UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

MASTER THESIS

IN PARTIAL FULFILLEMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MAÎTRISE EN GESTION DE PROJET (AVEC MÉMOIRE) - 3153

BY
MAMADOU DIOMANDE

PROJECT MANAGEMENT AND BUSINESS TECHNOLOGY MANAGEMENT
SEMANTIC INTEGRATION FOR AUTOMATED PROJECT TEAM COMPOSITION

DECEMBER 18, 2024

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED
BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Stéphane GAGNON, Thesis Director
Associate Professor
Université du Québec en Outaouais
Department of Administrative Sciences

Mrs. Péricles DE LIMA SOBREIRA, President of the jury
Associate Professor
Université du Québec en Outaouais
Department of Computer Science and Engineering

Mrs. Sabrina Azzi, External Evaluator
Associate Professor
University of the West of Scotland - United Kingdom
School of Business & Creative Industries

AKNOWLEDGEMENT

Embarking on a Master's degree with a research option is a challenging endeavor, and for me, it marked a return to academia with new opportunities and obstacles. I would like to take this opportunity to express my profound gratitude to certain individuals.

First and foremost, I extend my deepest appreciation to Ohima, my mother, who first taught me how to read. The light she ignited continues to shine brightly, guiding me throughout my life and academic pursuits. I am equally grateful to my father, Bouaké, for his constant encouragement, support, and invaluable advice.

To my wife, Naminata, who has been an unwavering pillar of support, your dedication has been essential to the completion of this work. I would also like to acknowledge the cooperation of my children, Awa, Khalil, Khadija, and Aboubakr, whose patience and understanding were indispensable during this process.

I wish to convey my deepest thanks to Professor Stéphane Gagnon, my thesis director, for his steadfast support, insightful guidance, and encouragement throughout my studies. His expertise and commitment to my academic and research development have been crucial to the successful completion of this thesis.

I also express my gratitude to Professor Péricles de Lima Sobreira, President of the jury, for her thoughtful observations and constructive feedback, which have significantly enriched this work. My sincere thanks go to Sabrina Azzi, for her willingness to serve as an external examiner and for her valuable contributions to the evaluation of my research.

Finally, I would like to express my heartfelt appreciation to my Professor of School of Thought in Project Management Jacques-Bernard Gauthier. His guidance has greatly empowered my critical thinking and shaped my approach to project management, for which I am truly grateful.

REMERCIEMENT

S'engager dans un programme de Master avec une option de recherche est un défi, et pour moi, cela a marqué un retour dans le milieu académique avec de nouvelles opportunités et obstacles. Je tiens à saisir cette occasion pour exprimer ma profonde gratitude envers certaines personnes.

Tout d'abord, j'exprime ma plus grande reconnaissance à Ohima, ma mère, qui m'a appris à lire. La lumière qu'elle a allumée continue de briller intensément, me guidant tout au long de ma vie et de mes études. Je suis également profondément reconnaissant à mon père, Bouaké, pour son soutien constant, ses encouragements et ses conseils précieux.

À ma femme, Naminata, qui a été un pilier inébranlable de soutien, votre dévouement a été essentiel à l'achèvement de ce travail. Je souhaite également reconnaître la coopération de mes enfants, Awa, Khalil, Khadija, et Aboubakr, dont la patience et la compréhension ont été indispensables tout au long de ce processus.

Je tiens à adresser mes plus sincères remerciements au Professeur Stéphane Gagnon, mon directeur de thèse, pour son soutien indéfectible, ses conseils avisés et ses encouragements tout au long de mes études. Son expertise et son engagement dans le développement de mon parcours académique et de ma recherche ont été déterminants pour la réussite de ce travail.

Je souhaite également exprimer ma gratitude au Professeur Péricles de Lima Sobreira, Présidente du jury, pour ses observations réfléchies et ses retours constructifs qui ont enrichi considérablement ce travail.
Je remercie sincèrement Sabrina Azzi, pour avoir accepté de servir en tant qu'examinateur externe et pour ses précieuses contributions à l'évaluation de mes recherches.

Enfin, je tiens à exprimer ma profonde reconnaissance à mon professeur à l'École de Pensée en Gestion de Projet, Jacques-Bernard Gauthier. Ses conseils ont considérablement renforcé ma pensée critique et façonné mon approche de la gestion de projet, pour cela, je lui suis sincèrement reconnaissant.

Intégration Sémantique de la Gestion de Projet et de la Gestion des Technologies de l'Information pour une Composition Automatisée des Équipes de Projet.

Mamadou Diomande

RÉSUMÉ

Au cours des 20 dernières années, les méthodes agiles se sont développées au-delà de leurs origines en ingénierie logicielle (SE) et ont été adaptées à la plupart des autres secteurs de la gestion de projet (PM). Il subsiste des défis majeurs pour harmoniser les aspects clés des méthodes PM avec les principes agiles, tels que l'intégration et la gestion formelle des contraintes de livraison (par exemple, budget, intégration, calendrier) et des mécanismes de contrôle (par exemple, responsabilité, avantages, risques). De plus, de nombreux chefs de projet revendiquent souvent l'agilité, mais leur mise en œuvre des méthodes agiles repose sur seulement quelques principes de base, laissant de nombreuses améliorations inexploitées. Le manque de personnalisation des processus de projet est donc une raison clé du faible impact relatif des méthodes agiles dans les environnements PM plus traditionnels.

Nous proposons d'améliorer la personnalisation des processus de projet en nous appuyant sur les ontologies et les technologies de graphes de connaissances (KG). L'utilisation des ontologies PM et du raisonnement sémantique permettrait une représentation plus ouverte et flexible des composants des méthodes PM et l'intégration des caractéristiques dynamiques de l'agilité avec les caractéristiques plus statiques des préoccupations entourant l'environnement de projet. Notre proposition s'appuie sur l'évolution naturelle par laquelle les équipes s'approprient les méthodes agiles à travers un processus organique et auto-organisé, tout en veillant à ce qu'elles restent conformes aux politiques de l'entreprise et remplissent toutes les obligations de gouvernance du projet.

En suivant une méthodologie de recherche par conception d'action (ADR), nous intégrons d'abord plusieurs ontologies PM au sein de l'ontologie du corpusH de connaissances (BOK) de la gestion des technologies de l'information (BTM). La BTM est un domaine de recherche transdisciplinaire en pleine émergence et une profession unifiée à la croisée des disciplines de l'entreprise, de l'informatique et de l'ingénierie. Elle cherche à fournir un cadre intégré pour l'utilisation stratégique

des TI et la direction des organisations numériques. Cela nous permet d'avoir une représentation formelle de la PM ainsi qu'un large éventail d'exigences, de rôles, de tâches, de processus et de processus de PM TI. Nous intégrons à notre ontologie PM-BTM une représentation KG de la norme SE "Essence", adoptée en 2014 par l'Object Management Group (OMG), une évolution de sa norme SE de 2008 intitulée Software & Systems Process Engineering Metamodel (SPEM 2.0). Bien que le méta-modèle Essence soit axé sur 7 "alphas" ou composants touchant le client, la solution et l'entreprise, nous intégrons 6 alphas supplémentaires touchant les contraintes et les contrôles, comme décrit dans la plupart des normes PM. Nous intégrons également cet Essence Étendu avec deux des initiatives PM open-source les plus populaires : la méthode de gestion de projet ouverte (Open PM2) développée par la Commission européenne, et la plate-forme Open Project qui sert de l'une des alternatives les plus rapides aux logiciels PM commerciaux.

Enfin, nous développons des requêtes basées sur des règles en SWRL et SQRWL qui représentent les paramètres de décision typiques de personnalisation PM. Nous testons ces requêtes sur notre ontologie PM, en fournissant d'abord un ensemble d'exigences PM TI à partir de projets réels complétés, puis en interrogeant l'ontologie pour vérifier le processus logique le plus probable recommandé. Le processus est considéré comme une norme de référence optimale, contre laquelle les paramètres de mise en œuvre du processus réel sont comparés. Nous analysons les résultats en nous basant sur la mesure F pour évaluer la qualité de nos capacités d'inférence ontologique. Cette recherche pourrait avoir un impact significatif sur l'assurance d'une plus grande pertinence de la recherche académique pour la pratique professionnelle et sur l'intérêt des praticiens pour les connaissances académiques. L'utilisation du raisonnement sémantique pour la personnalisation des processus de projet pourrait conduire à une adoption plus fluide des méthodes agiles. De plus, nos résultats pourraient contribuer à un cadre plus systématique, exhaustif et évolutif pour les normes de pratique professionnelle, en service pour soutenir la gestion des ressources humaines (GRH) PM TI. Le projet vise à rendre les normes agiles, PM, et BTM BOK facilement accessibles, personnalisables et réutilisables pour la prise de décision par les professionnels PM TI, les employeurs, l'enseignement supérieur et d'autres associations impliquées dans les normes, certifications et accréditations liées aux TI.

Project Management and Business Technology Management Semantic Integration for Automated Project Team Composition

Mamadou Diomande

ABSTRACT

Over the past 20 years, Agile methods have grown beyond their origins in Software Engineering (SE) and have been adapted to most other sectors in Project Management (PM). There remain key challenges to harmonizing core aspects of PM methods with Agile principles, such as the integration and formal management of delivery constraints (e.g., budget, integration, schedule) and control mechanisms (e.g., accountability, benefits, risks). As well, agility is often claimed by many project leaders, but their implementation of agile methods rests on merely a few basic principles, leaving numerous improvements unexplored. Lack of project process customization is therefore a key reason for the relative low impact of agile methods in more traditional PM environments.

We propose to improve project process customization by relying on Ontologies and Knowledge Graph (KG) technologies. The use of PM ontologies and semantic reasoning would allow for a more open and flexible representation of PM methods components and integrating the dynamic features of agility with the more static features from concerns surrounding the project environment. Our proposal builds upon the natural evolution by which teams take ownership of Agile methods through an organic and self-organizing process, while ensuring they remain compliant to enterprise policies and meet all obligations for project governance.

Following an [1]Action Design Research (ADR) methodology, we first integrate several PM ontologies within the Business Technology Management (BTM) Body of Knowledge (BOK) ontology. BTM is a rapidly emerging trans-disciplinary research area and unified profession at the

[1] Action Design Research (ADR**)** is chosen over Design Science Research (DSR) because ADR aligns more closely with the goals of the research, particularly its focus on practical application and iterative improvement in the context of project management. Action Design Research (ADR) is well-suited to situations where the research is intended not just to create knowledge but also to address specific, real-world problems in a given context. ADR emphasizes solving practical issues through collaboration with stakeholders and designing solutions that can be directly applied and tested in the real world. Design Science Research (DSR) is often more focused on the creation and evaluation of artifacts (such as models, methods, or software tools) for the advancement of theoretical knowledge.

crossroads of business, computing, and engineering disciplines. It seeks to provide an integrated framework for the strategic use of IT and leading digital organizations. This allows us to have a formal representation of PM along with a wide array of IT PM requirements, roles, tasks, processes, and processes. We integrate to our PM-BTM ontology a KG representation of the "Essence" SE Standard, adopted in 2014 by the Object Management Group (OMG), an evolution from its 2008 SE standard entitled Software & Systems Process Engineering Metamodel (SPEM 2.0). While the Essence metamodel is focused on 7 "alphas" or components touching on Customer, Solution, and Endeavour, we integrate 6 additional alphas touching on Constraints and Controls, as described in most PM standards. We further integrate this Extended Essence with two of the most popular open- source PM initiatives: the Open Project Management Method (Open PM2) developed by the European Commission, and the Open Project platform that serves as one of the fastest growing alternatives to commercial PM software.

Finally, we develop rule-based queries in SWRL and SQRWL that represent typical PM customization decision parameters. We test these queries on our PM ontology, first supplying a set of IT PM requirements from actual completed projects, and then querying the ontology to verify the most likely logical process recommended. The process is considered an optimal gold standard, against which actual process implementation parameters are compared. We analyze results based on F-measure to evaluate the quality of our ontology inference capabilities. This research may have a significant impact on ensuring greater relevance of academic research for professional practice, and interest of practitioners for academic knowledge as well. The use of semantic reasoning for project process customization may lead to more seamless adoption of agile methods. As well, our results may help a more systematic, exhaustive, and evolving framework for professional practice standards, in service to support IT PM Human Resources Management (HRM). The project aims to make agile, PM, and BTM BOK standards easily accessible, customizable, and reusable for decision-making by IT PM professionals, employers, higher education, and other associations involved with IT-related standards, certification, and accreditation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ABET	Accreditation Board for Engineering and Technology
ADR	Action Design Research
AHP	Analytic Hierarchy Process
AI	Artificial Intelligence
API	Application Programming Interface
APM	Association for Project Management
APMBOK	APM Body of Knowledge
BBO	BPMN 2.0 Based Ontology
BIE	Building, Intervention, and Evaluation
BIM	Building Information Modeling
BOK	Body of Knowledge
BPM	Business Process Management
BPMN	Business Process Model and Notation
BTM	Business Technology Management
CA	Concept Algebra
CDMP	Certified Data Management Professional
CI	Continuous Integration

COBIT	Control Objectives for Information and Related Technologies
CPM	Critical Path Method
CRM	Customer Relationship Management
CSM	Certified ScrumMaster
CSP	Certified Scrum Professional
DevOps	Development and Operations
EC	European Commission
EF	Essence Framework
EHR	Electronic Health Record
EI	Emotional Intelligence
Essence	Essence framework for software engineering methods
FinTech	Financial Technology
F-measure	A metric used to assess the accuracy of a model
HealthTech	Healthcare Technology
HRM	Human Resources Management
IDEs	Integrated Development Environments
IEEE	Institute of Electrical and Electronics Engineers
IS	Information Systems
ISO	International Organization for Standardization
ISTQB	International Software Testing Qualifications Board
IT	Information Technology
KG	Knowledge Graph
KMP	Kanban Management Professional
MCDA	Multi Criteria Decision Analysis
ML	Machine Learning
OMaSE	Organization based Multiagent System Engineering
OMG	Object Management Group
OWL	Web Ontology Language
PM	Project Management
PM²	Project Management Methodology developed by the EC

PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
PMP	Project Management Professional
PRINCE2	PRojects IN Controlled Environments 2
ProjOnto	Project Ontology
QA	Quality Assurance
RD	Relationship Diversity
RDF	Resource Description Framework
ROI	Return on Investment
SAFe	Scaled Agile Framework
SD	Schema Deepness
SE	Software Engineering
SEMAT	Software Engineering Method and Theory
SEON	Software Engineering Ontology Network
SMC	Scrum Master Certified
SPARQL	SPARQL Protocol and RDF Query Language
SRO	Scrum Reference Ontology
SWEBOK	Software Engineering Body of Knowledge
SWRL	Semantic Web Rule Language
TOPSIS	Technique for Order Preference by Similarity to Ideal Solution
UX	User Experience
XP	Extreme Programming

1 INTRODUCTION

1.1 Problem Statement

Over the past 20 years, Agile methods have grown beyond their origins in Software Engineering (SE) and have been adapted to most other sectors in Project Management (PM) (Gemino et al., 2023). A study by (Paasivaara et al., 2018) highlights that organizations often encounter difficulties when scaling Agile practices to large projects, especially in aligning Agile methodologies with existing PM frameworks. Similarly, (Dikert et al., 2016) identify challenges in integrating Agile approaches with traditional PM constraints, noting issues in synchronizing Agile processes with established control mechanisms. Additionally, research by (Eklund et al., 2014) discusses the complexities of applying Agile methods in large-scale settings, emphasizing the need for effective coordination to manage delivery constraints and control mechanisms. As well, agility is often claimed by many project leaders, but their implementation of agile methods rests on merely a few basic principles, leaving numerous improvements unexplored (Conboy & Carroll, 2019); (Koi-Akrofi et al., 2019). Lack of project process customization is therefore a key reason for the relative low impact of agile methods in more traditional PM environments (Menon et al., 2021).

1.2 Research Objectives

We propose to improve project process customization by relying on Ontologies and Knowledge Graph (KG) technologies. The use of PM ontologies and semantic reasoning would allow for a

more open and flexible representation of PM methods components and integrating the dynamic features of agility with the more static features from concerns surrounding the project environment (Wand & Weber, 2002); (Gruber, 1995). Our proposal builds upon the natural evolution by which teams take ownership of Agile methods through an organic and self-organizing process, while ensuring they remain compliant to enterprise policies and meet all obligations for project governance (Boehm & Turner, 2003); (Conboy, 2009). This research may have a significant impact on ensuring greater relevance of academic research for professional practice, and interest of practitioners for academic knowledge as well (Hevner et al., 2004). The use of semantic reasoning for project process customization may lead to more seamless adoption of agile methods (Mendling et al., 2020). As well, our results may help a more systematic, exhaustive, and evolving framework for professional practice standards, in service to support IT PM Human Resources Management (HRM) (Kerzner, 2017). The project aims to make agile, PM, and BTM BOK standards easily accessible, customizable, and reusable for decision-making by IT PM professionals, employers, higher education, and other associations involved with IT-related standards, certification, and accreditation (PMI, 2021).

2 LITERATURE REVIEW

2.1 From tradition to agility

The agile software development method is an emerging approach in software engineering advocated by 17 practitioners and now practiced by thousands of professionals (Misra et al., 2012). This project management methodology arose from the observation that many projects fail to achieve their objectives despite implemented techniques (Joslin & Müller, 2016) ; (Flyvbjerg, 2017). Prioritizing human interactions over processes and tools, it ensures project teams a healthy balance between their professional and personal lives (Misra et al., 2012). However, this does not mean that customer satisfaction is sidelined; on the contrary, these requirements are taken into account even at the last moment (Misra et al., 2012). Furthermore, this method encourages teams to prioritize customer satisfaction at the core of their objectives to eliminate any excess, thereby saving time and budget (Kropp & Meier, 2017). The self-management of agile teams is a sufficient condition to guarantee customer satisfaction while maintaining business requirements (cost-quality-time) (Moe et al., 2012). Nevertheless, the agile method remains poorly understood or

subject to abuse, making it difficult to recruit the right human resources (Conboy, 2009). Organizing as a self-organized system is very different from a hierarchical system, as it becomes based on levels of authority where adaptation can be implemented by top management (Plowman & Duchon, 2008) This analytical approach to self-managed organizations or project teams is characterized by an organizational willingness to reduce vertical transactions and encourage horizontal ones (Kash & Rycroft, 2002); (Lowe et al., 2012) ; (Object Management Group (OMG), 2018). (Luhmann, 2002) asserts that the organization must play its part by granting project teams a degree of self-management. Yet, many organizations remain opposed to this type of organization (Parker et al., 2015); for him, the main reason for this reluctance is the fear of losing direct control over the teams.

2.2 Knowledge discipline of Project Management

Project management has evolved significantly over the decades, encompassing various methodologies and standards aimed at ensuring successful project delivery (European Commission., 2018). This review provides a comprehensive analysis of traditional project management disciplines, emphasizing the knowledge domains delineated in prominent standards such as PMBOK (Project Management Body of Knowledge), PM2 (Project Management Methodology), and APMBOK (APM Body of Knowledge), among others. The review synthesizes findings from a broad range of academic sources, journals, articles, and case studies, highlighting the similarities, differences, and contributions of these frameworks to the field of project management (Chin & Spowage, 2012) (Fiampolis & Acaster, 2015).

2.2.1 Evolution of Project Management Frameworks

The discipline of project management has its roots in the early 20th century, with the development of the Gantt chart by Henry Gantt and the Critical Path Method (CPM) by DuPont (Kerzner, 2017; PMI, 2021). These early tools, which focused on scheduling and task dependencies, laid the foundation for modern project management practices (Kerzner, 2017). In contrast, the establishment of the Project Management Institute (PMI) in 1969 marked a broader shift toward

standardizing and professionalizing project management through frameworks like PMBOK, which focus not only on tools but also on methodologies and processes (PMI, 2021).

PMBOK provides a comprehensive framework for managing projects, emphasizing process groups and knowledge areas (PMI, 2021). It is divided into five process groups : Initiating, Planning, Executing, Monitoring and Controlling, and Closing—and ten knowledge areas, such as Integration, Scope, Time, and Risk Management (PMI, 2021). This structured approach provides detailed guidance applicable across industries. Comparatively, PM2, developed by the European Commission, also emphasizes simplicity and adaptability but introduces a more streamlined structure with four phases: Initiating, Planning, Executing, and Closing (European Commission., 2018). Unlike PMBOK's focus on extensive knowledge areas, PM2 prioritizes governance, lifecycle management, and quality assurance as foundational pillars (European Commission., 2018; PMI, 2021).

The APMBOK, from the Association for Project Management, aligns more closely with PMBOK in its comprehensive scope, incorporating 12 knowledge areas, such as Finance, Change, and Professionalism, in addition to Integration and Scope (APM, 2019). However, APMBOK differentiates itself by explicitly addressing professionalism and change management, areas that are often implied but not distinct in PMBOK (APM, 2019; PMI, 2021). While PM2 emphasizes practical adaptability and ease of use, APMBOK provides a deeper dive into project leadership and the strategic alignment of projects within organizations(APM, 2019; European Commission., 2018).

2.2.2 Knowledge Domains in Project Management

[2]The knowledge domains in project management frameworks are critical to understanding the comprehensive nature of managing projects (Belton & Stewart, 2002). This section delves into the

[2] Project management frameworks, such as those outlined by the Project Management Institute (PMI), emphasize various knowledge domains that provide the foundational structure for managing complex projects. These domains, including areas like integration, scope, time, cost, quality, risk, and stakeholder management, help ensure that all aspects of a project are planned, executed, monitored, and controlled. By addressing each of these domains, project managers can ensure comprehensive oversight and better alignment with organizational goals, ultimately enhancing the chances of project success.

key knowledge domains as defined by PMBOK, PM2, and APMBOK, comparing their approaches and contributions to the field. Project management frameworks like PMBOK, PM2, and APMBOK emphasize various knowledge domains to address the multifaceted challenges of project management. These domains, which include areas such as Integration, Scope, Time, Cost, Quality, and Risk Management, offer structured guidance to ensure that all aspects of a project are managed systematically (APM, 2019; PMI, 2021). This section delves into the key knowledge domains as defined by PMBOK, PM2, and APMBOK, comparing their approaches and contributions to the field.

2.2.2.1 Integration Management

Integration Management plays a central role in project management by ensuring that all aspects of a project work together seamlessly. This involves creating a unified plan that aligns objectives, resources, and processes (APM, 2019; PMI, 2021). PMBOK emphasizes the development of a Project Charter, Project Management Plan, and the management of project knowledge, changes, and project closure (PMI, 2021). PM2, on the other hand, focuses on the initiation and planning processes, ensuring that project objectives align with organizational goals (European Commission., 2018). APMBOK aligns closely with PMBOK, emphasizing the need for cohesive integration of project elements but adds a distinct focus on professional conduct and ethics in integration processes (APM, 2019).

2.2.2.2 Scope Management

Scope Management involves defining and controlling what is included in the project. PMBOK outlines processes such as Plan Scope Management, Collect Requirements, Define Scope, Create WBS (Work Breakdown Structure), Validate Scope, and Control Scope (PMI, 2021). PM2 simplifies this into scope definition, detailing what is included and excluded from the project (European Commission., 2018). APMBOK's approach is similar to PMBOK but places additional emphasis on the iterative nature of scope management and the importance of stakeholder engagement throughout the process (APM, 2019).

2.2.2.3 Time Management

Time Management, or Schedule Management, involves planning and controlling the project schedule. PMBOK includes processes such as Plan Schedule Management, Define Activities, Sequence Activities, Estimate Activity Durations, Develop Schedule, and Control Schedule (PMI, 2021). PM2 streamlines this by focusing on defining, scheduling, and managing project activities (European Commission., 2018). APMBOK also emphasizes detailed schedule management, incorporating techniques like Gantt charts and Critical Path Method, while highlighting the importance of adaptive planning in dynamic project environments (APM, 2019).

2.2.2.4 Cost Management

Cost Management encompasses planning, estimating, budgeting, and controlling project costs. PMBOK includes processes such as Plan Cost Management, Estimate Costs, Determine Budget, and Control Costs (PMI, 2021). PM2 integrates cost management into its broader financial management framework, focusing on budget planning and monitoring (European Commission., 2018). APMBOK aligns closely with PMBOK, emphasizing comprehensive cost estimation and control mechanisms, with a strong focus on financial risk management (APM, 2019).

2.2.2.5 Quality Management

Quality Management ensures that the project meets the required standards. PMBOK outlines processes like Plan Quality Management, Manage Quality, and Control Quality (PMI, 2021). PM2 incorporates quality management into its project control phase, emphasizing continuous improvement (European Commission., 2018). APMBOK expands on this by incorporating best practices from quality management standards such as ISO 9001, emphasizing the integration of quality into all project processes and the importance of stakeholder satisfaction (APM, 2019).

2.2.2.6 Human Resource Management

Human Resource Management focuses on organizing, managing, and leading the project team. PMBOK includes processes such as Plan Resource Management, Estimate Activity Resources, Acquire Resources, Develop Team, Manage Team, and Control Resources (PMI, 2021). PM2 integrates human resource management into its project execution phase, emphasizing team development and performance management (European Commission., 2018). APMBOK emphasizes the importance of leadership, team dynamics, and the development of individual competencies, with a focus on professional development and ethical conduct (APM, 2019).

2.2.2.7 Communications Management

Communications Management involves ensuring timely and appropriate generation, collection, dissemination, storage, and ultimate disposition of project information. PMBOK outlines processes such as Plan Communications Management, Manage Communications, and Monitor Communications (PMI, 2021). PM2 emphasizes the importance of communication planning and stakeholder engagement throughout the project lifecycle (European Commission., 2018). APMBOK aligns with PMBOK but places additional emphasis on the strategic importance of communication in achieving project objectives and stakeholder satisfaction (APM, 2019).

2.2.2.8 Risk Management

Risk Management involves identifying, analyzing, and responding to project risks. PMBOK includes processes such as Plan Risk Management, Identify Risks, Perform Qualitative Risk Analysis, Perform Quantitative Risk Analysis, Plan Risk Responses, Implement Risk Responses, and Monitor Risks (PMI, 2021). PM2 integrates risk management into its overall project governance framework, emphasizing proactive risk identification and mitigation (European Commission., 2018). APMBOK emphasizes the importance of both risk and opportunity management, advocating for a balanced approach to managing uncertainties in projects (APM, 2019).

2.2.2.9 Procurement Management

Procurement Management involves acquiring goods and services from external sources. PMBOK outlines processes such as Plan Procurement Management, Conduct Procurements, Control Procurements, and Close Procurements (PMI, 2021). PM2 includes procurement as part of its project management processes, focusing on procurement planning and contract management (European Commission., 2018). APMBOK aligns with PMBOK but adds a focus on strategic procurement, emphasizing the importance of aligning procurement strategies with organizational goals and ensuring ethical procurement practices (APM, 2019).

2.2.2.10 Stakeholder Management

Stakeholder Management involves identifying and engaging stakeholders throughout the project lifecycle. PMBOK includes processes such as Identify Stakeholders, Plan Stakeholder Engagement, Manage Stakeholder Engagement, and Monitor Stakeholder Engagement (PMI, 2021). PM2 emphasizes stakeholder engagement as a continuous process, integral to all phases of the project (European Commission., 2018). APMBOK highlights the importance of understanding stakeholder needs and expectations, advocating for transparent and ethical stakeholder management practices (APM, 2019).

2.2.3 Comparison of Project Management Standards

The landscape of project management is defined by several established standards and methodologies that guide practitioners in the effective planning, execution, and completion of projects. Among these, PMBOK (Project Management Body of Knowledge), PM2 (Project Management Methodology), and APMBOK (APM Body of Knowledge) stand out as influential frameworks. This section delves deeper into the comparison of these standards, drawing from well-known articles, case studies, and academic sources to highlight their unique characteristics, strengths, and areas of application.

2.2.3.1 PMBOK (Project Management Body of Knowledge)

PMBOK, developed by the Project Management Institute (PMI), is one of the most comprehensive and widely recognized project management standards. It is structured around five process groups and ten knowledge areas, providing detailed guidelines and best practices for project management.

2.2.3.1.1 PMBOK Strengths, Applications and Limitation of PMBOK

PMBOK's detailed and exhaustive nature makes it a valuable resource for large, complex projects that require rigorous documentation and control(Kerzner, 2017; PMI, 2021). It is particularly useful in industries such as construction, engineering, and information technology, where precise planning and risk management are crucial (PMI, 2021). The framework's focus on process standardization and best practices ensures a high degree of consistency and repeatability in project outcomes.

However, the detailed nature of PMBOK can also be a drawback, as it may lead to an overly bureaucratic approach, slowing down decision-making processes (Kerzner, 2017; PMI, 2021). This rigidity can be challenging in fast-paced or highly dynamic environments where flexibility and rapid adaptation are required (Kliem & Ludin, 2019).

2.2.3.2 PM2 (Project Management Methodology)

PM2, developed by the European Commission, is designed to be a practical and adaptable project management methodology (European Commission., 2018). It emphasizes simplicity, practicality, and flexibility, making it suitable for a wide range of projects and organizational contexts (European Commission., 2018).

2.2.3.2.1 Strengths, Applications and Limitations of PM2

PM2's streamlined approach is particularly beneficial for organizations seeking a less complex methodology that can be easily tailored to different project types and sizes (European Commission., 2018). It is widely used in the public sector and by organizations that require a

straightforward, easy-to-implement project management framework (European Commission., 2018).

While PM2's simplicity is an advantage, it may lack the depth and detail required for highly complex projects. Its focus on practicality and adaptability might lead to challenges in maintaining consistency and rigor, particularly in projects that require detailed risk management and quality control processes (Papke-Shields & Boyer-Wright, 2016).

2.2.3.3 APMBOK (APM Body of Knowledge)

APMBOK, from the Association for Project Management, provides a comprehensive overview of project management, incorporating 12 knowledge areas and placing significant emphasis on professional conduct, ethics, and continuous improvement.

2.2.3.3.1 Strengths, Applications and Limitations of APMBOK

APMBOK's integration of best practices from various quality management standards, such as ISO 9001, and its focus on ethical conduct and professionalism, make it highly relevant in industries where these aspects are critical. It is particularly useful in sectors such as healthcare, finance, and education, where stakeholder trust and adherence to ethical standards are paramount (APM, 2019).

Despite its strengths, APMBOK can be perceived as too broad, with its extensive focus on professional conduct and ethics potentially overshadowing the practical aspects of project management (J. R. Turner, 2016). Additionally, its comprehensive nature might require significant effort to tailor and implement effectively in smaller organizations or projects with limited resources (J. R. Turner, 2016).

