Knowledge Management Ontologies in Software Engineering: Applications, Challenges and Future Research Directions - A Systematic Literature Review

Edward N. Udo and Ini J. Umoeka

Department of Computer Science, Faculty of Computing, University of Uyo, Uyo, Nigeria

DOI: https://doi.org/10.51244/IJRSI.2026.1306000362

Received: 22 June 2026; Accepted: 28 June 2026; Published: 10 July 2026

ABSTRACT

Knowledge management has become increasingly important in software engineering due to the knowledge intensive nature of software development processes. However, traditional knowledge management approaches often face challenges related to knowledge representation, retrieval, sharing, and reuse. Ontologies have emerged as a promising solution by providing formal and semantic representations of knowledge that facilitate interoperability, knowledge integration, and intelligent decision support. This study presents a systematic literature review of knowledge management ontologies in software engineering, with the aim of synthesizing existing research on their applications, benefits, challenges, and future development opportunities. Relevant studies were identified through a structured search of major scientific databases and analyzed using predefined inclusion and exclusion criteria. The findings reveal that knowledge management ontologies have been applied across various software engineering activities, including requirements engineering, software design and development, software testing, software maintenance, project management, and software process improvement. Reported benefits include enhanced knowledge sharing, improved knowledge reuse, better decision making, increased software quality, and strengthened organizational learning. Despite these advantages, challenges such as ontology development complexity, knowledge acquisition difficulties, scalability limitations, integration issues, and user adoption barriers continue to hinder broader implementation. The review further identifies emerging research directions involving artificial intelligence, knowledge graphs, semantic interoperability, and intelligent software engineering environments. In conclusion, the study highlights the growing significance of ontology based knowledge management in software engineering and provides a comprehensive synthesis of current knowledge while outlining opportunities for future research and industrial adoption.

Keywords: Knowledge Management; Ontologies; Software Engineering; Knowledge Representation; Semantic Technologies; Systematic Literature Review.

INTRODUCTION

Software engineering is widely recognized as a knowledge intensive discipline in which the success of software projects depends largely on the effective creation, acquisition, sharing, storage, and reuse of knowledge throughout the software development lifecycle. Software development activities generate substantial amounts of knowledge related to requirements elicitation, system design, coding practices, testing procedures, project management, maintenance, and organizational experiences. The ability of software organizations to manage this knowledge effectively influences software quality, project performance, innovation, and long term organizational competitiveness (Rus & Lindvall, 2002; Bjørnson & Dingsøyr, 2008).

Knowledge Management (KM) has emerged as a critical approach for capturing and leveraging organizational knowledge assets. KM encompasses the processes through which knowledge is created, stored, shared, transferred, and applied to achieve organizational objectives (Alavi & Leidner, 2001). Ammirato et al., (2021) recently defined KM as the comprehensive process of identifying, organizing, transferring, and utilizing information and skills. Within software engineering, KM facilitates the preservation of valuable expertise,

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

reduces knowledge loss caused by employee turnover, enhances collaboration among development teams, and promotes the reuse of software artefacts and best practices (Dingsøyr & Conradi, 2002). As software systems become increasingly complex and development teams become more geographically distributed, the importance of effective KM strategies continues to grow because knowledge generation is a major benefit of any KM approach (Idrees et al., 2023).

Despite the recognized benefits of KM, many software organizations struggle to manage knowledge efficiently due to fragmented repositories, inconsistent documentation practices, and the limited ability of traditional information management systems to capture semantic relationships among software artefacts and development processes (Aurum et al. 2008). Conventional databases and document management systems primarily focus on storing information rather than representing the underlying meaning and interconnections between knowledge resources. Consequently, retrieving, integrating, and reusing knowledge across software projects often remains difficult and time consuming.

To address these limitations, researchers have increasingly explored the application of ontologies as a semantic foundation for knowledge management systems. An ontology can be defined as an explicit specification of a conceptualization that formally represents concepts, relationships, and constraints within a domain. Ontologies facilitate a shared understanding of domain knowledge and provide machine interpretable representations that support intelligent information retrieval, semantic interoperability, reasoning, and knowledge reuse. These characteristics make ontologies particularly suitable for managing the complex and interconnected knowledge generated during software development activities.

Ontology is a structured framework used to define and organize knowledge within a particular area (Biagetti, 2021; Osman, Noah, & Saad, 2022). It offers a formal representation of concepts, their attributes as well as their relationships, to guarantee clarity and shared understanding among users (Pliatsios et al., 2023; Said, et al., 2023). Ontologies help in reducing ambiguities and improving communication across different systems and fields by allowing consistent vocabulary and structure (Fraga et al., 2020; Guizzardi and Guarino, 2024).

Over the past two decades, ontology based approaches have been applied to various software engineering processes, including requirements engineering, software design, project management, software maintenance, software testing, and software process improvement (Alrumaih et al., 2020; Husáková and Bureš, 2020)).

Ontologies have been used to formalize software development knowledge, facilitate collaboration among stakeholders, support decision making, improve traceability, and enable the reuse of software assets. Furthermore, the emergence of Semantic Web technologies, knowledge graphs, artificial intelligence, and intelligent software development environments has expanded the potential applications of ontology driven knowledge management systems in software engineering.

Although the literature on ontology applications in software engineering has grown substantially, existing studies are dispersed across different domains, methodologies, and application contexts. As a result, there remains limited consolidated evidence regarding the types of knowledge management ontologies employed, their specific applications within software engineering processes, the benefits achieved, and the challenges affecting their implementation. Previous reviews have often focused on either software engineering knowledge management practices or ontology engineering techniques independently, providing limited synthesis of their intersection.

A systematic examination of the existing literature is therefore necessary to develop a comprehensive understanding of how knowledge management ontologies contribute to software engineering activities. Such a review can provide valuable insights into current research trends, identify existing challenges, highlight knowledge gaps, and establish priorities for future investigations. Moreover, a consolidated synthesis can support researchers and practitioners seeking to design, implement, and evaluate ontology based knowledge management solutions in software development environments.

Therefore, this study presents a systematic literature review of knowledge management ontologies in software engineering. The review aims to synthesize existing research, analyze ontology applications across software engineering processes, identify reported benefits and implementation challenges, and propose future research

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

directions. By integrating evidence from the existing body of knowledge, the study contributes to a deeper understanding of ontology enabled knowledge management and its role in supporting effective software engineering practices.

METHODOLOGY

This study adopted a Systematic Literature Review (SLR) approach to identify, evaluate, and synthesize existing research on the application of knowledge management ontologies in software engineering. The systematic review methodology was selected because it provides a transparent, rigorous, and reproducible process for summarizing evidence from published studies while minimizing selection bias (Kitchenham and Charters, 2007; Petersen et al., 2015).

The review process was guided by established systematic review procedures commonly employed in software engineering research.

A review protocol was established before commencing the literature search. The protocol specified the research questions, databases, search strategy, eligibility criteria, quality assessment procedure, and data extraction process. The review was conducted in accordance with the PRISMA 2020 reporting guidelines to ensure methodological transparency, reproducibility, and consistency throughout the review process.

Review Protocol

A predefined systematic review protocol was developed before commencing the literature search to ensure methodological rigor, transparency, and reproducibility. The protocol specified the review objectives, research questions, search strategy, electronic databases to be searched, search strings, publication time frame, study selection procedures, inclusion and exclusion criteria, quality assessment framework, and data extraction process.

