs are converted into direct semantic relationships during serialization. This is an example of the design principle that the **semantic graph should reflect domain meaning, not incidental graphical layout**.

The technical architecture uses ADOxx as the visual environment and AdoScript for model content management, validation, and deterministic RDF serialization. The resulting Turtle or TriG content can be transferred to GraphDB. Python orchestration and LLM services provide the GraphRAG interaction, while internal scripting supports operations such as security exposure scoring that do not require generative AI. The scoring mechanism matches the threats associated with DFD elements against applicable mitigation possibilities, enabling basic auditing functions while remaining compatible with different threat catalogs. This separation is appropriate: deterministic model processing is used where the necessary rules are explicit; probabilistic language-model operations are reserved for tasks such as query interpretation and explanation.

CTM4ADO provides a rich setting for investigating GraphRAG retrieval because security questions often concern relations several steps away from a named component. A question may refer to a processing element but ask about its data flows, threats, affected assets, trust boundaries, or mitigations. The relevant information can be connected through paths of varying length, making the size and composition of the retrieved subgraph a critical design variable.

The thesis implements **fixed-radius graph traversal**, in which entity identification is followed by breadth-first expansion to a selected number of hops. SPARQL queries retrieve surrounding triples, including relevant blank-node properties, and the assembled subgraph is supplied as context to an LLM. The evaluated depths include one, two, three, and four hops. This family of strategies is relatively predictable, but its central weakness is that radius-based expansion may retrieve many statements that are structurally nearby yet irrelevant to the actual question.

A second strategy, **dynamic hops**, allows the LLM to determine an appropriate expansion depth based on question interpretation. A third strategy, **ontology traversal**, uses the graph schema to select semantically promising predicates and follow paths toward relevant evidence. Unlike fixed-radius retrieval, ontology-guided traversal attempts to privilege relations that matter to the question rather than treating all neighboring edges as equally informative. Its potential benefit is a more targeted retrieval context; its risk is that an incorrect early path choice can prevent recovery of necessary evidence altogether. A **naïve textual RAG** baseline and, in a second experiment, retrieval over **OWASP Threat Dragon JSON exports** provide points of comparison with other forms of knowledge organization.

This range of strategies allows the research to examine a question with implications beyond cybersecurity: **How much graph context is enough?** A deeper neighborhood increases the amount of potentially available evidence, but it also increases ambiguity, latency, token use, and the difficulty of grounding an answer in the correct subset of facts. The results consequently bear on the design of GraphRAG systems wherever large, interconnected enterprise models are used as knowledge sources.

The method is demonstrated with cybersecurity scenarios in which processes and technical information flows are connected with threats and mitigations. One principal evaluation employs an **e-commerce scenario**, together with question categories covering direct relations and more complex structural paths. The goal is to assess graph retrieval strategies against an equivalent textual baseline, using correctness judgments, runtime and cost measures, RAGAS-based quality indicators, and semantic similarity to reference answers.

In this first experiment, **ontology traversal achieved the highest fully correct answer rate, at 73.94%**. The fixed **two-hop** strategy achieved **61.82%**, three- and four-hop strategies each achieved **61.21%**, naïve RAG achieved **57.58%**, dynamic hops achieved **56.36%**, and one-hop retrieval achieved **50.30%**. These percentages must be read alongside their error profiles. Ontology traversal had relatively few partially correct answers but a substantial proportion of outright incorrect answers, indicating a tendency to either reach the needed path or fail decisively. Greater fixed traversal depth reduced the share of outright incorrect answers but increased partial answers and the amount of context supplied to the generator.

Time and cost further differentiate the methods. Naïve RAG remained relatively inexpensive, averaging approximately **2.70 seconds and USD 0.005 per question**. Ontology traversal averaged **5.98 seconds and USD 0.016**, while fixed two-hop traversal averaged **5.71 seconds and USD 0.056**. Three- and four-hop traversal imposed considerably greater costs, reflecting the growth of retrieved neighborhoods. The results support ontology traversal as an effective strategy when its path-selection assumptions hold, but they also support **two hops as a comparatively stable default** when accuracy, robustness, and bounded context need to be balanced.

RAGAS results introduce a necessary distinction between retrieval breadth and grounded generation. Ontology traversal obtained relatively strong faithfulness (**0.871**) and context recall (**0.770**) in the first experiment. Two hops produced high answer relevancy (**0.885**) but lower faithfulness (**0.656**), while three and four hops exhibited further drops in faithfulness to **0.427** and **0.396**, respectively. This non-monotonic pattern is one of the chapter's most interesting findings. Retrieving additional relationships may provide more evidence in principle, yet make it harder for the language model to isolate the relevant facts and construct a faithful answer.

A second experiment uses a **medical-sector threat-modeling scenario** and compares graph retrieval with an adapted retrieval pipeline over **OWASP Threat Dragon JSON**. The experimental cases are designed to be reasonably compatible with the comparator's supported concepts, so the evaluation is not simply a test of missing features. The JSON baseline requires a dedicated parser and entity-indexing mechanism, a practical implementation cost not reflected in per-query API expenditure. The experiment therefore contrasts the capability of distinct knowledge representations and their retrieval procedures within a relatively comparable task setting.

In this second evaluation, **two-hop graph retrieval achieved 82.54% fully correct answers, 17.46% partially correct answers, and no incorrect answers**. Dynamic hops achieved **71.43%** full correctness, ontology traversal **65.08%**, and the Threat Dragon JSON baseline **66.67%**. Two-hop retrieval also achieved particularly strong RAGAS results, including **faithfulness of 0.969**, **context recall of 0.984**, **context precision of 1.000**, and **generation F1 of 0.957**. The graph-based representation therefore showed a meaningful advantage on this task, including the preservation of relationships around trust boundaries and structured threat–mitigation associations.

The JSON baseline remained inexpensive, averaging approximately **USD 0.007 per question**, compared with **USD 0.021** for two-hop graph retrieval. Nevertheless, the latter was faster on average in the reported experiment (**3.56 seconds**, compared with **3.91 seconds** for the JSON-based approach). The conclusion is not that RDF retrieval is always cheaper or superior under all conditions. Rather, explicit semantic relationships can offer a substantial advantage for security questions that require structured and multi-step traceability, while the most appropriate retrieval strategy depends on the model and task.