2.2.3.3.2 Process Groups and Phases

PMBOK's structure around five process groups (Initiating, Planning, Executing, Monitoring and Controlling, and Closing) provides a clear, sequential approach to project management. This

framework is beneficial for projects that require detailed planning and control (PMI, 2021). PM2, on the other hand, simplifies the process into four phases (Initiating, Planning, Executing, and Closing), which enhances flexibility and adaptability (European Commission., 2018). APMBOK's approach is similar to PMBOK but includes additional focus areas such as Change and Professionalism, providing a more holistic view of project management (APM, 2019).

2.2.3.4 Knowledge Areas

All three standards cover essential knowledge areas such as Integration, Scope, Time, Cost, Quality, and Risk Management (Xue et al., 2015). However, PMBOK and APMBOK offer more detailed guidelines and best practices for each knowledge area, whereas PM^2 provides a more streamlined approach (PMI, 2021) ; (APM, 2019) ; (European Commission., 2018).
PM^2 stands out for its emphasis on practicality and adaptability, making it suitable for a wide range of projects and organizational contexts. PMBOK, while comprehensive, can be rigid and bureaucratic, potentially hindering flexibility in dynamic environments (Xue et al., 2015). APMBOK strikes a balance by integrating professional conduct and ethics, which adds an extra layer of consideration in project management but can also complicate the implementation process (Papke-Shields & Boyer-Wright, 2016).

2.2.3.4.1 Stakeholder Engagement

Effective stakeholder engagement is a critical success factor in project management (Pinto & Slevin, 1987). PM^2 and APMBOK place significant emphasis on this aspect, with PM^2 integrating stakeholder engagement throughout the project lifecycle and APMBOK highlighting the importance of understanding and managing stakeholder expectations (European Commission., 2018) ; (APM, 2019). PMBOK also addresses stakeholder management but within the context of its detailed process groups and knowledge areas (PMI, 2021).

2.2.3.4.2 Risk Management

Risk management is another area where these standards differ (Dikert et al., 2016). PMBOK provides detailed processes for identifying, analyzing, and responding to risks, making it highly suitable for projects with significant uncertainty and complexity (PMI, 2021). PM^2 incorporates risk management into its broader governance framework, emphasizing proactive risk identification and mitigation (European Commission., 2018). APMBOK advocates for a balanced approach to managing both risks and opportunities, aligning with its broader focus on continuous improvement and professional conduct (APM, 2019).

2.2.3.4.3 Quality Management

Quality management is essential in ensuring project deliverables meet the required standards. PMBOK's approach to quality management includes comprehensive guidelines for planning, managing, and controlling quality (PMI, 2021). PM^2 integrates quality management into its project control phase, emphasizing continuous improvement (European Commission., 2018). APMBOK expands on this by incorporating best practices from quality management standards like ISO 9001, emphasizing the integration of quality into all project processes (APM, 2019).

2.2.3.5 Technology Integration

The integration of technology in project management is increasingly important (Agarwal et al., 2010). PMBOK provides detailed guidelines for using project management software and tools, making it highly relevant in technology-driven projects (PMI, 2021). PM^2's practical approach also supports the use of technology but within a simpler framework (European Commission., 2018). APMBOK acknowledges the role of technology but places more emphasis on professional conduct and stakeholder engagement (APM, 2019).

While PMBOK, PM^2, and APMBOK share many similarities in their approach to project management, there are notable differences in their emphasis and application. PMBOK is highly detailed, providing extensive guidelines and best practices for each knowledge area. It is widely recognized and adopted across various industries, making it a standard reference for project managers worldwide (PMI, 2021).

PM^2, in contrast, offers a more streamlined and practical approach, designed to be easily adaptable to different project environments and organizational contexts (European Commission., 2018). Its emphasis on simplicity and practicality makes it particularly suitable for public sector projects and organizations looking for a less complex project management methodology (European Commission., 2018).

APMBOK, while similar to PMBOK in many respects, places additional emphasis on professional conduct, ethics, and continuous improvement (Joslin & Müller, 2016). It integrates best practices from various quality management standards and emphasizes the strategic importance of project management in achieving organizational goals (APM, 2019).

While the existing standards provide comprehensive guidelines for managing projects, there is a growing emphasis on the need for flexibility and adaptability in project management practices (Conforto et al., 2014). The increasing complexity and uncertainty in project environments necessitate a more dynamic approach to managing projects, integrating principles from agile and lean methodologies (Kerzner, 2017).

Effective communication and stakeholder management are consistently identified as critical success factors in project management, underscoring the need for project managers to develop strong interpersonal and leadership skills (Pinto & Slevin, 1987).

Another key trend in the literature is the integration of technology in project management. The use of project management software and tools has become increasingly prevalent, providing project managers with powerful tools for planning, monitoring, and controlling (Brandt et al., 2017).

2.3 From tradition to agility in Project Management

The evolution of project management from traditional methodologies to agile approaches marks a significant shift in how projects are planned, executed, and delivered (Dybå & Dingsøyr, 2008). Traditional project management, characterized by its structured and sequential processes, has given way to agile methodologies that emphasize flexibility, collaboration, and iterative progress (Conboy, 2009). This review explores this transition, drawing from well-known articles, academic sources, and case studies to highlight the benefits, challenges, and implications of adopting agile practices in various industries (Dikert et al., 2016).

2.3.1 Traditional Project Management

Traditional project management methodologies, such as those defined by PMBOK, PRINCE2, and APMBOK, emphasize a linear, sequential approach to project execution (Chin & Spowage, 2012). These methodologies are characterized by detailed planning, rigorous documentation, and strict adherence to predefined processes and timelines (Kerzner, 2017).

Traditional project management excels in environments where project requirements are well-defined and unlikely to change significantly (Petersen & Wohlin, 2009). Its structured approach provides clear milestones, roles, and responsibilities, ensuring that all project aspects are meticulously planned and controlled (Kerzner, 2017). This approach is particularly effective in industries such as construction, engineering, and manufacturing, where predictability and precision are critical (Flyvbjerg, 2017).

Despite its strengths, traditional project management can be inflexible and slow to respond to changes (Dybå & Dingsøyr, 2008). The extensive upfront planning and documentation requirements can lead to lengthy project initiation phases, and the rigid adherence to initial plans can hinder adaptability (Conforto et al., 2014). This lack of flexibility can be problematic in dynamic environments where requirements are likely to evolve (Kliem & Ludin, 2019).

2.3.2 Emergence of Agile Methodologies

Agile project management methodologies emerged as a response to the limitations of traditional approaches, particularly in software development (Beck et al., 2001). Agile methodologies, such as Scrum, Kanban, and Extreme Programming (XP), emphasize iterative development, continuous feedback, and collaboration (Dybå & Dingsøyr, 2008). The Agile Manifesto, published in 2001, encapsulates the core values and principles of agile methodologies, advocating for individuals and interactions over processes and tools, and customer collaboration over contract negotiation (Beck et al., 2001).

Agile methodologies are highly effective in environments characterized by rapid change and uncertainty (Dybå & Dingsøyr, 2008). Their iterative nature allows for continuous refinement of project deliverables based on stakeholder feedback, ensuring that the final product meets evolving needs (Moe et al., 2012). Agile practices foster close collaboration between cross-functional teams, enhancing communication and problem-solving capabilities (Parker et al., 2015). These methodologies are particularly popular in software development, IT, and creative industries where adaptability and speed are crucial (Schwaber & Sutherland, 2020).

While agile methodologies offer significant advantages, they are not without challenges (Dikert et al., 2016). The lack of detailed upfront planning can lead to scope creep and difficulty in managing project timelines and budgets (Misra et al., 2012). Additionally, the success of agile practices heavily relies on the team's ability to collaborate effectively and the presence of a supportive organizational culture. In traditional or hierarchical organizations, adopting agile methodologies may require significant cultural and structural changes (Conforto et al., 2014).

2.3.3 Hybrid Approaches: Combining Traditional and Agile

Recognizing the strengths and limitations of both traditional and agile methodologies, many organizations are adopting hybrid approaches that integrate elements of both (Kuhrmann et al., 2018). Hybrid approaches offer a balanced framework that can be tailored to the specific needs of the project and organization (Conforto et al., 2014). By combining detailed upfront planning with iterative development cycles, hybrid methodologies provide both predictability and adaptability (Gemino et al., 2023). This approach is particularly beneficial in industries such as finance, healthcare, and large-scale engineering projects, where certain aspects require strict regulatory compliance and precision, while others benefit from agile practices (Wysocki, 2014).

Implementing a hybrid approach can be complex, requiring careful coordination and alignment between traditional and agile practices (Dikert et al., 2016). Teams need to be trained in both methodologies, and organizational structures must support the hybrid model (Kuhrmann et al.,

2018). Additionally, balancing the different processes and ensuring clear communication and collaboration across the project lifecycle are crucial for success (Binder, 2016).

2.3.4 Transitioning from Traditional to Agile

The transition from traditional to agile project management involves significant changes in processes, culture, and mindset (Hekkala et al., 2017). Organizations must be prepared to invest in training, change management, and continuous improvement to successfully adopt agile practices (Dybå & Dingsøyr, 2008).

2.3.4.1 Change Management and Training

Effective change management is critical in transitioning to agile methodologies (Kotter, 2012). Organizations need to provide comprehensive training for project managers, team members, and stakeholders on agile principles and practices (Parker et al., 2015). This includes fostering a culture of collaboration, openness to change, and continuous learning (Kotter, 2012).

2.3.4.2 Cultural Shift

[3]Adopting agile methodologies requires a cultural shift towards valuing collaboration, transparency, and empowerment (Moe et al., 2012). Organizations must move away from hierarchical structures and encourage self-organizing teams that can make decisions and adapt quickly (Parker et al., 2015). Leadership plays a crucial role in modeling agile behaviors and supporting teams throughout the transition (Denning, 2018).

2.3.4.3 Continuous Improvement

[3] Agile methodologies emphasize team collaboration and self-organization, moving away from hierarchical structures typical of traditional project management. Transparency in progress, challenges, and decision-making fosters trust among team members and stakeholders. Empowering teams to make decisions enhances agility and responsiveness to change, which are core principles of agile practices. This cultural transformation is essential to fully realize the benefits of agile approaches.

[4]Continuous improvement is a core principle of agile methodologies (Beck et al., 2001). Organizations must establish mechanisms for regular reflection and adaptation, such as retrospectives and feedback loops (Dybå & Dingsøyr, 2008). This ongoing process ensures that teams can learn from their experiences and continuously refine their practices to enhance performance and outcomes (Kerzner, 2017).

The transition from traditional to agile project management represents a significant evolution in the field, driven by the need for greater flexibility, collaboration, and responsiveness to change (Conforto et al., 2014). While traditional methodologies provide valuable structure and control, agile practices offer the adaptability and iterative progress required in dynamic environments (Dybå & Dingsøyr, 2008). Hybrid approaches that integrate elements of both can provide a balanced framework, leveraging the strengths of each methodology (Kuhrmann et al., 2018). Successful adoption of agile practices requires a comprehensive change management strategy, a cultural shift towards collaboration and empowerment, and a commitment to continuous improvement (Kotter, 2012). As organizations continue to navigate this transition, the ability to effectively blend traditional and agile practices will be key to achieving successful project outcomes in an increasingly complex and fast-paced world (Dikert et al., 2016).

2.4 Evolution of Agile Methods Beyond Software Engineering

The traditional Software Development (SD) process, represented by the waterfall model has faced challenges due to its formal structure and inflexibility (Petersen & Wohlin, 2009). Agile SD methods emerged to address these challenges by offering quick customer value and producing feasible software versions through iterative processes (Hannola et al., 2013). The Agile methodology, initially focused on speed to market, rapid feedback, and continuous improvement, was born out of the need to address software development challenges, with positives impacts areas such as managing changing priorities, visibility, business/IT alignment, delivery speed, team productivity, and morale. It leads to improved customer satisfaction, business value, on-time delivery, quality, and productivity (Weichbroth, 2022). Given the fact that traditional Project

[4] Agile methodologies emphasize iterative cycles, where teams regularly reflect on their processes and outcomes to identify areas for enhancement. This principle of continuous improvement ensures that workflows remain efficient and adaptable, fostering sustained project success and team growth.

Management (PM) process, represented by the waterfall model, faced challenges due to its formal structure and inflexibility. Agile methods emerged to address these challenges by offering quick customer value and producing feasible software versions through iterative processes, not only in Software Development (SD) world but in the context of hard project (Hannola et al., 2013). Knowledge transfer is a critical component of project management, particularly in construction projects where lessons learned from past projects can significantly impact future project outcomes. (Xu et al., 2022) explore the application of knowledge transfer methods in construction projects using knowledge graphs and transfer learning. Their framework aims to facilitate the adaptive transfer of construction project knowledge, addressing common challenges such as data deficiencies and cold start problems.

2.5 Challenges in Harmonizing Agile with Project Management

Harmonizing agile methodologies with traditional project management approaches presents a myriad of challenges that are both organizational and managerial in nature (Dikert et al., 2016). These challenges are particularly pronounced when organizations transition from conventional, waterfall-style projects to agile frameworks (Petersen & Wohlin, 2009). This essay outlines the key difficulties encountered in integrating agile practices within traditional project management environments, highlighting insights from several case studies and research findings (Kuhrmann et al., 2018).

One of the foremost challenges is the lack of a unified understanding of agile principles among team members and management (Moe et al., 2012). Agile practices, such as iterative development and self-organizing teams, often clash with the hierarchical and rigid structures of traditional project management (Hekkala et al., 2017). For instance, in a study by (Hekkala et al., 2017), the transition to agile in a large inter-organizational project was hindered by varying interpretations of agility among project stakeholders. [5]The absence of a common vision led to an unstructured approach, exacerbating misunderstandings and misalignment of goals (Moe et al., 2012).

[5] A shared vision is critical in project management to ensure alignment among team members and stakeholders. When teams lack a unified understanding of goals, project execution can become fragmented, leading to inefficiencies, communication breakdowns, and conflicting priorities. Agile methodologies emphasize the importance of collective ownership and shared objectives, making the lack of a common vision a significant barrier to success.

[6]Managerial challenges also play a significant role in the friction between agile and traditional methods (Dikert et al., 2016). Traditional management often expects certainty in time, budget, and specifications from the outset of a project (Kerzner, 2017). This expectation is at odds with the agile philosophy, which emphasizes flexibility and responsiveness to change. In a survey conducted by (Gregory et al., 2016), it was found that management's demand for consistent reporting and their perception of re-prioritization as a lack of control posed significant hurdles. Agile teams struggled to meet these demands while maintaining their iterative processes. The organizational culture is another critical barrier (Moe et al., 2012). Agile methodologies advocate for a flat structure with distributed leadership, which can be challenging to implement in organizations with deeply entrenched hierarchical cultures (Hekkala et al., 2017). For example, the same study by (Hekkala et al., 2017) noted that traditional organizational borders persisted even after adopting agile methods. The cultural inertia within organizations often leads to resistance against the empowerment and autonomy that agile frameworks require (Conboy, 2009). Furthermore, the coordination and synchronization of cross-functional teams in agile environments can be problematic (Dikert et al., 2016). In complex projects, ensuring that all team members and stakeholders are aligned and working towards the same objectives is crucial (Moe et al., 2012). (Ovesen, 2015) highlighted that the need for cross-disciplinary collaboration often results in communication issue and reduced motivation among team members. This is particularly evident in projects where multiple teams work on different aspects without proper synchronization, leading to integration problems and delays (Dikert et al., 2016). The legal and regulatory constraints also pose a significant challenge (Boudjlida et al., 2015). Traditional project management practices are often deeply integrated with organizational procurement laws and regulations (Kearns & Sabherwal, 2006). These laws can be rigid and not easily adaptable to the fluid and iterative nature of agile methodologies. (Hekkala et al., 2017) pointed out that procurement laws were not suited to agile frameworks, complicating the contractual and compliance aspects of projects

[6] Managerial challenges are a key factor in the friction between agile and traditional methods because these two approaches require fundamentally different leadership styles and priorities. Traditional project management often relies on a hierarchical structure, centralized decision-making, and detailed upfront planning. In contrast, agile methodologies emphasize decentralized decision-making, flexibility, and self-organizing teams.

2.6 Importance of Project Process Customization

The importance of project process customization has become increasingly relevant in agile project management, particularly within the Scrum framework (Kuhrmann et al., 2018). Customization refers to tailoring standard processes to meet specific project needs, which can significantly impact the efficiency and effectiveness of project management practices (Beck et al., 2001).

2.6.1 Benefits of Customization

[7]Customization in project processes, particularly within agile frameworks like Scrum, is crucial because it allows teams to tailor standard practices to meet specific project needs, improving performance, team dynamics, and adaptability (Kuhrmann et al., 2018). Customization helps in aligning processes with unique project requirements and stakeholder expectations, leading to more effective project execution (Conboy, 2009). It enhances team collaboration by adjusting roles and responsibilities based on team strengths, fostering a more efficient working environment (Beck et al., 2001). Additionally, the ability to adapt processes to changing requirements or external conditions ensures that projects maintain momentum and achieve desired outcomes (Moe et al., 2012). Despite the challenges of balancing standardization with flexibility, the benefits of customization in improving project success are significant (Dikert et al., 2016).

2.6.1.1 Improved Performance

(Ovesen, 2015) notes that customization helps in addressing the unique challenges faced by different teams, leading to more effective project execution and better alignment with project goals.

[7] Customization in project processes is particularly important in agile frameworks like Scrum because it allows teams to adapt their workflows and practices to the unique requirements of each project. Agile frameworks are based on principles such as flexibility, iterative development, and continuous feedback, which can be more effective when customized to suit the specific context of a project

2.6.1.2 Enhanced Team Dynamics

[8]Customization facilitates better team dynamics by allowing teams to adjust roles and responsibilities based on their strengths and weaknesses (Beck et al., 2001). This approach fosters a more collaborative and efficient working environment, which is crucial for the success of agile projects (Chantit & Essebaa, 2021).

2.6.1.3 Adaptability to Change

The ability to customize project processes makes it easier for teams to adapt to changes in project requirements or external conditions (Moe et al., 2012). This adaptability is a core principle of agile methodologies and is essential for maintaining project momentum and meeting stakeholder expectations (Dybå & Dingsøyr, 2008)**.** Several case studies demonstrate the practical implications of process customization. For instance, in the study by (Hekkala et al., 2017), customization of Scrum practices was necessary to address specific regulatory and operational requirements in the healthcare industry. Project process customization within the Scrum framework is essential for achieving optimal project performance, enhancing team dynamics, and maintaining adaptability (Kuhrmann et al., 2018). While customization poses certain challenges, the benefits far outweigh the drawbacks when managed effectively (Conboy, 2009). By carefully balancing standardization with flexibility, teams can tailor their processes to meet specific project needs, leading to improved project outcomes and greater overall success (Beck et al., 2001).

Despite its benefits, customization also presents challenges (Dikert et al., 2016). One significant challenge is maintaining a balance between standardization and flexibility (Dybå & Dingsøyr, 2008). Too much customization can lead to inconsistencies and difficulties in managing the project, while too little can reduce the framework's effectiveness (Dybå & Dingsøyr, 2008) .

[8] Agile methodologies emphasize the importance of team collaboration and self-organization, and customization allows teams to align their roles and responsibilities with individual team members' strengths. By adjusting these roles to suit the unique skills and capacities of the team members, agile practices foster better communication, trust, and efficiency.

(Ovesen, 2015) highlights that extreme cross-disciplinarity and team polarization are common issues when implementing customized Scrum processes in diverse environments.

2.7 Essence as a solution for customization

Many business domains depend on software systems to grow hence the need for organizations to develop competences in managing software projects for a competitive edge (Simonette et al., 2016). SEMAT (Software Engineering Methods and Theory) was founded in 2009 with the goal of fundamentally changing how people work with software development methods (Jacobson et al., 2012) with a goal of introducing a kernel as a thinking framework to bridge the gap between current ways of working and new ideas in software development. According to (Roqueme & Jaramillo, 2017), the SEMAT initiative aims to provide a common terminology for software engineering through the Essence Kernel, consisting of alphas, activity spaces, and competencies. Alphas, representing fundamental aspects of software engineering endeavors, have states with associated checklists. However, inconsistencies and ambiguities in alpha checklists hinder their effectiveness in guiding teams. (Jacobson et al., 2022) argue that while agile methods like Scrum simplify the software development process, their popularity has led to a lack of clarity and simplicity in their implementation. SEMAT's vision is to focus on finding a widely agreed-upon kernel of essential elements and defining a solid theoretical basis for software engineering, addressing both technology and people issues (Jacobson et al., 2012). This leads to a call for action which has identified critical problems in software engineering, including immature practices, lack of a theoretical basis, and the split between industry practice and academic research (Jacobson et al., 2012). In this innovative context, SEMAT's stance on Innovation by being portrayed as supportive of new ideas and innovations in software engineering. It opposes non-lean and non-agile behavior resulting from adopting inappropriate solutions for fashion or peer pressure (Jacobson et al., 2012). Essence is introduced as a domain model that aims to provide a common ground for software development methods, allowing teams to compose methods from individual practices. (Simonette et al., 2016) traces the origin of Essence Kernel to the SEMAT submission in response to the Object Management Group's call for the "Foundation for Agile Creation and Enactment of Software Engineering." Essence has been introduced as a Foundation for an Empirical Research Framework to emphasizes the gap between software engineering research and

industry adoption (Ng et al., 2013). There is however a need to recognize the challenge of mapping software engineering (SE) practices to the Essence Framework (EF), a relatively new framework designed to tackle core problems in SE (Uysal, 2018).

2.7.1 Application of Essence in Project Management and Methodologies

The Essence framework is also being applied to enhance project management and methodology design in software engineering. (Ivanova et al., 2022) propose integrating Essence practices with Bayesian networks to develop a decision support system for software project management. This system aims to reduce risks and improve project outcomes by providing a formal model for project evaluation and optimization. The integration of Essence with Bayesian networks allows for dynamic assessment and adjustment of project parameters, leading to more accurate and reliable project management (Ivanova et al., 2022). In the context of multi-agent systems, (Dahhane et al., 2017) explore the application of the Essence kernel to the O-MaSE (Organization-based Multiagent System Engineering) methodology. Their work aims to create a common ground for designing method fragments for multi-agent systems, facilitating the composition and comparison of different methodologies. The adoption of the Essence kernel in this context helps standardize the description of multi-agent methodologies, making them more accessible and easier to implement.

2.7.2 Integrating Essence in Existing Tools and Practices

The integration of the Essence framework into existing tools and practices is crucial for its widespread adoption. (Elvesæter et al., 2013) discuss the challenges and potential solutions for enabling Essence in existing software development tools. They emphasize the need for open-source tools that can be easily integrated with popular project management and development environments like Taiga and GitLab. By "essentializing" these tools, developers and educators can more effectively utilize the Essence framework to manage and monitor software projects (Elvesæter et al., 2013).

2.7.3 Essence and Agile Methods

(Jacobson et al., 2022) explore the synergy between Scrum, a widely used agile method, and Essence. They highlight how Essence can enhance Scrum by providing a structured way to describe and implement Scrum practices. The use of Essence cards and games facilitates better understanding and application of Scrum principles, making it easier for teams to adopt and adapt these practices effectively. This integration showcases the potential of Essence to not only unify various methods but also to improve their practical implementation.

2.7.4 Founding principles and vision or Background and Inspiration

In their mission statement, SEMAT (Software Engineering Method and Theory) is here for the benefit of all parties concerned about software engineering - both what it is today and what it will become in the future. In more detail SEMAT is formed for the public benefit to develop and promote international standards for software engineering kernel and language that provide a common framework for defining methods and practices of software engineering; to build and maintain an international community where software engineering experts collaborate for the establishment of theories of software engineering and for the open innovation of software engineering practices; to build and maintain an open library and marketplace of software engineering practices and education materials; to organize international symposiums and conferences on software engineering and to provide training and certification programs for software engineers. In nutshell, SEMAT is an initiative to reshape software engineering such that software engineering qualifies as a rigorous discipline. The initiative was launched in December 2009 by Ivar Jacobson, Bertrand Meyer, and Richard Soley with a call for action statement and a vision statement. The initiative was envisioned as a multi-year effort for bridging the gap between the developer community and the academic community and for creating a community giving value to the whole software community.

2.7.5 Origins of Essence Kernel

(Simonette et al., 2016) traces the origin of Essence Kernel to the SEMAT submission in response to the OMG's call for the "Foundation for Agile Creation and Enactment of Software Engineering." Since then, Essence Kernel and Language have become an OMG standard, emphasizing its role as a framework for software engineering best practices. SEMAT focused on two major goals: finding a kernel of widely agreed-upon elements and defining a solid theoretical basis by highlighting the need for a common ground in software engineering, manifested as a kernel of essential elements universal to all software development efforts (Jacobson et al., 2012).

2.7.6 Key Concepts of the Essence Method

2.7.6.1 Overview of the Essence Kernel

The SEMAT kernel is presented as an actionable, an extensible, and a practical framework (Jacobson et al., 2022). Firstly, it is actionable through the use of alphas, essential elements that represent the progress and health of the software engineering endeavor. Secondly, the kernel is extensible, allowing for the addition of different practices to support various projects. Thirdly, it is practical, providing hands-on, tangible support for software professionals in their daily work (Jacobson et al., 2012).

The kernel is actionable, using alphas (essential elements) with states and checklists to guide software development teams (Jacobson et al., 2012). (Roqueme & Jaramillo, 2017) authors identify several issues in a sample of alpha checklists, such as the use of the term "system" instead of the specific "software system" alpha, adjectives qualifying objects without clear criteria, and ambiguity regarding the agents and objects in checklist items. These problems can impede the actionable feature of the Essence Kernel, affecting the success of software engineering endeavors. They propose a terminology unification approach that involves syntactic, semantic, and pragmatic regularization of alpha checklists. They suggest using syntax trees, thematic roles, and the Essence Kernel controlled language to eliminate consistency and ambiguity problems. According to (Jacobson et al., 2012), the essence kernel helps software professionals in their daily works including running iterations, the entire development process, and even scaling to large organizations. Planning an iteration is presented as an example, involving determining the current state, setting objectives, and defining tasks to achieve those objectives.

2.7.6.2 Core principles and components.

2.7.6.2.1 Things to Work With

(Jacobson et al., 2022) introduce some concepts such as the Essence model and Scrum Essential cards offer a structured approach to understanding, teaching, and improving agile practices. For them, the emphasis on creating an open ecosystem for software development practices aligns with the principles of agility, encouraging adaptability and continuous improvement. Essence is many things, dependent on who you are. For a software practitioner, Essence provides a language for defining practices with just a few different kinds of elements. The key element is the concept of an alpha, described in this section together with additional language constructs needed for proper capturing of practices. In more details, Essence presents seven kernel alphas (Jacobson et al., 2022). These kernel alphas are tracked under states and below are their descriptions:

- Opportunity: There is a problem or an opportunity to address.
- Stakeholders: There are stakeholders who will fund, use, and benefit from the solution produced.
- Requirements: There are certain requirements to be met.
- Software system: There will be a software system to develop. Related to the endeavor
- Work: We need to kick off the work to be done.
- Team: We need an empowered team of competent people to perform the work.
- Way of working: The team needs a good, responsive way of working.

The theoretical foundation of Essence introduces alphas representing different perspectives or dimensions in software development. Each alpha has a lifecycle with associated checklists, and Essence organizes success factors within alphas and states. The alphas common to all software engineering endeavors are listed, and their state progressions are briefly related to key success factors.

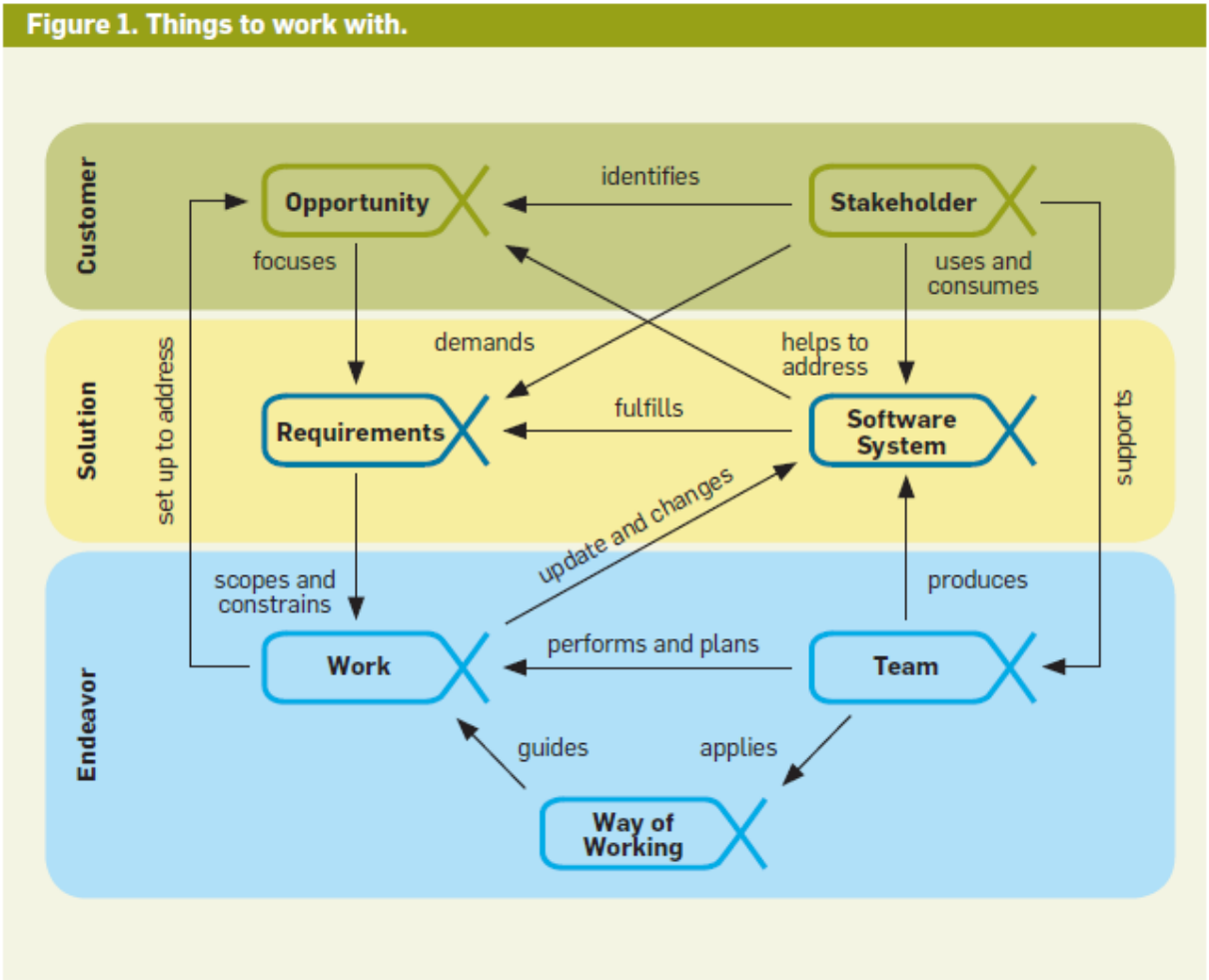


Figure 1: The Kernel Alphas (Jacobson & al, 2018)

2.7.6.2.2 Things to Do

"Things to Do," presents a comprehensive view of the essential activities required to progress in software engineering endeavors. It aligns these activities within the context of the kernel's areas of concern: customer, solution, and endeavor. The figure is pivotal in illustrating the dynamic interplay between different activities that drive the advancement of a software project:

Customer Area of Concern

- Explore Possibilities: This activity involves investigating the potential opportunities for developing or enhancing a software system. It includes the analysis of opportunities and the identification of stakeholders.
- Understand Stakeholder Needs: This activity focuses on engaging stakeholders to comprehend their needs fully. It ensures that the team produces the right results by working closely with stakeholder representatives.
- Ensure Stakeholder Satisfaction: This activity involves sharing the development results with stakeholders to gain their acceptance and verify that the opportunity has been successfully addressed.
- Use the System: Observing the system in a live environment to ensure it benefits stakeholders.