The review was conducted in accordance with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA 2020) guidelines (Page et al., 2021). The protocol guided each stage of the review, including literature identification, duplicate removal, title and abstract screening, full-text eligibility assessment, quality appraisal, and narrative synthesis of the selected studies. Adherence to this structured protocol minimized selection bias and enhanced the reliability, transparency, and reproducibility of the review findings.

Review Design

The review was conducted in four stages: literature search, study selection, quality assessment, and data synthesis. These stages were designed to ensure that only relevant and high-quality studies were included in the final analysis. The review focused on studies that investigated ontology-based approaches for knowledge management within software engineering environments, including software development, project management, maintenance, testing, and process improvement activities.

The review was guided by the following research questions:

RQ1: What types of knowledge management ontologies have been applied in software engineering?

RQ2: How are knowledge management ontologies utilized across software engineering processes?

RQ3: What benefits have been reported from the adoption of knowledge management ontologies in software engineering?

RQ4: What challenges affect the implementation and effectiveness of ontology-based knowledge management approaches?

RQ5: What future research directions can enhance the application of knowledge management ontologies in software engineering?

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Search Strategy

A comprehensive literature search was conducted using major scientific databases that index software engineering, information systems, artificial intelligence, and knowledge management research. The selected databases included:

- Scopus

- Web of Science

- IEEE Xplore Digital Library

- ACM Digital Library

- ScienceDirect

- SpringerLink

These databases were selected because they contain a substantial proportion of peer-reviewed publications relevant to software engineering and ontology research.

The search process utilized combinations of keywords related to knowledge management, ontologies, and software engineering.

Additional keyword combinations were employed where necessary to capture relevant studies that may not have been retrieved through the primary search string.

Search String and Search Period

The literature search was conducted between January and March 2026 across Scopus, Web of Science, IEEE Xplore, ACM Digital Library, ScienceDirect, and SpringerLink. The search strategy was designed to identify studies addressing the application of ontology-based knowledge management approaches within software engineering environments.

The primary search string was:

("knowledge management" OR "knowledge sharing" OR "knowledge reuse") AND (ontology OR ontologies OR "semantic technology" OR "semantic web") AND ("software engineering" OR "software development" OR "software process" OR "software project management")

To improve retrieval coverage, the search syntax was adapted to meet the indexing requirements of individual databases. Additional combinations of keywords such as "knowledge graph", "semantic repository", "ontology engineering", "software maintenance", and "requirements engineering" were also employed where necessary.

The search was restricted to English-language publications published between 2000 and 2025. This period was selected because ontology-based knowledge management research in software engineering gained significant momentum following the emergence of Semantic Web technologies in the early 2000s.

The review process followed the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA 2020) reporting guidelines (Page et al., 2021) to ensure transparency, consistency, and reproducibility.

A total of 520 records were retrieved from six electronic databases. After removing 85 duplicate records, 435 studies remained for titles and abstracts screening. Following the screening process, 380 were excluded because they did not meet the inclusion criteria. The remaining 55 full text articles were assessed for eligibility, of which, 47 studies satisfied all inclusion and quality assessment criteria and were included in the final qualitative synthesis. Figure 1 and Table 1 illustrate the complete PRISMA study selection process.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

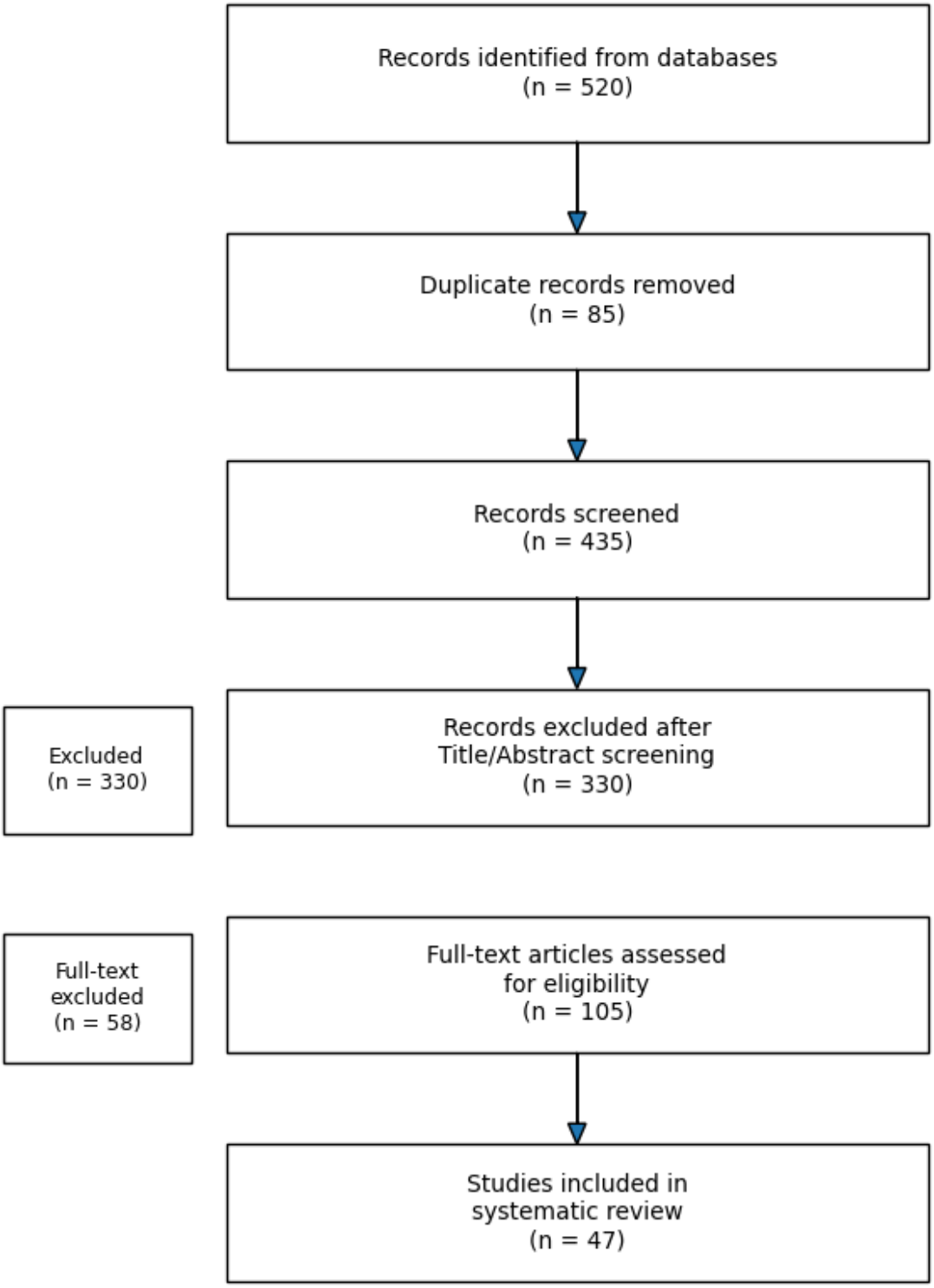

Figure 1 - PRISMA Flow Diagram for the Selection Process

Table 1: Summary of the Study Selection Process Following the PRISMA 2020 Guidelines

PRISMA Stage	Number of Records
Records Identified	520
Duplicates Record	85
Records Screened	435
Records Excluded	380
Full-text Article Assessed	55
Studies Included	47

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Inclusion and Exclusion Criteria

To ensure the relevance and quality of the selected studies, predefined inclusion and exclusion criteria were applied.