The thesis dedicated investigation space to why faithfulness can decline even when additional traversal steps make more supporting triples available. In a diagnostic analysis of **55 questions**, the median serialized retrieval context grew from approximately **800 tokens** at one hop to **24200** at two hops, **59800** at three hops, and **78100** at four hops. Mean measured faithfulness fell from **0.770** to **0.416** across that range. At shallow depths, a significant number of answers were refusals or abstentions because necessary evidence had not been reached; these refusals could receive favorable faithfulness scores even though they did not answer the question.

At deeper levels, refusals became less frequent, but the retrieved subgraphs included many related yet distracting entities and relations. The subsequent errors were differentiated between failures in generation, where the answer did not correctly use the available evidence, and limitations of the LLM-based evaluator when processing very long contexts. The analysis connects these effects to published observations about distraction by relevant-looking evidence and the reduced usability of information buried within long contexts (Shi et al., 2023; Liu et al., 2024). Importantly, **greater evidence availability is not equivalent to better evidence**

use. This distinction limits simplistic interpretations of raw context recall or averaged faithfulness.

The main contribution of CTM4ADO is consequently twofold. First, it demonstrates a practical method for creating **semantically connected cybersecurity knowledge** from domain-appropriate diagrams and taxonomies. Second, it provides empirical evidence that **GraphRAG retrieval must be matched to the structure of the question**. Ontology-aware paths can be accurate and cost-effective but brittle; shallow retrieval is efficient but may miss essential relations; deep traversal reaches more knowledge but may overwhelm the generator. These findings motivate hybrid and adaptive retrieval mechanisms rather than a universal fixed-hop setting.

6.3 Visual Knowledge Graph Building Instance Method

The third instance method is implemented in the **GRAPHxx** tool (Chiș et al., 2025a), a visual environment for building general RDF-oriented knowledge graphs. Unlike WSF4ADO and CTM4ADO, its primary conceptualization is not a domain theory or professional practice but the **RDF knowledge representation model** itself. Its scope is therefore technology-specific: it offers an intuitive authoring surface for knowledge structures that may serve very different business, educational, and technical purposes. This generality makes it a complementary proof of the blueprint's flexibility.

The motivation is the gap between the ease of sketching a graph and the effort required to create a correct, reusable semantic graph in specialist tools. An expert can write Turtle or edit an ontology in Protégé, but that expertise should not be a prerequisite for expressing basic relationships among identifiable entities. Conversely, producing a diagram in a conventional presentation application does not create a usable RDF knowledge asset. GRAPHxx aims to combine **free-form visual expression with explicit and exportable semantics**, following a low-code philosophy in which common tasks are available with little exposure to standard-specific complexity (Bock & Frank, 2021; Chiș et al., 2025a).

Requirements were informed by educational experience, knowledge engineering practice, and discussions with practitioners such as SEO specialists. The tool should support simple drag-and-drop construction; graphs mixing schema and content; optional rather than mandatory schema-first development; flexible display and hiding of details; terminology that does not presuppose mastery of OWL; multiple serialization formats; and interoperability with triplestores and LLM services. The intended experience is progressive: basic graph representations should be straightforward, while advanced RDF constructs remain available when appropriate.

The GRAPHxx metamodel expresses familiar ideas such as **things (instances), thing types (classes), properties, relations, and values**. It supports separate but permeable spaces for

schema-oriented knowledge and content-oriented assertions. Graph Schema emphasizes classes, properties, and their conceptual relationships; Graph Content emphasizes resources, literals, and assertions. Users can nonetheless mix relevant constructs or connect them across diagrams, avoiding an inflexible TBOX/ABOX separation. This is particularly useful when teaching RDF principles, prototyping an unfamiliar domain, or creating a small graph without first defining a complete ontology.

The visual language uses approachable names for formal RDF relationships. Constructs corresponding to *rdf:type*, domain and range, and subclass or subproperty relationships are represented in terms such as **IsOfType**, **From**, **To**, and **IsAKindOf**. Individual identifiers, typed literal values, blank-node-like helper constructs, and ordinary entity relations allow users to build meaningful graph patterns. Appropriate identifiers and namespaces can be entered explicitly, selected from available vocabularies, or generated by the tool where necessary. This approach retains formal interoperability while reducing unnecessary early exposure to syntax and terminology.

Several usability mechanisms embody the method's design assumptions. Prefixes and local identifiers can be managed through model attributes and reusable selections. Types and data properties can be represented as arrows or as annotations, depending on visual complexity. Domain-specific icons can be attached to otherwise minimalist nodes. Containers support grouped or ordered structures; helper nodes allow nested RDF patterns; and visual objects can be linked across diagrams. These decisions are grounded in the trade-off between **graphical economy and semantic transparency** articulated by Moody (2009): diagrams should remain manageable while providing enough differentiation for users to interpret their content.

The design also respects the difference between **presentation constructs** and **semantic statements**. Layout, shape placement, and interface aids should not inadvertently become domain claims in the exported graph. Serialization scripts transform the model's substantive nodes, relations, identities, hyperlinks, and annotations into RDF structures while translating simplified user-facing concepts into standard predicates when relevant. This makes the graph transferable rather than locking it to the editor's internal representation.

GRAPHxx supports **Turtle, TriG, and JSON-LD serialization**, selected according to downstream needs. Turtle and TriG can be used with RDF triplestores and GraphRAG systems; JSON-LD provides a particularly accessible export for structured web content. A triplestore such as GraphDB handles storage, indexing, reasoning, and SPARQL querying, while the editor supplies the visual authoring layer. Additional Python and LLM-based components allow selected graph content to be retrieved and used in natural-language question answering. The architecture therefore deliberately separates authoring from scalable graph processing.