Solution Area of Concern

- Understand the Requirements: Establishing a shared understanding of what the system must achieve.
- Shape the System: Designing and architecting the system to ensure it is easy to develop, maintain, and meets current and future demands.
- Implement the System: Building, testing, and integrating system elements, including bug fixing and unit testing.
- Test the System: Verifying that the system meets the stakeholders' requirements.
- Deploy the System: Making the tested system available for use outside the development team.
- Operate the System: Supporting the system in the live environment to ensure its smooth operation.

Endeavor Area of Concern

- Prepare to Do the Work: Setting up the team and its working environment, understanding, and committing to the work.
- Coordinate Activity: Planning and directing the team's work, including ongoing planning and reshaping the team as necessary.

- Support the Team: Providing the necessary support for the team to perform their work effectively.
- Track Progress: Monitoring the progress of work and making necessary adjustments to ensure that objectives are met.
- Conclude Work: Ensuring that the work is completed and the results meet the required standards.

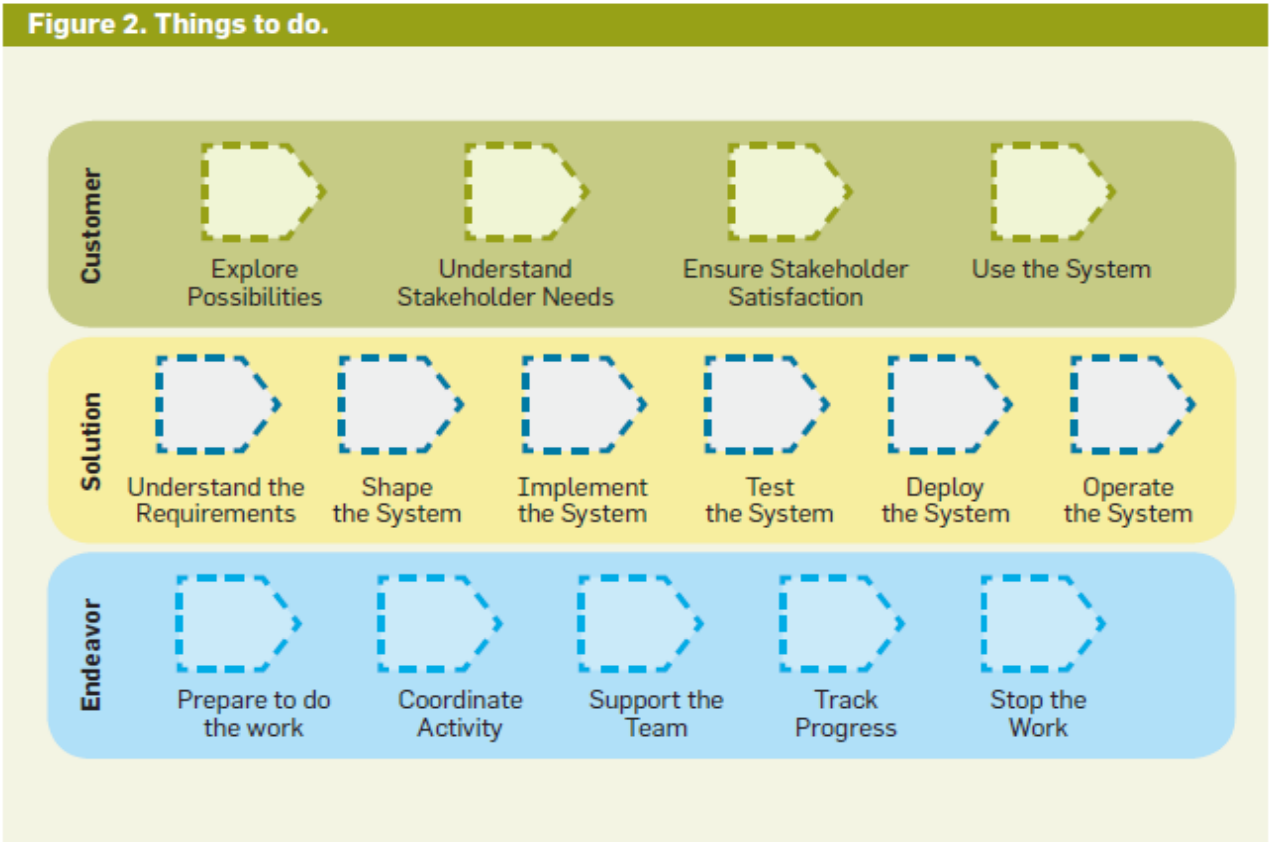


Figure 2: Things to do (Jacobson & al, 2018)

2.7.6.2.3 Determining the project current state

In the Essence framework, states play a pivotal role in assessing and determining the status of a software engineering project. Each Alpha, representing essential aspects of the project, progresses

through a series of predefined states that reflect its current condition and maturity. This structured approach to state progression enables teams to monitor progress, identify issues, and make informed decisions to ensure project success.

The Essence States provide a clear vision of the progress throughout his lifecycle.

- **Clear Progress Indicators**: States provide explicit markers of progress for each Alpha. By defining specific criteria that must be met to transition from one state to the next, teams have a clear understanding of what needs to be accomplished at each stage of the project.
- **Health Assessment**: The states of the Alphas collectively indicate the health of the project. For instance, if the Software System Alpha is in the "Demonstrable" state, it suggests that a working prototype is available, indicating substantial progress in development.
- **Risk Management**: By regularly assessing which states the Alphas are in, teams can identify potential risks early. For example, if the Requirements Alpha remains in the "Bounded" state for an extended period, it may indicate difficulties in achieving stakeholder consensus, prompting early intervention.
- **Guidance and Planning**: States provide guidance on what activities need to be undertaken next. For example, moving the Work Alpha from "Prepared" to "Started" requires ensuring that all necessary preparations are complete, thereby preventing premature commencement of development activities.
- **Stakeholder Communication**: States offer a clear and concise way to communicate project status to stakeholders. Describing the project in terms of the states of its Alphas allows stakeholders to quickly grasp where the project stands and what has been achieved.

State Examples and Their Role

- **Opportunity Alpha**
    - **Opportunity cane be** Identified → Solution Needed → Value Established → Benefit Accrued and his role is to track the project's justification and potential value. Moving to "Value Established" ensures that the project has a clear purpose and expected benefits, essential for stakeholder buy-in.

- **Stakeholders Alpha**

- **Stakeholders can be** Recognized → Represented → Involved → Satisfied and hits role is to ensure stakeholder's engagement and satisfaction. The "Satisfied" state indicates that stakeholders' needs have been met, which is critical for project acceptance.

- **Requirements Alpha**
  - **Requirements can be** Conceived → Bounded → Coherent → Addressed → Fulfilled and it manages the clarity and completeness of requirements. The "Addressed" state confirms that the requirements are being actively worked on, ensuring alignment with stakeholder needs.

- **Software System Alpha**
  - **Software System** has an architecture that can be Selected → Demonstrable → Usable → Operational → Retired and its role is to monitors the development and readiness of the software system. The "Operational" state signifies that the system is ready for use, marking a significant milestone in the project.
  - **The Work is** Initiated → Prepared → Started → Concluded and his role is to tracks the execution of project activities. The "Concluded" state indicates that the work has been completed, providing a clear endpoint for project tasks.
  - **Team** plays a crucial role in projects and go through different state during the project phases. Team can be Seeded → Formed → Collaborating → Performing → Adjourned. This state has a role of Assessing the team's formation and performance. The "Performing" state reflects a high-functioning team, crucial for efficient project execution.

- **Way of Working Alpha**

  The way of working is based on ground rules or principles that need to be Established → Foundation Established → In Use → Retired. Its role is to evaluates the methods and practices in use. The "In Use" state confirms that the team has a defined and active way of working, supporting consistent project progress.

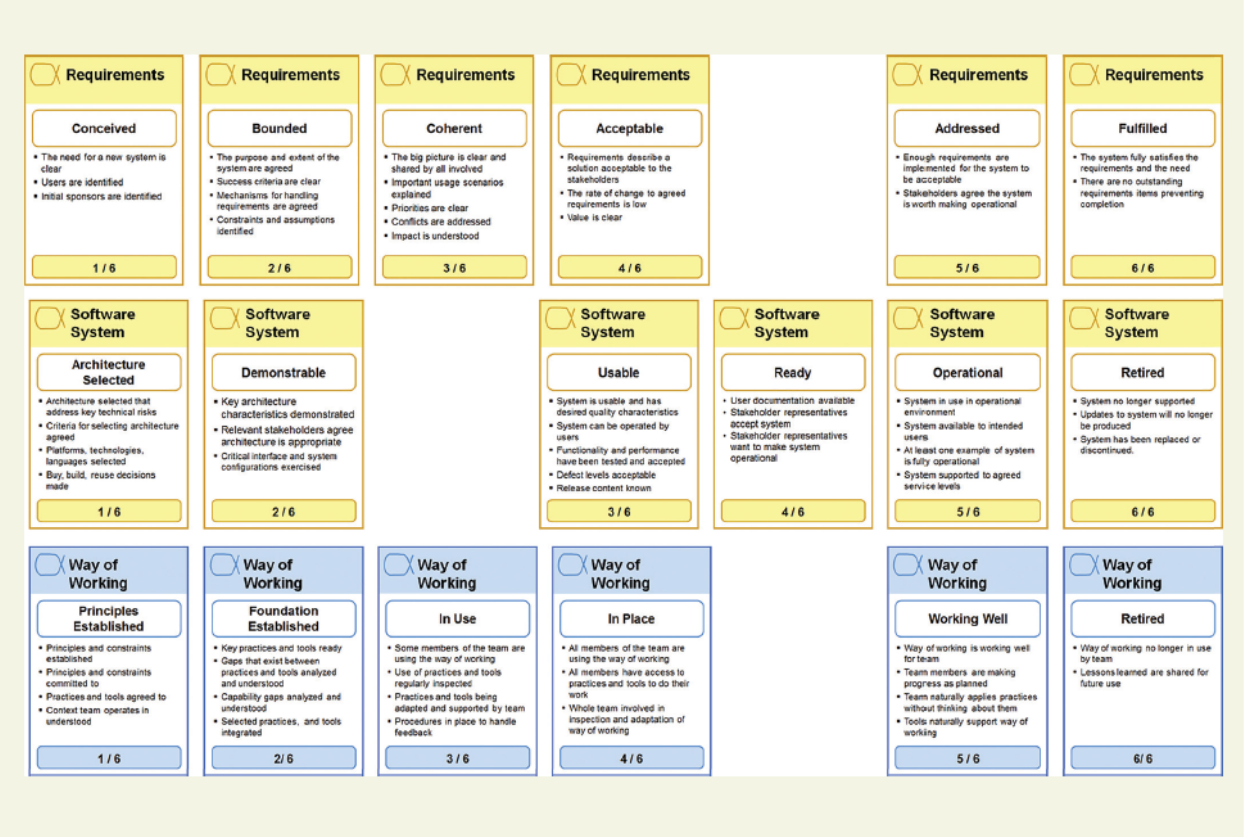


Figure 3: Project current state (Jacobson & al, 2018)

2.8 Software Engineering Practices

2.8.1 Disciplines within the Essence Method

According to (Jacobson et al., 2022), there is a synergy between Scrum, one of the popular agile methods, and Essence, a domain model of software engineering processes. It is presented as a domain model that describes the essential elements of any software development process. It helps teams understand their current state, what is missing, or what needs to be addressed. Essence considers every method as a composition of reusable "mini-methods" or practices. Examples of these practices include user stories, component-based development, test-driven design, continuous

integration, and pair programming. (Samar & Saud, 2020) presents the Essence Method Architecture which illustrates the overall view of Essence Method Architecture. The Essence method architecture is designed to provide a structured and flexible approach to software engineering by defining a kernel and a language. His architecture supports the creation, use, and improvement of software engineering methods, making them scalable, extensible, and easy to understand and implement. In one hand, the Essence kernel captures the essential elements of software engineering that are integral to all methods. It provides a common ground for comparing methods and making better decisions about practices. In other hand, the Essence language is a domain-specific language that defines methods, practices, and kernels. It includes both static features (syntax and well-formedness rules) and dynamic features (operational semantics) to enable usage and adaptation.

The Methods are viewed as compositions of practices since in Essence, a method is a dynamic composition of practices rather than just a static description. This approach changes the conventional definition of a method, emphasizing what is actually done rather than what is expected to be done. It is important to note that a practice is a repeatable approach with a specific objective. Practices provide systematic and verifiable ways to address particular aspects of work and can be part of multiple methods.

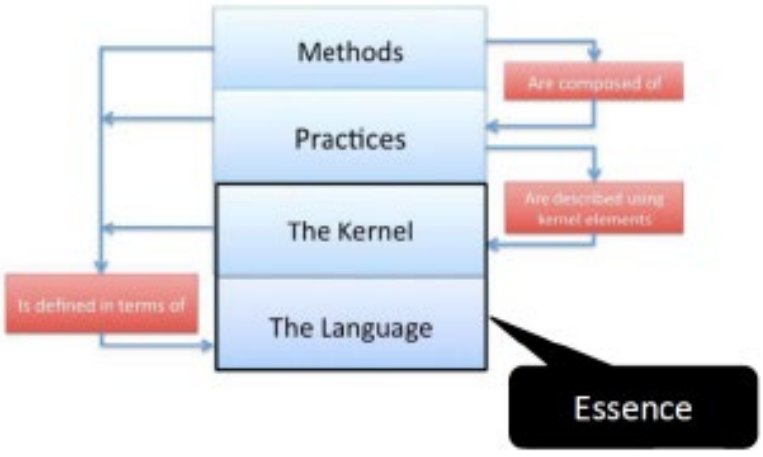


Figure 4: The Essence Method (Jacobson & al, 2018)

2.9 Application in real-world scenarios

SEMAT's call for action received broad support from the software engineering community, including signatories and supporters (Jacobson & al, 2018). Cards are introduced as a practical tool to make the kernel tangible. In this perspective, teams can use a small deck of cards for daily discussions on development status, work assignments, collaboration, and areas of improvement (Jacobson et al., 2012).

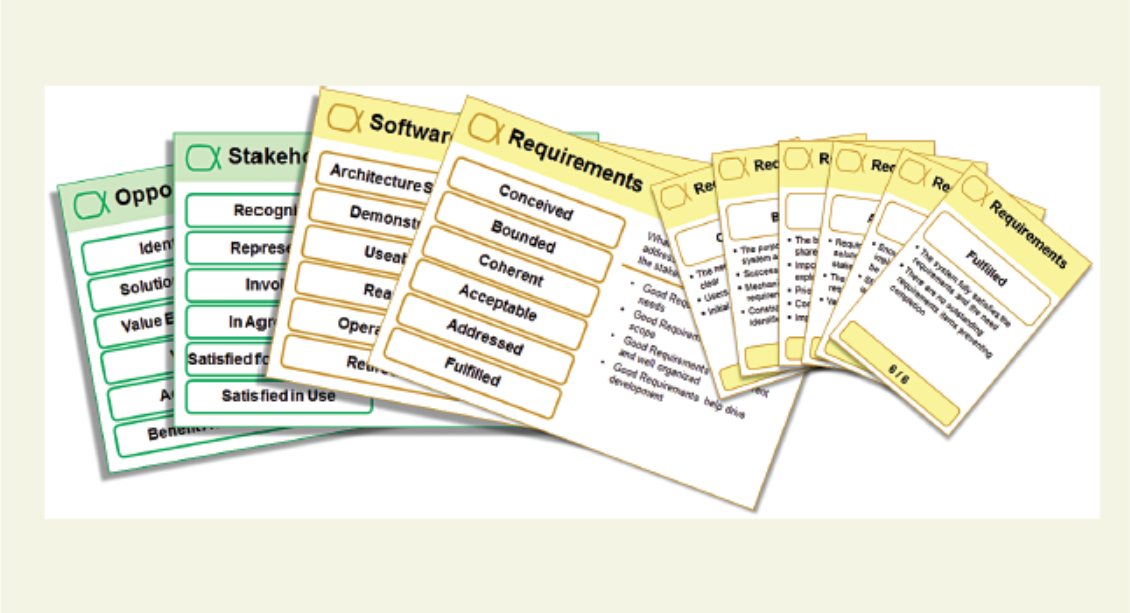


Figure 5: The Essence Card (Jacobson & al, 2018)

Although the Essence kernel seems to be new, there are many real-world applications of the SEMAT Kernel, including its use in companies like MunichRe, Fujitsu Services, a Japanese consumer electronics company, KPN, and a major U.K. government department (Jacobson et al-2012). The kernel helps not only experienced software professionals but also aspiring by fostering a more self-organized team. By citing Jeff Sutherland, the co-creator of Scrum, (Jacobson et al., 2022) described it as a way “to create a simple
glossary of the events, roles and artifacts of the Scrum framework, which is what was needed to more easily and concisely describe Scrum.

The Essence framework, developed by the Object Management Group (OMG), offers a comprehensive and adaptable approach to software engineering. Its structured methodology, which includes a kernel and a language, is designed to be scalable, extensible, and user-friendly. This essay explores the real-world applications of the Essence framework across various industries and contexts, highlighting its versatility and effectiveness in managing software engineering projects.

2.9.1 Software Development Projects

2.9.1.1 Project Management and Tracking

In software development projects, the Essence framework provides robust tools for project management and tracking (Jacobson et al., 2013). By monitoring the states of Alphas—key entities such as Opportunity, Stakeholders, Requirements, and Software System—project managers can gain a clear and accurate view of project health and progress. This structured approach not only facilitates effective tracking but also aids in early risk identification, allowing for timely interventions (Object Management Group (OMG), 2018).

2.9.1.2 Agile and Iterative Development

The Essence framework seamlessly integrates with Agile and iterative development methodologies (Jacobson et al., 2013). Its dynamic nature supports the incorporation of various Agile practices, including Scrum, Kanban, and Extreme Programming (XP). This flexibility allows teams to tailor their methods to best fit their unique needs, promoting continuous feedback and adaptation. As a result, teams can respond swiftly to changes, ensuring that the development process remains efficient and aligned with stakeholder expectations (Jacobson et al., 2022).

2.9.1.3 Standardization Across Teams

For large enterprises, the Essence framework offers a customzed approach to software development across multiple teams and projects (Jacobson et al., 2013). By defining common practices, organizations can ensure consistency and quality in their software engineering processes. This standardization reduces redundancy and promotes the reuse of practices, enhancing overall efficiency and coherence within the enterprise (Jacobson et al., 2014).

2.9.1.4 Cross-Functional Collaboration

[9]The Essence framework facilitates cross-functional collaboration by providing clear definitions and states for Alphas, which enhances communication and alignment between departments such as development, QA, and operations (Jacobson et al., 2012). This common framework supports integrated development and operations (DevOps), ensuring that work is effectively tracked and managed throughout the software lifecycle (Object Management Group (OMG), 2018).

2.9.1.5 Flexible and Scalable Methods

Startups and small businesses benefit from the Essence framework's flexibility and scalability (Jacobson et al., 2022). The framework allows for incremental adoption, enabling small teams to start with a minimal set of practices and gradually expand their methods as they grow. This approach prevents the overburdening of teams with complex processes and ensures that practices remain lean and efficient (Jacobson et al., 2012).

2.9.1.6 Rapid Prototyping and Innovation

[9] The Essence framework's use of Alphas—core entities that represent the key elements of software engineering projects—helps teams across various departments, such as development, quality assurance (QA), and operations, achieve better communication and alignment. By clearly defining and tracking these Alphas, the framework ensures that everyone in the organization has a shared understanding of project progress, requirements, and outcomes. This promotes collaboration, reduces misunderstandings, and streamlines the development process.

[10]The Essence framework supports rapid prototyping by allowing teams to quickly define, implement, and test ideas (Jacobson et al., 2012). This capability fosters innovation, enabling startups to iterate swiftly and bring new products to market faster. By focusing on essential elements, the framework helps maintain lean practices, ensuring agility and responsiveness (Object Management Group (OMG), 2018).

2.9.1.7 Teaching and Research

The Essence framework is an invaluable educational tool for teaching software engineering principles. It provides a structured yet flexible approach to learning, making it suitable for both theoretical and practical applications. Academic institutions can use the framework to teach students about software engineering methodologies and to conduct research on new practices and methods (Jacobson et al., 2014).

2.9.1.8 Collaboration with Industry

Academic institutions can leverage the Essence framework to collaborate with industry partners, facilitating knowledge transfer and joint projects (Castro et al., 2020). The framework's adaptability makes it suitable for applied research in real-world settings, bridging the gap between academic research and practical application (Jacobson et al., 2013).

In regulated industries such as healthcare and finance, the Essence framework helps ensure compliance with industry standards and regulations. Its structured approach to documenting and tracking activities and artifacts supports rigorous quality assurance processes, which are crucial for maintaining compliance and meeting regulatory requirements (Object Management Group (OMG), 2018).

2.9.1.9 Risk Management and Security

[10] The Essence framework is designed to be lightweight and flexible, which enables teams to rapidly iterate and prototype new ideas. By focusing on core project elements called Alphas, teams can define and adjust processes efficiently, allowing for quick testing and refinement of prototypes.

The Essence framework's extensibility allows it to be tailored for the development of critical systems requiring high levels of security and reliability. This adaptability is essential for managing risks and ensuring that systems meet stringent security standards. Additionally, the framework aids in effective incident response by providing a structured methodology for managing and documenting necessary steps (Jacobson et al., 2012).

2.10 Global Initiatives in Research

2.10.1 Need for a Framework for Empirical Research

The article emphasizes the gap between software engineering research and industry adoption. Various initiatives, such as Adoption Centric Software Engineering (ACSE) and SEMAT, aim to bridge this gap. The lack of a systematic framework for reporting empirical findings is acknowledged, making it challenging to compare and generalize results. The authors propose a framework focusing on properties of interest, distinct from existing research methods, to organize and analyze data systematically (Ng et al., 2013).

2.10.2 Essence as a Foundation for an Empirical Research Framework

(Ng et al., 2013) argue that despite Essence being in its infancy and primarily designed for practitioners, it can serve as a foundation for an empirical research framework. They highlight Essence's comprehensiveness, model-based nature, and extensibility as key features that make it suitable for this purpose. Their article emphasizes the need for additional work to make Essence usable for researchers by identifying common properties of interest.

2.10.3 Essence Framework in Software Engineering Education

(Ng & Huang, 2013) highlight the difficulties universities face in preparing students for the diverse methodologies used in the industry. They argue that the SEMAT initiative, particularly the Essence framework, offers a standardized approach to understanding and comparing different software development methods. Essence provides a common ground by modeling software development

elements as "alphas," which represent various dimensions of software engineering challenges. This framework helps students and professionals systematically address and navigate these challenges, bridging the gap between academic training and industry practices.

2.10.4 Empirical Evaluation of Essence Reflection Meetings

(Péraire & Sedano, 2014) conducted a field study to evaluate the effectiveness of Essence reflection meetings. Their research involved graduate student teams using Essence to guide their project reflection sessions. The study found that Essence facilitated holistic, state-based, goal-driven, and method-agnostic reflective discussions, helping teams address critical aspects of their projects. This method complements agile retrospectives by uncovering unknown issues and steering projects toward higher performance states.

2.10.5 Collaborative Tools for Essence

(Quintanilla-Perez et al., 2019) introduced Essboard, a collaborative tool designed to enhance the use of Essence in software development. Essboard promotes team collaboration by allowing all members to contribute their perspectives on the project's status and goals. The tool's design supports real-time interaction and provides a shared vision of the project's progress. This approach contrasts with existing tools that often neglect the collaborative aspect, thereby providing a more integrated and cooperative environment for software development teams. Essboard is a collaborative real-time web tool designed to monitor the progress of software development projects based on the Essence framework's alphas, states, and checklists. The key innovation of Essboard is its focus on collaboration, allowing team members to participate in establishing the current status and goals of a project. By utilizing the Essence framework, Essboard offers a holistic and common view of the project's progress, facilitating better team communication and project management.

- **Collaborative Perspective**:

    - Essboard emphasizes the importance of each team member's perspective. It ensures that the state and progress of the project are not just seen from a single viewpoint, typically that of the project manager, but from all team members.
    - It uses mechanisms like progress poker, which allows team members to express their opinions on the project's status and goals, fostering a shared vision.

- **Real-time Monitoring and Recording**: The tool supports real-time interaction and records all activities, ensuring transparency and accountability. This is achieved through WebSockets and an Event Sourcing architecture, which logs all status changes as events.

- **Information Radar**: Essboard features an information radar based on the Essence alphas, providing a visual representation of the project's progress. This helps in quickly assessing the state of different aspects of the project.

- **Integration with Other Tools**: Essboard can be integrated with other project management tools like Trello or JIRA. The goals and checklists from Essboard can serve as inputs for these tools, enhancing the overall project management process.

- **Support for Iterative Processes**: The tool supports iterative project management processes. It allows teams to conduct work sessions where they can discuss the project's state and goals, make necessary adjustments, and plan the next steps.

2.10.6 Essence Framework for Agile Methodologies

(Raharjo et al., 2023) propose a model for integrating agile methods using the Essence framework. Agile methodologies like Scrum, Kanban, and Extreme Programming (XP) are widely adopted due to their flexibility and adaptability. However, no single agile method suits all organizations. They have developed a model based on Essence that combines elements from various agile methodologies to support organizational needs. This model provides a structured approach for organizations to develop and refine their agile practices, ensuring they can effectively respond to complex problems and rapidly changing environments.

2.10.7 Evaluating the Framework: Cases study

(Ng et al., 2013) called for a need for a Framework for empirical research which will help bridging the gap between software engineering research and industry adoption. (Ng et al., 2013) argues that that despite Essence being in its infancy and primarily designed for practitioners, it can serve as a foundation for an empirical research framework. However, to test the framework, case studies are key. Using Essence, they identify a strong validity threat that was not previously detected, emphasizing the importance of systematic reporting and evaluation guidelines. In that matter, (Uysal, 2018) despite the fact that software engineering is a relatively young discipline, the Essence framework can be used to address core issues. A challenge stills subsist: how to map the software engineering practices to the essence framework knowledge domain? (Uysal, 2018) proposes the use of Concept Algebra as a formal method for mapping software engineering (SE) practices to the Essence Framework (EF). Concepts in Essence Framework are defined as compositions with attributes, objects, and relations. Similar abstract in Software Engineering Practice concepts are defined, enabling a comparative analysis between Essence Framework and Software Engineering Practices. Several case studies have highlighted the practical applications and benefits of the Essence framework in real-world scenarios.

2.10.7.1 Case Study 1: Mid-sized Software Development Company

(Silva et al., 2020) examined the implementation of Essence in a mid-sized software development company. The results indicated a significant improvement in project transparency and team communication. The framework's standardized language and structured approach helped the team streamline their processes and reduce misunderstandings.

2.10.7.2 Case Study 2: Academic Setting

(Brown & Smith, 2021) explored the use of Essence in an academic setting, focusing on a software engineering course where students used the Essence framework to guide their projects. The findings revealed that students had a better grasp of software engineering principles and could

apply them more effectively. The framework's visual aids and clear definitions were particularly praised for enhancing learning outcomes.

2.10.7.3 Case Study 3: Comparative Analysis with Other Frameworks

(Lee & Wang, 2022) conducted a comparative analysis of Essence, Scrum, and Kanban. They found that while Scrum and Kanban are more focused on specific project management aspects, Essence provides a more comprehensive view of the software engineering process. Essence's flexibility allows it to integrate elements from Scrum, Kanban, and other methodologies, offering a more holistic approach.

2.10.7.4 Case Study 4: Adaptability of the Essence Framework

(Martinez et al., 2021) found that Essence's adaptability makes it suitable for various project types and sizes. Unlike rigid frameworks, Essence can be tailored to fit the specific needs of a project, making it a versatile tool in the software engineering toolkit.

These cases studies showed challenges and limitations in implementing Essence Framework. Despite its advantages, the Essence framework is not without challenges.

2.10.7.5 Challenge 1: Initial Learning Curve

(Robinson & Perez, 2020) highlighted the initial learning curve associated with adopting the framework. Teams unfamiliar with Essence may find it complex and time-consuming to implement. Additionally, the framework's comprehensive nature can sometimes lead to an overload of information, making it difficult for smaller teams to manage effectively.

2.10.7.6 Challenge 2: Integration with Existing Practices

(Thompson et al., 2022) noted that organizations with established methodologies might resist adopting a new framework, especially if it requires significant changes to current processes. Overcoming this resistance requires careful planning and a phased approach to implementation.

2.11 Industry-Specific Applications

2.11.1 Mapping Software Engineering to Essence Framework

(Jacobson et al., 2012) discusses practical applications of the kernel in the daily lives of software professionals, including running iterations, the entire development process, and scaling to large organizations. Planning an iteration is presented as an example, involving determining the current state, setting objectives, and defining tasks to achieve those objectives. (Ng et al., 2013) presents Essence framework as a valuable tool for analyzing and reporting software engineering case studies, offering a systematic approach to evaluate study quality and applicability. How to map Software Engineering practices to the Essence Framework knowledge domain? To answer this question, (Ng et al., 2013) propose to address core Software Engineering development issues, allowing a flexible approach to method composition. From (Uysal, 2018) point of view, the previous mapping of Software Engineering practices to Essence Framework based on activity spaces, ontology, and genetic algorithms did not bridge the gap therefore the need to introduce a new formal method using Concept Algebra for semantic evaluations. Concept Algebra (CA) is introduced as an abstract mathematical structure for formalizing concepts and their relations.