Inclusion Criteria

Studies were included if they:

i. Focused on ontology-based knowledge management in software engineering.

ii. Examined the development, implementation, evaluation, or application of ontologies within software engineering contexts.

iii. Were published in peer-reviewed journals, conference proceedings, or book chapters.

iv. Were written in English.

v. Provided sufficient methodological or technical information to support analysis.

Exclusion Criteria

Studies were excluded if they:

i. Focused exclusively on ontology engineering without a knowledge management component.

ii. Addressed knowledge management in domains unrelated to software engineering.

iii. Were editorials, abstracts, posters, tutorials, dissertations, or non-peer-reviewed publications.

iv. Were duplicate records retrieved from multiple databases.

v. Lacked sufficient information for data extraction and analysis.

The application of these criteria helped to ensure that the final dataset remained directly relevant to the objectives of the review.

Study Selection Process

The study selection process involved several stages. Initially, all retrieved records were exported into a reference management system and duplicate publications were removed. The titles and abstracts of the remaining studies were subsequently screened against the inclusion and exclusion criteria.

Studies that satisfied the initial screening requirements proceeded to full-text assessment. During this stage, each article was examined in detail to determine its relevance to the objectives of the review. Only studies that explicitly addressed ontology-supported knowledge management activities within software engineering were retained for the final synthesis.

The selection procedure followed the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework, which provides a structured approach for documenting literature identification, screening, eligibility assessment, and inclusion decisions (Page et al., 2021).

Quality Assessment

To improve the reliability of the review findings, the selected studies were subjected to quality assessment. Each study was evaluated using a set of criteria adapted from established software engineering review guidelines (Kitchenham and Charters, 2007).

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

The quality assessment considered the following questions:

- Was the study objective clearly defined?

- Was the ontology approach adequately described?

- Was the software engineering context clearly specified?

- Were evaluation methods or validation procedures reported?

- Were findings and conclusions supported by evidence?

Each study was independently assessed against the five quality criteria using a three point scoring system: **Yes (1 point), Partially (0.5 points), and No (0 points).** The maximum attainable quality score was five. Studies scoring below three points were excluded from the review because they did not provide sufficient methodological quality or relevance to the study objectives. The quality assessment ensured that only robust and relevant studies contributed to the final synthesis.

Data Extraction and Synthesis

A structured data extraction framework was developed to ensure consistency in the collection of information from the selected studies. The following information was extracted:

- Author(s) and publication year

- Research objectives

- Ontology type and characteristics

- Software engineering application domain

- Knowledge management function addressed

- Reported benefits

- Identified challenges

- Evaluation approach

- Key findings

The extracted data were analyzed using a narrative synthesis approach. This method was considered appropriate because of the diversity of research designs, ontology models, software engineering contexts, and evaluation methods reported in the literature. Studies were grouped according to ontology type, software engineering application area, benefits, and challenges to facilitate comparison and thematic analysis.

The synthesis process enabled the identification of recurring patterns, emerging trends, research gaps, and opportunities for future development. The findings are presented in subsequent sections through thematic discussions supported by summary tables and conceptual illustrations.

The methodological framework adopted in this study provides a systematic and transparent basis for synthesizing current knowledge on the application of ontologies in software engineering knowledge management. By following established review procedures, the study aims to produce reliable evidence that can support both academic research and practical implementation efforts.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

OVERVIEW OF KNOWLEDGE MANAGEMENT ONTOLOGIES IN SOFTWARE ENGINEERING

Evolution of Ontology Based Knowledge Management

Knowledge management has become an integral component of software engineering due to the knowledge intensive nature of software development activities. Software projects continuously generate diverse forms of knowledge, including technical specifications, design decisions, coding standards, testing procedures, project experiences, and organizational best practices. Early knowledge management approaches in software engineering primarily relied on document repositories, databases, expert directories, and lessons learned systems to capture and disseminate organizational knowledge (Rus & Lindvall, 2002; Dingsøyr & Conradi, 2002). Although these approaches improved information storage and accessibility, they often lacked mechanisms for representing semantic relationships among knowledge resources.

The emergence of Semantic Web technologies introduced new opportunities for addressing these limitations through ontology based knowledge representation. Ontologies provide structured and machine interpretable descriptions of concepts and relationships within a domain, thereby facilitating knowledge sharing, integration, and reuse. Their adoption in software engineering was motivated by the need to improve communication among stakeholders, enhance knowledge retrieval, support decision making, and promote software process improvement.

During the early 2000s, researchers began developing ontology driven frameworks for modelling software development knowledge and supporting software engineering activities. These efforts focused on formalizing software concepts, documenting development processes, and enabling semantic interoperability among software tools and repositories (Happel and Seedorf, 2006). As ontology engineering techniques matured, their application expanded beyond knowledge representation to include requirements management, software architecture design, project management, software maintenance, and collaborative software development environments.

Recent advances in artificial intelligence, cloud computing, knowledge graphs, and intelligent software systems have further increased interest in ontology based knowledge management solutions. They are commonly applied in areas such as artificial intelligence, semantic web, software development, and data management to support knowledge sharing, integration, and reasoning (Chaccour et al., 2024; Fu et al., 2022; Olan, et al., 2022).

Contemporary ontology driven systems increasingly integrate semantic reasoning, automated knowledge discovery, and intelligent recommendation capabilities to support software engineering decision making and organizational learning (Wongthongtham et al. 2009). Ontologies enable better interoperability of data, enhance decision-making, and also promote the development of automated systems by organizing and linking information in a systematic way (De Nicola and Villani, 2021)' Consequently, ontologies have evolved from simple knowledge modelling tools to strategic enablers of intelligent knowledge management in software development environments.

Types of Knowledge Management Ontologies

Knowledge management ontologies employed in software engineering can be classified into several categories based on their scope, purpose, and application domain.

Domain Ontologies

Domain ontologies describe concepts and relationships associated with a specific software engineering domain. They provide a shared vocabulary that facilitates communication among stakeholders and supports knowledge reuse within specialized areas. Examples include ontologies developed for requirements engineering, software testing, software architecture, and software maintenance (Gómez Pérez et al., 2004).

Task Ontologies

Task ontologies represent knowledge associated with specific software engineering activities or tasks. These

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

ontologies define procedures, workflows, inputs, outputs, and decision criteria related to particular development activities. Task ontologies are commonly used to support requirements analysis, defect management, testing processes, and software project planning.

Process Ontologies

Process ontologies focus on modelling software development methodologies, workflows, and lifecycle activities. They provide structured representations of software processes and facilitate process standardization, monitoring, and improvement. Process ontologies are particularly useful in organizations seeking to implement software process improvement frameworks and quality assurance programmes.

Enterprise Ontologies

Enterprise ontologies capture organizational knowledge, business rules, development practices, and institutional experiences. They support knowledge sharing across departments and development teams by providing a unified representation of organizational assets and processes. Enterprise ontologies are frequently employed in large software organizations where knowledge integration across multiple projects is essential.