Two demonstration scenarios illustrate its range. In a **GraphRAG scenario**, visualized and linked graph fragments become explicit knowledge that can be published and queried, with appropriate subgraphs supplied to an LLM. This supports a human-to-graph-to-language-model flow similar to the other instance methods, but without constraining the content to work systems or cybersecurity. In a **semantic SEO scenario**, a user creates a graph of entities, types, and properties corresponding to structured website information and obtains JSON-LD suitable for embedding into an HTML page. This demonstrates that the same modeling principles support not only AI retrieval but also established machine-readable web publishing practices.

The two scenarios emphasize a distinctive advantage of the method: **one semantic authoring operation can serve several destinations**. A diagram constructed for understanding or explanation can also become a triplestore input, a machine-readable web artifact, or a source for LLM-supported interaction. The generality is enabled by RDF interoperability; the usability is supported by a DSML engineered specifically for graph authoring.

The thesis evaluates GRAPHxx using tasks that compare the creation of equivalent graph content through several means: GRAPHxx modeling, free drawing in PowerPoint, direct editing of Turtle or JSON-LD, and use of the ontology editor Protégé. The protocol measures task completion time, an interaction-based effort score, errors, and qualitative observations. The reported group included **seven educators**, four familiar with RDF and three unfamiliar with RDF but experienced in diagramming. Tasks requiring direct RDF serialization or Protégé were assigned only to the knowledgeable participants, reflecting the specialized knowledge needed by those approaches.

The results indicate that visual semantic authoring can reduce important forms of friction, particularly when compared with a general-purpose drawing program or direct serialization. Graphical constraints prevent some mistakes that are possible when writing syntax manually, while the tool allows users to avoid repeating low-level namespace and RDF statements. The comparison also reveals qualifications. Prior experience matters: RDF-knowledgeable participants can perform technical tasks more efficiently, and GRAPHxx still incurs interface overhead from opening annotation sheets, selecting relationships, and replicating recurring patterns. Familiarity with PowerPoint or Protégé influences measured times, so the results are more appropriately interpreted as formative design evidence than as a definitive population-wide ranking of tools.

The evaluation feeds directly into design refinements. The language evolved toward simpler terminology and less rigid separation between schema and content. The next iteration should improve bulk duplication of graph patterns, grouped object selection, and other interactions that generate repetitive effort. The methodology also points to broader assessments with novice and industrial users, so that the intended **citizen-oriented knowledge graph development** claim can be evaluated more convincingly.

The feature-oriented review positions GRAPHxx between several established tool categories. Domain-specific model-to-KG converters may generate valid graphs but bind users to a predefined domain conceptualization. Ontology editors provide sophisticated semantic capabilities but may emphasize ontology-first workflows and technical terminology. RDF visualizers support inspection of knowledge created elsewhere, while general drawing tools do not guarantee formal RDF semantics. GRAPHxx attempts to combine **graphically accessible authoring, metamodel-based validity, semantic serialization, and downstream knowledge reuse** within one method.

The method also enables some structured querying within the ADOxx environment through its query language, including inquiries that follow model relations and inter-model links. More extensive querying and inference remain the responsibility of the RDF repository. This allocation of responsibilities demonstrates architectural restraint: the modeling environment is not presented as a replacement for scalable graph infrastructure but as a user-friendly way to populate and manage interoperable knowledge structures.

Synthetic performance tests assessed the limits of representing large graph examples in the chosen metamodeling environment. One test created up to **5000 ThingIdentifier nodes** in a diagram, with population times increasing markedly beyond approximately **3000 objects**. Another generated up to **1500 ThingIdentifier nodes with two relations per node**, yielding a total of **4500 modeling constructs**; at the upper end, diagram population required substantial time whereas RDF export remained comparatively short. A further test increased the number of data attributes per node and found that export work rose with the number of RDF statements. The conclusion resembles that of WSF4ADO: **semantic serialization is technically feasible at useful illustrative scales, while large-scale visual object creation can be the limiting factor**.

The evaluations are subject to several validity limitations. The sample of users is small and not representative of all possible enterprise modelers. Some participants already possessed relevant skills, while long-term learning effects were not assessed. The complexity and diversity of tested graph examples were limited, and the interaction-based effort measure could be refined to distinguish classes of error. The thesis accordingly places GRAPHxx at an advanced but still evolving stage of the Design Science process, where evidence supports the direction of the method and informs a substantial future-development agenda.

GRAPHxx demonstrates that a model-driven knowledge graph building method can retain RDF interoperability while offering an authoring experience closer to graphical idea organization than to manual triple construction. Its distinguishing contribution is the **translation between accessible visual concepts and formal graph semantics**, with flexibility for mixed schema and instance content. By supporting both RDF publishing and LLM-oriented knowledge flows, the tool makes the general Model-Driven GraphRAG blueprint applicable

beyond one specific enterprise discipline. The resulting evidence is strongest for functional feasibility, progressive usability improvements, and performance within the examined modeling scales; claims of broad adoption and universal efficiency require additional evaluation.

7 Discussions and Implications

7.1 Summary

The seventh chapter interprets the research findings at the level of the overall methodological contribution rather than merely recounting the functionality of the three tools. Its central argument is that **enterprise modeling can be repositioned as an upstream component of AI-supported knowledge management**. Enterprise diagrams become more valuable when their concepts, relationships, and annotations can be transformed into interoperable semantic representations and reused during retrieval. WSF4ADO, CTM4ADO, and GRAPHxx demonstrate this principle in different contexts, from managerial work-system descriptions through cybersecurity analysis to general RDF knowledge creation.

The chapter relates the resulting artifacts to **Active Knowledge Modeling**, emphasizing that models should be available for computational operations during knowledge work. In the three cases, activation is achieved by moving from visual modeling to RDF structures that can support queries, reasoning, and natural-language interaction. The methods also address Hybrid Intelligence characteristics: human-centered augmentation, common semantic representations, collaborative use, and potentially more explainable responses. The resulting knowledge infrastructures remain subject to human oversight, particularly because the completeness of the modeled content and the relevance of its interpretations cannot be established by technical conversion alone.

7.2 Discussion of the Research Questions

RQ1: How can tacit enterprise knowledge be captured by means of metamodels acting as domain-specific mediators in the knowledge transfers between enterprise stakeholders and retrieval-augmented AI systems?