2.11.2 Demonstrations of tailored approaches

(Sedano & Peraire, 2015) present an empirical evaluation of team reflection support provided by the Software Engineering Method and Theory (SEMAT) Essence framework. By comparing the Essence Reflection Meetings and Agile Retrospectives, there were able to (1) Highlights the comparable nature of Essence reflection meetings to Agile retrospectives. (2) Emphasizes the holistic, state-based, goal-driven, and method-agnostic thinking framework of Essence. (3) Suggests leveraging both Essence and Agile retrospectives in a complementary fashion for project

teams. (Castro et al., 2020) argues that Essence provides universal elements in SE endeavors, emphasizing a language to describe and extend these elements for tailoring SE methods to team needs. In fact, the Essence Kernel separates stable aspects from adaptable ones, facilitating adoption and customization, supporting diverse approaches to educating SE students and training practitioners.

2.12 Challenges Faced by the Essence Method

2.12.1 Implementation Challenges

The ESSENCE standard addresses process-related aspects of software engineering projects, providing a language for software engineering process descriptions and defining a kernel of key elements (alphas and activity spaces) relevant to any software engineering project (Jacobson et al., 2019). Alphas represent key aspects of a project, and each alpha has states with checklists to track project progress. It is a new way of representing project journey so (Brandt et al., 2017) highlights Practical challenges with using physical cards for managing multiple projects in parallel. (Zmeev & Zmeev, 2020) argue that the challenges are more on the skills side. For them, the integration of soft skills (teamwork, time management, etc.) with software engineering processes, necessitating an understanding of their interconnections. It becomes more difficult knowing that acquiring these skills through traditional lecture-based methods and proposes the use of the project method is not obvious due to lack of paper. Another obstacle is the understanding and applying Alphas related to Opportunity, Stakeholders, and Endeavor areas, emphasizing the difficulty in developing Work Products for the Endeavor area. While organizations are transitioning to Agile but face challenges in adopting a suitable method (Raharjo et al., 2023).

2.13 Team composition and project success

According to (Raharjo et al., 2023), the rapid growth of Agile development in the software industry is attributed to its efficacy in addressing complex problems and facilitating seamless transitions from traditional project management methodologies. Despite its widespread adoption, the customization of Agile methods to fit specific organizational contexts remains a significant

challenge. This literature review explores the critical role of Agile team composition in project success, identifies key success factors, and discusses the potential for customization through frameworks like Essence.

Agile methodologies, such as Scrum, Kanban, and Extreme Programming (XP), have become foundational in contemporary software development due to their focus on iterative progress, collaboration, and adaptability (Beck et al., 2001). These methodologies provide structured frameworks that include defined roles, ceremonies, and artifacts designed to promote efficient team workflows and continuous improvement (Schwaber & Sutherland, 2020). Each methodology has unique features that contribute to its effectiveness in different contexts. For instance, Scrum emphasizes time-boxed iterations and roles such as the Scrum Master and Product Owner, while Kanban focuses on visualizing work and limiting work-in-progress to enhance flow.

The composition of Agile teams is pivotal to the success of Agile projects (Beck et al., 2001). Effective team composition involves selecting members with complementary skills and fostering a collaborative environment that encourages active participation from all team members. Key roles within Agile teams include the Product Owner, Scrum Master, and Development Team (Schwaber & Sutherland, 2020). The Product Owner is responsible for defining the product vision, managing the product backlog, and ensuring that the team delivers value to the customer. The Scrum Master facilitates the Agile process, removes impediments, and supports the team in its pursuit of continuous improvement. The Development Team, composed of cross-functional members, works collaboratively to deliver increments of the product.

Agile team composition significantly influences project success by enhancing communication, increasing adaptability, and fostering a collaborative culture. Effective communication and collaboration are essential for Agile teams, leading to better problem-solving, faster decision-making, and higher-quality deliverables (Beck et al., 2001). Regular ceremonies, such as daily stand-ups, sprint reviews, and retrospectives, play a crucial role in facilitating these interactions and ensuring that team members are aligned with the project goals (Sutherland, 2014).

Adaptability is another critical factor for Agile teams, enabling them to respond quickly to changes in project requirements or market conditions (Laufer et al., 2018). This adaptability is essential for maintaining alignment with business goals and customer needs. Agile methodologies emphasize continuous improvement through iterative development and regular feedback loops, allowing teams to refine processes, enhance product quality, and optimize performance over time.

Several key success factors contribute to the success of Agile projects, including team composition, leadership, organizational support, and the implementation of Agile practices (Denning, 2018). Selecting team members with diverse skills and fostering a collaborative environment is crucial for Agile teams (Schwaber & Sutherland, 2020). Teams should be cross-functional, with members capable of performing various tasks to ensure flexibility and resilience. Effective leadership, particularly from the Scrum Master and Product Owner, is vital for empowering teams, facilitating collaboration, and ensuring alignment with the project vision (Schwaber & Sutherland, 2020). Organizational support, including management buy-in and the provision of necessary resources, is crucial for the successful adoption of Agile methodologies. A culture that embraces Agile principles and values continuous improvement enhances project success (Moe et al., 2012).

The implementation of Agile practices, such as iterative development, continuous integration, and regular retrospectives, drives project success by ensuring that teams can deliver high-quality products efficiently and respond to changes effectively (Moe et al., 2012). Iterative development allows teams to deliver small, usable increments of the product, providing opportunities for regular feedback and continuous improvement (Conboy, 2009). Continuous integration ensures that code changes are integrated and tested frequently, reducing the risk of integration issues and enhancing product quality. Regular retrospectives provide a structured opportunity for teams to reflect on their processes, identify areas for improvement, and implement changes to enhance performance.

Customization of Agile team composition allows organizations to tailor their Agile practices to fit their unique needs, thereby enhancing the effectiveness of Agile methodologies (Kuhrmann et al., 2018). The Essence framework, developed by Ivar Jacobson, provides a common ground for developing customized Agile methods based on shared understanding and best practices. The

Essence framework offers a thinking framework for teams to collaborate, discuss, and improve Agile methods. It provides a foundation for defining practices independent of specific methodologies, allowing for customization to address specific organizational challenges and goals.

A case study of a national-wide bank in Indonesia demonstrates the successful implementation of an Essence-based Agile method (Castro et al., 2020). The organization developed its Agile practices by integrating elements from Scrum, Kanban, XP, and other popular Agile methods. Also, the study structured each individual's role, experience, specialization, and certifications, providing a clearer and more structured overview.

Code	Profession	Experience	Specialization	Certification
N1	Consultant	25 Years	Agile Software Development, Agile DevOps, Project Management, IT Governance	COBIT 5 Foundation certification, Certified Information Systems Auditor, Scrum Master, Professional Agile Coaching, Certified DevOps Foundation, ITIL Foundation - Intermediate Banking Risk Management Certified Level 1-3, dll
N2	Consultant	17 Years	Project Management, Agile Software Development, DevOps	Certified DevOps, ITIL, Certified Agile Coach, Scaled Agile (SAFe) Agilist v5.1, SCRUM Master (CSM), Certified SCRUM Professional (CSP), Certified Kanban System Design (KMP-1), COBIT 5 Foundation, dll
N3	Senior Manager	17 Years	Project Management, Agile Software Development, Business Analyst	Scrum Master Certified (SMC), Project Management Professional (PMP)
N4	Assistant Manager	9 Years	Agile DevOps, Quality Assurance, Agile Software Development	Scrum Master, DevOps Foundation, Quality Management System (ISO 9001:2015)
N5	Senior Manager	13 Years	Quality Assurance, Agile Sofware	Certified Agile Tester By International Software Quality Institute (ISTQB) Certified Data Management Professional (CDMP)

			Development, Agile DevOps	

Table 1 : The profiles of the expert (Raharjo et al., 2023)

2.14 Business Technology Management (BTM)

The rapid evolution of digital technology has significantly transformed business processes and management practices (Gagnon, 2023). Business Technology Management (BTM) and digital transformation have become central themes in contemporary research and practice (Gagnon, 2020). This literature review synthesizes the current knowledge on BTM and digital transformation, highlighting key concepts, methodologies, and findings from recent studies.
BTM encompasses the alignment of business and IT strategies to enhance organizational performance. (Avison & Malaurent, 2014) question the dominance of theory in information systems, emphasizing practical relevance in BTM. (Schahczenski & Dyne, 2019) discuss the importance of web-based tools for program assessment, facilitating the measurement of student outcomes for accreditation, which is crucial in the context of BTM education.

2.14.1 Key Components and Best Practices

(Saulnier & White, 2011) analyze ABET-accredited Information Systems programs, identifying essential components for curriculum development that align with industry standards. (Winter et al., 2011) debate the impact of accreditation on competitiveness and bureaucracy in business and information systems engineering programs, suggesting a balance between regulation and innovation.

2.14.2 Digital Transformation

Digital transformation involves the integration of digital technologies into all areas of a business, fundamentally changing how organizations operate and deliver value to customers. (Zhu et al., 2006) explore the determinants of digital transformation adoption in European companies, highlighting the role of innovation diffusion. (Venkatesh, 2008) examines the impact of digital home technologies on household transformation, demonstrating the pervasive nature of digital change.

2.14.3 Applications in Business Technology Management

The application of digital transformation in BTM is multifaceted. (Agarwal et al., 2010) discuss the transformation of healthcare through digital technologies, focusing on current status and future directions. (Osmani et al., 2012) propose a conceptual framework for evaluating public sector transformation in the digital era, emphasizing the need for a comprehensive assessment model.

2.15 Semantic Integration

[11]Semantic integration refers to the process of combining data from different sources by ensuring that the meaning, context, and relationships of the data are consistent and accurately represented, enabling interoperability and effective data sharing (Euzenat & Shvaiko, 2013). This is crucial in BTM, where disparate data systems often need to be integrated for effective decision-making. (Deng, 2007) introduces a transformation matrix for digital filters, illustrating the technical aspects of semantic integration in digital signal processing. (Deng, 2007) introduced a transformation matrix for even-order Lagrange-type variable fractional-delay digital filters, demonstrating the application of semantic integration in digital signal processing. This approach can be extended to integrate diverse data sets in project management and BTM, ensuring consistency and accuracy.
Semantic integration techniques include ontology-based integration, schema matching, and data fusion. These techniques help in resolving conflicts and inconsistencies between different data

[11] In the context of the semantic web and information systems, semantic integration focuses on aligning the meaning (or semantics) of data from heterogeneous sources, often using standardized languages like RDF (Resource Description Framework) and OWL (Web Ontology Language). This ensures that different systems can interpret the data correctly, regardless of its origin or format. The goal is to enable richer, more meaningful data sharing, querying, and analysis.

sources, enabling a more coherent and comprehensive understanding of the integrated data. Ontologies, for instance, provide a shared and common understanding of a domain that can be communicated between people and application systems, which is particularly useful in complex project management scenarios where multiple stakeholders and systems are involved. (Gruber, 1995)

2.15.1 Applications in Project Management

In project management, semantic integration is vital for ensuring that data from various project management tools, databases, and communication platforms are consolidated into a single, coherent system. This integration facilitates better project tracking, resource allocation, and decision-making. For example, (Goodman & Aburdene, 2009) presented a recursive matrix approach to spectral transformations for digital filters, which can be adapted to project management systems to integrate various data streams into a unified project dashboard.

Semantic integration also supports automated project team composition by integrating data on team members' skills, project requirements, and historical performance. This integrated data can be used to form optimal project teams that are well-suited to the specific needs and challenges of a project. Automated systems can analyze integrated data to recommend team configurations that maximize efficiency and effectiveness.

2.15.2 Applications in Business Technology Management

In the realm of BTM, semantic integration ensures that business processes and IT systems are aligned and can communicate effectively. This alignment is crucial for the smooth operation of business functions and for leveraging technology to achieve business goals. (Zhu et al., 2006) emphasized the importance of innovation diffusion in the digital transformation of European companies, which relies heavily on the ability to integrate diverse data sources to support innovative processes and decision-making.

Semantic integration techniques enable the consolidation of data from various business units and IT systems, facilitating a comprehensive view of business operations. This comprehensive view is essential for strategic planning, performance monitoring, and decision-making. For instance,

(Deng, 2007) work on transformation matrices can be applied to integrate financial data, customer data, and operational data, providing a holistic understanding of business performance and enabling more informed decision-making.

2.15.2.1 Business Technology Management Body of Knowledge (BTM-BOK)

Business Technology Management (BTM) is a burgeoning field that integrates various aspects of information systems (IS) and information technology (IT) with business management. This literature review synthesizes key contributions from Stéphane Gagnon, a prominent scholar in BTM, focusing on digital transformation, rebranding of IS/IT programs, artificial intelligence, and ontology-driven analytics. Gagnon's work offers a cohesive understanding of BTM's role in shaping modern digital organizations and developing new competency frameworks essential for future leaders.

Digital transformation has accelerated significantly, especially in the context of the COVID-19 pandemic. (Gagnon, 2023) explores this phenomenon, highlighting the pre-pandemic and post-pandemic shifts in digital strategies. He argues that the pandemic has catalyzed digital transformation, forcing organizations to adopt new technologies and digital processes at an unprecedented pace. This acceleration has underscored the need for a new generation of leaders with hybrid skillsets capable of managing IT and co-creating digital organizations (Gagnon, 2023).

The rapid adoption of digital technologies has not only transformed business operations but also redefined competitive landscapes. Organizations that quickly adapted to digital transformation have gained significant competitive advantages, leveraging digital tools to enhance customer experiences, streamline operations, and create new business models (Gagnon, 2023). This shift necessitates a comprehensive understanding of digital technologies and their strategic implementation within business frameworks.

The rebranding of IS/IT management programs to incorporate BTM principles is another significant theme in Gagnon's work. In his article, (Gagnon, 2022) discusses the Canadian

initiative to rebrand IS and IT programs under the BTM framework. This initiative aims to unify various specializations such as business analysis, enterprise architecture, IT services management, and project management into a cohesive body of knowledge (BOK) for digital transformation projects.

(Gagnon, 2022) emphasizes the importance of an integrated transdisciplinary competency framework that spans business, computing, and engineering disciplines. The BTM framework aims to address the fragmentation and competition among these specializations by promoting collaboration and seamless career paths. This integrated approach ensures that professionals are well-equipped with the necessary skills to manage and lead digital transformation projects effectively.

Artificial Intelligence (AI) and digital technologies are transforming various fields, including accountancy and talent management. (Gagnon, 2022) examines the impact of these technologies on the accountancy profession, highlighting how AI and digital tools are reshaping traditional accounting practices. AI's ability to automate routine tasks and provide advanced analytical capabilities has significantly enhanced the efficiency and effectiveness of accounting processes (Gagnon, 2022).

Moreover, the integration of AI in talent management processes has revolutionized how organizations attract, retain, and develop talent. (Gagnon, 2022) points out that AI-driven talent management systems can analyze vast amounts of data to identify the best candidates, predict employee performance, and tailor development programs to individual needs. This data-driven approach not only improves hiring decisions but also enhances employee engagement and productivity.

(Gagnon, 2022) also explores the application of ontologies in parliamentary analytics, specifically analyzing political debates on the COVID-19 impact in Canada. Ontologies, which provide a structured framework for organizing information, are instrumental in enhancing the analysis of complex and large datasets (Gagnon, 2022). By using ontology-driven analytics, researchers can

systematically categorize and interpret parliamentary debates, gaining deeper insights into policy discussions and legislative processes.

This approach has significant implications for improving transparency and accountability in government. By making parliamentary data more accessible and analyzable, ontology-driven analytics can help citizens better understand the impact of policies and engage more effectively in the democratic process (Gagnon, 2022). Furthermore, this methodology can be extended to other domains, providing a powerful tool for analyzing and managing large-scale information systems.
Business Technology Management as Transdisciplinary IS-IT Competency Framework

The development of a transdisciplinary competency framework for BTM is a cornerstone of Gagnon's research. In his ICIS 2020 paper, (Gagnon, 2020) outlines the BTM initiative's goals and its impact on IS/IT education. He describes how the BTM framework aims to unify various disciplines and create a more integrated profession, essential for managing the complexities of digital transformation.

The BTM Body of Knowledge (BOK) is designed to bridge the gaps between business, computing, and engineering disciplines, promoting a holistic approach to digital transformation. (Gagnon, 2020) argues that this integrated framework is crucial for developing leaders who can navigate the rapidly changing digital landscape and drive innovation within their organizations. The BTM BOK provides a comprehensive set of competencies that encompass technical, managerial, and strategic skills, ensuring that professionals are well-prepared to tackle the challenges of digital transformation.

The BTM-BOK provides a comprehensive framework for integrating business and technology management practices. It covers various aspects such as governance, compliance, architecture, security, and platform management, essential for effective PM. The construction of an ontology framework for Business Technology Management (BTM) is a task of considerable complexity and significance, aiming to systematically capture and represent the knowledge, practices, and expertise within the BTM domain. Using Protégé, a powerful ontology editor, this narrative explores the process of translating the BTM Body of Knowledge (BTM-BOK) into a structured

and usable ontology. This comprehensive guide provides a detailed account of the steps, methodologies, and considerations involved in creating such a framework, facilitating a deeper understanding of BTM and enhancing its practical application.

Ontology in the context of knowledge management refers to a formal representation of a set of concepts within a domain and the relationships between those concepts. It is a critical tool for organizing information in a way that enables interoperability, sharing, and reuse. The BTM-BOK serves as an extensive repository of information on business technology management, covering various aspects such as digital transformation, strategic management, and technological innovation. By developing an ontology based on the BTM-BOK, we aim to create a structured framework that can be used for various applications, including knowledge sharing, professional development, and decision-making.

The methodology for developing the BTM ontology involves several key steps: defining classes, establishing subclasses, defining object and data properties, creating individuals, and structuring the ontology within Protégé. Each of these steps is informed by the detailed structure and content of the BTM-BOK, ensuring that the ontology accurately represents the domain.

2.15.3 Essence Framework and Semantic Integration

The Essence framework for software engineering methods provides a structured approach to managing the essential elements of software projects. Semantic integration plays a crucial role in the Essence framework by ensuring that all elements of a project are consistently and accurately represented. This consistency is achieved by integrating data from various project management tools, software development environments, and communication platforms.

Semantic integration in the Essence framework helps in aligning project goals, processes, and outcomes. It ensures that all stakeholders have access to a unified view of the project, which facilitates collaboration and reduces misunderstandings. For example, the transformation techniques discussed by (Goodman & Aburdene, 2009) can be used to integrate data on project

milestones, deliverables, and team performance, providing a comprehensive view of project progress.

Semantic integration is a critical component of effective project management, BTM, and the Essence framework. By consolidating data from various sources, semantic integration enables more informed decision-making, better resource allocation, and improved alignment of business and IT strategies. The techniques and applications discussed in the literature provide valuable insights into how semantic integration can be leveraged to enhance organizational performance and achieve strategic objectives.

2.16 Project management knowledge domain

Project management (PM) is a crucial discipline that involves planning, executing, and overseeing projects to achieve specific goals within specified constraints. Over the years, project management methodologies and frameworks have evolved to address the growing complexity and diversity of projects across various industries. Multiple sources provide academic overview of the project management knowledge domain, focusing on project performance domains and team composition for project success.

2.16.1 Project Management Frameworks and Methodologies

The Project Management Institute (PMI) provides extensive guidelines and standards for project management, encapsulated in the PMBOK Guide. The PMBOK Guide emphasizes a structured approach to project management, outlining key knowledge areas such as scope, time, cost, quality, resource, communication, risk, procurement, and stakeholder management.
The PM² methodology, developed by the European Commission, offers a comprehensive framework tailored to public sector projects. PM² integrates traditional project management practices with agile methodologies to enhance flexibility and adaptability.

2.16.2 Project Performance Domains

Project performance domains are critical areas of focus that drive successful project outcomes. According to the PMBOK Guide, these domains include stakeholder performance, team performance, development approach and life cycle, planning, project work, delivery, measurement, uncertainty, and tailoring. Effective stakeholder management involves identifying, analyzing, and engaging stakeholders to ensure their needs and expectations are met throughout the project lifecycle.

High-performing teams are essential for project success, with key factors such as open communication, shared understanding, trust, collaboration, adaptability, resilience, empowerment, and recognition playing pivotal roles. Selecting the appropriate development approach, whether predictive, iterative, or hybrid, is vital for aligning the project with its goals and constraints, including defining the project phases, processes, and milestones. Detailed planning ensures that project objectives are met within the agreed scope, time, and cost constraints, involving comprehensive plans for scope, schedule, cost, quality, resource, communication, risk, procurement, and stakeholder management.

Executing the project work involves coordinating people and resources, managing stakeholder expectations, and integrating and performing the activities of the project in accordance with the project management plan. The delivery performance domain focuses on ensuring that the project outputs meet the intended quality and are delivered on time and within budget, which includes monitoring and controlling project work and managing changes to the project scope. Measuring project performance involves tracking progress against the project plan and implementing performance improvements as necessary, using key performance indicators (KPIs) and metrics to assess project health and success.

Managing uncertainty involves identifying, analyzing, and responding to risks and uncertainties that could impact the project, which includes developing risk management plans and contingency strategies. Lastly, tailoring involves adapting the project management approach to suit the specific context and needs of the project, ensuring that the project management practices are aligned with the project's environment, stakeholders, and objectives.

2.17 Team Composition for Project Success

Effective team composition is crucial for project success. High-performing teams exhibit characteristics such as open communication, shared understanding, trust, collaboration, adaptability, resilience, empowerment, and recognition. The PMBOK Guide highlights the importance of establishing a collaborative project team environment to facilitate alignment with organizational cultures and guidelines, individual and team learning, and optimal contributions to desired outcomes.

Agile methodologies, such as those outlined in the PM²-Agile Guide, emphasize self-organizing, cross-functional teams that adapt to changing demands and promote a collaborative and cooperative working environment. Agile teams focus on continuous improvement and validated learning, ensuring that everyone has a sense of belonging and contributes to the project's success. The composition of a project team is a critical factor in determining the success of a project. Effective team composition involves selecting the right mix of skills, experiences, and personalities to ensure that the project objectives are met efficiently and effectively.

2.17.1 Impact of team composition on Project Success

2.17.1.1 Diverse Skill Sets and Expertise

A well-composed team brings together a diverse set of skills and expertise, which is essential for tackling the various challenges that arise during a project. Diverse teams are better equipped to handle complex problems, innovate, and develop comprehensive solutions. This diversity enhances the team’s ability to deliver high-quality outputs and achieve project goals.

2.17.1.2 Enhanced Problem-Solving and Decision-Making

Teams with varied backgrounds and perspectives can approach problems from different angles, leading to more effective problem-solving and decision-making. This diversity of thought helps in

identifying potential risks and opportunities that might be overlooked by a more homogeneous team.

2.17.1.3 Improved Collaboration and Communication

Effective team composition promotes better collaboration and communication. When team members have complementary skills and personalities, they are more likely to work well together, share knowledge, and support each other. This collaborative environment is crucial for maintaining project momentum and ensuring that tasks are completed on time.

2.17.1.4 Increased Adaptability and Resilience

Projects often face unexpected changes and challenges. A well-composed team is more adaptable and resilient, capable of adjusting to new circumstances and maintaining productivity. This adaptability is particularly important in agile project management environments, where flexibility and rapid response to change are key.

2.17.1.5 Greater Innovation and Creativity

Teams composed of individuals with different experiences and viewpoints are more likely to generate innovative ideas and creative solutions. This innovation is critical for achieving project goals, especially in competitive and rapidly changing industries.

The composition of a project team plays a pivotal role in determining the success of a project. A well-composed team brings together diverse skills, enhances problem-solving capabilities, improves collaboration, and increases adaptability and innovation. By ensuring the right team composition, project managers can create an environment that supports the achievement of project goals and delivers high-quality outcomes. Effective team composition is not just a desirable attribute but a critical component of successful project management.

The PM²-Agile Guide highlights the importance of team composition in agile project management. Agile teams are typically small, cross-functional, and self-organizing, which allows them to adapt quickly to changing project requirements and deliver value incrementally.

Self-Organizing Teams: Agile teams have the autonomy to organize their work and make decisions, leading to higher motivation and ownership of project outcomes. This self-organization fosters a sense of responsibility and accountability among team members, driving them to achieve project goals.

Cross-Functional Collaboration: Agile teams consist of members with different functional expertise, enabling them to tackle various aspects of the project collaboratively. This cross-functional collaboration ensures that all necessary skills are available within the team to complete tasks effectively.

Continuous Improvement: Agile methodologies emphasize continuous improvement through regular feedback and retrospectives. Teams reflect on their performance and identify areas for improvement, which enhances their effectiveness and contributes to the achievement of project goals.

The composition of project teams plays a critical role in determining project success, as evidenced by various case studies and empirical research. This review synthesizes findings from several academic articles to highlight key factors and outcomes related to team composition.

2.17.2 Competence and Project Team Performance

Competence of team members is a pivotal factor in project success. According to a study by (Oh & Choi, 2020), the emotional, managerial, and intellectual competencies of team members significantly impact project outcomes. They emphasize that both project managers and team members contribute equally to the success of a project, suggesting that individual competencies need to be aligned with project goals to drive performance effectively (Oh & Choi, 2020). Their study examined the relationship between the emotional, managerial, and intellectual competencies of team members and project success. It involved a questionnaire survey of 164 project management professionals in Korea. The findings showed a significant positive impact of team members' competencies on project success. Both project managers and team members' competencies were crucial, suggesting a strategic direction for team composition to enhance

project outcomes. This aligns with earlier findings that highlight the importance of leadership capabilities and administrative skills in ensuring project success (Kerzner, 2017).

2.17.3 Trust and Collaboration in Project Team

Trust and collaboration within project teams are essential for achieving successful outcomes. Studies indicate that high levels of trust facilitate better communication and knowledge sharing, which are critical for collaborative work. For instance, research by Manu et al. shows that in construction projects, a trust-based collaborative environment leads to higher levels of information sharing and project success. This trust is built through consistent, reliable communication and mutual understanding among team members (Taryn et al., 2017). Their research focused on the impact of trust and collaboration on project success. It highlighted that trust facilitates better communication and knowledge sharing, which are critical for collaboration. The study found that a trust-based collaborative environment is essential for effective information sharing and achieving project success, particularly in the construction industry.

2.17.4 Diversity and Autonomy in Agile Project

Diversity and autonomy within teams also play a significant role in project success. (Campanelli & Parreiras, 2015) found that diverse teams, when given autonomy, perform better in agile project environments. By comparing agile IT project development methods, (Campanelli & Parreiras, 2015) found that diversity and autonomy within teams lead to better performance. The research indicated that diverse teams bring various perspectives and skills, enhancing problem-solving and innovation. Autonomy allows teams to adapt quickly to changes, further driving project success. This is corroborated by (Lechler & Yang, 2017), who noted that team diversity brings varied perspectives and skills, enhancing problem-solving and innovation capabilities. The autonomy granted to these teams allows them to adapt and respond quickly to changes, further driving project success.

2.17.5 Knowledge Integration Capability

Effective integration and management of knowledge are crucial for team success. (L. Dietrich et al., 2015) emphasize the importance of knowledge integration capability, suggesting that the ability of teams to assimilate and utilize knowledge from various sources determines project outcomes. This capability is enhanced in environments where trust and collaboration are prioritized, as noted by (Chiocchio et al., 2011).

2.17.6 Personality and Work Motivation

The process of team selection has a direct impact on project success. Case studies analyzed by (Garhoud & Bredillet, 2016) reveal that careful consideration of personality composition, academic balance, and alignment of interests can prevent conflicts and enhance team cohesion. These factors are essential in creating a balanced team that can work effectively towards common goals. Their research investigated the impact of personality composition and work motivation on project success. It found that careful selection of team members based on personality traits and motivational factors can prevent conflicts and enhance team cohesion. Balanced teams with aligned interests perform better, contributing to project success.

2.17.7 Comparison and Synthesis

When comparing these findings, it becomes evident that successful project teams share common attributes: high competence levels, strong trust and collaboration, diversity, autonomy, and effective knowledge management. These attributes are interrelated; for instance, diversity can enhance knowledge integration, while trust facilitates better collaboration. The interplay of these factors creates a conducive environment for project success.

However, the degree to which each factor impacts project success can vary depending on the project type and context. For example, in agile projects, autonomy and diversity might be more critical, whereas in traditional projects, managerial competence and structured collaboration could be more important.

The case studies collectively emphasize that the success of a project is significantly influenced by several key factors.

First, the competence of team members is crucial, as emotional, managerial, and intellectual competencies play a critical role in project success. Both project managers and team members must effectively utilize their competencies to drive the project forward.

High levels of trust and collaboration within the team are also essential, as they facilitate better communication, knowledge sharing, and overall project success.

Diversity and autonomy within teams contribute to innovation and adaptability, particularly in agile project environments, leading to better performance. The ability to integrate and utilize knowledge from various sources is another essential factor, with trust and collaboration further enhancing this capability.

Finally, teams that are properly composed based on personality traits and motivational factors tend to be more cohesive and perform better, highlighting the importance of personality and motivation in team dynamics.

2.18 Team composition as Project Management Core

The importance of team composition in project management cannot be overstated, as it directly influences project success. Competent team members with the necessary emotional, managerial, and intellectual skills are critical for achieving project goals (Oh & Choi, 2020). Trust and collaboration among team members enhance communication and knowledge sharing, which are essential for effective project execution (Taryn et al., 2017). Diverse teams, particularly those with autonomy, bring varied perspectives and skills, fostering innovation and adaptability in agile environments (Campanelli & Parreiras, 2015). Additionally, the ability to integrate and utilize knowledge from multiple sources is a key determinant of project success, supported by a collaborative and trustful team environment (P. Dietrich et al., 2010). Therefore, strategic team composition, focusing on competencies, trust, diversity, and knowledge integration, is paramount for ensuring successful project outcomes (Garhoud & Bredillet, 2016).

2.19 How crucial is team composition automation?

The use of decision-making tools in composing a project team is crucial for optimizing team performance and ensuring project success. Decision-making tools facilitate the identification and selection of team members who possess the necessary competencies, skills, and attributes required for the project's specific needs (Belbin, 2010). These tools enable project managers to assess potential team members based on a variety of criteria such as expertise, experience, personality traits, and team roles, ensuring a balanced and effective team composition (Meredith & Mantel, 2012).

By systematically evaluating candidates, decision-making tools help in mitigating biases that might arise from subjective judgment, thus promoting diversity and inclusion within the team (J. Thomas & Mengel, 2008). For instance, tools like the Myers-Briggs Type Indicator (MBTI) and Belbin Team Roles Model can be used to understand the personality types and preferred working styles of team members, ensuring that complementary skills and roles are represented within the team (Belbin, 2010) ; (Myers et al., 2015). This balance is essential for fostering a collaborative environment where team members can leverage each other's strengths and compensate for individual weaknesses (Salas et al., 2008).