Application Ontologies

Application ontologies are designed to support specific software applications or knowledge management systems. They typically combine concepts from multiple ontology categories and are tailored to meet particular operational requirements. Such ontologies often support intelligent search, recommendation systems, semantic repositories, and decision support platforms.

The diversity of ontology types demonstrates the broad applicability of ontology based knowledge management across software engineering activities as shown in Table 2. Many modern systems employ hybrid approaches that integrate multiple ontology categories to achieve greater flexibility and effectiveness.

Table 2: Classification of Knowledge Management Ontologies Used in Software Engineering

Ontology Type	Primary Purpose	Typical Software Engineering Applications
Domain Ontology	Knowledge representation within a domain	Requirements engineering, testing
Task Ontology	Activity-specific knowledge modelling	Defect management, testing
Process Ontology	Workflow and lifecycle representation	Process improvement, quality management
Enterprise Ontology	Organizational knowledge integration	Project management, knowledge sharing
Application Ontology	Application-specific implementation	Semantic repositories, decision support

Source: Adapted from Gómez Pérez et al., 2004); Happel and Seedorf (2006).

Ontology Technologies and Tools

The development and implementation of ontology based knowledge management systems rely on a variety of technologies, standards, and software tools. These technologies provide the foundation for representing, storing, querying, and reasoning over software engineering knowledge.

One of the most widely used ontology representation standards is the Resource Description Framework (RDF), which enables the structured description of resources and their relationships. RDF provides the basis for semantic data representation and interoperability across heterogeneous systems. Building upon RDF, the RDF Schema (RDFS) introduces mechanisms for defining classes, properties, and hierarchical relationships among concepts.

The Web Ontology Language (OWL) has become the dominant ontology representation language due to its expressive capabilities and support for logical reasoning. OWL enables ontology developers to define complex

relationships, constraints, and inference rules, thereby enhancing the semantic richness of knowledge management systems (McGuinness & van Harmelen, 2004).

To facilitate ontology querying and retrieval, the SPARQL Protocol and RDF Query Language (SPARQL) is commonly employed. SPARQL allows users and software agents to retrieve and manipulate semantic information stored within ontology repositories.

Several ontology development environments have also contributed to the adoption of ontology based knowledge management in software engineering. Among these, Protégé remains the most widely used ontology development platform because of its flexibility, extensibility, and support for OWL based ontology modelling (Musen, 2015). Other tools such as TopBraid Composer, WebProtégé, Apache Jena, and OntoStudio have also been employed in ontology engineering projects.

In recent years, ontology technologies have increasingly been integrated with artificial intelligence, machine learning, and knowledge graph frameworks. This integration has expanded the capabilities of ontology based systems by enabling automated knowledge extraction, semantic reasoning, intelligent recommendation, and adaptive decision support. Such developments have strengthened the role of ontologies as a foundational technology for knowledge management in software engineering.

Overall, the evolution of ontology based knowledge management, the diversity of ontology types, and the availability of advanced semantic technologies have contributed significantly to the growing adoption of ontologies within software engineering. These developments provide the conceptual and technological foundation upon which ontology applications across software development processes are built.

APPLICATIONS OF KNOWLEDGE MANAGEMENT ONTOLOGIES IN SOFTWARE ENGINEERING

Knowledge management ontologies have been applied across various phases of the software development lifecycle to improve knowledge representation, facilitate collaboration, enhance decision making, and support software process improvement. By providing formal semantic structures for organizing and retrieving information, ontologies enable software organizations to capture and reuse valuable knowledge assets more effectively. This section examines the major application areas of knowledge management ontologies within software engineering processes.

Requirements Engineering

Requirements engineering is one of the earliest and most extensively studied application areas of ontology based knowledge management. The requirement process is one of the most critical factors in determining whether the software development process is successful. It is crucial to consider the function that ontology plays in the requirements of software engineering development (Fauzan et al., 2025. The requirements phase involves gathering, analyzing, documenting, validating, and managing stakeholder requirements. Given the complexity and ambiguity often associated with requirements elicitation, ontologies have been employed to establish shared vocabularies and improve communication among stakeholders (Noy and McGuinness, 2001).

Ontology based approaches facilitate the identification and classification of requirements by providing structured representations of domain concepts and their relationships. These semantic models help reduce misunderstandings between clients, analysts, and developers while improving requirements consistency and completeness. Furthermore, ontologies support requirements traceability by linking requirements to design elements, implementation components, and testing activities, thereby enhancing change management and project monitoring. The use of ontology in recent years has the biggest opportunity in the requirement phase of software engineering methodology (Farghaly et al., 2023; Tudorache, 2020).

Several studies have demonstrated that ontology driven requirements management systems improve knowledge reuse by enabling analysts to retrieve similar requirements from previous projects. Such capabilities reduce duplication of effort and accelerate software development processes.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Software Design and Development

Software design and development activities generate substantial amounts of technical knowledge, including architectural decisions, design patterns, coding standards, and implementation practices. Managing this knowledge effectively is essential for maintaining software quality and ensuring consistency across development teams.

Knowledge management ontologies have been used to model software architecture knowledge, document design rationale, and support component reuse. Ontology based repositories provide developers with access to previously developed software artefacts and design solutions, thereby facilitating knowledge sharing and reducing development costs (Aurum et al. 2008).

In software development environments, ontologies also support semantic code retrieval and component discovery. Developers can search repositories using conceptual descriptions rather than simple keyword matching, resulting in more accurate identification of relevant software components. This capability enhances software reuse and promotes the development of modular and maintainable systems.

Furthermore, ontology driven development environments have been employed to integrate knowledge from multiple software tools and repositories, enabling greater semantic interoperability across development platforms. Such integration improves coordination among development teams and enhances access to organizational knowledge resources.

Software Testing and Maintenance

Software testing and maintenance represent critical activities that require extensive knowledge management due to the large volume of information generated during system validation and evolution. Testing processes involve the creation and management of test cases, defect reports, validation procedures, and quality assurance records.

Ontologies have been utilized to organize testing knowledge and support intelligent defect classification systems. By representing relationships among software components, defects, test cases, and corrective actions, ontology based systems facilitate knowledge retrieval and improve fault diagnosis processes (Wongthongtham, et al., 2009).

In software maintenance, ontologies support the preservation and transfer of knowledge related to legacy systems. Maintenance engineers often face challenges when working with poorly documented systems developed by previous teams. Ontology based knowledge repositories provide structured representations of system architecture, component interactions, and historical modifications, thereby reducing the effort required to understand and modify existing software systems.

Additionally, ontology driven maintenance systems enhance change impact analysis by identifying dependencies among software artefacts and highlighting potential consequences of proposed modifications. Such capabilities contribute to improved software reliability and maintainability.

Project and Process Management

Software project management relies heavily on the effective management of organizational knowledge, including project experiences, lessons learned, risk assessments, resource allocation strategies, and process documentation. Ontology based knowledge management systems have been developed to capture and disseminate this information across software projects.

Project management ontologies facilitate the formal representation of project knowledge and support decision making throughout the project lifecycle. By structuring information related to schedules, resources, risks, and performance indicators, these ontologies enable project managers to access relevant knowledge and apply lessons learned from previous projects (Dingsøyr & Conradi, 2002).