The research supports a two-stage account of how enterprise knowledge may become machine-processable. First, domain experts articulate selected aspects of their understanding through diagrams whose concepts and connections are constrained by metamodels. Second, model-processing algorithms transform the resulting diagrams into RDF knowledge graphs that preserve the modeled semantic relations. WSF4ADO makes WST concepts accessible to business-oriented users; CTM4ADO makes technical-security relations available in process and data-flow terms; GRAPHxx permits more generic graph authoring. Metamodels function both as **guidance for expression** and as **formal mediation rules** that maintain continuity between what users mean and what systems retrieve.

The findings do not imply that tacit knowledge can be exhaustively converted into explicit statements. Human experience, contextual judgment, situated practice, and assumptions that

remain unarticulated are outside the reach of automatic serialization. What the methods provide is a disciplined pathway for **capturing and formalizing expressible aspects** of organizational understanding. The modeler's interpretation remains indispensable, and the resulting graph should be treated as a maintained representation of knowledge rather than as a complete substitute for domain expertise.

RQ2: How can enterprise KG development be facilitated for enterprise modelers familiar with domain-specific conceptualizations and enterprise modeling, but lacking technical KG skills?

The thesis suggests that lowering the adoption barrier requires more than replacing code with boxes and arrows. Five connected design properties matter: familiar domain-specific concepts, metamodel-based constraints, visual notations and interactions, automated semantic transformations, and infrastructure interoperability that hides unnecessary technical operations. Managers can work with work-system concepts, cybersecurity specialists with threats and data flows, and general knowledge modelers with intuitive things-and-relations concepts. Technical complexity is shifted from the user interaction into the engineered method, while advanced controls remain available when needed. This design is not entirely independent of user expertise, but it makes RDF knowledge creation possible through competencies users already possess.

RQ3: How do traditional unstructured descriptions of enterprise content compare with DSML-based diagrammatic descriptions, in terms of LLM answering quality, using different retrieval strategies for model-based content?

The evidence favors model-derived graph representations particularly when questions require explicit relationships, structural joins, cross-model dependencies, or multiple-step traceability. WSF4ADO's template-guided SPARQL approach produced a higher proportion of fully correct answers than its textual RAG comparators. CTM4ADO likewise showed that graph-based retrieval can support accurate security explanations, including when compared with retrieval from a structured JSON export. However, the results also warn against a blanket assertion that GraphRAG is always superior. Text retrieval may remain efficient and adequate for simple descriptive questions. Graph retrieval can fail through incorrect entity grounding, poorly chosen paths, incomplete ontology assumptions, or excessive context expansion. The appropriate claim is **conditional superiority in knowledge-intensive relational tasks**, not universal replacement of text-based RAG.

7.3 Discussion of the Main Artifact Contribution

The first set of implications concerns **knowledge management and the SECI framework**. Conceptual modeling is presented as a means of supporting externalization, while model-to-RDF transformation and graph integration enhance explicit-to-explicit knowledge

combination. LLM-mediated retrieval can subsequently provide accessible content for human interpretation and possible internalization. This offers a contemporary account of how enterprise knowledge may pass through visual, semantic, and textual forms during AI-supported work. The thesis does not present a complete causal evaluation of enterprise learning through all four SECI modes; rather, it contributes an interpretation of the role that engineered modeling methods may play in those modes (Nonaka, 1994; Nonaka & von Krogh, 2009), with a particular focus on Externalization and Combination.

A second set of implications concerns **Hybrid Intelligence**. Domain experts remain the primary sources of situated knowledge and retain responsibility for deciding what relationships are accurate, relevant, and sufficiently complete. The technical system enables formalization and retrieval, while the LLM translates selected evidence into natural-language answers. This division of labor creates the possibility of traceable AI-supported reasoning grounded in explicit models. It may improve the ability to inspect a recommendation's underlying assumptions, although graph grounding alone does not remove LLM hallucinations or guarantee responsible use. Hybrid Intelligence therefore depends as much on governance and validation as on architecture.

A third set of implications concerns **Modeling Method Engineering**. Traditionally, modeling methods emphasize communication, analysis, documentation, simulation, or code and configuration generation. The thesis adds a further role: methods can produce **AI-consumable semantic knowledge infrastructures**. Under this interpretation, semantic export, ontology mappings, graph retrieval, and LLM interactions are not incidental add-ons but legitimate modeling-method mechanisms. The contribution is thus reusable methodological design knowledge for an emerging class of modeling environments situated between human knowledge capture and generative AI.

7.4 Discussion of Evaluation Results

For **enterprise knowledge management**, the main practical lesson is to treat maintained diagrams as reusable knowledge assets. Enterprises frequently invest in process maps, architecture descriptions, system designs, and specialist domain models. Their value is reduced when they remain isolated documents. Semantic transformation permits controlled querying and integration, allowing portions of that already-explicit knowledge to be reused in new information services. However, enterprises must maintain the accuracy and currency of their models, otherwise AI services will simply make outdated assumptions easier to retrieve.

For **managers and work-system practitioners**, the thesis indicates that structured work-system representations can support analytical questions about strategies, resources, processes, and value delivery that are awkward to answer from separate snapshot templates. For **cybersecurity analysts**, the developed method offers the possibility of linking security

exposures to business processes, trust boundaries, technical components, and applied mitigations. For **knowledge engineers**, the GRAPHxx results suggest a way to reduce reliance on manual RDF syntax for early knowledge capture, teaching, prototyping, and stakeholder collaboration. Across all three roles, the value proposition is not primarily a visually attractive interface; it is the conversion of human modeling decisions into **semantic objects that remain queryable outside the original editor**.

Architecturally, the findings emphasize the need to separate graph construction from graph retrieval and text generation, and to evaluate each stage independently. Good model-to-RDF conversion does not guarantee correct entity identification. Correct entity identification does not guarantee useful subgraph selection. Relevant retrieved triples do not ensure that an LLM will combine them faithfully. The strongest designs therefore require carefully governed ontologies or vocabularies, transparent transformation rules, retrieval strategies suited to question structure, and measures that distinguish retrieval effectiveness from answer quality. The experiments also motivate adaptive GraphRAG capable of selecting among direct graph queries, limited-hop neighborhoods, ontology-guided paths, and textual retrieval rather than relying on a universal strategy.