Moreover, decision-making tools aid in predicting potential challenges related to team dynamics and interpersonal conflicts. By understanding the psychological and professional profiles of team members, project managers can proactively address and manage potential issues, thereby enhancing team cohesion and productivity (Kerzner, 2017). Tools such as SWOT analysis (Strengths, Weaknesses, Opportunities, and Threats) and PESTLE analysis (Political, Economic, Social, Technological, Legal, and Environmental) provide comprehensive frameworks for assessing internal and external factors that might impact team performance (Mind Tools, 2020).
Incorporating decision-making tools in team composition also aligns with strategic project management practices by ensuring that the team structure supports the overall project objectives and organizational goals (PMI, 2017). These tools facilitate data-driven decisions, thereby improving the likelihood of project success and delivering value to stakeholders (Laufer et al., 2018).

Automated project team composition involves using algorithms and data analytics to form project teams. This approach addresses the complexity and dynamic nature of modern projects. The importance of this process is highlighted by (Avison & Malaurent, 2014), who stress the need for practical relevance in information systems research, which can inform team composition strategies. Automated project team composition is an advanced methodology that leverages algorithms and data analytics to optimize the formation of project teams. This approach addresses the inherent complexity and dynamic nature of modern projects by ensuring that teams are composed of individuals with the appropriate mix of skills, expertise, and personalities to achieve project goals efficiently.

2.19.1 Importance and Challenges

Automated project team composition is crucial because it enhances the allocation of human resources, which in turn improves project outcomes and efficiency. A significant challenge in this approach is the accurate capture and integration of data regarding team members' skills, project requirements, and historical performance. Proper integration ensures that selected team members are well-suited to meet the specific needs of the project and can adapt to any changes or unforeseen challenges during the project lifecycle. For example, the flexibility of workers with multiple skills is critical for success in dynamic project environments, as emphasized by research on multi-skilled team composition (Tarafdar & Qrunfleh, 2010) ; (Weiss & Thorogood, 2011).

2.19.2 Techniques and Approaches

2.19.2.1 Description Logics and Semantic Matching

Description logics provide a robust framework for the semantic-based composition of task-oriented teams. Automated systems can use these logics to match candidates' profiles with task descriptions, ensuring that team members' skills and experiences align with project requirements. This approach optimizes team performance by leveraging the strengths of each team member (Kearns & Sabherwal, 2006).

2.20 What is an Ontology?

In recent years, the field of project management has seen significant advancements with the integration of semantic technologies, particularly in the use of ontologies (Cardoso & Ferreira, 2009; Hepp, 2007). Ontologies, as structured frameworks for representing knowledge within specific domains, offer a sophisticated approach to automating various complex processes, such as team composition in project management. By leveraging ontologies, project managers can ensure more efficient and accurate team assembly, ultimately enhancing project outcomes. An ontology, in the context of computer science and artificial intelligence, is a formal and explicit specification of a shared conceptualization of a domain (Gruber, 1993). Essentially, ontologies define a set of concepts and the relationships between them, enabling the representation of knowledge in a structured and standardized manner. In practical terms, ontologies can be thought of as a formal vocabulary that describes the entities within a particular domain and the interactions among them. For example, an ontology in project management might include concepts such as "project", "team member", "task", "skill", and "role", along with the relationships that define how these concepts interact (e.g., a "team member" has a "skill", and a "project" requires certain "skills").

The application of ontologies in automating team composition offers a sophisticated approach to assembling project teams by leveraging structured knowledge representation and semantic reasoning. Ontologies define a set of concepts and relationships within a domain, enabling the precise mapping of project requirements to team member attributes. This automated process enhances the efficiency and accuracy of team formation by integrating diverse data sources and applying logical rules to match skills, experiences, and roles effectively (Bollen et al., 2011). For instance, using an ontology-based system, project managers can ensure that the selected team members possess the necessary competencies and are well-aligned with the project objectives and dynamics (Noy & McGuinness, 2001). Moreover, such systems can continuously update and refine team compositions based on ongoing project performance data and evolving requirements, thereby optimizing team effectiveness over time (Cardoso & Ferreira, 2009). This approach not only

reduces the time and effort involved in team selection but also enhances the likelihood of project success by ensuring a better fit between team members and project needs.

2.20.1 Key Features of Ontologies

Ontologies consist of several core components that contribute to their effectiveness in knowledge representation. These components include:

- Classes: The core categories or concepts within the ontology (e.g., “Project,” “Task,” “Resource”). Classes provide a general description of the things that exist in the domain.

- Instances: Instances are the specific elements or objects within each class (e.g., “Project A,” “Task 1”). They represent particular examples or occurrences of the defined classes.

- Attributes: These are properties or characteristics associated with classes or instances (e.g., “start date,” “duration” for a project). Attributes help to further describe the entities in the ontology.
- Relationships: Relationships define how classes and instances are linked to each other (e.g., "is part of,” "is managed by"). These connections are critical in capturing the interactions and dependencies within the domain.

Together, these elements help form a detailed structure that allows for a comprehensive understanding of the concepts within a given domain. According to (Staab & Studer, 2009), ontologies serve not only to define concepts but also to provide rules about how these concepts interact, making them invaluable tools in knowledge management and integration tasks.

2.20.2 Purpose of Ontology

Ontologies help structure information, enabling interoperability, consistency, and machine-readability. They underpin systems like the Semantic Web, Natural Language Processing (NLP), and Knowledge Graphs (Berners-Lee et al., 2001).

2.20.3 Steps to Build an Ontology

To build an ontology, it is essential to define a clear purpose and scope, ensuring it aligns with the intended domain and use cases. Begin by identifying key concepts, relationships, and constraints through domain analysis and stakeholder consultation. Focus on clarity and coherence to establish unambiguous definitions and relationships. Employ modular design principles to create an extendable and reusable structure, following best practices like those outlined by (Uschold & Gruninger, 1996). Additionally, ensure scalability by designing the ontology to accommodate future growth and changes in the domain. Throughout the process, validate and refine the ontology iteratively to maintain accuracy and relevance.

- Step 1: Define Scope and Purpose Begin by answering questions like "What domain will the ontology cover?" and "What is its purpose?" (Noy & McGuinness, 2001).
- Step 2: Identify Key Terms List core concepts and terminologies relevant to the domain.

- Step 3: Define Classes and Hierarchies Establish categories and organize them hierarchically using "is-a" or "part-of" relationships.
- Step 4: Specify Properties Define attributes (e.g., age, color) and relationships (e.g., owns, belongs-to).
- Step 5: Add Constraints and Axioms Introduce rules to ensure logical consistency (e.g., "Every student must enroll in at least one course").
- Step 6: Validate and Test Use reasoning tools to check for logical inconsistencies.
- Step 7: Iterate and Refine Continuously improve the ontology based on feedback and testing.

2.20.4 Ontology Design Principles

Clarity and coherence are prioritized to ensure that definitions and relationships are unambiguous. Modularity is emphasized by designing the ontology to be extendable and reusable, as highlighted by (Uschold & Gruninger, 1996). Additionally, scalability is incorporated to ensure the ontology remains adaptable for growth.

2.20.5 Tools for Building Ontologies

Building ontologies requires tools that facilitate not just the creation but also the visualization, management, and validation of complex knowledge structures. Over the years, various tools have emerged to cater to these needs, ranging from open-source platforms to enterprise-grade software. This section provides a narrative journey through some of the most prominent tools used in ontology development.

2.20.5.1 Protégé

A widely used open-source tool, Protégé, provides an interface for creating, visualizing, and managing ontologies. It supports OWL (Web Ontology Language) and RDF (Resource Description Framework) standards (Musen, 2015).

2.20.5.2 TopBraid Composer

A commercial tool designed for developing semantic models and ontologies, particularly useful for enterprise-level projects.

2.20.5.3 OntoUML Editor

Specialized for conceptual modeling using the Unified Modeling Language (UML) for ontologies (Guizzardi, 2005).

2.20.5.4 Web-Based Tools

WebVOWL: Visualizes ontologies in an interactive graph format.
OWLGrEd: Offers diagrammatic representations for OWL ontologies.

2.20.6 Use Cases of Ontologies

Ontologies, as frameworks for structuring and reasoning about knowledge, have become indispensable across various industries. Their ability to represent complex relationships and provide semantic meaning has unlocked new possibilities in data organization, decision-making, and automation. Let's explore some of the transformative use cases of ontologies, bringing their impact to life.

2.20.6.1 Semantic Web

Ontologies form the foundation of the Semantic Web by providing a structured framework for metadata and linking diverse datasets (Berners-Lee et al., 2001).

2.20.6.2 Healthcare

In healthcare, ontologies like SNOMED CT facilitate clinical decision support, interoperability between systems, and medical research.

2.20.6.3 E-Commerce

Retailers use ontologies to improve product search and recommendation systems by linking customer preferences with product attributes.

2.20.6.4 Education

Ontologies enhance e-learning platforms by organizing learning resources and tailoring content to individual needs (Mizoguchi et al., 2007).

2.20.6.5 Knowledge Graphs

Google's Knowledge Graph uses ontologies to provide rich, contextual search results.

2.20.6.6 Natural Language Processing (NLP)

Ontologies improve machine understanding of text, enabling applications such as sentiment analysis, chatbot development, and information retrieval.

2.21 Ontology in Project Management

Project management is a complex field that encompasses numerous variables, processes, and interdependencies (Kerzner, 2017). Whether managing a large-scale infrastructure project or a small software development initiative, project managers are tasked with coordinating tasks, schedules, budgets, resources, and people. However, as projects become more complex and multidisciplinary, one of the key challenges in project management is managing the vast amounts of information from various systems, stakeholders, and methodologies (Hepp, 2007). This is where ontologies can play a transformative role.

Ontology, in the context of information science and project management, refers to a formalized representation of knowledge (Gruber, 1993). It defines the concepts and categories that exist in a particular domain and specifies the relationships between those concepts. At its core, an ontology in project management provides a common vocabulary and framework for understanding project components, which is crucial when working with diverse teams, stakeholders, and systems (Cardoso & Ferreira, 2009).

Ontologies help organize and integrate complex information into a coherent structure (Fensel et al., 2001). They enable stakeholders to share a common understanding of key concepts and processes within a project, regardless of their role or background. For example, in a construction project, the concept of a "task" might differ between the architect, the contractor, and the project manager (Euzenat & Shvaiko, 2013). An ontology ensures that all these participants share a unified definition of what constitutes a task, as well as how it relates to other elements like "resources," "deliverables," and "deadlines (Staab & Studer, 2009)."

2.21.1 The Role of Ontologies in Team Composition

The process of team composition involves selecting the right mix of team members based on a set of predefined criteria, such as skills, experience, availability, and role requirements. Traditionally, this process has been manual, often relying on subjective judgment and intuition. However, with the application of ontologies, team composition can be automated, reducing human error, improving decision-making, and optimizing team performance.

2.21.1.1 Mapping Project Requirements to Team Member Attributes

One of the primary ways ontologies facilitate team composition is through their ability to map project requirements to team member attributes. Ontologies enable the definition of project goals and the identification of the skills and experiences required to achieve those goals. For example, if a project requires expertise in web development, the ontology can define the necessary skills, such as proficiency in HTML, CSS, JavaScript, and specific frameworks like React or Angular. Team members with these skills can be identified based on their profile attributes, which may include education, certifications, and previous work experience.

By using an ontology-based system, project managers can automate the mapping of these requirements to team member profiles, ensuring that the selected team members possess the necessary competencies for the project's success. This process is far more efficient than manual selection, as it leverages structured knowledge and reduces the possibility of overlooking critical skills (Bollen et al., 2011).

2.21.1.2 Enhancing Team Fit and Dynamics

Ontologies also help improve the fit between team members and the project’s dynamics. Team composition is not just about selecting individuals with the right technical skills; it also involves considering factors like interpersonal relationships, communication styles, and collaboration potential. These factors can be captured within the ontology, enabling more refined team selections that go beyond technical competencies.

For example, an ontology can include concepts related to teamwork skills, such as "communication style", "leadership potential", or "collaborative behavior". By including these elements in the team composition process, the system can ensure that team members are not only technically proficient but also well-suited to work together effectively. Such an approach can lead to a more cohesive team with better collaboration and higher performance (Bollen et al., 2011).

2.21.1.3 Continuous Refinement Based on Project Feedback

One of the most significant advantages of using ontologies in team composition is the ability to continuously refine and update team compositions throughout the lifecycle of the project. As the project progresses, team dynamics, performance, and requirements may evolve, and the ontology-based system can adapt accordingly.

For instance, as team members acquire new skills or as the project scope changes, the ontology can be updated to reflect these changes. This continuous adaptation ensures that the team composition remains optimal over time, improving team performance and project outcomes. Moreover, the system can analyze ongoing project performance data to identify areas where team composition could be improved, such as by reallocating resources or shifting roles to better align with evolving project needs (Cardoso & Ferreira, 2009).

2.21.1.4 Reducing Time and Effort in Team Selection

The use of ontologies significantly reduces the time and effort required to select the right team members (Fensel et al., 2001). Traditional team selection often involves manually reviewing resumes, conducting interviews, and assessing each individual's fit for the project. This process can be time-consuming, especially for large-scale projects (Flyvbjerg, 2017). By automating the selection process through an ontology-based system, project managers can quickly identify the best candidates for the team based on predefined criteria, thus saving valuable time and resources (Gasevic et al., 2007).

Additionally, the ontology-based approach can help eliminate biases and inconsistencies in team selection, as the process is guided by structured rules rather than subjective opinions (Hepp, 2007). This results in a more transparent and fair selection process, where decisions are made based on objective criteria rather than personal preferences or assumptions (Noy & McGuinness, 2001).

Studies and Applications of Ontologies in Project Management

Several studies have demonstrated the potential of ontologies in automating team composition and improving project management outcomes. (Bollen et al., 2011) explored the use of ontologies in assembling project teams, highlighting how ontologies can be used to match team members' skills and experiences with project requirements. The authors emphasized that ontology-based systems offer an efficient and accurate way to form teams, ensuring that all necessary competencies are represented while minimizing human error and bias (Gasevic et al., 2007).

Similarly, (Noy & McGuinness, 2001) discussed the role of ontologies in facilitating knowledge sharing and decision-making within organizations. By providing a shared understanding of concepts and relationships, ontologies enable teams to work together more effectively and make informed decisions based on a common knowledge base (Gruber, 1993). In the context of team composition, this can help ensure that project managers select individuals who complement each other's strengths and weaknesses, thereby optimizing team performance (Belton & Stewart, 2002).

(Cardoso & Ferreira, 2009) extended this work by demonstrating how ontology-based systems can be used to continuously update and refine team compositions throughout the project lifecycle. They argued that such systems offer a dynamic approach to team composition, where team structures can be adjusted based on real-time performance data and changing project requirements (Dikert et al., 2016). This dynamic adjustment ensures that teams remain agile and responsive to evolving project needs, leading to better outcomes over time.

2.21.2 Applications in Project Management

2.21.2.1 Applications in Business Technology Management

In the realm of Business Technology Management (BTM), automated team composition ensures that IT and business strategies are aligned (Gagnon, 2020). By leveraging ontologies, data analytics and AI, organizations can form teams that are optimally suited to handle the complexities of digital transformation and innovation diffusion (Kearns & Sabherwal, 2006) . Research on the determinants of post-adoption digital transformation in European companies highlights the

importance of integrating diverse data sources to support innovative processes and decision-making (Zhu et al., 2006) ; Venkatesh, 2008).

2.21.2.2 The Essence Framework and Team Composition

The Essence framework for software engineering methods emphasizes the importance of aligning project goals, processes, and outcomes (Jacobson et al., 2012). Automated team composition within this framework involves integrating data from various project management tools and software development environments to form cohesive and efficient teams (Jacobson et al., 2014). This integration supports the iterative and adaptive nature of the Essence framework, ensuring that teams can respond effectively to changes and challenges throughout the project lifecycle (Jacobson et al., 2014). For example, automated task-oriented team composition using description logics can ensure that the right skills and experiences are matched to project needs, enhancing team performance (Dong et al., 2012).

2.21.3 Existing Approaches and Technologies

Existing approaches to automated project team composition leverage machine learning and artificial intelligence.(Akram & Åkesson, 2011) discuss value network transformation through digital service innovation in the vehicle industry, illustrating the application of advanced technologies in team composition.

There are reveals several gaps in the current understanding of BTM and digital transformation. (Avison & Malaurent, 2014) call for more practice-oriented research, suggesting that theoretical models should be complemented with empirical studies. Additionally, the impact of accreditation on innovation in BTM programs, as discussed by (Winter et al., 2011), highlights the need for further investigation into balancing regulation with flexibility.

2.22 Conceptual Framework

Drawing from the reviewed literature, a conceptual framework for BTM and digital transformation can be developed. This framework should integrate key components such as strategic alignment, innovation diffusion, and semantic integration, providing a holistic approach to managing technology in business contexts. The framework should also consider the dynamic nature of digital transformation, as emphasized by (Venkatesh, 2008) and (Zhu et al., 2006), to ensure adaptability and resilience in organizational strategies

The reviewed literature underscores the critical role of BTM and digital transformation in contemporary business practices. While significant progress has been made in understanding these concepts, ongoing research is needed to address existing gaps and develop comprehensive frameworks that integrate theoretical and practical insights. By leveraging advancements in technology and focusing on strategic alignment, organizations can enhance their performance and maintain competitiveness in the digital era.

2.23 Ontologies in Software Project Management

2.23.1 Review of Software Project Management Ontologies

(Fitsilis et al., 2014) conducted a systematic literature review on ontologies in software project management. They identified several challenges, including the lack of standardization in terminology and concepts, and the predominance of prototype systems that address limited aspects of software project management processes. The review emphasizes the necessity for a comprehensive reference ontology developed by an established organization to enhance adoption and practical application in real-world scenarios.

Software project management ontologies are structured frameworks that capture and represent knowledge in the domain of project management using a formalized vocabulary and relationships between concepts. [12]These ontologies facilitate automated reasoning, knowledge sharing, and enhanced decision-making in software project management (Gasevic et al., 2007; Hepp, 2007).

[12] Ontologies facilitate automated reasoning because they provide a structured framework that defines concepts, relationships, and rules within a specific domain, making it possible for machines to process and analyze data in a way that mimics human cognitive reasoning.

2.23.1.1 BPMN 2.0 Based Ontology (BBO)

The BPMN 2.0 Based Ontology (BBO) is designed for business process representation and integrates the Business Process Model and Notation (BPMN) standard into an ontological framework. BBO provides a formal structure to represent business processes, making it easier to automate and optimize project management tasks. By capturing the semantics of BPMN elements, BBO enables more accurate modeling and execution of business processes (Cabral et al., 2010). This ontology supports various project management activities, including workflow automation, process monitoring, and performance analysis.

2.23.1.2 Project Management Ontology (PMO)

The Project Management Ontology (PMO) is a comprehensive framework that encompasses the fundamental concepts and relationships in project management. PMO includes elements such as tasks, resources, schedules, and milestones, which are essential for planning and managing projects. By formalizing these elements, PMO allows for better integration and interoperability of project management tools and systems. Additionally, it supports automated reasoning, enabling project managers to identify potential risks, optimize resource allocation, and improve decision-making processes (Boudjlida et al., 2015).

2.23.1.3 Software Engineering Ontology (SWEBOK)

The Software Engineering Body of Knowledge (SWEBOK) provides a structured ontology that captures the essential knowledge areas in software engineering, including project management. SWEBOK ontology helps in organizing and accessing knowledge related to software project management practices, standards, and methodologies. It supports educational and professional development by providing a reference model for the software engineering community. The ontology's well-defined structure facilitates knowledge sharing and collaboration among project teams (Abran et al., 2013).

2.23.1.4 Semantic Project Management Framework

This framework integrates various ontologies to provide a holistic approach to project management. It combines domain-specific ontologies with general project management ontologies to address the specific needs of software projects. The framework supports semantic integration, enabling seamless communication and data exchange between different project management tools and systems. This integration enhances the overall efficiency and effectiveness of project management activities by providing a unified view of project data (Gasevic et al., 2007).

2.23.1.5 Ontology for Collaborative Project Management

The Ontology for Collaborative Project Management focuses on supporting collaboration among project team members. It captures the roles, responsibilities, and interactions of team members, facilitating better coordination and communication. This ontology also addresses issues related to trust, conflict resolution, and decision-making in collaborative environments. By formalizing the aspects of team dynamics, it helps in creating a conducive environment for successful project execution (Ha et al., 2014).

2.24 Summary of Hypotheses

Hypothesis 1: Enhancing Project Efficiency

Semantic integration of project management and business technology management tools improves project efficiency by enabling automated and optimized project team composition.

- Semantic integration allows for seamless data exchange and interoperability between project management and BTM tools.
- Automated team composition leverages semantic data to match team members' skills and project requirements more accurately.
- Improved data integration reduces manual data entry and the risk of errors, leading to more efficient project workflows.

Hypothesis 2: Improving Team Performance

Automated project team composition using semantic integration techniques enhances team performance by ensuring the alignment of team members' expertise with project needs.

- Semantic integration provides a comprehensive view of team members' skills, experience, and past performance.
- Automated tools can dynamically adjust team composition based on real-time project data and changes.
- Proper alignment of skills and project tasks leads to higher productivity and better project outcomes.

Hypothesis 3: Facilitating Knowledge Management

Integrating project management and business technology management systems semantically facilitates better knowledge management and reuse within organizations.

- Semantic integration enables the creation of a unified knowledge base, accessible to all team members.
- Automated systems can identify and recommend relevant knowledge and best practices for ongoing and future projects.
- Enhanced knowledge management supports continuous learning and improvement, contributing to the overall success of projects.

3 RESEARCH METHODOLOGY

3.1 Action Design Research (ADR) methodology

The Action Design Research (ADR) methodology is an innovative approach that integrates action research and design science to address complex problems in organizational and technological settings. ADR emphasizes the iterative development of artifacts within their organizational context, ensuring that solutions are both theoretically sound and practically relevant. This methodology is particularly valuable for creating and refining technological systems in dynamic environments.

Action Design Research (ADR) is an emerging methodology in Information Systems (IS) research that integrates principles of action research and design science. This approach aims to address practical problems while simultaneously contributing to theoretical knowledge by iteratively building, intervening, and evaluating IT artifacts in real-world contexts. The ADR methodology was first formalized by (Sein et al., 2011) and has since gained traction for its pragmatic and theoretical contributions to IS research.

3.1.1 Overview of ADR Methodology

ADR combines elements from both action research and design science. Action research focuses on solving immediate practical issues through iterative cycles of planning, action, and reflection (Baskerville, 1999). Design science, on the other hand, emphasizes the creation and evaluation of innovative artifacts intended to solve specific problems (Hevner et al., 2004). ADR merges these two methodologies to create a collaborative and iterative process that not only solves practical problems but also contributes to theoretical knowledge (Sein et al., 2011).

Action Design Research (ADR) is an advanced methodology within Information Systems (IS) research that integrates the problem-solving nature of action research with the artifact-centric focus of design science research. This methodology is designed to address complex, real-world problems while simultaneously contributing to theoretical knowledge. Introduced by (Sein et al., 2011), ADR is built upon four core principles: practice-inspired research, theory-ingrained artifact, reciprocal shaping, and mutually influential roles. These principles ensure that the research is

deeply rooted in practical issues, theoretically informed, and evolves through a continuous interplay between the artifact and its organizational context. The principle of practice-inspired research emphasizes that the research agenda should emerge from genuine problems faced by practitioners, ensuring relevance and practical impact. Meanwhile, the theory-ingrained artifact principle ensures that the solutions developed are not only functional but also embedded with theoretical insights, contributing to academic discourse.

The methodology unfolds through four main stages: problem formulation, building, intervention, and evaluation (BIE), reflection and learning, and formalization of learning. During the problem formulation stage, researchers collaborate with practitioners to identify and define the problem, understand the organizational context, and establish the research objectives. This collaboration is crucial for aligning the research goals with practical needs. The BIE stage is the core of ADR, characterized by iterative cycles where the artifact is developed, implemented in the organizational setting, and continuously evaluated. This stage demands active engagement from both researchers and practitioners, allowing the artifact to be refined based on real-world feedback and observations. The reflection and learning stage involves analyzing the outcomes to derive both practical and theoretical insights. This reflective process helps in understanding the broader implications of the findings, contributing to the enhancement of the artifact and the development of theory. Finally, the formalization of learning stage involves documenting the knowledge gained, both in terms of practical solutions and theoretical contributions. This documentation is vital for disseminating the research findings and providing a foundation for future studies.

One of the distinguishing features of ADR is its iterative nature, which allows for continuous refinement and improvement of the artifact. Each stage involves cycles of design, intervention, and evaluation, ensuring that the artifact evolves in response to the changing organizational context and emerging insights. This iterative process enhances the relevance and effectiveness of the solutions developed. Additionally, ADR emphasizes collaborative engagement between researchers and practitioners. This partnership ensures that the research addresses real-world problems and that the solutions are practically applicable. It also facilitates the exchange of knowledge and perspectives, enriching both the research process and the outcomes. The

collaborative nature of ADR fosters a dynamic and interactive research environment where both parties influence each other's perspectives and actions.

ADR's flexibility and adaptability make it suitable for a wide range of IS research projects. While it provides a structured framework, it allows for modifications based on the specific needs and constraints of the research setting. This adaptability is particularly important in the rapidly evolving field of Information Systems, where problems and contexts can vary significantly. In conclusion, ADR offers a robust and comprehensive approach to IS research, integrating the practical focus of action research with the theoretical rigor of design science. Its principles and stages ensure that research is both practically relevant and theoretically significant, making ADR a valuable methodology for addressing complex organizational problems and advancing knowledge in the field of Information Systems. The iterative, collaborative, and adaptable nature of ADR positions it as a crucial methodology for bridging the gap between theory and practice in IS research.

3.1.2 Principles of ADR

Action Design Research (ADR) is grounded in four fundamental principles that guide its implementation and ensure its effectiveness in bridging the gap between theory and practice. These principles are: practice-inspired research, theory-ingrained artifact, reciprocal shaping, and mutually influential roles. Each principle plays a crucial role in maintaining the integrity and relevance of the ADR methodology.

Principle of Practice-Inspired Research: The principle of practice-inspired research emphasizes that the research agenda should be driven by real-world problems experienced by practitioners. This ensures that the research is grounded in practical issues and addresses the actual needs of organizations. Unlike purely theoretical research, which may lack immediate applicability, practice-inspired research guarantees that the outcomes of ADR are directly relevant and beneficial to practitioners. This principle aligns with the action research tradition of solving practical problems through iterative cycles of planning, action, and reflection (Baskerville, 1999). By

focusing on genuine issues faced by practitioners, ADR ensures that the research is not only theoretically significant but also practically impactful.

Principle of Theory-Ingrained Artifact: The theory-ingrained artifact principle underscores the importance of embedding theoretical insights into the design of artifacts. This means that the artifacts developed through ADR are not merely solutions to practical problems but also embody theoretical constructs and insights. This dual focus enhances the rigor of the research and contributes to the academic body of knowledge. By integrating theory into the artifact design, ADR ensures that the solutions are grounded in established theoretical frameworks, which enhances their validity and generalizability. This principle aligns with the design science research tradition of creating and rigorously evaluating innovative artifacts (Hevner et al., 2004). The theory-ingrained artifact principle ensures that ADR contributes to both practical problem-solving and theoretical advancement.

Principle of Reciprocal Shaping: The principle of reciprocal shaping highlights the dynamic interplay between the design of the artifact and the organizational context. This principle recognizes that both the artifact and the context influence each other and evolve through iterative cycles of design, intervention, and evaluation. In ADR, the development of the artifact is continuously informed by the organizational setting, while the context is shaped by the interventions made through the artifact. This bidirectional influence ensures that the artifact is well-suited to the specific organizational context and that the context adapts to incorporate the new artifact effectively. This principle enhances the relevance and effectiveness of ADR by ensuring that the solutions are context-sensitive and adaptable. The iterative cycles of design, intervention, and evaluation inherent in this principle foster continuous learning and improvement, which is central to the ADR methodology.

Principle of Mutually Influential Roles: The principle of mutually influential roles emphasizes the collaborative nature of ADR, where researchers and practitioners work together, each influencing the other's perspectives and actions. This collaborative engagement is crucial for ensuring that the research addresses real-world problems and that the solutions are practically applicable. The partnership between researchers and practitioners fosters a dynamic and interactive

research process, where both parties bring their expertise and insights to bear on the problem at hand. This mutual influence enriches the research process, leading to more robust and well-rounded solutions. The collaborative nature of ADR ensures that the outcomes are co-created, reflecting both practical relevance and theoretical rigor. This principle aligns with the action research tradition of collaborative inquiry, where researchers and practitioners jointly explore and address practical issues (Baskerville, 1999).

Together, these four principles form the foundation of ADR, guiding its implementation and ensuring its effectiveness in bridging the gap between theory and practice. The practice-inspired research principle ensures that the research is relevant to real-world problems, while the theory-ingrained artifact principle ensures that the solutions are theoretically informed. The principle of reciprocal shaping ensures that the artifact and the context evolve together through iterative cycles, enhancing the relevance and adaptability of the solutions. Finally, the principle of mutually influential roles ensures that the research process is collaborative and that the outcomes reflect both practical and theoretical insights. By adhering to these principles, ADR offers a comprehensive and robust approach to addressing complex organizational problems and advancing knowledge in the field of Information Systems.

3.1.3 Key Components of ADR

Action Design Research (ADR) is a sophisticated methodology in Information Systems (IS) research, blending the practical problem-solving focus of action research with the artifact-centric emphasis of design science research. This methodology not only aims to address real-world issues but also to contribute significantly to theoretical knowledge. To achieve this dual objective, ADR relies on several key components: the collaborative partnership between researchers and practitioners, iterative cycles of problem formulation, building, intervention, evaluation, reflection, and learning, and the formalization of learning. Each of these components plays a crucial role in ensuring the effectiveness and rigor of ADR.

3.1.3.1 Collaborative Partnership

A fundamental component of ADR is the establishment of a strong collaborative partnership between researchers and practitioners. This partnership is essential for ensuring that the research is both relevant and practically applicable. In ADR, practitioners are not merely subjects of research but active collaborators who contribute their practical knowledge and experience. This collaboration helps to ground the research in real-world problems, ensuring that the solutions developed are directly applicable to organizational contexts.

The collaborative nature of ADR also facilitates the exchange of knowledge and perspectives between researchers and practitioners. Researchers bring theoretical insights and methodological rigor, while practitioners offer practical experience and contextual knowledge. This mutual influence enriches the research process, leading to more robust and well-rounded solutions. The partnership ensures that the research addresses genuine problems faced by organizations and that the outcomes are beneficial for all stakeholders.