Knowledge management ontologies also play an important role in software process improvement initiatives.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Process ontologies provide formal descriptions of software development methodologies, workflows, and best practices, thereby supporting process standardization and compliance with quality frameworks such as Capability Maturity Model Integration (CMMI) and ISO software quality standards (Table 3).

In agile and distributed software development environments, ontology based systems support collaboration by facilitating knowledge sharing among geographically dispersed teams. Semantic repositories enable team members to access project information, development guidelines, and organizational expertise regardless of their physical location. This capability contributes to improved communication, coordination, and project performance.

The growing adoption of DevOps practices has further expanded the application of ontologies in project and process management. Ontology based knowledge management systems can integrate information from development, testing, deployment, and operational environments, thereby promoting continuous learning and supporting data driven decision making across the software delivery pipeline.

Generally, the literature demonstrates that knowledge management ontologies have become valuable tools for supporting software engineering activities throughout the software development lifecycle. Their applications extend from requirements engineering to software maintenance and project management, highlighting their versatility in facilitating knowledge capture, sharing, reuse, and decision support. The benefits derived from these applications, as well as the challenges associated with ontology implementation, are discussed in the following section.

Table 3: Applications of Knowledge Management Ontologies across Software Engineering Processes

Software Engineering Process	**Ontology Application**	**Knowledge Management Function**	**Expected Benefits**
Requirements Engineering	Requirements modelling and traceability	Knowledge capture and reuse	Improved consistency and stakeholder communication
Software Design and Development	Architecture and component modelling	Knowledge sharing and reuse	Reduced development effort and improved software quality
Software Testing	Defect classification and test management	Knowledge organization	Improved fault diagnosis and testing efficiency
Software Maintenance	Legacy system documentation and change analysis	Knowledge preservation	Enhanced maintainability and system understanding
Project Management	Lessons learned and risk management	Knowledge transfer	Better decision making and project performance
Process Improvement	Workflow and methodology modelling	Organizational learning	Improved process standardization and maturity

Source: Compiled from Dingsøyr and Conradi (2002); Aurum et al., (2008); Wongthongtham, et al., (2009).

BENEFITS AND CHALLENGES OF KNOWLEDGE MANAGEMENT ONTOLOGIES IN SOFTWARE ENGINEERING

Benefits

The growing adoption of ontology based knowledge management systems in software engineering has generated considerable interest among researchers and practitioners. Existing studies report numerous benefits associated with ontology implementation, particularly in the areas of knowledge sharing, knowledge reuse, collaboration, and decision support. However, despite these advantages, several technical, organizational, and operational challenges continue to limit the widespread adoption of ontology driven solutions. This section synthesizes the major benefits and challenges identified in the literature.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Enhanced Knowledge Sharing

One of the most frequently reported benefits of ontology based knowledge management systems is their ability to facilitate knowledge sharing among software development stakeholders. Software projects often involve multidisciplinary teams comprising analysts, developers, testers, project managers, and domain experts. Differences in terminology and understanding can create communication barriers that negatively affect project outcomes.

Ontologies provide a shared conceptual framework that standardizes vocabulary and defines relationships among concepts, thereby improving communication and collaboration among team members (Studer et al. 1998). Through semantic representation of knowledge, stakeholders can access, interpret, and exchange information more effectively, reducing ambiguity and enhancing mutual understanding.

In distributed software development environments, ontology based systems support collaboration by enabling geographically dispersed teams to access consistent and well-structured knowledge repositories. Such capabilities improve coordination and contribute to more efficient project execution.

Improved Knowledge Reuse

Knowledge reuse is a critical objective of software engineering knowledge management initiatives. Reusing existing knowledge resources reduces duplication of effort, accelerates development activities, and improves software quality. Ontology based repositories facilitate knowledge reuse by organizing information in a structured and semantically meaningful manner.

Developers can retrieve previously documented requirements, software components, design patterns, testing procedures, and project experiences through semantic search mechanisms rather than relying solely on keyword based searches (Rus & Lindvall, 2002). This improves the accuracy of information retrieval and increases the likelihood that valuable organizational knowledge will be reused in future projects.

Knowledge reuse also contributes to cost reduction and productivity improvement by minimizing the need to recreate existing solutions.

Enhanced Decision Making

Software engineering decision making often involves evaluating complex technical and managerial alternatives under conditions of uncertainty. Ontology based knowledge management systems provide decision support by integrating information from multiple sources and representing relationships among relevant concepts.

The semantic structure provided by ontologies enables stakeholders to analyze dependencies, identify potential risks, and evaluate alternative courses of action more effectively. In project management contexts, ontology driven decision support systems can assist in resource allocation, risk assessment, project planning, and process optimization (Aurum et al. 2008).

Furthermore, ontology based reasoning mechanisms can support automated recommendations and intelligent knowledge discovery, thereby improving the quality and consistency of decisions.

Improved Software Quality and Consistency

The formal representation of knowledge through ontologies contributes to greater consistency across software engineering activities. By establishing standardized definitions, procedures, and relationships, ontologies help ensure that software artefacts are developed according to agreed specifications and organizational standards.

In requirements engineering, ontologies reduce ambiguity and improve requirement consistency. In software testing, they facilitate systematic defect classification and quality assurance activities. During maintenance, ontology based documentation improves system understanding and reduces the likelihood of introducing errors during software modification.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

Consequently, ontology driven knowledge management systems contribute to improved software quality, reduced defects, and enhanced maintainability.

Knowledge Preservation and Organizational Learning

Software organizations frequently face knowledge loss resulting from employee turnover, retirement, or project discontinuation. Valuable tacit knowledge may disappear when experienced personnel leave an organization.

Ontology based knowledge repositories help preserve organizational knowledge by converting implicit expertise into structured and reusable forms. These repositories capture lessons learned, best practices, project experiences, and technical knowledge that can be utilized by future teams (Dingsøyr and Conradi, 2002).

The preservation of organizational knowledge promotes continuous learning and enables organizations to build cumulative expertise over time. This capability is particularly important in knowledge intensive industries where innovation and adaptability are critical success factors.

Challenges

Complexity of Ontology Development

Despite their advantages, ontology based knowledge management systems are often difficult and resource intensive to develop. Constructing comprehensive ontologies requires extensive domain expertise, stakeholder involvement, and careful knowledge elicitation processes.

Developers must identify relevant concepts, define relationships, establish constraints, and ensure consistency across the ontology structure. This process can be time consuming and may require specialized ontology engineering skills that are not readily available in many software organizations (Gómez Pérez et al., 2004).

The complexity associated with ontology construction remains one of the most significant barriers to adoption.

Knowledge Acquisition Difficulties

The effectiveness of ontology based systems depends heavily on the quality and completeness of the knowledge they contain. Acquiring knowledge from experts and organizational sources can be challenging because much software engineering knowledge exists in tacit form and is difficult to articulate explicitly.

Stakeholders may also be unwilling or unable to devote sufficient time to knowledge capture activities. Consequently, ontology developers often encounter incomplete, inconsistent, or outdated information during ontology construction and maintenance.

These challenges can limit the accuracy and usefulness of ontology based knowledge repositories.

Scalability and Maintenance Issues

As software organizations grow and generate increasing volumes of information, ontology repositories can become difficult to manage. Large ontologies may contain thousands of concepts and relationships, making them complex to maintain and update.