7.5 Theoretical Implications

The thesis extends the interpretation of the **SECI knowledge conversion spiral** by introducing diagrammatic modeling, **Knowledge Graphs**, and **GraphRAG** as mechanisms supporting enterprise knowledge conversion. Socialization remains a human-centered process, while externalization is facilitated through domain-specific modeling methods that transform stakeholders' knowledge into structured representations. Combination is enhanced through RDF-based knowledge integration, enabling the reuse of knowledge across different systems, while internalization is supported by AI-assisted question answering. Consequently, **Model-Driven GraphRAG** introduces a semantic and computational dimension to the traditional SECI spiral, facilitating knowledge sharing and reuse in AI-enabled enterprise environments.

The proposed approach contributes to **Hybrid Intelligence** by establishing a complementary relationship between human expertise and artificial intelligence. Human stakeholders remain responsible for knowledge conceptualization, modeling, validation, and interpretation, while AI systems support knowledge retrieval, explanation, and decision-making. By grounding LLM-generated answers in model-derived knowledge graphs, the approach also facilitates traceability and explainability. Thus, Model-Driven GraphRAG provides an infrastructure for human-AI collaboration in which AI augments, rather than replaces, human knowledge and judgment.

The research extends the traditional role of **modeling methods** beyond knowledge representation, communication, and documentation toward the development of AI-ready

knowledge infrastructures. By integrating metamodels, semantic mappings, RDF serialization, and GraphRAG mechanisms, modeling methods become instruments for generating and operationalizing enterprise knowledge. The proposed blueprint contributes reusable design principles for developing modeling methods that connect human-oriented visual representations with machine-oriented knowledge processing, expanding the scope of Modeling Method Engineering in the context of enterprise AI.

7.6 Practical Implications

The research demonstrates how enterprise diagrams can evolve from static documentation into operational knowledge assets supporting semantic querying, traceability, and AI-assisted knowledge retrieval. By transforming diagrammatic representations into **RDF Knowledge Graphs**, enterprises can improve the accessibility and reuse of knowledge distributed across different enterprise systems. However, successful implementation requires appropriate knowledge governance practices, including model validation, maintenance, and clearly defined responsibilities for keeping enterprises knowledge accurate and up to date.

WSF4ADO demonstrates how the **Work System Framework** can be operationalized through diagrammatic modeling and semantic knowledge representations. Managers can capture enterprise knowledge using familiar business concepts and subsequently explore relationships between processes, participants, technologies, information, products, services, and other work-system elements through AI-supported querying. This facilitates enterprise analysis, process improvement, and system redesign without requiring stakeholders to possess technical knowledge of RDF or SPARQL.

CTM4ADO enables cybersecurity analysts to connect business processes, data flows, cyber threats, and mitigation measures within a unified semantic representation. The resulting knowledge graphs support traceability, contextual risk analysis, and natural-language interaction with cybersecurity knowledge. This creates opportunities for more informed threat assessment and security decision support. Nevertheless, AI-generated recommendations require expert validation, particularly given the operational risks associated with incomplete or inaccurate cybersecurity information.

GRAPHxx facilitates RDF knowledge graph development through a visual modeling environment, reducing the need to manipulate technical serialization formats directly. The approach supports knowledge elicitation, rapid prototyping, collaborative conceptualization, and the development of semantic knowledge structures suitable for **GraphRAG** applications. Rather than replacing established ontology engineering environments, GRAPHxx complements them by providing an accessible bridge between visual conceptual modeling and formal knowledge graph engineering.

7.7 Technical and Architectural Implications

The research identifies several architectural considerations for developing **Model-Driven GraphRAG** systems. First, the quality of the underlying knowledge representation, including metamodels, ontologies, and semantic relationships, directly influences retrieval and answer quality. Second, retrieval mechanisms should adapt to the complexity and relational structure of user questions, as deeper graph traversal does not necessarily produce more accurate results. The findings also support combining graph-based and textual retrieval mechanisms within hybrid architectures. Furthermore, model-derived knowledge enables greater explainability by linking generated answers to their originating knowledge structures. Finally, enterprise implementations require mechanisms for knowledge validation, versioning, updates, and governance to maintain alignment with evolving enterprises environments.

7.8 Methodological Implications for Design Science Research

From a methodological perspective, the thesis demonstrates the suitability of combining **Design Science Research (DSR)** with **Agile Modeling Method Engineering (AMME)** for developing AI-oriented modeling methods. The iterative Produce-Use cycle enables the progressive refinement of modeling languages, software implementations, semantic transformations, and GraphRAG integration mechanisms. The research also emphasizes that these components should be designed and evaluated as an integrated knowledge management pipeline rather than as independent artifacts. Beyond the three developed instance methods, the resulting blueprint provides transferable design knowledge that can inform the development of similar modeling methods in other application domains.

7.9 Limitations

The research presents several limitations concerning the scale, generalizability, and evaluation of the developed artifacts. Although the modeling methods were demonstrated through multiple scenarios, broader validation in industrial environments and with more diverse user groups remains necessary. Furthermore, automated LLM evaluation metrics and LLM-as-judge approaches may not fully reflect practical usefulness and can introduce measurement inconsistencies.

The CTM4ADO experiments also revealed that increasing graph traversal depth can negatively affect faithfulness despite improved information coverage. Further analysis attributed this behavior to increasing contextual noise, generation errors, and limitations of LLM-based evaluation when processing large contexts, consistent with findings reported by Shi et al. (2023) and Liu et al. (2024). Finally, stronger interoperability between WSF4ADO, CTM4ADO, and GRAPHxx remains a future research direction, particularly regarding shared vocabularies, cross-model integration, and federated knowledge retrieval.