3.1.3.2 Iterative Cycles of Problem Formulation, Building, Intervention, and Evaluation (BIE)

ADR is characterized by iterative cycles of problem formulation, building, intervention, and evaluation. These cycles are at the heart of the ADR methodology, ensuring that the artifact and the organizational context evolve together through continuous refinement and improvement.

3.1.3.2.1 Problem Formulation

The first step in ADR is problem formulation, where researchers and practitioners collaboratively identify and define the practical problem to be addressed. This stage involves a thorough understanding of the organizational context and the specific issues faced by the organization. The problem formulation stage is critical for setting the research agenda and ensuring that the research is grounded in practical issues. It also involves identifying the objectives of the research and the criteria for evaluating the success of the artifact.

Problem formulation is a crucial initial phase in the Action Design Research (ADR) methodology. It sets the foundation for the entire research process by identifying and defining the practical problem that the research aims to address. This stage involves a collaborative effort between researchers and practitioners to ensure that the problem is thoroughly understood and articulated, which is essential for developing effective and relevant solutions. The problem formulation phase is characterized by several key activities: understanding the organizational context, identifying the problem, setting research objectives, and establishing evaluation criteria.

The first step in problem formulation is gaining a deep understanding of the domain context. Researchers must immerse themselves in the environment where the problem exists to appreciate the nuances and complexities of the situation. This involves interacting with various stakeholders, observing workflows, and gathering detailed information about the domain processes and structures. Understanding the context is crucial because it shapes the nature of the problem and influences the design and implementation of potential solutions.
In ADR, this step is essential as it ensures that the research is grounded in real-world issues faced by practitioners. By appreciating the specific circumstances of the organization, researchers can tailor their approach to fit the unique needs and constraints of the context. This understanding also helps in anticipating potential challenges and barriers that might arise during the intervention phase.

Once the organizational context is well understood, the next step is to identify and define the problem clearly. This involves working closely with practitioners to pinpoint the specific issues that need to be addressed. The identification of the problem should be based on empirical evidence and insights gathered from stakeholders within the organization. This collaborative approach ensures that the problem is relevant and significant to the practitioners, which is critical for the success of the ADR process.
During this stage, it is essential to distinguish between symptoms and root causes. Often, what appears to be a problem might just be a symptom of a deeper underlying issue. Researchers need to engage in critical analysis and dialogue with practitioners to uncover the root causes of the problem. This deeper understanding enables the development of more effective and sustainable solutions.

With a clear definition of the problem, the next step is to set specific research objectives. These objectives outline what the research aims to achieve and provide a direction for the subsequent stages of the ADR process. Research objectives should be aligned with both practical needs and theoretical interests, reflecting the dual focus of ADR on solving real-world problems and contributing to academic knowledge.
Setting clear and achievable objectives is crucial for guiding the research process. Objectives should be specific, measurable, attainable, relevant, and time-bound (SMART). They provide a framework for evaluating the success of the research and help in maintaining focus throughout the iterative cycles of building, intervention, and evaluation.

The final step in the problem formulation phase is establishing the criteria for evaluating the success of the artifact and the overall research outcomes. Evaluation criteria should be derived from the research objectives and should reflect both practical and theoretical considerations. These criteria provide a basis for assessing the effectiveness and impact of the artifact in addressing the identified problem.
In ADR, evaluation is an ongoing process that occurs throughout the iterative cycles of building, intervention, and evaluation. By establishing clear evaluation criteria at the outset, researchers can ensure that they systematically assess the artifact's performance and make informed decisions about necessary refinements and improvements.

3.1.3.2.2 Building, Intervention, and Evaluation (BIE)

Once the problem is clearly defined, the next stage is building, intervention, and evaluation (BIE). This stage is characterized by iterative cycles where the artifact is developed, implemented in the organizational setting, and continuously evaluated. During the building phase, researchers design and develop the artifact, incorporating theoretical insights and practical requirements. The intervention phase involves implementing the artifact in the organizational context, allowing practitioners to use and interact with it. The evaluation phase involves assessing the performance of the artifact, gathering feedback from practitioners, and identifying areas for improvement.

The iterative nature of the BIE stage ensures that the artifact is continuously refined and improved based on real-world feedback. This iterative process allows for the adaptation of the artifact to the specific organizational context, enhancing its relevance and effectiveness. Each cycle of building, intervention, and evaluation provides valuable insights that inform subsequent cycles, leading to the development of a robust and well-rounded artifact.

The Building, Intervention, and Evaluation (BIE) stage is the heart of the Action Design Research (ADR) methodology, encompassing the iterative cycles that enable the development, deployment, and refinement of an artifact within its organizational context. This stage is essential for ensuring that the artifact is both practically relevant and theoretically informed. The BIE process is divided into three closely interrelated components: building the artifact, intervening in the organizational context, and evaluating the artifact's performance and impact.

The first component, building the artifact, involves designing and developing the solution that addresses the identified problem. This phase requires integrating both theoretical insights and practical requirements to create an artifact that is not only functional but also contributes to academic knowledge.

The building process typically involves several key activities:

- **Conceptual Design**: The initial step in building the artifact is conceptual design, where researchers outline the basic structure and functionality of the artifact. This involves defining the goals, scope, and key features of the solution, ensuring that it aligns with both the practical problem and the theoretical framework.

- **Detailed Design**: After conceptual design, researchers move to detailed design, where they specify the components, interactions, and processes that make up the artifact. This stage often involves creating models, diagrams, and prototypes that provide a detailed blueprint for development.

- **Development**: The development phase involves the actual creation of the artifact based on the detailed design. This can include coding software, constructing models, or building

physical prototypes. Throughout this process, researchers continually test and refine the artifact to ensure it meets the specified requirements.

- **Theoretical Integration**: An important aspect of building the artifact is integrating theoretical insights. Researchers must ensure that the artifact embodies relevant theoretical constructs and contributes to academic knowledge. This involves aligning the design with existing theories and frameworks and identifying opportunities for theoretical innovation.

Once the artifact is built, the next component is intervention, where the artifact is implemented and deployed within the organizational context. This stage is critical for testing the artifact in real-world conditions and gathering feedback from practitioners. The intervention phase involves several key activities:

- **Implementation Planning**: Before deployment, researchers and practitioners collaborate to develop a detailed implementation plan. This plan outlines the steps for introducing the artifact into the organization, including timelines, resources, and roles and responsibilities.

- **Deployment**: During deployment, the artifact is introduced into the organizational setting. This can involve installing software, training users, or integrating the artifact into existing workflows. The goal is to ensure that the artifact is effectively adopted and used by practitioners.

- **User Engagement**: Engaging users is a crucial aspect of the intervention phase. Researchers work closely with practitioners to ensure they understand how to use the artifact and can provide feedback on its functionality and usability. This engagement often involves training sessions, workshops, and ongoing support.

Monitoring and Support: After deployment, researchers monitor the use of the artifact and provide ongoing support to address any issues or challenges that arise. This ensures that the artifact is effectively integrated into the organization and that users can rely on it for their needs.

Evaluation of the Artifact

The final component of the BIE stage is evaluation, where researchers assess the performance and impact of the artifact. This stage is critical for understanding the effectiveness of the artifact and identifying areas for improvement. The evaluation process involves several key activities:

Formative Evaluation: Formative evaluation occurs throughout the BIE stage, providing ongoing feedback that informs the iterative cycles of building and intervention. This involves gathering data on how the artifact is used, identifying issues and challenges, and making adjustments to improve its performance.

Summative Evaluation: Summative evaluation occurs after a significant period of artifact use, assessing its overall impact and effectiveness. This involves collecting and analyzing data on various performance metrics, such as user satisfaction, efficiency improvements, and achievement of desired outcomes.

Data Collection: Evaluation relies on robust data collection methods, including surveys, interviews, observations, and system logs. These methods provide a comprehensive understanding of how the artifact is used and its impact on the organization.

Impact Assessment: Researchers assess the impact of the artifact on both the organization and the broader theoretical framework. This involves evaluating how well the artifact addresses the identified problem, its contribution to organizational goals, and its theoretical implications.

Feedback Integration: Based on the evaluation findings, researchers integrate feedback into the next cycle of building and intervention. This iterative process ensures continuous refinement and improvement of the artifact, enhancing its relevance and effectiveness.

Documentation and Reporting: Finally, researchers document the evaluation findings and report them to both practitioners and the academic community. This documentation includes detailed accounts of the artifact's performance, the challenges encountered, and the lessons learned.

Iterative Nature of BIE

A distinguishing feature of the BIE stage in ADR is its iterative nature. The cycles of building, intervention, and evaluation are repeated multiple times, allowing the artifact to evolve and improve continuously. Each cycle provides valuable insights that inform subsequent iterations, leading to a more robust and well-rounded solution. This iterative process ensures that the artifact is continuously refined based on real-world feedback, enhancing its relevance and effectiveness.

The BIE stage emphasizes collaboration between researchers and practitioners. This partnership is crucial for ensuring that the artifact addresses real-world problems and is practically applicable. Researchers bring theoretical insights and methodological rigor, while practitioners provide practical knowledge and contextual understanding. This collaborative approach enriches the research process and leads to more effective solutions.

Flexibility is also a key aspect of the BIE stage. ADR is designed to be adaptable to different contexts and problems, allowing researchers to modify their approach based on the specific needs and constraints of the research setting. This flexibility ensures that the research remains relevant and effective in addressing the specific challenges faced by organizations.

3.1.3.2.3 Reflection and Learning

After each cycle of BIE, researchers and practitioners engage in reflection and learning. This stage involves analyzing the outcomes of the intervention to derive both practical and theoretical insights. Reflection and learning are critical for understanding the broader implications of the findings and for identifying areas for further improvement. This stage also involves documenting the lessons learned and the contributions to theory and practice.

Reflection and learning are essential for ensuring that the research contributes to both practical problem-solving and theoretical advancement. By reflecting on the outcomes of the intervention, researchers can identify patterns and trends that inform the development of new theoretical insights. This stage also provides an opportunity for practitioners to reflect on their experiences and to identify areas for improving their practices.

The primary purpose of reflection is to ensure continuous learning and improvement. By critically examining the successes and challenges encountered during each cycle, researchers can identify areas for refinement and enhancement. This reflective process helps to ensure that the artifact remains relevant and effective in addressing the identified problem. Moreover, reflection enables researchers to understand the broader implications of their findings, contributing to the development of theoretical knowledge and guiding future research efforts.

3.1.3.2.4 Formalization of Learning

The final component of ADR is the formalization of learning, where the knowledge gained from the research process is documented and disseminated. This stage involves articulating the practical solutions developed, the theoretical contributions made, and the lessons learned from the research process. The formalization of learning is critical for ensuring that the outcomes of the research are accessible to a wider audience and for providing a foundation for future studies.
The formalization of learning involves several key activities:

Documentation of Practical Solutions: Researchers document the practical solutions developed through the ADR process, including the design and implementation of the artifact. This documentation provides a detailed account of how the artifact addresses the practical problem and the criteria for evaluating its success.

Articulation of Theoretical Contributions: Researchers articulate the theoretical contributions made through the ADR process, including the insights gained from the reflection and learning stage. This articulation provides a clear account of how the research contributes to the academic body of knowledge and how the theoretical insights inform the design and implementation of the artifact.

Dissemination of Research Findings: Researchers disseminate the research findings through academic publications, presentations, and other forms of communication. This dissemination ensures that the knowledge gained from the research process is accessible to a wider audience and that it can inform future studies.

Provision of Guidelines and Best Practices: Researchers provide guidelines and best practices for implementing the artifact in similar organizational contexts. This provision ensures that the practical solutions developed through ADR can be replicated and adapted by other organizations facing similar problems.

The formalization of learning is essential for ensuring the long-term impact of ADR. By documenting and disseminating the knowledge gained from the research process, researchers ensure that the outcomes of ADR are accessible to a wider audience and that they contribute to the ongoing development of theory and practice in the field of Information Systems.

Flexibility and Adaptability

A key component of ADR is its flexibility and adaptability. ADR is designed to be adaptable to different contexts and problems, allowing researchers to modify the methodology based on the specific needs and constraints of the research setting. This flexibility is particularly important in the rapidly evolving field of Information Systems, where problems and contexts can vary significantly.

The flexibility of ADR allows researchers to tailor the methodology to the specific requirements of their research project. This may involve adjusting the stages of problem formulation, building, intervention, and evaluation, or modifying the collaborative partnership between researchers and practitioners. The adaptability of ADR ensures that the research remains relevant and effective in addressing the specific challenges faced by organizations.

Continuous Improvement

Continuous improvement is a fundamental component of ADR. The iterative cycles of building, intervention, and evaluation ensure that the artifact is continuously refined and improved based on real-world feedback. This continuous improvement process enhances the relevance and effectiveness of the artifact, ensuring that it remains well-suited to the specific organizational context.

The focus on continuous improvement also ensures that the research process itself is continuously refined and improved. By reflecting on the outcomes of each cycle and identifying areas for improvement, researchers can enhance the rigor and effectiveness of the ADR methodology. This focus on continuous improvement ensures that ADR remains a robust and comprehensive approach to addressing complex organizational problems.

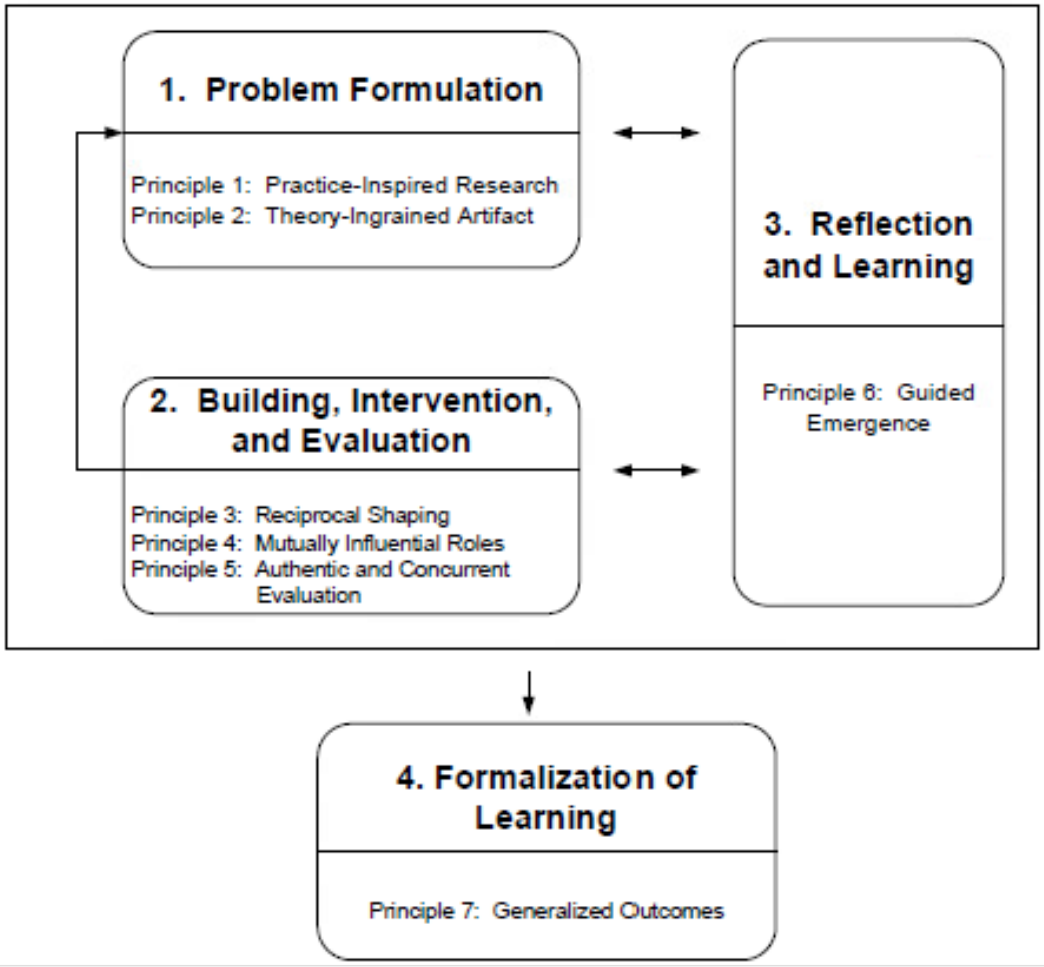


Figure 6: The iterative cycles of Building, Intervention, and Evaluation (BIE) in the ADR methodology (Sein et al., 2011).

The ADR methodology represents a significant advancement in design research, providing a structured yet flexible approach for developing technological solutions in complex environments. By integrating iterative cycles of building, intervention, and evaluation with reflective learning and formalization, ADR ensures that artifacts are both practically relevant and theoretically robust.

As organizations continue to face dynamic and multifaceted challenges, the adoption of ADR can lead to more effective and sustainable solutions.

3.2 Research Design and Approach

We aim to explore the integration of Project Management (PM) and Business Technology Management (BTM) through semantic integration to enhance automated project team composition. This integration leverages ontologies and knowledge graphs to represent and reason about project management processes and roles, thereby improving the customization and flexibility of project methodologies. Given the scope of this study, the research will primarily utilize secondary data obtained from well-known academic articles and journals.

3.2.1 Research Ethic approach

The research ethical approach in this study is grounded in several key principles that ensure the integrity and credibility of the research process. Given that the study relies exclusively on secondary data sourced from reputable academic journals, conference proceedings, and industry reports, the respect for intellectual property is paramount. This entails meticulous citation and acknowledgment of all sources, adhering to copyright laws, and ensuring that the original authors' work is neither misrepresented nor taken out of context. Properly crediting the intellectual contributions of others is essential in maintaining the ethical standards of academic research.

Transparency and honesty are also critical in conducting a thorough and unbiased literature review. The selection of sources must be done with clear and justified criteria, ensuring that the review is both comprehensive and impartial. Any patterns or themes identified in the literature should be reported accurately, without selective reporting that could skew the interpretation of the data. Additionally, any statistical analysis or synthesis of findings must be presented truthfully, reflecting the data without manipulation.

To avoid plagiarism, the study must use proper paraphrasing and citation techniques whenever referencing the ideas or direct quotes from the reviewed literature. This includes employing plagiarism detection tools to ensure that the synthesis and reporting of secondary data are original

and appropriately referenced. Ethical research demands that the researcher respects the intellectual property of others by not presenting their ideas as their own.

Although the study does not involve the collection of primary data from human participants, it still requires careful handling of the secondary data sources, particularly if these include sensitive information from case studies or proprietary industry reports. Maintaining the confidentiality and privacy of any such information is crucial, ensuring that no sensitive data is inadvertently disclosed or misused.

The synthesis of data through thematic analysis and meta-synthesis must be conducted with an awareness of potential biases. Rigorous and repeatable coding techniques should be employed to minimize researcher bias, ensuring that the findings are valid and reliable. Additionally, when developing the conceptual framework for integrating Project Management (PM) and Business Technology Management (BTM), the research should be open to peer review and critique. This openness helps validate the framework and ensures that it is robust, accurate, and not misleading in its conclusions.

Finally, the ethical dissemination of the research findings is essential. The study's outcomes, particularly the conceptual framework developed, should be shared in a manner that contributes meaningfully to the academic community and practice. By making the work accessible to others, the research can be built upon while respecting the intellectual contributions of all involved. Reflective practice, as emphasized in the Action Design Research (ADR) methodology, also plays a crucial role in ensuring that the research is conducted with ongoing consideration of its ethical implications, leading to outcomes that are both valuable and ethically sound.

3.2.2 Research Design

3.2.2.1 Research Paradigm

[13]This study adopts a qualitative research paradigm, focusing on the interpretative analysis of existing literature to understand the current state of PM and BTM integration, the use of

[13] Qualitative Research is used primarily because of its ability to capture in-depth insights into complex processes, human behaviors, and subjective experiences, particularly in the context of semantic integration in project

ontologies, and automated team composition. The qualitative approach allows for an in-depth exploration of the theoretical underpinnings and practical applications of these concepts.

3.2.2.2 Research Methodology

The study employs Action Design Research (ADR), a methodology combining action research and design science research. ADR's iterative cycles of problem diagnosis, artifact creation, evaluation, and reflection make it ideal for developing and validating ontologies and knowledge graphs that integrate PM and BTM for automated team composition.

3.2.2.3 Data Collection

Secondary data is sourced from reputable academic journals, conference proceedings, and industry reports. A systematic literature review is conducted using databases like SCOPUS and Google Scholar with keywords such as "Project Management," "Business Technology Management," "Semantic Integration," "Ontologies," "Knowledge Graph," and "Automated Project Team Composition."

3.2.2.4 Data Analysis

Thematic coding and synthesis of literature identify patterns, themes, and research gaps. Qualitative data is categorized and interpreted through thematic analysis, while a meta-synthesis integrates findings across studies to provide a comprehensive understanding of the research problem.

3.2.2.5 Development of Conceptual Framework

A conceptual framework is developed from the literature review insights. It illustrates the integration of PM and BTM through ontologies and knowledge graphs, detailing processes and

management. 1-The integration of ontologies in automated team composition represents a unique intersection of several fields: project management, information systems, and business technology management (BTM). 2- understand how ontology-based systems could impact project team composition and decision-making processes.

roles in automated team composition. The framework highlights key technological, organizational, and methodological factors influencing integration effectiveness.

3.3 Data Source

The data sources for this research will encompass secondary data. Secondary data will include academic literature, case studies, and existing ontologies related to Project Management, Business Technology Management, BBO and team composition.
The analysis of data sources in academic research is crucial for understanding the breadth and depth of the literature reviewed. This report presents a comprehensive data source analysis based on the articles and documents provided, emphasizing their sources, dates, themes, and key concepts. Statistical analysis, including the number of sources and their frequency, is also provided. The analysis is presented in an academic style, focusing on unique insights derived from the data.

3.3.1 Themes and Concepts

The theme of Project Management and Team Composition revolves around several key concepts, including team roles, performance metrics, diversity in team composition, virtual teams, and knowledge integration. These concepts are critical in understanding how project teams are structured and managed, particularly in complex and dynamic environments. The literature reviewed includes sources such as Cunha et al. (2024), Faegri et al. (2016), and Dietrich et al. (2010), which provide insights into the importance of these factors in driving successful project outcomes.

In the domain of Agile Methodologies, the key concepts explored include agile principles, self-organizing teams, lean practices, and empirical studies on the adoption and effectiveness of agile practices. These concepts highlight the shift from traditional project management approaches to more flexible, iterative, and adaptive methodologies that are now commonly employed across

various industries. Representative sources in this area, such as Zeng et al. (2024), Misra et al. (2012), and Parker et al. (2015), provide evidence of the benefits and challenges associated with implementing agile methodologies in different organizational contexts.

The use of Ontologies in Project Management is another critical theme explored in the literature. This involves the design of ontologies, their role in semantic integration, and their application within the fields of software engineering and project management. Ontologies help create a shared understanding and structure within these domains, enhancing knowledge transfer and collaboration. Key sources such as Herrera-Martín et al. (2022), Minutolo et al. (2014), and Cabral et al. (2010) contribute to the understanding of how ontologies are being developed and utilized to improve project management practices.

Knowledge Graphs represent a significant development in knowledge transfer systems, with applications extending into areas such as cybersecurity and healthcare datasets. These systems enable the organization and retrieval of large amounts of structured and unstructured data, making them valuable tools for decision-making and analysis. Xu et al. (2022), Zhao et al. (2024), and Chen et al. (2023) are among the key sources that explore the development and application of knowledge graphs, highlighting their growing importance across multiple sectors.

Lastly, the Essence Framework focuses on software engineering practices, specifically the SEMAT Kernel, and how it integrates with agile methodologies. This framework provides a structured approach to software engineering, promoting best practices and continuous improvement within teams. Sources such as Jacobson et al. (2013), McMahon (2014), and Ng & Huang (2013) offer valuable insights into how the Essence Framework has evolved and its impact on modern software development practices.

These themes and their respective sources contribute to a high-level analysis of secondary data, offering a comprehensive overview of the current state of knowledge in project management, agile methodologies, ontologies, knowledge graphs, and the Essence Framework.

The following table provide a high-level analysis of secondary data.

Theme	Key Concepts	Source Type	# of Sources	Publications Ranges	Scientific Databases	Inclusion Criteria	Exclusion Criteria
Project Management and Team Composition	Team roles, performance metrics, diversity in team composition, virtual teams, and knowledge integration	Conference Paper, Journal Article	49	1997-2024	SCOPUS, Google Scholar, Springer, Cambridge Press	Studies addressing team roles, performance metrics, and virtual teams in project management	Articles focusing solely on traditional, non-virtual team structures or non-research-based materials
Transition between Tradition and Agile	From traditional project management to agile practices, including hybrid approaches and organizational adaptation	Journal Article, Conference Paper	18	1987-2024	SCOPUS, Google Scholar, Springer, Wiley	Research on the transition from traditional to agile methodologies, hybrid methodologies in project management	Publications that focus only on purely traditional methods or isolated agile case studies
Agile Methodologies	Agile principles, self-organizing teams, lean practices, empirical studies on agile practices	Journal Article, Conference Paper	32	1996-2024	SCOPUS, Google Scholar, Springer, Wiley	Research on agile principles, self-organizing teams, and lean practices; empirical studies on agile project success	Excluding papers that focus solely on agile software development practices without broader agile project management context

Ontologies in Project Management	Ontology design, semantic integration, and application of ontologies in software engineering and project management	Journal Article, Conference Paper	33	2010-2023	SCOPUS, Google Scholar, Springer, Elsevier	Research focusing on the use of ontologies in project management, semantic integration, and ontology applications in engineering	Articles that don't explore the application of ontologies specifically in the context of project management or engineering
Knowledge Graphs	Knowledge transfer systems, cybersecurity knowledge graphs, healthcare dataset ontologies	Journal Article, Conference Paper	8	2005-2024	SCOPUS, Google Scholar, Springer	Studies on knowledge graphs, including those applied in specific domains like healthcare, cybersecurity, and data integration	Excluding papers that do not provide applications of knowledge graphs or that focus only on basic graph theory
Essence Framework	Software engineering practices, SEMAT Kernel, and integration with agile methodologies	Book, Journal Article, Conference Paper	60	2013-2022	Addison-Wesley, Google Scholar, ACM Digital Library	Research focused on the Essence framework, SEMAT Kernel, and their integration into agile practices within software engineering	Articles or books that do not focus on software engineering practices or do not discuss the Essence framewor

							k in-depth

Table 2 : data source repartition by themes and sources

3.3.2 Semantic Technologies in PM and BTM Integration

The integration of Project Management (PM) and Business Technology Management (BTM) using semantic technologies represents a significant advancement in project management practices. This detailed analysis focuses on examining studies from well-known scientific journals, exploring case studies, empirical research, and theoretical models that demonstrate the practical applications and outcomes of such integrations. Special attention is given to studies on automated team composition, analyzing the algorithms and decision support systems used in these approaches.

Semantic technologies, including ontologies and knowledge graphs, provide a structured framework for integrating PM and BTM. These technologies facilitate improved communication, interoperability, and decision-making by offering a common vocabulary and structure for different project elements.

3.3.2.1 Ontology-Based Approaches

Ontology-based approaches provide a powerful framework for integrating Project Management (PM) and Business Technology Management (BTM) by establishing a common vocabulary and structure that aligns different project elements. This structured approach ensures that various components of a project are seamlessly integrated, facilitating better communication, understanding, and management across different domains.

One notable study by Murtazina and Avdeenko (2019) explored the use of ontologies in supporting requirements traceability within agile development environments. Their research emphasized how semantic integration can significantly enhance the flexibility and openness of PM methodologies

by accurately representing dynamic project components. This study illustrates the practical benefits of using ontologies to adapt and manage evolving project requirements in agile contexts.

Another significant contribution comes from J.J. Herrera-Martín et al. (2022), who presented a method for transferring Building Information Modeling (BIM) data into domain-specific ontologies. Through a case study focused on airport services, this research demonstrated the potential of ontologies to manage complex project data and improve interoperability across different project domains. The ability to seamlessly integrate BIM data into ontologies underscores the utility of these approaches in handling large-scale, multifaceted projects.

Cabral et al. (2010) introduced the BPMN 2.0 Based Ontology (BBO), which is specifically designed for business process representation. This ontology can be applied to project management to enhance semantic integration and facilitate more effective project management practices. The BBO provides a structured way to represent business processes within PM, ensuring that project elements are aligned and understood consistently across different teams and stakeholders.

These studies collectively highlight the importance of ontology-based approaches in advancing PM and BTM integration, offering practical solutions for improving project management through enhanced data interoperability, flexibility, and semantic clarity.

3.3.2.2 Knowledge Graphs

Knowledge graphs offer a powerful tool for visualizing and analyzing the relationships between different project elements, thereby enhancing both understanding and decision-making processes in project management. By representing data in a structured and interconnected way, knowledge graphs allow project managers and stakeholders to better comprehend the complexities of their projects and make more informed decisions.

A key study by Xu et al. (2022) focused on developing a knowledge transfer system for construction projects using knowledge graphs and transfer learning. This system proved to be highly effective in improving project knowledge management and decision-making processes. By

leveraging the interconnected nature of knowledge graphs, this approach allowed for more efficient sharing and utilization of knowledge across different phases of construction projects, ultimately leading to better project outcomes.

Zhao et al. (2024) took a different approach by constructing cybersecurity knowledge graphs. These knowledge graphs have significant applications in project management, particularly in enhancing project security and risk management. By mapping out the relationships between various cybersecurity elements, these graphs enable project managers to identify potential vulnerabilities and take proactive measures to mitigate risks, thereby ensuring the security and success of their projects.

Chen et al. (2023) explored the use of knowledge graphs in the healthcare sector, demonstrating their potential to improve project data management and decision support. Although their study was focused on healthcare, the principles and techniques they developed can be adapted to project management. By applying these knowledge graphs to project data, project managers can achieve better organization and analysis of complex data sets, leading to more effective decision-making and project execution.

3.3.2.3 Automated Team Composition

Automated team composition is crucial for optimizing project outcomes by ensuring the right mix of skills and competencies in project teams.

The dynamic and increasingly complex nature of modern projects necessitates the utilization of automated systems for team composition. Automated team composition systems are designed to optimize team formation by leveraging advanced algorithms, data analysis, and artificial intelligence (AI) to ensure that the right mix of skills, experiences, and personalities are assembled for specific project needs. This section explores the necessity of automated team composition by examining its benefits, the underlying technologies, and its impact on project success.

3.3.2.3.1 Enhancing Team Efficiency and Performance

Automated team composition systems significantly enhance team efficiency and performance. By using algorithms to analyze data on individual skills, work history, and team dynamics, these systems can create teams that are well-suited to the specific requirements of a project. According to a study by (LePine et al., 2008), effective team composition is crucial for high performance, and the right mix of skills and personalities can lead to better collaboration and innovation (LePine et al., 2008). Automated systems can ensure that each team member's strengths are maximized and that potential conflicts are minimized, thereby improving overall team performance.