Changes in business processes, software technologies, and organizational structures often require continuous ontology revision. Failure to maintain ontologies adequately may result in outdated knowledge representations that reduce system effectiveness (Noy & McGuinness, 2001).

Scalability concerns therefore remain a major challenge for long term ontology deployment.

Integration Challenges

Modern software organizations typically utilize a variety of tools and platforms for software development, project management, testing, and documentation. Integrating ontology based knowledge management systems with these

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

heterogeneous environments can be technically challenging.

Differences in data formats, terminology, and system architectures may hinder interoperability. Although Semantic Web standards have improved integration capabilities, practical implementation challenges persist, particularly in large and complex software ecosystems.

Successful ontology adoption often requires substantial investment in system integration and data transformation activities.

User Acceptance and Organizational Resistance

The success of any knowledge management initiative depends largely on user participation and organizational support. Employees may resist ontology based systems if they perceive them as difficult to use, time consuming, or disruptive to established work practices.

Resistance may also arise from concerns about knowledge ownership, additional documentation responsibilities, or uncertainty regarding the benefits of ontology adoption. Without adequate training and management support, organizations may struggle to achieve effective utilization of ontology driven knowledge management systems (Bjørnson and Dingsøyr, 2008).

Therefore, human and organizational factors remain critical determinants of implementation success.

Synthesis of Benefits and Challenges

The literature demonstrates that knowledge management ontologies offer substantial potential for improving software engineering practices through enhanced knowledge sharing, reuse, decision support, software quality, and organizational learning (Table 4). Their ability to provide semantic representations of software development knowledge makes them valuable tools for managing increasingly complex software projects.

However, the successful implementation of ontology based knowledge management systems requires addressing several persistent challenges, including ontology development complexity, knowledge acquisition difficulties, scalability concerns, integration problems, and user acceptance issues. These challenges highlight the need for more practical, scalable, and user friendly ontology solutions that can be effectively deployed within real world software engineering environments.

Understanding both the benefits and limitations of ontology driven knowledge management is essential for guiding future research and supporting broader industrial adoption.

Table 4. Major Benefits and Challenges of Knowledge Management Ontologies in Software Engineering

Category	Key Findings
Benefits	Improved knowledge sharing, knowledge reuse, decision support, software quality, and organizational learning
Technical Challenges	Ontology development complexity, scalability limitations, maintenance burden, and integration difficulties
Organizational Challenges	User resistance, lack of ontology expertise, and knowledge acquisition barriers
Strategic Implications	Need for scalable, intelligent, interoperable, and user friendly ontology frameworks

Source: Synthesize from Rus and Lindvall (2002); Bjørnson and Dingsøyr, (2008); Aurum et al., (2008)

DISCUSSION

The findings of this review demonstrate that ontology-based knowledge management has evolved from a

specialized semantic modelling approach into a strategic mechanism for supporting software engineering activities. Across the reviewed studies, ontologies consistently contributed to improved knowledge representation, sharing, reuse, and decision support throughout the software development lifecycle.

A notable finding is the widespread application of ontologies in requirements engineering, software maintenance, and project management. These domains rely heavily on the effective capture and transfer of knowledge among stakeholders. By providing formal semantic representations, ontologies reduce ambiguity, improve traceability, and facilitate organizational learning. These findings align with previous knowledge management studies that emphazise the importance of structured knowledge repositories in improving software development performance (Bjørnson & Dingsøyr, 2008; Hogan et al. 2021).

The review further reveals that ontology-based approaches increasingly support intelligent software engineering environments. The integration of semantic technologies with artificial intelligence, machine learning, and knowledge graphs is transforming traditional knowledge repositories into intelligent decision-support systems. This trend reflects the broader movement toward data-driven and knowledge-driven software development practices.

Despite these advantages, several challenges continue to constrain adoption. Ontology development remains complex and resource intensive, requiring substantial domain expertise and stakeholder participation. Scalability concerns, interoperability limitations, and user resistance also remain significant barriers. These findings suggest that future research should focus not only on technical improvements but also on organizational and human-centred implementation strategies (Noy et al. 2023).

From a practical perspective, software organizations can benefit from ontology-based knowledge management by improving knowledge preservation, reducing duplication of effort, and enhancing collaboration across development teams. However, successful implementation requires adequate governance structures, training programmes, and integration mechanisms capable of embedding ontology technologies into existing software engineering workflows.

Overall, the evidence suggests that ontology-based knowledge management represents a promising foundation for intelligent software engineering. Nevertheless, broader industrial adoption will depend on the development of scalable, interoperable, and user-friendly ontology frameworks capable of supporting rapidly evolving software development environments (Singh et al. 2023).

FUTURE RESEARCH DIRECTIONS

The findings of this review indicate that knowledge management ontologies have made significant contributions to software engineering by facilitating knowledge representation, sharing, reuse, and decision support. Despite these advancements, several limitations continue to constrain their widespread adoption and practical effectiveness. Existing studies reveal persistent challenges related to ontology development complexity, maintenance requirements, interoperability, scalability, and organizational acceptance. Consequently, further research is required to improve the design, implementation, and sustainability of ontology based knowledge management systems in software engineering environments (Bjørnson and Dingsøyr, 2008; Wongthongtham et al., 2009).

One important research direction involves the integration of artificial intelligence techniques with ontology based knowledge management systems. Traditional ontology development often depends on manual knowledge acquisition and expert intervention, making ontology construction both time consuming and resource intensive. Recent advances in machine learning, natural language processing, and large language models provide opportunities to automate ontology generation, concept extraction, semantic annotation, and ontology evolution (Figure 2). Future studies should investigate hybrid frameworks that combine ontology driven semantic reasoning with artificial intelligence capabilities to support intelligent knowledge discovery, automated decision making, and adaptive learning within software engineering environments (Tamburis et al., 2023).

Another promising area of investigation is the integration of knowledge graphs with ontology based knowledge

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

management systems. Knowledge graphs have emerged as effective mechanisms for representing complex relationships among entities across heterogeneous information sources. Unlike traditional ontology repositories, knowledge graphs enable dynamic and interconnected representations of software engineering knowledge derived from requirements documents, source code repositories, testing databases, issue tracking systems, and project management platforms. Integrating ontologies with knowledge graphs could significantly improve semantic interoperability, traceability analysis, and intelligent information retrieval across software development processes. Future research should therefore explore scalable architectures that combine the formal semantics of ontologies with the flexibility and extensibility of knowledge graph technologies (Pan et al., 2022; Hogan et al., 2021).

The growing adoption of Agile methodologies and DevOps practices presents another important research opportunity. Most existing ontology based knowledge management frameworks were developed within the context of traditional software development models and may not adequately address the dynamic and iterative nature of modern software engineering environments (Aurum et al. 2008). Agile and DevOps practices generate large volumes of rapidly evolving knowledge through sprint planning, continuous integration, continuous delivery, automated testing, and operational monitoring activities. Future studies should focus on developing lightweight and adaptive ontology frameworks capable of supporting real time knowledge capture, sharing, and reuse in fast paced development environments.