8 Conclusions and Future Work

The final chapter concludes that modeling methods can be engineered as a **disciplined bridge between human enterprise knowledge and graph-grounded AI**. The research begins with the observation that enterprise knowledge is dispersed across individual expertise, diagrams, documents, information systems, and databases. In response, it proposes a way for domain-relevant visual modeling to become an explicit source of machine-readable knowledge. The main outcome is the **Model-Driven GraphRAG blueprint**, supported by three modeling methods and their tool implementations: WSF4ADO, CTM4ADO, and GRAPHxx.

The first tool demonstrates how a managerial conceptualization can support structured work-system descriptions, semantic traceability, and high-quality ontology-informed retrieval. The second demonstrates the value of connecting business processes, data-flow analysis, cyberthreats, and mitigations within semantically queryable models, while exposing the importance of choosing appropriate retrieval depth. The third demonstrates a more general approach to visual RDF graph construction, enabling users to author interoperable semantic content through simplified concepts and low-code interactions. Across the three cases, the thesis shows that **the knowledge captured through diagrams can be transformed into reusable RDF structures and employed as evidence for LLM-based question answering**.

Theoretical conclusions concern knowledge conversion, Hybrid Intelligence, and modeling method engineering. The proposed methods support the **externalization of articulable domain knowledge**, its **combination through formal semantic structures**, and its subsequent access through natural-language AI systems. The contribution is compatible with a human-centered understanding of intelligence because expert stakeholders remain responsible for the meaning, validity, and maintenance of the knowledge represented. It also expands the scope of modeling methods: models can serve not only communication and design, but also the ongoing knowledge needs of enterprise AI architectures.

The practical conclusion is that effective enterprise GraphRAG begins **before retrieval**. It depends on selecting useful conceptualizations, enabling informed stakeholders to describe relevant facts and relations, and preserving those semantics through deterministic transformations. A sophisticated retriever cannot fully compensate for missing or ambiguous knowledge. At the same time, a well-structured knowledge graph does not guarantee excellent responses unless retrieval and generation are aligned with the question. The comparative experiments make both parts of this interdependence visible.

The final chapter identifies several directions for further research. One is **adaptive, question-sensitive retrieval GraphRAG retrieval**, capable of choosing among direct SPARQL queries, fixed-hop traversal, ontology-guided exploration, and textual or hybrid strategies. A second is **broader longitudinal evaluation**, involving more diverse groups of managers, enterprise

architects, educators, knowledge engineers, and cybersecurity practitioners to understand usability, learning, enterprise adoption, and model maintenance over time. A third is stronger **quality assurance** across the modeling and retrieval chain, from metamodel constraints and semantic consistency to evidence provenance and expert review of generated answers. Additional work should improve **interoperability among the three methods**, including shared concepts and cross-domain knowledge integration, and assess performance under more demanding real-world enterprise workloads.

In conclusion, the thesis contributes a reusable methodological perspective on enterprise knowledge streamlining in the age of generative AI. Its central proposition is not that enterprise should abandon documents or replace experts with language models. It is that **diagrammatic modeling, formal knowledge representation, and GraphRAG can be engineered as complementary parts of a human-centered knowledge system**. By turning selected domain knowledge into typed, traceable, and queryable semantic structures, modeling methods become an enabling infrastructure for enterprise AI rather than an isolated activity preceding its deployment.

Selected References from the Thesis Bibliography

This concise selection lists foundational and directly relevant works cited in the summary. The complete bibliography remains in the original thesis.

Akata, Z., Balliet, D., de Rijke, M., Dignum, F., Dignum, V., Eiben, G., Fokkens, A., Grossi, D., Hindriks, K., Hoos, H., Hung, H., Jonker, C., Monz, C., Neerincx, M., Oliehoek, F., Prakken, H., Schlobach, S., van der Gaag, L., van Harmelen, F., van Hoof, H., van Riemsdijk, B., van Wynsberghe, A., Verbrugge, R., Verheij, B., Vossen, P., Welling, M., (2020), A Research Agenda for Hybrid Intelligence: Augmenting Human Intellect With Collaborative, Adaptive, Responsible, and Explainable Artificial Intelligence, Computer, 53(8), pp. 18-28, IEEE Computer Society. DOI: https://doi.org/10.1109/MC.2020.2996587.

Alter, S., Bork, D., (2019), Work System Modeling Method with Different Levels of Specificity and Rigor for Different Stakeholder Purposes, in Proceedings: 14th International Conference on Wirtschaftsinformatik, AIS eLibrary, pp. 124 – 138. URL: https://aisel.aisnet.org/wi2019/track03/papers/1/.

Alter, S., Bork, D., (2020), Systems analysis and design toolkit based on work system theory and its extensions, Journal of Database Management, 31(3), pp. 1–13, IGI Global. DOI: https://doi.org/10.4018/JDM.2020070101.

Alter, S., (2006), The Work System Method: Connecting People, Processes, and IT for Business Results, First Edition, Work System Press.

Alter, S., (2013), Work System Theory: Overview of Core Concepts, Extensions, and Challenges for the Future, Journal of the Association for Information Systems, AIS Journals, vol. 14(2), p. 72. DOI: https://doi.org/10.17705/1jais.00323.

Alter, S., (2020), Ten Lightweight SA&D Tools Based on Work System Theory and Its Extensions, in Proceedings: The 26th Americas Conference on Information Systems (AMCIS 2020), virtual, August 15-17, 4, AIS eLibrary. Available: https://aisel.aisnet.org/amcis2020/adv_info_systems_research/adv_info_systems_research/4.

Becerra-Fernandez, I., Sabherwal, R., (2014), Knowledge Management: Systems and Processes, 2nd Edition, Routledge, Taylor & Francis Group. DOI: https://doi.org/10.4324/9781315715117.

Bock, A.C., Frank, U., (2021), Low-Code Platform, Business & Information Systems Engineering vol. 63, pp. 733–740, Springer. DOI: https://doi.org/10.1007/s12599-021-00726-8

Bork, D., Alter, S., (2020), Satisfying four requirements for more flexible modeling methods: theory and test case, Enterprise Modelling and Information Systems Architectures Journal (EMISAJ), vol. 15, pp. 3–1. DOI: https://doi.org/10.18417/emisa.15.3.