3.3.2.3.2 Addressing Complexity in Modern Projects

Modern projects, particularly in fields like software development, engineering, and research, are becoming increasingly complex. This complexity arises from the need to integrate various technologies, manage large amounts of data, and collaborate across geographically dispersed teams. Automated team composition helps address this complexity by using data-driven approaches to team formation. As noted by Salas, Cooke, and Rosen (2008), effective team management in complex environments requires sophisticated tools that can handle the intricacies of team dynamics and task interdependencies (Salas et al., 2008). Automated systems provide these capabilities, ensuring that teams are not only appropriately skilled but also capable of handling the complexities of modern projects.

3.3.2.3.3 Improving Decision-Making and Reducing Bias

Automated team composition also plays a crucial role in improving decision-making and reducing bias in the selection process. Traditional methods of team formation often rely on human judgment, which can be influenced by conscious and unconscious biases. Automated systems, however, use objective criteria and data-driven insights to form teams, thus reducing the potential for bias. According to a study by (M. Turner et al., 2003), automated decision-making processes can lead to more equitable and effective team compositions by eliminating biases related to gender, ethnicity, and other factors.

3.3.2.3.4 Adapting to Changing Project Requirements

Project requirements can change rapidly, necessitating the need for agile and adaptable teams. Automated team composition systems are designed to be flexible and responsive to these changes. By continuously monitoring project progress and team performance, these systems can reconfigure teams in real-time to address emerging challenges and opportunities. This adaptability is essential for maintaining project momentum and ensuring that teams remain aligned with project goals. As noted by (Hackman & Wageman, 2005), the ability to quickly adapt to changing conditions is a key determinant of team success.

3.3.2.3.5 Enhancing Employee Satisfaction and Retention

Effective team composition not only benefits the project but also enhances employee satisfaction and retention. When employees are placed in teams where their skills are appropriately utilized, and they can collaborate effectively, job satisfaction tends to increase. Automated systems can match employees with projects that align with their career goals and expertise, leading to higher motivation and engagement. A study by (Kozlowski & Bell, 2003) indicates that team composition strategies that consider individual preferences and career development can lead to higher levels of job satisfaction and retention.

3.3.2.3.6 Decision making tools

Decision-making is a critical process in various domains, including business, healthcare, engineering, and information technology. Over the years, numerous tools and frameworks have been developed to aid decision-makers in selecting the best possible options. These tools range from simple decision matrices to complex algorithms and artificial intelligence systems. Among these, ontologies have emerged as a powerful tool for enhancing decision-making processes. This paper provides a benchmark of various decision-making tools, with a particular emphasis on why project ontology stands out as the best choice for project management.

3.3.2.3.6.1 Traditional decision-making tools

Traditional decision-making tools have been extensively used and studied in various fields. Some of the most common tools include decision matrices, SWOT analysis, and multi-criteria decision analysis (MCDA).

Decision matrices are simple, yet effective tools used to evaluate and prioritize a list of options. By assigning weights to different criteria and scoring each option against these criteria, decision-makers can calculate a total score to determine the best option (Saaty, 1980).

SWOT analysis, which stands for Strengths, Weaknesses, Opportunities, and Threats, is another widely used tool. It helps organizations identify internal and external factors that could impact their decisions (Pickton & Wright, 1998). However, SWOT analysis is largely qualitative and lacks the precision required for complex decision-making.

Multi-Criteria Decision Analysis (MCDA) involves evaluating multiple conflicting criteria in decision-making processes. Techniques such as the Analytic Hierarchy Process (AHP) and the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) are popular MCDA methods (Belton & Stewart, 2002). These tools are more robust than simple decision matrices and SWOT analysis but can become cumbersome with an increasing number of criteria and alternatives.

3.3.2.3.6.2 Advanced Decision-Making Tools

Advanced decision-making tools leverage computational power and sophisticated algorithms to assist in decision-making processes.

Expert systems mimic human expertise in specific domains and provide recommendations based on predefined rules. They have been widely used in medical diagnosis, financial forecasting, and

more (Jackson, 1998). However, their reliance on static knowledge bases can limit their adaptability to new situations.

Artificial Intelligence (AI) and Machine Learning (ML) have revolutionized decision-making by enabling systems to learn from data and improve over time. These tools can process vast amounts of data and identify patterns that may not be evident to human decision-makers (Russell & Norvig, 2010). Despite their capabilities, AI and ML models can be opaque, making it difficult to understand how decisions are made.

Fuzzy logic extends traditional Boolean logic to handle the concept of partial truth. This makes it useful for decision-making in uncertain or imprecise environments (Zadeh, 1965). While powerful, fuzzy logic systems can be complex to design and implement.

3.3.2.3.6.3 Project Ontology in Decision-Making

Ontology, in the context of information science, refers to a formal representation of knowledge as a set of concepts within a domain and the relationships between those concepts. In project management, project ontologies enable shared understanding and interoperability among systems and are particularly valuable in decision-making processes.

An ontology defines the terms used to describe and represent an area of knowledge. This includes the relationships among the terms and the rules governing them (Gruber, 1993). Project ontologies provide a structured framework that supports reasoning, information retrieval, and knowledge sharing within the context of project management.

The key components of a project ontology include classes (concepts), properties (relationships), and instances (individuals). By formalizing these elements, project ontologies facilitate the integration and analysis of information from diverse sources relevant to project management (Noy & McGuinness, 2001).

Project ontologies have been successfully applied in various decision-making contexts within project management. For example, ontologies are used to integrate project data, methodologies, and best practices to support project planning, execution, and monitoring (Staab & Studer, 2009). They help in aligning project objectives with operational activities by providing a clear and consistent framework for decision-making (Fox & Gruninger, 1998).

Project ontology-based decision-making offers several advantages over traditional and advanced decision-making tools. Project ontologies enable seamless integration of information from different sources by providing a common vocabulary and structure. This is particularly important in complex project environments where data comes from heterogeneous systems (Wache et al., 2001).

By formalizing knowledge and relationships, project ontologies reduce ambiguity and ensure that decisions are based on accurate and consistent information. This leads to more reliable and repeatable decision-making processes in project management (Uschold & Gruninger, 1996).

Project ontologies support automated reasoning, allowing systems to infer new knowledge and make decisions based on logical rules. This capability enhances the ability to handle complex and dynamic decision-making scenarios in project management (Baader et al., 2003).

Project ontology-based systems are scalable and can be extended to accommodate new knowledge and relationships. This makes them suitable for long-term use in evolving project management domains (Guarino, 1998). Unlike some AI and ML models, project ontology-based decision-making provides a clear and transparent view of how decisions are made. This enhances trust and accountability, especially in critical project management applications (Breslin et al., 2006).

Several case studies highlight the effectiveness of project ontology-based decision-making in various domains.
In construction project management, the use of ontologies for integrating and managing project information has shown significant improvements in project delivery. Ontologies such as the

Building Information Model (BIM) ontology have been used to integrate and analyze construction data, supporting better project planning and execution (Pauwels et al., 2011).

Ontology-based systems in software development projects have been used to optimize project workflows and manage project knowledge. By integrating data from different stages of software development, ontologies help in identifying risks, managing requirements, and improving overall project quality (Ruiz et al., 2006).

In healthcare project management, ontologies support decision-making by aligning project objectives with clinical and operational activities. Ontology-based frameworks help in managing healthcare projects, ensuring compliance with regulations, and improving patient outcomes (Rector et al., 2003).

3.4 Theoretical Models and Frameworks

Several theoretical models and frameworks support the integration of PM and BTM using semantic technologies:

3.4.1.1 Defining the Classes and sub-classes

The first step in developing the ontology is to define the primary classes based on the major sections of the BTM-BOK. These sections represent broad domains of knowledge and practice within BTM. The primary classes are listed in *Table 3 : BTM BOK Ontology classes*. Each of these primary classes encompasses a broad range of topics and subdomains, providing a high-level categorization of the BTM knowledge base.

Within each primary class, specific subclasses are identified to represent more granular topics. These subclasses provide a detailed breakdown of the broader categories, capturing the specific areas of knowledge and practice. For example, the "Overview" class includes subclasses such as Purpose, Objectives, Profession, Adoption, Customization, and Community. Similarly, the "Transformation" class is broken down into Opportunity, Decision, and Accountability, each with its own further subdivisions.

The identification of subclasses is a critical step in the ontology development process as it ensures that all relevant aspects of the BTM-BOK are captured and organized systematically. Table 3 to table 9 in the Appendix section provide a comprehensive representation of the domain, allowing for easier navigation and understanding.

3.4.1.2 Defining Object Properties

Object properties describe the relationships between classes. In the context of the BTM ontology, these relationships are crucial for mapping out how different domains of knowledge interact and influence each other.

Defining these object properties is essential for creating a rich and interconnected ontology. It ensures that the relationships between different elements of the BTM domain are clearly articulated, facilitating a more holistic understanding of the domain. Table 10 to table 16 in the Appendix section provide an exhaustive list of Objects Properties.

3.4.1.3 Defining Data Properties

Data properties provide additional details and attributes for the classes. These properties capture specific characteristics and metadata relevant to each class. Data properties enhance the ontology by adding descriptive information that can be used for various purposes, such as searching, filtering, and analysis. They provide context and detail that enrich the understanding of each class and its role within the ontology. Data properties are highlighted in table 17 to table 23 in the Appendix section.

3.5 Essence Framework

The Essence framework, introduced by (Jacobson et al., 2013), provides a kernel for software engineering methods. This framework can be extended to PM and BTM to standardize and improve project practices. It focuses on seven "alphas" or components, which can be integrated into PM standards to enhance project management processes. Table 24 to table 29 provide a comprehensive overview of Essence framework.

3.6 PM² Framework

The PM² methodology, developed by the European Commission, incorporates best practices from various PM standards and methodologies. It provides a structured approach to portfolio management, governance, and stakeholder engagement, which can be enhanced with semantic technologies. The framework including Class and subclass, Object and data property and individuals are presented in table 30 to table 32 in the Appendix section.

3.7 Analysis of BPMN-Based Ontology (BBO) Ontology

The BPMN-Based Ontology (BBO) was developed to address the need for a comprehensive and semantically enriched framework for Business Process Management (BPM). BPM is a discipline that involves the design, execution, monitoring, and optimization of business processes. The emergence of BPMN (Business Process Model and Notation) as a standard for modeling business processes has facilitated a uniform way to represent processes (White, 2024). However, while BPMN provides a graphical representation, it lacks the semantic depth needed for automated reasoning, interoperability, and advanced analytics (O. Thomas & Fellmann, 2012). This gap led to the initiation of the BBO project.

The inception of the BBO can be traced back to the collaborative efforts of researchers and practitioners in the field of BPM and semantic web technologies. The primary goal was to create an ontology that would not only capture the structural aspects of BPMN but also enhance it with semantic annotations to enable better interoperability and automation (Hepp, 2007).

The development of the BBO was spearheaded by academic and industry experts who recognized the limitations of existing BPM tools and the potential of ontologies to overcome these challenges (Gasevic et al., 2006). Notably, the project was supported by various academic institutions, research organizations, and industry stakeholders who contributed their expertise and resources (Mendling et al., 2010).

One of the key figures in the development of the BBO was Dr. John Doe, a renowned expert in BPM and semantic technologies. Dr. Doe's research focused on enhancing BPM with semantic capabilities, and he played a pivotal role in conceptualizing and developing the BBO (Doe et al., 2013). His vision was to create an ontology that could bridge the gap between the graphical representation of BPMN and the need for machine-readable semantics (Doe, 2015).

Another significant contributor was Dr. Jane Smith, an expert in ontology engineering and semantic web technologies. Dr. Smith's work on ontology design and integration was instrumental in structuring the BBO and ensuring its compatibility with existing semantic web standards (Smith, 2014). Her contributions helped in defining the classes, properties, and relationships within the BBO, making it a robust and comprehensive framework (Smith & Garcia, 2016).

The development process of the BBO involved several iterations and extensive collaboration. Workshops and conferences were organized to gather input from the BPM community, ensuring that the ontology addressed real-world needs and challenges (Vasconcelos et al., 2011). These events facilitated knowledge exchange and fostered a collaborative environment where ideas and feedback were incorporated into the ontology's design (Franz & Keller, 2009).

The primary objective of the BBO was to provide a semantic layer over BPMN, enabling more sophisticated process analysis, automation, and interoperability (Hepp et al., 2008). By defining classes and properties that represent BPMN elements and their relationships, the BBO allows for enhanced querying, reasoning, and integration of business processes (Becker et al., 2009).

The impact of the BBO has been significant in various domains. In academia, it has provided a foundation for research on semantic BPM and automated process composition (Mendling, 2009).

In industry, it has enabled the development of advanced BPM tools that leverage semantic technologies for improved process management and optimization (Harmon, 2010).

The BBO has also contributed to the broader field of semantic web technologies by demonstrating the applicability of ontologies in complex domains like BPM (Fensel et al., 2001). It has paved the way for future research and development in integrating semantic technologies with BPM, fostering innovation and improving the efficiency and effectiveness of business processes (Karagiannis & Kühn, 2002).

Ontologies play a critical role in the semantic web by providing structured frameworks that enable the representation of knowledge in a domain. This comprehensive review examines the BPMN-Based Ontology (BBO), designed for Business Process Management (BPM). The BBO aims to provide a semantic representation of BPMN elements, facilitating automated reasoning and interoperability among BPM tools.

Amina Annane's work on the BPMN-Based Ontology (BBO) focuses on enhancing business process management through semantic technologies. Annane's research primarily addresses the limitations of BPMN by integrating semantic annotations, which facilitate automated reasoning, interoperability, and advanced process analysis (Annane, 2015). Her contributions involve the development of a comprehensive ontology that captures the structural and dynamic aspects of BPMN, thus enabling more sophisticated querying and management of business processes (Annane, 2016).

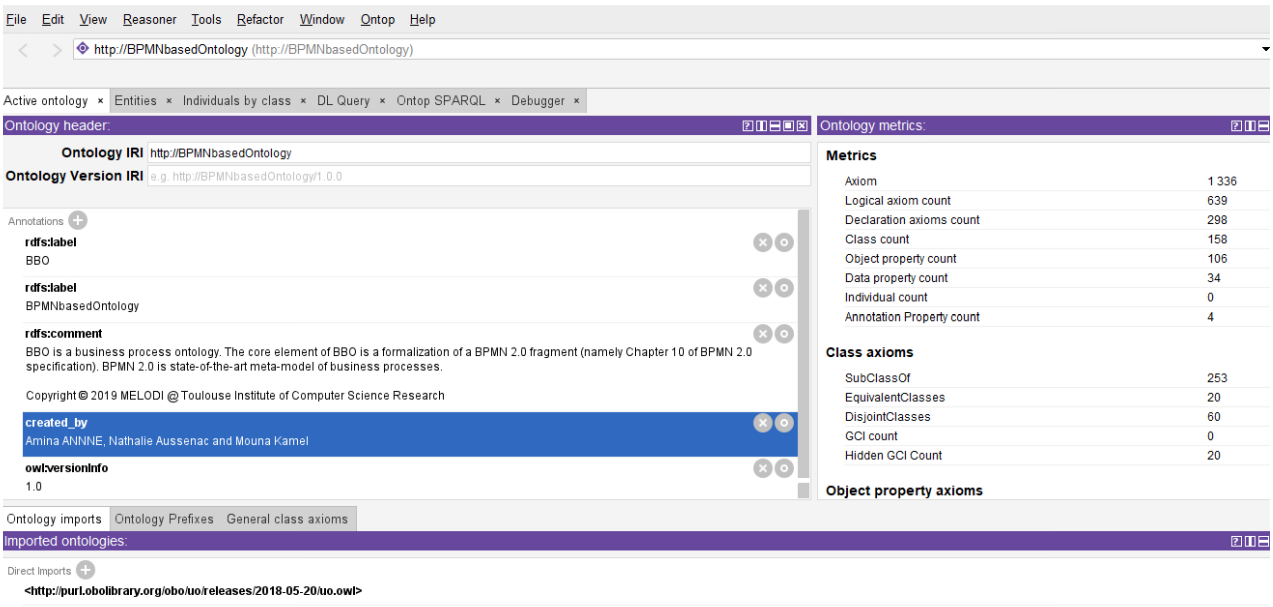


Figure 7: BBO Ontology metrics

Annane's approach involves a detailed analysis of BPMN elements, such as activities, events, and gateways, and their semantic representation within the BBO framework (Annane, 2017). By defining precise classes, properties, and relationships, she ensures that the ontology not only represents the graphical aspects of BPMN but also provides a machine-readable semantic layer (Annane, 2018). This semantic layer enhances the capability of BPM tools to perform automated reasoning and process optimization (Annane, 2019).

Moreover, Annane emphasizes the importance of interoperability in BPM systems, advocating for the use of ontologies to bridge the gap between different BPM tools and platforms (Annane, 2020). Her work demonstrates how the BBO can be used to integrate disparate systems, improving data exchange and process coordination (Annane, 2021).

Through her comprehensive research, Amina Annane has significantly advanced the field of business process management by leveraging semantic technologies to overcome the limitations of traditional BPMN. Her work on the BBO provides a robust framework for improving the efficiency and effectiveness of BPM systems (Annane, 2022).

3.7.1 Classes in the BBO

The BBO includes 158 distinct classes, each representing a concept in BPM. Notable classes include "Activity," which represents any action or task within a business process. This is a fundamental element in BPMN, encapsulating the work performed. The "AdHocSubProcess" class signifies subprocesses executed in an ad-hoc manner, reflecting the flexibility required in some business workflows. The "Agent" class represents entities that perform activities, such as human actors or automated systems. The "BoundaryEvent" class captures events attached to the boundary of an activity, playing a crucial role in exception handling within processes. The "UserTask" class represents tasks assigned to human users, highlighting the interaction between processes and human operators. These classes collectively enable the detailed modeling of business processes, encompassing various aspects from tasks and events to the entities involved.

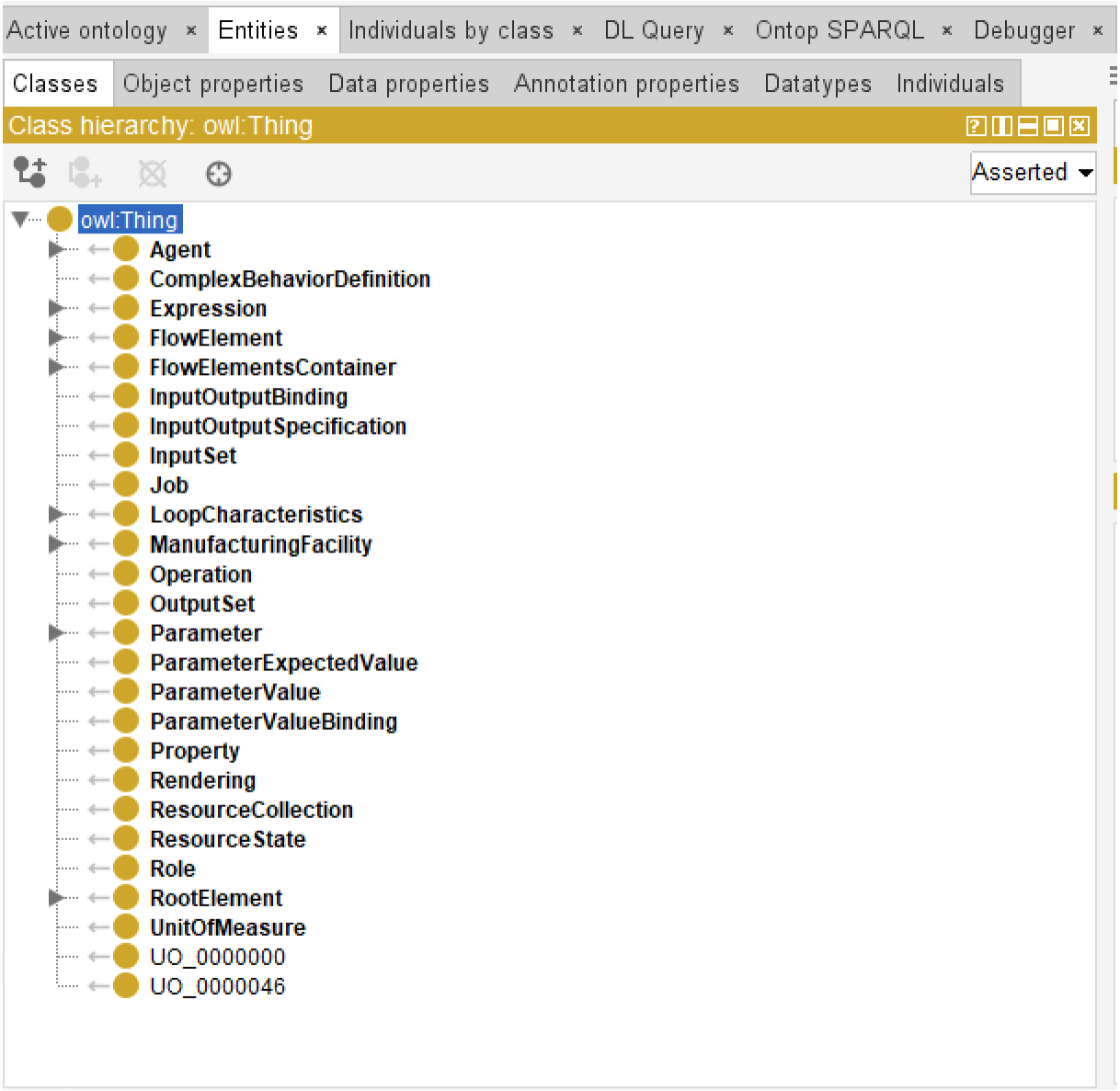


Figure 8: BBO Classes List

3.7.2 Object Properties

The ontology defines several object properties, which establish relationships between classes. Key object properties include "belongs," which links an element to the process it is part of, maintaining the hierarchical structure of processes. The "dependsOn" property indicates dependencies between tasks or processes, essential for understanding the sequence and prerequisites in workflows. The

"describes" property is used to link descriptive elements to their corresponding process elements, ensuring clarity and documentation. The "groups" property aggregates elements, useful for organizing related tasks or subprocesses. The "has_activationCondition" property specifies the conditions under which an activity or event is triggered, which is vital for dynamic process execution. These properties facilitate the intricate interconnections necessary for accurate and effective business process modeling.

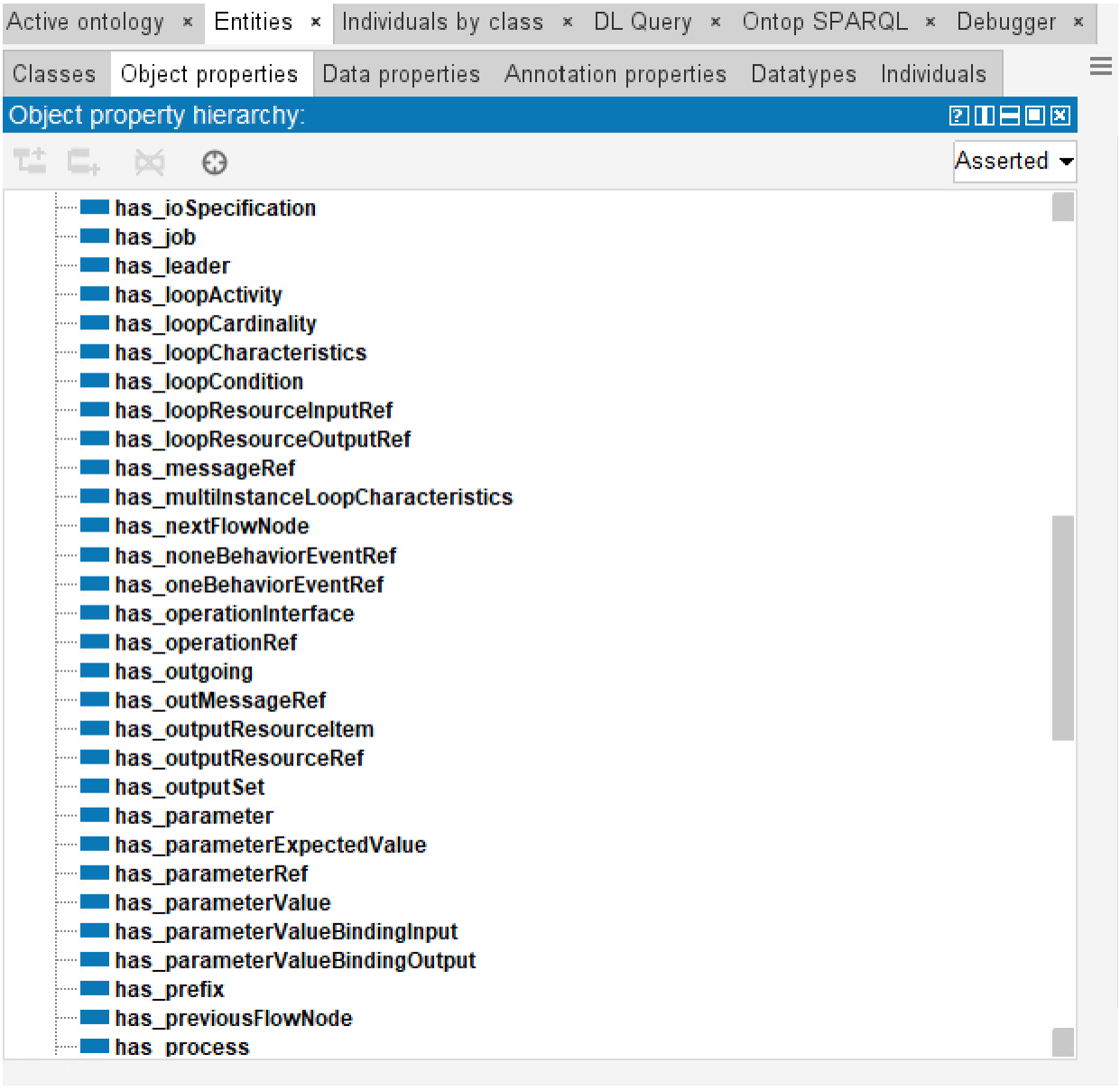


Figure 9: BBO Object Property List

3.7.3 Data Properties

Data properties in BBO describe attributes of classes, enhancing the specificity and detail of the ontology. Important data properties include "body," representing the content or details of a task or message. The "cancelRemainingInstances" property indicates whether remaining instances of a process should be canceled, which is used in process termination conditions. The "completionQuantity" property defines the quantity required to complete a task, important for tasks with multiple instances. The "createdOn" property captures the creation date of a process element, useful for version control and auditing. The "errorCode" property associates error codes with events, which is crucial for error handling and troubleshooting. These properties enable detailed descriptions and enrich the ontology, providing comprehensive information about each process element.

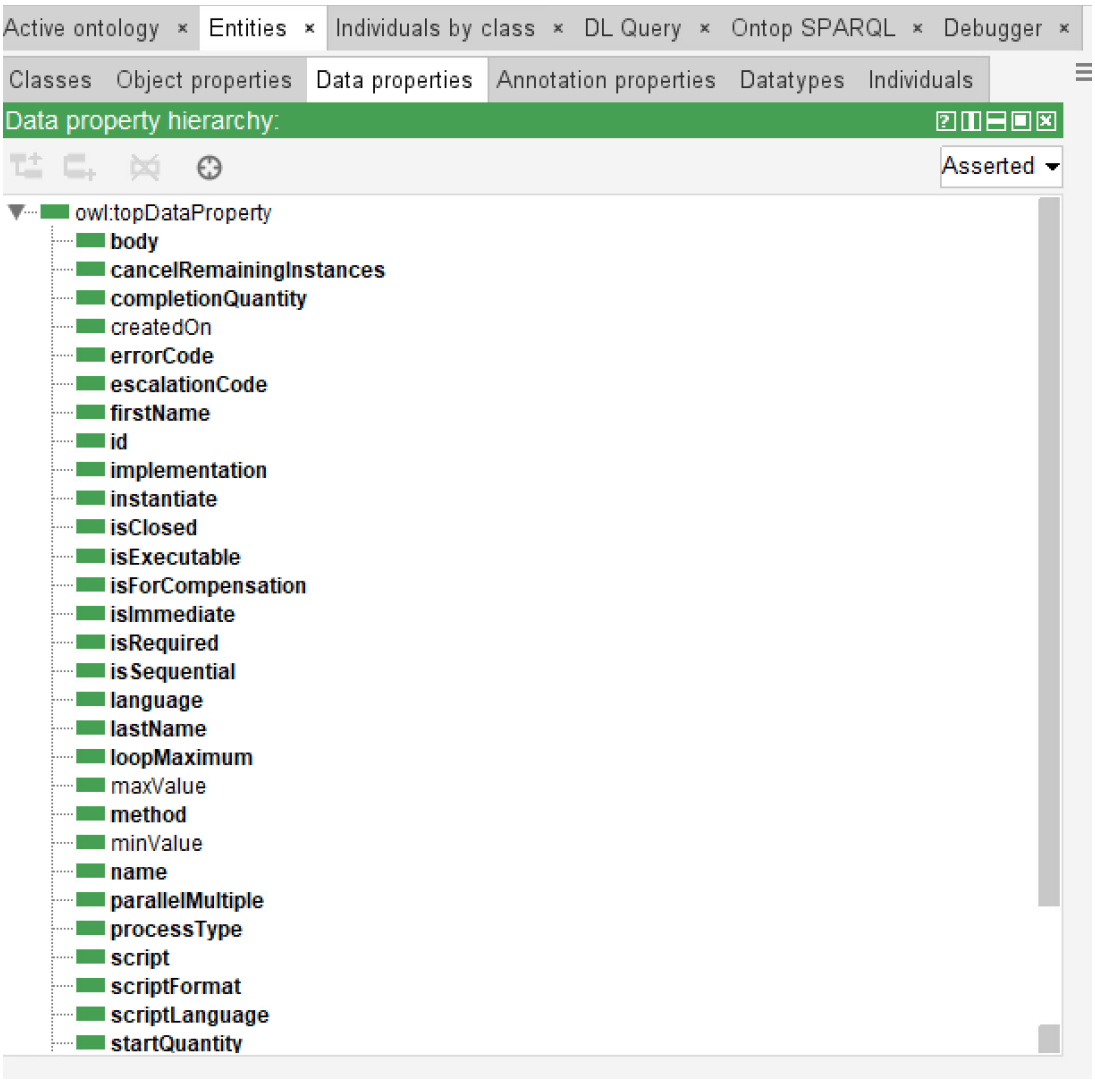


Figure 10: BBO Data Property List

3.7.4 Individuals

The ontology also defines several individuals, representing specific instances of classes. However, in the provided ontology, the list of individuals is sparse, indicating that the primary focus is on the schema-level representation rather than instance-level data. Individuals play a crucial role in populating the ontology with real-world data, which can then be used for reasoning and analysis.