Ontology evolution and maintenance remain critical challenges that require sustained research attention. Software engineering knowledge is inherently dynamic because development practices, technologies, standards, and organizational structures continuously evolve. As a result, static ontologies can quickly become outdated, reducing their usefulness and reliability (Noy and McGuinness, 2001). Future investigations should explore automated ontology maintenance techniques that employ machine learning, semantic reasoning, and intelligent agents to identify changes in knowledge structures and update ontology repositories accordingly. Such approaches could enhance ontology scalability and reduce maintenance costs (Felderer et al., 2023).

The literature also highlights the need for greater attention to human and organizational factors affecting ontology adoption. Although many studies focus on technical ontology design, relatively few investigate user experience, usability, stakeholder engagement, and organizational readiness. User resistance has consistently been identified as a barrier to successful knowledge management implementation in software organizations (Bjørnson and Dingsøyr, 2008). Future research should therefore examine how ontology based systems can be designed to improve usability, encourage knowledge sharing behaviours, and align with existing organizational workflows. Human centred ontology engineering approaches may contribute significantly to improving acceptance and long term sustainability.

A further gap identified in the literature concerns the limited empirical validation of ontology based knowledge management systems in industrial settings. Many existing studies remain conceptual or rely on prototype implementations evaluated under controlled conditions. While such studies provide valuable theoretical insights, they offer limited evidence regarding the practical effectiveness of ontology based systems in real world software development environments (Dingsøyr and Conradi, 2002). Future research should prioritize longitudinal case studies, industrial experiments, and large scale evaluations that assess the impact of ontology adoption on software quality, productivity, project performance, and organizational learning outcomes.

Another emerging area involves semantic interoperability among software engineering tools and platforms. Modern software development relies on numerous tools for requirements management, source code control, testing, deployment, project management, and maintenance. The lack of semantic integration among these tools often leads to fragmented knowledge repositories and information silos. Ontology based frameworks have the potential to address these challenges by providing common semantic models that enable seamless information exchange across heterogeneous environments (Euzenat and Shaviko, 2024). Further research is needed to develop interoperable ontology standards capable of supporting integrated software engineering ecosystems.

In addition, future research should explore the development of intelligent knowledge management ecosystems that combine ontologies, knowledge graphs, artificial intelligence, cloud computing, and software analytics technologies (Figure 2). Such ecosystems could support continuous knowledge acquisition, automated reasoning,

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XIII Issue VI June 2026

predictive analytics, and intelligent recommendation services throughout the software development lifecycle. By leveraging advances in semantic technologies and intelligent computing, next generation knowledge management systems could transform software engineering from a largely knowledge intensive activity into a knowledge driven and learning oriented discipline (Tamuris et al., 2023; Noy et al., 2023).

Finally, the future of knowledge management ontologies in software engineering lies in the convergence of semantic technologies, artificial intelligence, and intelligent software development practices. Addressing current technical, organizational, and methodological challenges will be essential for achieving broader industrial adoption and maximizing the benefits of ontology based knowledge management systems. Continued research in these areas is expected to contribute significantly to more efficient, collaborative, and knowledge driven software engineering environments.

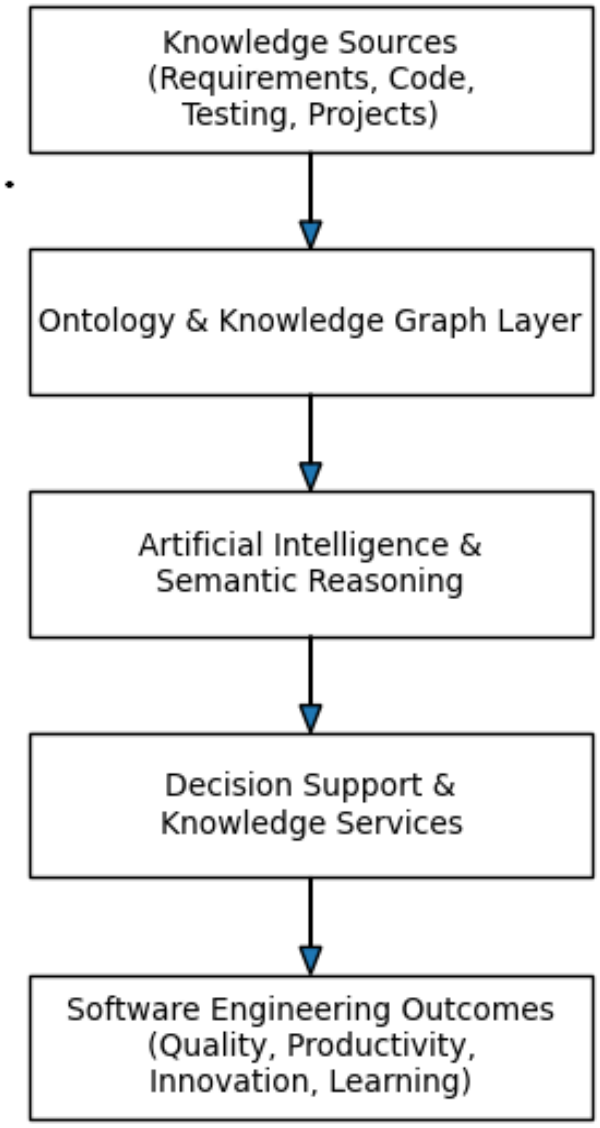

Figure 2: Future Research Framework for Knowledge Management Ontologies in Software Engineering

CONCLUSION

This systematic literature review examined the role of knowledge management ontologies in software engineering, focusing on their applications, benefits, challenges, and future research directions. The review demonstrates that ontology based approaches have become valuable tools for supporting knowledge management activities across the software development lifecycle, including requirements engineering, software design, testing, maintenance, and project management.

The findings indicate that knowledge management ontologies contribute significantly to improved knowledge sharing, knowledge reuse, decision support, software quality, and organizational learning. By providing structured and semantic representations of software engineering knowledge, ontologies facilitate collaboration among stakeholders and enhance the accessibility of organizational knowledge assets. However, their widespread adoption continues to be constrained by challenges such as ontology development complexity, knowledge acquisition difficulties, scalability concerns, integration issues, and user acceptance barriers.

The review also highlights emerging opportunities associated with artificial intelligence, knowledge graphs, semantic interoperability, and intelligent software engineering environments. These developments have the potential to enhance ontology capabilities and address many of the limitations identified in existing studies. Nevertheless, further empirical research is needed to validate ontology based knowledge management systems in real world industrial settings and to evaluate their long term organizational impact.

Consequently, knowledge management ontologies provide a promising foundation for supporting knowledge driven software engineering. Continued advances in semantic technologies and intelligent computing are expected to strengthen their role in improving software development processes and fostering more effective knowledge management practices within software organizations.