Brinkkemper, S., (1996), Method engineering: engineering of information systems development methods and tools, Information and Software Technology, 38 (4), Elsevier, p. 275-280. DOI: https://doi.org/10.1016/0950-5849(95)01059-9.

Buchmann, R.A., Ghiran, A.M., (2023), Teaching knowledge graphs: A journey from logic to Web development and semantics-driven engineering, in: The SCME23: 7th Symposium on Conceptual Modeling Education @ The 42nd International Conference on Conceptual Modeling (ER 2023), Lisbon, Portugal, November 6-9, 2023, vol. 3618, CEUR-WS. https://ceur-ws.org/Vol-3618/scme_paper_4.pdf.

Buchmann, R., Eder, J., Fill, H.G., Frank, U., Karagiannis, D., Laurenzi, E., Mylopoulos, J., Plexousakis, D., Santos, M.Y., (2024), Large language models: Expectations for semantics-driven systems engineering, Data & Knowledge Engineering, vol. 152, Elsevier, 102324. DOI: https://doi.org/10.1016/j.datak.2024.102324.

Chiș, A., Ghiran, A.M., Buchmann, R.A., (2025), GRAPHxx: DSML engineering for knowledge graph building and streamlining with GraphRAG, Software and Systems Modeling Journal (SOSYM), Springer. DOI: https://doi.org/10.1007/s10270-025-01297-y.

Chiș, A., Ghiran, A.M., Buchmann, R.A., (2025), WSF4ADO: An ADOxx Deployment of the Work Systems Framework, in: Proceedings of CAiSE 2025 Forum and Doctoral Consortium, International Conferences on Advanced Information Systems Engineering (CAiSE), Vienna, Austria, June 16-20, 2025, Intelligent Information Systems, Lecture Notes in Business Information Processing, vol 557, Springer, Cham. DOI: https://doi.org/10.1007/978-3-031-94590-8_6.

Chiș, A., Ghiran, A.M., Buchmann, R.A., (2026), A model-driven knowledge graph treatment for process-centric cyber threat mitigation, Knowledge and Information Systems - An International Journal, 68, 210, Springer. DOI: https://doi.org/10.1007/s10115-026-02819-w.

Chiș, A., Ghiran, A.M., Alter, S., (2024), Informing Enterprise Knowledge Graphs with a Work System Perspective, Enterprise Modelling and Information Systems Architectures (EMISAJ), vol. 19, German Informatics Society. DOI: https://doi.rog/10.18417/emisa.19.7.

Dellermann, D., Ebel, P., Söllner, M., Leimeister, J.M., (2019), Hybrid Intelligence, Business & Information Systems Engineering, 61, pp. 637-643. DOI: https://doi.org/10.1007/s12599-019-00595-2.

Fill, H.G., Karagiannis, D., (2013), On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform, Enterprise Modelling and Information Systems Architectures (EMISAJ) - International Journal of Conceptual Modeling, vol. 8(1), German Informatics Society. DOI: https://doi.org/10.18417/emisa.8.1.1.

Frank, U., (2011), Some guidelines for the conception of domain-specific modelling languages, Enterprise modelling and information systems architectures (EMISA 2011), German Informatics Society.

Frank, U., (2013), Domain-Specific Modeling Languages: Requirements Analysis and Design Guidelines, in: Reinhartz-Berger, I., Sturm, A., Clark, T., Cohen, S., Bettin, J. (eds) Domain Engineering. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-36654-3_6.

Glaser, P.L., Ali, S.J., Sallinger, E., Bork, D., (2022), Model-Based Construction of Enterprise Architecture Knowledge Graphs, in: Proceedings of the Enterprise Design, Operations, and Computing: 26th International Conference (EDOC 2022), October 3-7, 2022, Bozen-

Bolzano, Italy, Lecture Notes in Computer Science, vol. 13585, Springer, Cham. DOI: https://doi.org/10.1007/978-3-031-17604-3_4.

Han, H., Wang, Y., Shormer, H., Guo, K., Ding, J., Lei, Y., Halappanavar, M., Rossi, R.A., Mukherjee, S., Tang X., He, Q., Hua, Z., Long, B., Zhao, T., Shah, N., Javari, A., Xia, Y., Tang, J., (2025), Retrieval-Augmented Generation with Graphs (GraphRAG), arXiv preprint arXiv:2501.00309v2, ArXiv. DOI: https://doi.org/10.48550/arXiv.2501.00309.

Harmsen, F., Brinkkemper, S., Oei, H., (1994), Situational Method Engineering for Information System Project Approaches, In: Proceedings of the IFIP WG8.1 Working Conference on Methods and Associated Tools for the Information Systems Life Cycle, Springer, p. 169–194.

Henderson-Sellers, B., Ralyté, J., Ågerfalk, P.J., Rossi, M., (2014), Situational Method Engineering, Springer Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-41467-1.

Hevner, A.R., (2007), A Three Cycle View of Design Science Research, Scandinavian Journal of Information Systems, vol. 19(2), Article 4. Available at: http://aisel.aisnet.org/sjis/vol19/iss2/4.

Jarrahi, M.H., Lutz, C., Newlands, G., (2022), Artificial intelligence, human intelligence and hybrid intelligence based on mutual augmentation, Big Data & Society, Sage Journals. DOI: https://doi.org/10.1177/20539517221142824.

Karagiannis, D., Kühn, H., (2002), Metamodelling Platforms, in: Bauknecht, K., Tjoa, A.M., Quirchmayr, G. (eds) E-Commerce and Web Technologies, EC-Web 2002, Lecture Notes in Computer Science, vol 2455. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/3-540-45705-4_19.

Karagiannis, D., (2015), Agile Modeling Method Engineering, in: Proceedings of the 19th Panhellenic Conference on Informatics, Association for Computing Machinery, p. 5–10. DOI: https://doi.org/10.1145/2801948.2802040.

Kohnfelder, L., Garg, P., (1999), The threats to our products, Microsoft Interface, Microsoft Corporation, 33.