REFERENCES

1. Alavi, M., and Leidner, D. E. (2001). Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues. MIS Quarterly, 25(1), 107–136. https://doi.org/10.2307/3250961
2. Alrumaih, H., Mirza, A., and Alsalamah, H. (2020). Domain Ontology for Requirements Classification in Requirements Engineering Context. IEEE Access, 8, 89899-89908. doi:https://doi.org/10.1109/ACCESS.2020.2993838
3. Ammirato, S., Linzalone, R., and Felicetti, A. M. (2021). Knowledge Management in Pandemics. A Critical Literature Review. Knowledge Management Research and Practice, 19 (4), 415–426.
4. Aurum, A., Daneshgar, F., and Ward, J. (2008). Investigating Knowledge Management Practices in Software Development Organizations. Information and Software Technology, 50(6), 511–533. https://doi.org/10.1016/j.infsof.2007.05.005
5. Bjørnson, F. O., & Dingsøyr, T. (2008). Knowledge Management in Software Engineering: A Systematic Review of Studied Concepts, Findings and Research Methods used. Information and Software Technology, 50(11), 1055–1068. https://doi.org/10.1016/j.infsof.2008.03.006
6. Chaccour, C., Saad, W., Debbah, M., Han, Z., and Poor, H. V. (2024). Less Data, More Knowledge: Building Next Generation Semantic Communication Networks. IEEE Communications Surveys & Tutorials. doi:https://doi.org/10.1109/COMST.2024.3412852
7. De Nicola, A., and Villani, M. L. (2021). Smart City Ontologies and Their Applications: A Systematic Literature Review. Sustainability, 13(10). doi:https://doi.org/10.3390/su13105578
8. Dingsøyr, T., & Conradi, R. (2002). A Survey of Case Studies of the Use of Knowledge Management In Software Engineering. International Journal of Software Engineering and Knowledge Engineering, 12(4), 391–414. https://doi.org/10.1142/S0218194002001024
9. Euzenat, J., and Shvaiko, P. (2024). Ontology Matching (3rd ed.). Springer. https://doi.org/10.1007/978-3-031-60694-2
10. Farghaly, K., Soman, R. K., and Zhou, S. A. (2023). The Evolution Of Ontology In AEC: A Two-Decade Synthesis, Application Domains, and Future Directions. Journal of Industrial Information Integration, 36. doi:https://doi.org/10.1016/j.jii.2023.100519
11. Fauzan, R., Hamidi, M, Z., Safitri, W. A., Siahaan, D. O., and Karimi, M. I. (2025). Ontology in Requirements Software Development Method: A Systematic Literature Review. Journal of Information Technology and Cyber Security, 3(1), 14-32. https://doi.org/10.30996/jitcs.12297
12. Felderer, M., Méndez Fernández, D., Travassos, G. H., Kalinowski, M., and Sarro, F. (2023). Artificial Intelligence and Data-Driven Software Engineering: Research and Practice. Journal of Systems and Software, 198, 111588. https://doi.org/10.1016/j.jss.2022.111588
13. Fraga, A. L., Vegetti, M. and Leone, H. P. (2020). Ontology-Based Solutions for Interoperability

among Product Lifecycle Management Systems: A Systematic Literature Review. Journal of Industrial Information Integration, 20. doi:https://doi.org/10.1016/j.jii.2020.100176
14. Fu, C., Jiang, H., and Chen, X. (2022). Modeling of an Enterprise Knowledge Management System Based on Artificial Intelligence. Knowledge Management Research & Practice, 1–13. doi:https://doi.org/10.1080/14778238.2020.1854632
15. Gómez Pérez, A., Fernández López, M., and Corcho, O. (2004). Ontological Engineering. London, United Kingdom: Springer. https://doi.org/10.1007/b106454
16. Guizzardi, G. and Guarino, N. (2024). Explanation, Semantics, and Ontology. Data & Knowledge Engineering, 153. doi:https://doi.org/10.1016/j.datak.2024.102325
17. Happel, H. J. and Seedorf, S. (2006). Applications of Ontologies In Software Engineering. In Proceedings of the International Workshop on Semantic Web Enabled Software Engineering (SWESE 2006) (pp. 1–14).
18. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngonga Ngomo, A. C., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S., and Zimmermann, A. (2021). Knowledge Graphs. ACM Computing Surveys, 54(4), 1–37. https://doi.org/10.1145/3447772
19. Husáková, M., and Bureš, V. (2020). Formal Ontologies in Information Systems Development: A Systematic Review. Information, 11, 66: 1 – 18. doi:https://doi.org/10.3390/info11020066
20. Idrees H., XU, J., Haider, S., Tehseen S. (2023). A Systematic Review of Knowledge Management and New Product Development Projects: Trends, Issues and Challenges. Journal of Innovation and Knowledge, 8, 100350, 1 – 10.
21. Kitchenham, B., and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. EBSE Technical Report EBSE-2007-01, Keele University and Durham University.
22. McGuinness, D. L., & van Harmelen, F. (2004). OWL Web Ontology Language Overview. World Wide Web Consortium (W3C) Recommendation. https://www.w3.org/TR/owl-features/
23. Musen, M. A. (2015). The Protégé Project: A Look Back and a Look Forward. AI Matters, 1(4), 4–12. https://doi.org/10.1145/2757001.2757003
24. Noy, N. F. and McGuinness, D. L. (2001). Ontology Development 101: A Guide to Creating your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05.
25. Olan, F., Arakpogun, E. O., Suklan, J., Nakpodia, F., Damij, N., and Jayawickrama, U. (2022). Artificial Intelligence and Knowledge Sharing: Contributing Factors to Organizational Performance. Journal of Business Research, 145, 605-615. doi:https://doi.org/10.1016/j.jbusres.2022.03.008
26. Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., ... Moher, D. (2021). The PRISMA 2020 Statement: An Updated Guideline for Reporting Systematic Reviews. BMJ, 372, n71. https://doi.org/10.1136/bmj.n71
27. Pan, J. Z., Vetere, G., Gomez-Perez, J. M., & Wu, H. (Eds.). (2022). Exploiting Linked Data and Knowledge Graphs in Large Organizations. Springer. https://doi.org/10.1007/978-3-030-82085-1
28. Petersen, K., Vakkalanka, S., & Kuzniarz, L. (2015). Guidelines for Conducting Systematic Mapping Studies in Software Engineering: An Update. Information and Software Technology, 64, 1–18. https://doi.org/10.1016/j.infsof.2015.03.007
29. Pliatsios, A., Kotis, K., and Goumopoulos, C. (2023). A Systematic Review on Semantic Interoperability in the IoE-enabled Smart Cities. Internet of Things, 22. doi:https://doi.org/10.1016/j.iot.2023.100754
30. Rus, I. and Lindvall, M. (2002). Knowledge Management in Software Engineering. *IEEE Software*, 19(3), 26–38. https://doi.org/10.1109/MS.2002.1003450
31. Said, A., Zhao, Y., Derr, T., Shabbir, M., Abbas, W., and Koutsoukos, X. (2023). A Survey of Graph Unlearning. doi:https://doi.org/10.48550/arXiv.2310.02164
32. Studer, R., Benjamins, V. R., and Fensel, D. (1998). Knowledge Engineering: Principles and Methods. Data and Knowledge Engineering, 25(1-2), 161–197. https://doi.org/10.1016/S0169-023X(97)00056-6
33. Tamburis, O., Esposito, C., and Taticchi, C. (2023). Artificial Intelligence and Ontology Engineering:

Opportunities and Challenges for Semantic Knowledge Systems. Information Systems Frontiers, 25(5), 1821–1838.

34. Tudorache, T. (2020). Ontology Engineering: Current State, Challenges, and Future Directions. Semantic Web, 11(1), 125-138. doi:https://doi.org/10.3233/SW-190382
35. Wongthongtham, P., Chang, E., Dillon, T., and Hussain, F. (2009). Ontology Based Knowledge Management Systems in Software Engineering. Journal of Universal Computer Science, 15(15), 2906–2933.