Krogstie, J., Sindre, G., Jørgensen, H., (2006), Process models representing knowledge for action: a revised quality framework, European Journal of Information Systems, vol.15, p. 91–102, Palgrave Macmillan. DOI: https://doi.org/10.1007/978-3-319-39417-6_1.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., Kiela, D., (2020), Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20), December 6-12, 2020, Virtual, Article no. 793, pp. 9459-9474, Association for Computing Machinery, Curran Associates Inc. DOI: https://dl.acm.org/doi/abs/10.5555/3495724.3496517.

Lillehagen, F., Krogstie, J., (2008), Active Knowledge Modeling of Enterprises, Springer Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-540-79416-5.

Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P., (2024), Lost in the Middle: How Language Models Use Long Contexts, Transactions of the Association for Computational Linguistics, vol. 12, pp. 157-173. DOI: 10.1162/tacl_a_00638.

Maier, R., (2007), Knowledge Management Systems: Information and Communication Technologies for Knowledge Management, Springer Berling, Heidelberg. DOI: https://doi.org/10.1007/978-3-540-71408-8.

Merali, Y., Davies, J., (2001), Knowledge capture and utilization in virtual communities, in: Proceedings of the 1st International Conference on Knowledge Capture (K-CAP 2001), Victoria, BC, Canada, October 21-23, 2001, pp. 92-99, Association for Computing Machinery. DOI: https://doi.org/10.1145/500737.500754.

Moody, D.L., (2009), The "Physics" of notations: towards a scientific basis for constructing visual notations in software engineering, IEEE Transactions on Software Engineering 35(5), 756–779, IEEE Computer Society. DOI: https://doi.org/10.1109/TSE.2009.67.

Nonaka, I., Takeuchi, H., (1995), The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation, Oxford Academic.

Nonaka, I., (1994), A Dynamic Theory of Organizational Knowledge Creation, Organization Science, vol. 5(1), p. 14-37, Informs. URL: https://www.jstor.org/stable/2635068.

Ontotext, What is Graph RAG?, (n.d.-b), URL: https://www.ontotext.com/knowledgehub/fundamentals/what-is-graph-rag/. Last accessed 17.01.2025.

Pan. S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X., (2024), Unifying Large Language Models and Knowledge Graphs: A Roadmap, IEEE Transactions on Knowledge and Data Engineering , 36(7), pp. 3580 - 3599. DOI: 10.1109/TKDE.2024.3352100.

Peffers, K., Tuunanen, T., Gengler C.E., Rossi, M., Hui, W., Virtanen, V., Bragge, J., (2006), The Design Science Research Process: A Model for Producing and Presenting Information Systems Research, arXiv: Software Engineering (Claremont Graduate University), p. 83-106. DOI: https://doi.org/10.48550/arXiv.2006.02763.

Reitemeyer, B., Fill, H.G., (2024), Leveraging LLMs in Semantic Mapping for Knowledge Graph-based Automated Enterprise Model Generation, in Proceedings: Modellierung 2024 Satellite Events, Digital Library, Gesellschaft für Informatik. Available: https://dl.gi.de/server/api/core/bitstreams/940c8d0a-3bce-42c5-8408-41142e8b0d07/content.

Sandkuhl, K., Barn, B.S., Barat, S., (2026), LLM support for enterprise modelling based on integrating prompt and enterprise modelling meta-models, Discover Artificial Intelligence, vol. 6, 303, Springer. DOI: https://doi.org/10.1007/s44163-026-01056-y.

Schinckus, M., Simonofski, A., Rosselló, N.B., (2026), Large Language Models for Process Knowledge Acquisition, Business & Information Systems Engineering, vol. 68, pp. 7-33, Springer. DOI: https://doi.org/10.1007/s12599-025-00976-w.

Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E., Schärli, N., Zhou, D., (2023), Large language models can be easily distracted by irrelevant context, in: Proceedings of 40th International Conference on Machine Learning, 2023 (ICML 2023), July 23-29, 2023, Honolulu, Hawaii, USA, 1291, pp. 31210-31227, ACM Digital Library. DOI: https://dl.acm.org/doi/10.5555/3618408.3619699.

Shostack, A., (2008), Experiences Threat Modeling at Microsoft, in: Proceedings of the Workshop on Modeling Security (MODSEC08), 2008 International Conference on Model Driven Engineering Languages and Systems (MODELS 2008), September 28,

2008, Tolouse, France, CEUR-WS, vol. 413, paper 12. Available: https://ceur-ws.org/Vol-413/paper12.pdf.

Smajevic, M., Bork, D., (2021), From Conceptual Models to Knowledge Graphs: A Generic Model Transformation Platform, in: Proceedings of the 2021 ACM/IEEE International Conference on Model Driven Enginnering Languages and Systems Companion (MODELS-C 2021), October 10 - 15, 2021, Fukuoka, Japan, pp. 610-614, IEEE. DOI: https://doi.org/10.1109/MODELS-C53483.2021.00093.

W3C, W3C Recommendation, SPARQL Query Language for RDF, (2008). URL: https://www.w3.org/TR/rdf-sparql-query/. Last accessed 17.01.2025.

W3C, Web Ontology Language (OWL), (2012). URL: https://www.w3.org/OWL/. Last accessed 23.06.2026.

W3C, W3C Recommendation, RDF, (2014a). URL: https://www.w3.org/RDF/. Last accessed 10.12.2025.

Wieringa, R.J., (2014), Design Science Methodology for Information Systems and Software Engineering, Springer Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-662-43839-8.

Wieringa, R.J., (2016), Design Science Research Methods. URL: https://wwwhome.ewi.ut-wente.nl/~roelw/DSM90minutes.pdf. Last accessed 20.07.2025.

Wuyts, K., Sion, L., Joosen, W., (2020), Linddun go: A lightweight approach to privacy threat modeling, in: Proceedings of the 5th IEEE European Symposium on Security and Privacy Workshops, EuroS&PW, Online, 7–11 September, IEEE Xplore, pp. 302–309. DOI: https://doi.org/10.1109/EuroSPW51379.2020.00047.