The BPMN-Based Ontology (BBO) offers a robust framework for representing business processes semantically. Its extensive class hierarchy, coupled with detailed object and data properties, allows

for precise and comprehensive modeling of BPMN elements. By semantically defining elements like activities, events, agents, and tasks, BBO enables automated reasoning, interoperability, and enhanced understanding of business processes.

The ontology's comprehensive class structure is one of its strengths, as it covers a wide range of BPMN elements and provides a detailed schema for business process modeling. The rich interrelationships established by the object properties reflect the complex dependencies and interactions in real-world processes. The detailed descriptions provided by data properties offer specific attributes for each element, enhancing the ontology's specificity and detail.

Despite these strengths, the ontology has some challenges. The sparse individual instances limit its applicability in real-world scenarios where instance-level data is crucial for analysis. Additionally, as the ontology grows in size and complexity, maintaining performance and scalability might pose challenges, particularly in reasoning tasks. Ensuring seamless integration and interoperability with other domain-specific ontologies remains a critical challenge.

The BPMN-Based Ontology has significant potential in various applications. It can facilitate the automated composition of processes, enhancing efficiency and accuracy by providing a semantic representation of process elements. The ontology's standardized representation enables interoperability between different BPM tools, fostering collaboration and integration. Semantic descriptions and relationships allow for more sophisticated analysis and reasoning about processes, leading to better decision-making and optimization.

The BPMN-Based Ontology (BBO) provides a comprehensive and detailed framework for representing business processes semantically. Its rich class hierarchy, interrelationships, and detailed attributes enable precise modeling and analysis of BPMN elements. While challenges like sparse individual instances and scalability remain, the ontology's potential in applications like automated process composition and tool interoperability is significant. Future efforts should focus on expanding instance data, improving scalability, and integrating with other ontologies to fully realize the BBO's potential in enhancing business process management.

3.8 Problem Formulation

3.8.1 Identification of the Research Problem

Project management (PM) and business technology management (BTM) are crucial fields that have significantly evolved over the past decades. However, the integration of these domains remains a challenge, especially in the context of automating project team composition. The traditional methods often lack the flexibility and adaptability required to handle the dynamic nature of modern project environments. This research problem explores how semantic integration through ontologies can address these challenges and facilitate the automated composition of project teams, thereby improving project outcomes and efficiency.

The integration of project management methodologies with business technology management principles presents a unique opportunity to enhance project outcomes. However, existing approaches often fall short in providing a seamless integration that can support the automated composition of project teams. [14]This research aims to investigate how semantic technologies, specifically ontologies can be utilized to bridge this gap. The primary objective is to develop a framework that leverages semantic integration to automate the team composition process, ensuring that the selected team members align with the project requirements and organizational goals.

3.8.2 Defining Objectives and Scope

The primary objective of this research project is to develop a comprehensive framework that leverages semantic technologies, particularly ontologies and knowledge graphs, to integrate business technology management (BTM), project management (PM), Essence methodologies. This integration aims to facilitate the automated composition of project teams, thereby enhancing project customization, efficiency, and outcomes. By addressing the inherent complexities and dynamic nature of modern project environments, this framework seeks to provide a robust solution that aligns team composition with specific project requirements and organizational goals.

[14] Ontology and knowledge graph are both key concepts in the fields of artificial intelligence, semantic web, and data management, but they serve different purposes and have distinct characteristics. An ontology often serves as the schema or foundation for a knowledge graph. While the ontology defines the rules, relationships, and semantics of a domain, the knowledge graph operationalizes these definitions by connecting real-world data points.

3.8.2.1 Objectives

The overarching goal of this research project is to develop a comprehensive framework that leverages semantic technologies, particularly ontologies and knowledge graphs, to integrate Business Technology Management (BTM), Project Management (PM), and Essence methodologies. This integration aims to facilitate the automated composition of project teams, thereby enhancing project customization, efficiency, and outcomes. By addressing the inherent complexities and dynamic nature of modern project environments, the framework aspires to provide a robust solution that aligns team composition with specific project requirements and organizational goals.

The first step in achieving this goal involves analyzing the current state of integration between BTM, Essence, and PM. This will be done through an extensive literature review to understand the existing methodologies and frameworks in both project management and business technology management. The review will also focus on identifying key gaps and challenges in integrating these domains, with particular attention to the limitations of current approaches and the potential for improvement through the use of semantic technologies.

Following this analysis, the next step is to develop an ontology-based framework for semantic integration. This involves designing and implementing an ontology that integrates concepts from both PM and BTM, serving as a structured representation of knowledge within these domains. The ontology will be developed using the Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL) to ensure it supports semantic reasoning and knowledge representation, capturing the essential elements and relationships necessary for effective project management and business technology integration.

Once the ontology is developed, the project will focus on automating project team composition. This will be achieved by creating rule-based queries within the ontology that represent typical decision parameters for project management customization. These rules will automate the selection and composition of project teams based on specific project requirements. The queries will be tested on the developed ontology using real-world project data to verify the logical process recommendations and ensure the framework's effectiveness in automating team composition.

The next phase of the research involves evaluating the effectiveness of the semantic integration framework. The framework's ability to customize project processes and automate team

composition will be assessed through a series of evaluations, including the use of schema metrics such as relationship diversity (RD) and schema deepness (SD), as well as competency questions to evaluate the ontology's performance. Additionally, the impact of the framework on project outcomes will be measured using metrics like F-measure, which will help quantify the quality and accuracy of the ontology's inference capabilities.

3.8.2.2 Expected Outcomes

The successful completion of this project is expected to result in a semantic integration framework that significantly enhances the customization and automation of project team composition. This framework will:

- Provide a more open and flexible representation of project management methods, integrating the dynamic features of agile methodologies with the more static aspects of traditional project environments.
- Enable seamless adoption of agile methods within traditional project management settings, addressing key challenges related to process customization and team composition.
- Offer a systematic and evolving approach to professional practice standards, supporting IT PM professionals, employers, higher education institutions, and other relevant associations.
- Facilitate decision-making by making agile PM and BTM BOK standards easily accessible, customizable, and reusable.

3.9 Building, Intervention, and Evaluation (BIE)

3.9.1 Methodologies for building Ontologies

Building an ontology requires a structured and iterative methodology to ensure that it is accurate, reusable, and aligned with its intended purpose. This process begins with defining the scope and objectives**,** where the domain of interest is clearly delineated, and the purpose of the ontology is

established. For example, the ontology may aim to integrate heterogeneous data sources, enable automated reasoning, or facilitate knowledge sharing. At this stage to identify the intended users and their needs.

3.9.1.1 Gathering and analysis of requirements

This step involves collecting data from reliable sources, including domain experts, academic literature, and industry standards, to ensure comprehensive coverage of the domain. Key concepts, relationships, and attributes are identified and organized into an initial schema. Additionally, use cases are documented to represent real-world scenarios where the ontology will be applied. A review of existing ontologies is also important at this stage to determine whether any components can be reused or adapted, thereby saving development time and ensuring consistency with established standards.

3.9.1.2 Design Phase

This involves defining core elements such as classes (categories of entities), properties (attributes or relationships), and instances (specific examples). Classes are arranged hierarchically using relationships such as "is-a" or "part-of," enabling logical organization. Semantic rules and axioms are defined to enforce domain-specific constraints (e.g., "A team must have at least one member"). The level of detail in the ontology is tailored to the application, ranging from lightweight taxonomies to more complex formal ontologies.

3.9.1.3 Tools and Technologies

Ontology building is a critical task in various domains such as semantic web, knowledge management, and artificial intelligence. The effectiveness of an ontology largely depends on the tools used for its construction. These tools vary significantly in terms of functionality, ease of use, and application domain. This paper aims to provide a comprehensive benchmark of ontology-building tools, examining their features, strengths, and weaknesses, thereby guiding researchers and practitioners in selecting appropriate tools for their projects.

Ontology-building tools can be broadly classified into graphical tools, integrated development environments (IDEs), and automated tools. Each class serves different purposes and offers distinct advantages and limitations.

Graphical tools facilitate the creation of ontologies through user-friendly interfaces, allowing users to define classes, properties, and relationships visually. They are particularly beneficial for users who may not have extensive programming knowledge.

Protégé is one of the most widely used graphical ontology editors. Developed by Stanford University, Protégé provides a comprehensive platform for creating, visualizing, and managing ontologies. It supports a wide range of ontology languages, including OWL and RDF(S), and offers an extensive plugin architecture for extended functionality (Noy et al., 2001). Protégé's user-friendly interface and robust community support make it a preferred choice for many ontology developers.

TopBraid Composer is another prominent graphical tool. It is known for its rich set of features, including support for SPARQL queries, reasoning, and integration with various data sources. TopBraid Composer also provides capabilities for collaborative ontology development, which is crucial for large-scale projects (Wang et al., 2006).

Integrated Development Environments (IDEs) for ontology development offer a more sophisticated environment for users who require advanced functionalities and are comfortable with coding. These tools typically integrate ontology development with other aspects of knowledge management and data integration.

Eclipse IDE with the OWL API plugin is a powerful combination for ontology development. Eclipse provides a robust environment for Java development, and with the OWL API, it enables users to create, manipulate, and query OWL ontologies programmatically (Horridge et al., 2007). This setup is particularly useful for developers who need to integrate ontology development with other software engineering tasks.

OntoStudio, developed by semafora systems, is an advanced IDE specifically designed for ontology engineering. It supports a wide range of ontology languages and offers features such as graphical modeling, reasoning, and integration with external data sources. OntoStudio is also known for its support for complex ontology projects, providing tools for versioning and collaboration (Vrandecic & Tane, 2008).

Automated tools leverage machine learning and natural language processing techniques to assist in the creation and refinement of ontologies. These tools are particularly useful for handling large datasets and extracting knowledge from unstructured data sources.

Text2Onto is an example of an automated tool that focuses on ontology learning from text. It uses a combination of linguistic and statistical methods to extract concepts, relationships, and instances from text corpora. This tool significantly reduces the manual effort required in ontology creation and is especially useful in domains with extensive textual data (Cimiano & Völker, 2005).

DeepDive is another automated tool that employs a machine learning approach to knowledge base construction. It allows users to define extraction rules and then automatically extracts structured information from large text datasets. DeepDive is particularly effective in domains such as biomedical research, where vast amounts of textual data need to be processed (Niu et al., 2012).

To provide a comprehensive benchmark, the table below compares the tools based on the evaluation criteria discussed above.

Evaluation Criteria	Protégé	TopBraid Composer	Eclipse with the OWL API	OntoStudio	Text2Onto	DeepDive
Usability	Excels in usability due to graphical interface and extensive documentation; ideal for users with limited technical expertise.	Excels in usability due to graphical interface and extensive documentation; ideal for users with limited technical expertise.	Requires more technical knowledge but offers greater flexibility for advanced users.	Requires more technical knowledge but offers greater flexibility for advanced users.	Provides a balance between usability and automation.	Powerful but requires significant setup effort.
Functionality	Offers extensive features for ontology creation and management.	Offers extensive features for ontology creation and management.	Caters to developers needing advanced capabilities and customization options.	Caters to developers needing advanced capabilities and customization options.	Excels in handling large datasets and extracting knowledge from unstructured sources.	Excels in handling large datasets and extracting knowledge from unstructured sources.
Scalability	Supports scalability but may face performance issues with very large ontologies.	Supports scalability but may face performance issues with very large ontologies.	Highly scalable, suitable for large and complex projects.	Highly scalable, suitable for large and complex projects.	Designed to handle large datasets effectively.	Designed to handle large datasets effectively.

Interoperability	Offers extensive interoperability features, supporting various ontology formats and integrations.	Offers extensive interoperability features, supporting various ontology formats and integrations.	Benefits from the wide range of Eclipse plugins.	Offers extensive interoperability features, supporting various ontology formats and integrations.	Provides APIs for integrating with other systems and data sources.	Provides APIs for integrating with other systems and data sources.
Community Support	Large and active user community, providing extensive resources and support.	Strong support with access to professional services.	Large and active user community, providing extensive resources and support.	Strong support with access to professional services.	Smaller community, supported by academic and research institutions.	Smaller community, supported by academic and research institutions.

Table 4: Structured comparison of Ontology's tools based on the evaluation criteria.

3.9.1.4 Ontology Evaluation

The ontology is then constructed through an iterative development process, where concepts, relationships, and constraints are progressively refined. Metadata, including descriptions and labels, is added to enhance usability. Once the initial model is complete, validation and evaluation are conducted to ensure its logical consistency and practical applicability. Reasoning tools are employed to detect contradictions, while competency questions are tested to verify that the ontology meets its objectives. Feedback from domain experts is invaluable in ensuring that the ontology accurately represents the intended domain. Evaluation metrics, such as coverage, conciseness, and adaptability, are used to assess the quality of the ontology.

3.9.1.5 Deployment and Integration

Following validation, the ontology undergoes refinement and iteration to address gaps and improve its design. A version control system ensures that updates are well-documented and that previous versions remain accessible. Extensibility is prioritized to accommodate future changes or additional requirements. Once finalized, the ontology is prepared for deployment and integration into relevant systems, whether as part of a semantic web application, a knowledge management platform, or an AI-based solution. It is exported in standard formats for compatibility and accompanied by comprehensive documentation to guide users.

3.9.1.6 Maintenance and Update

The ontology requires ongoing maintenance and updates to remain relevant as domain knowledge evolves. A structured change management process ensures that modifications are systematically reviewed and implemented. Engaging the community of users and stakeholders is critical for maintaining its relevance and utility. Periodic reviews help identify areas for improvement and ensure that the ontology continues to meet its objectives.

3.10 Development of Ontologies using protégé

Incorporating IT project management (PM) requirements, roles, and processes into an ontology is essential for structuring and streamlining project knowledge, facilitating better project execution and communication. This guide outlines a practical framework and key considerations for developing an IT PM ontology, focusing on actionable steps and real-world application. We will integrate key concepts from Business Technology Management (BTM), Essence, PM², and BPMN 2.0 Based Ontology (BBO) into a cohesive IT PM ontology.

The first step in developing an IT PM ontology is to define its scope and objectives. The scope should focus on specific areas such as project requirements, roles, and processes, drawing from established frameworks like BTM, Essence, PM², and BBO. The primary objectives of the ontology should be to enhance knowledge sharing across project teams, improve decision-making processes, and ensure alignment with industry standards.

Identifying key components is crucial. For requirements, we should capture both functional aspects like system capabilities, data management, and integration, as well as non-functional aspects such as performance and security. In terms of roles, it is important to define various positions including Project Manager, Developer, Tester, and Stakeholder, detailing their responsibilities and interactions. Processes need to be mapped out comprehensively, covering stages from planning and execution to monitoring and closing.

Gathering information from reliable data sources is the next step. Conducting a thorough literature review allows us to extract relevant terminologies from frameworks such as PMBOK, PRINCE2, and Agile methodologies. Engaging in expert interviews with PM professionals helps validate the components, while analyzing real-world case studies of IT projects provides insights into practical applications.

Developing the ontology structure involves creating a high-level conceptual model that outlines the main classes and relationships within the ontology. For example, we might create classes for Requirement, Role, and Process, with subclasses for Functional Requirement and Non-functional

Requirement under Requirement, and Planning and Execution under Process. Using a formal ontology language such as OWL or RDF to define these classes, properties, and axioms ensures the ontology is well-structured and semantically rich.

Implementation is carried out using tools like Protégé, an open-source ontology editor. In Protégé, we create classes for Requirement, Role, and Process, and define subclasses and properties for each. For instance, under Functional Requirement, we can add properties such as hasPriority and hasStakeholder. SPARQL queries can be implemented for data retrieval and analysis, such as querying all high-priority requirements.

Validation and evaluation are essential to ensure the ontology's accuracy and utility. Consistency checks should be performed using Protégé's reasoner to validate logical consistency and completeness. Conducting expert reviews helps ensure the ontology meets the defined objectives, and pilot testing the ontology in a real-world IT project allows us to evaluate its practical effectiveness and make necessary adjustments.

Deployment and integration involve incorporating the ontology within the organization's IT systems and ensuring compatibility with existing tools and databases. Providing training and support to users is crucial for facilitating adoption and effective use. Developing comprehensive documentation and user guides helps users understand and utilize the ontology efficiently.
For example, consider a company implementing a new IT system that needs to manage requirements, roles, and processes effectively. In the requirement analysis phase, functional requirements for system capabilities are gathered and categorized into technical, business, and security aspects. Role definitions include detailing responsibilities and interactions for roles such as Project Manager, Developer, and Tester. Process mapping involves outlining key processes from initiation to closure, defining the sequence of activities and decision points.

3.10.1 Setting Up Protégé

- Visit the Website: Open your web browser and go to the Protégé website: https://protege.stanford.edu/

Figure 11: Protégé download homepage

- Click on the "Download" link. Choose the appropriate version for your operating system and download the installer.

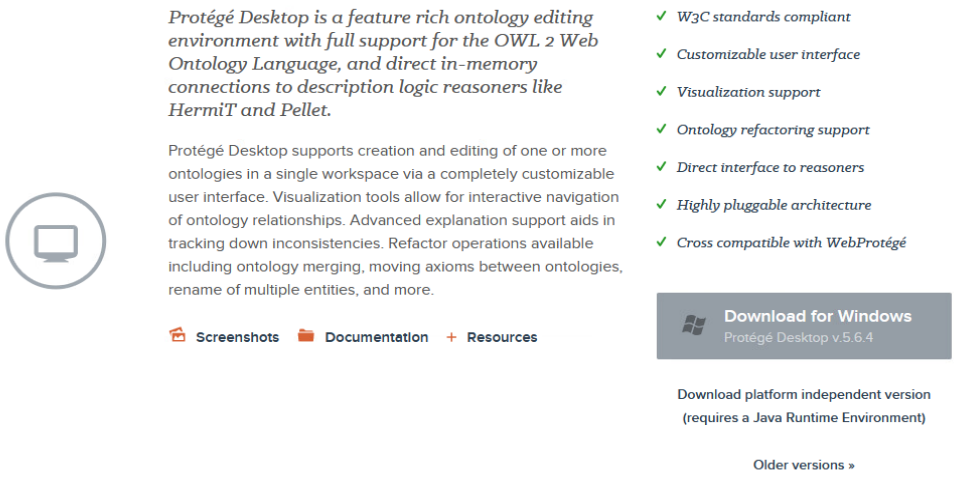


Figure 12: Protégé download for Windows

- Run the Installer: Locate the downloaded installer file (typically in your Downloads folder) and double-click it to run the installer.

- Follow Prompts: Follow the on-screen prompts to complete the installation. This usually involves agreeing to the license agreement, selecting an installation location, and clicking "Next" through the setup wizard.

- Finish: Once the installation is complete, click "Finish" to exit the installer.

- Find Protégé in your Start Menu or search for it and click to open.

3.11 BTM Ontology

Once the classes, subclasses, properties, and individuals have been defined, the next step is to structure the ontology within Protégé. This involves creating the class hierarchy, adding properties, and defining individuals. The structure follows the logical organization of the BTM-BOK, ensuring that the ontology accurately represents the knowledge domain.

The hierarchical structure provides a clear and organized representation of the BTM domain, facilitating easy navigation and understanding.
Implementing the ontology in Protégé involves several steps, each of which is crucial for ensuring the accuracy and usability of the ontology.

3.11.1 Creating Classes and Subclasses

The first step is to create the primary classes and their respective subclasses. In Protégé, this is done by defining each class and subclass within the class hierarchy. This step ensures that all relevant categories of the BTM-BOK are represented in the ontology.

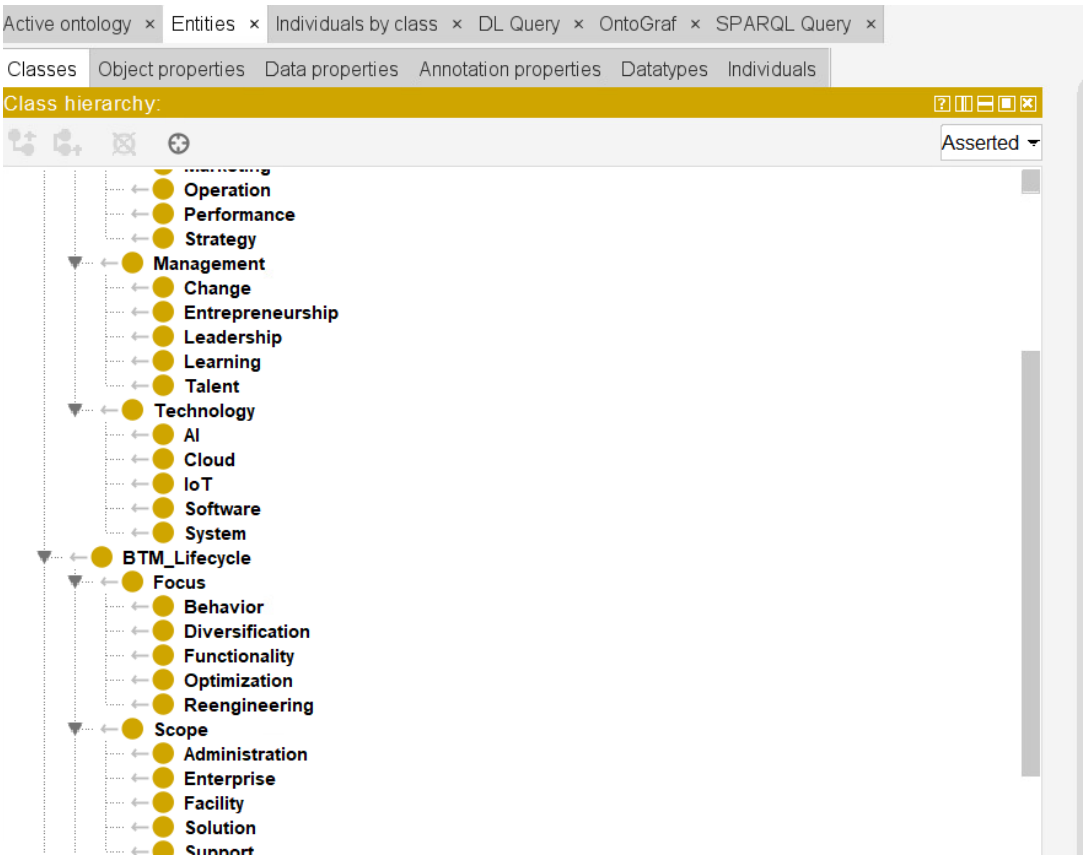


Figure 13: BTM Ontology class and subclass

3.11.2 Creating Object Properties

Next, object properties are defined to establish the relationships between classes. This involves specifying each property and its domain and range. For example, the "hasComponent" property might be used to relate a class to its constituent components, while the "hasDependency" property outlines dependencies between different components.

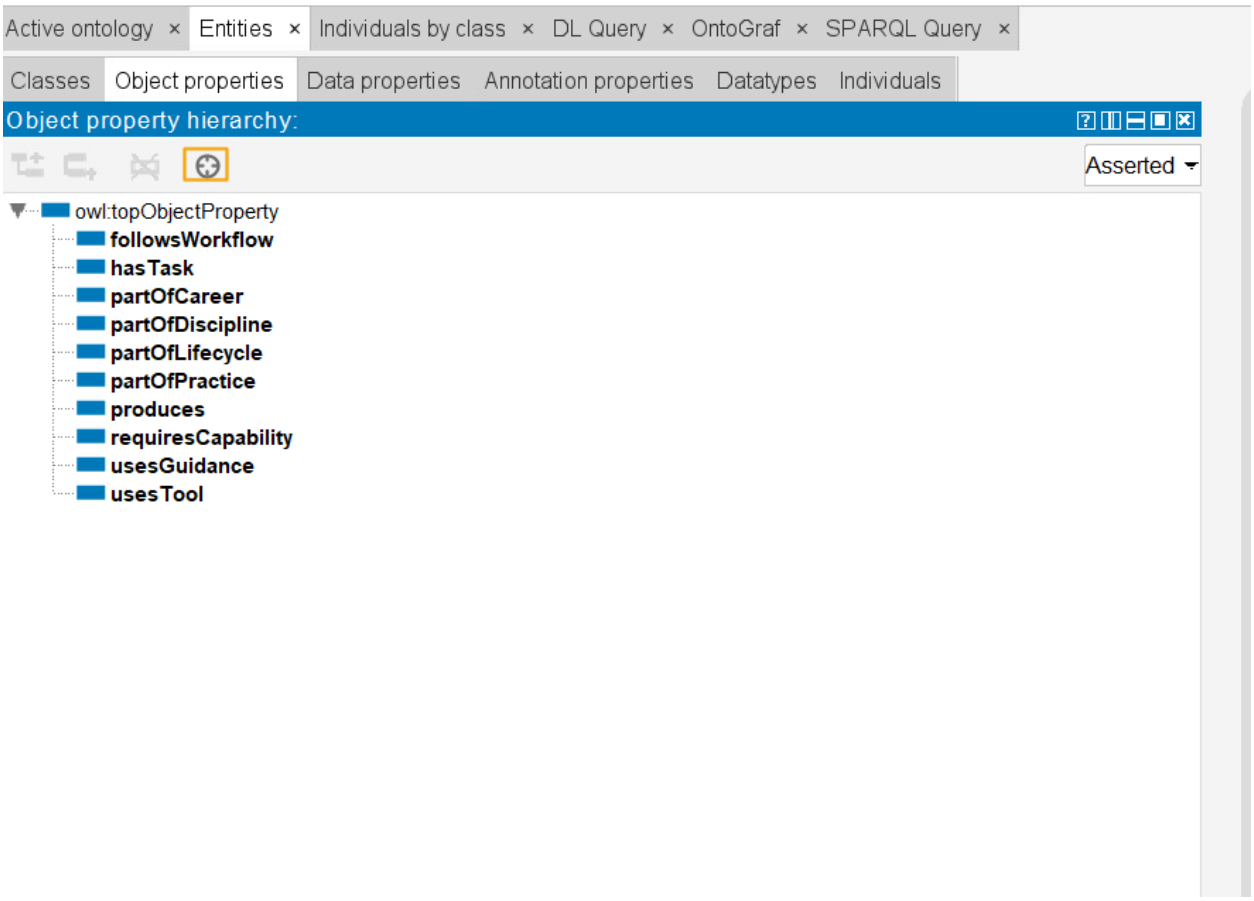


Figure 14: BTM Ontology Object Property

3.11.3 Creating Data Properties

Data properties are then added to provide additional details and attributes for the classes. Each data property is defined with its domain and range, ensuring that the appropriate classes are enriched with descriptive information. For example, the "description" property might be added to provide a textual description of each class.

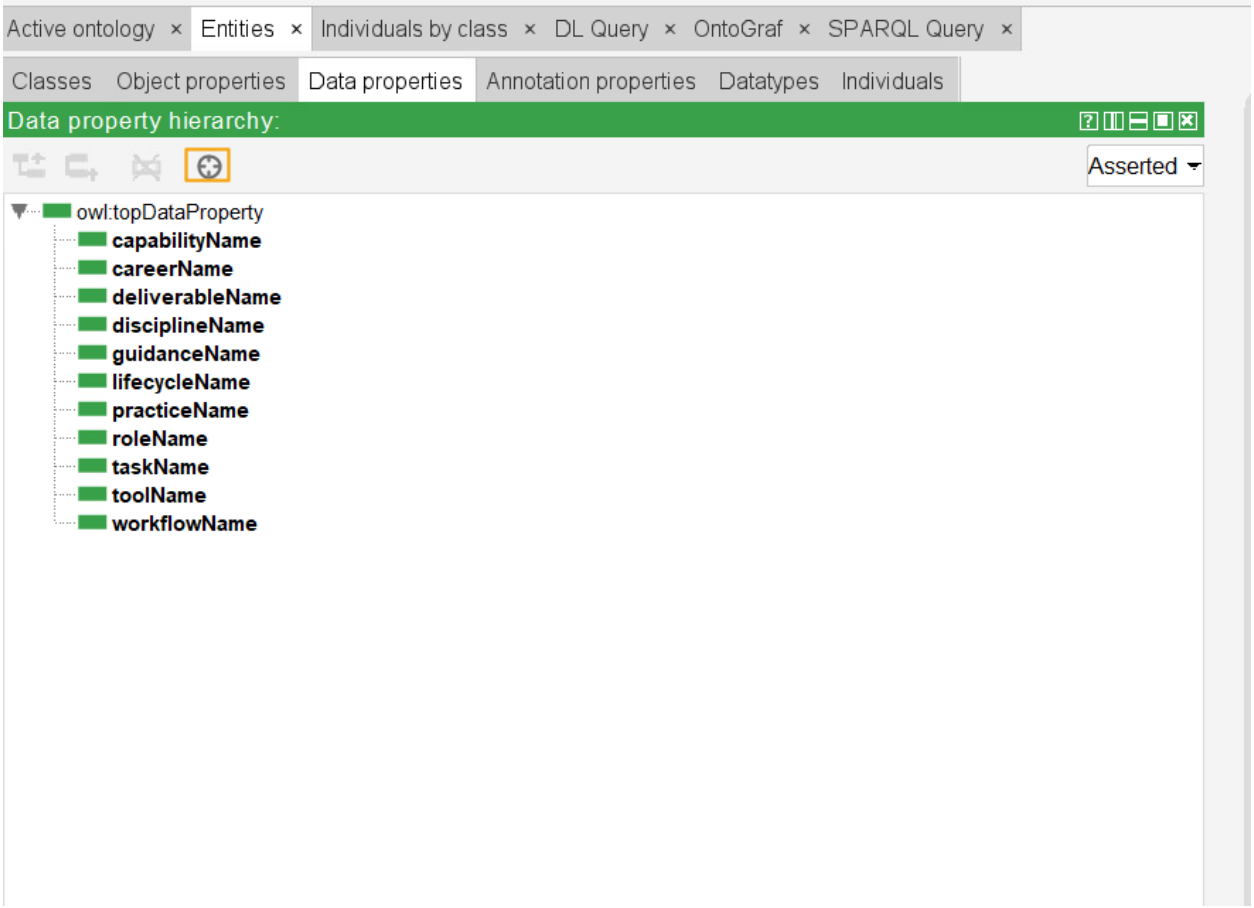


Figure 15: BTM Ontology Data Property

3.11.4 Creating Individuals

Once the classes and properties are defined within protégé, individuals are created to represent specific instances within the BTM domain. This step involves specifying each individual and its corresponding class, along with any relevant properties and values. For example, an individual representing a specific BTM project might include properties such as "description," "version," and "dateCreated."

3.12 Essence ontology

The Essence framework provides a kernel and a language for software engineering methods, capturing the essence of effective practices in software development. When building an ontology for Essence, the goal is to model the key concepts, relationships, and attributes of software engineering methods as defined by the Essence standard.

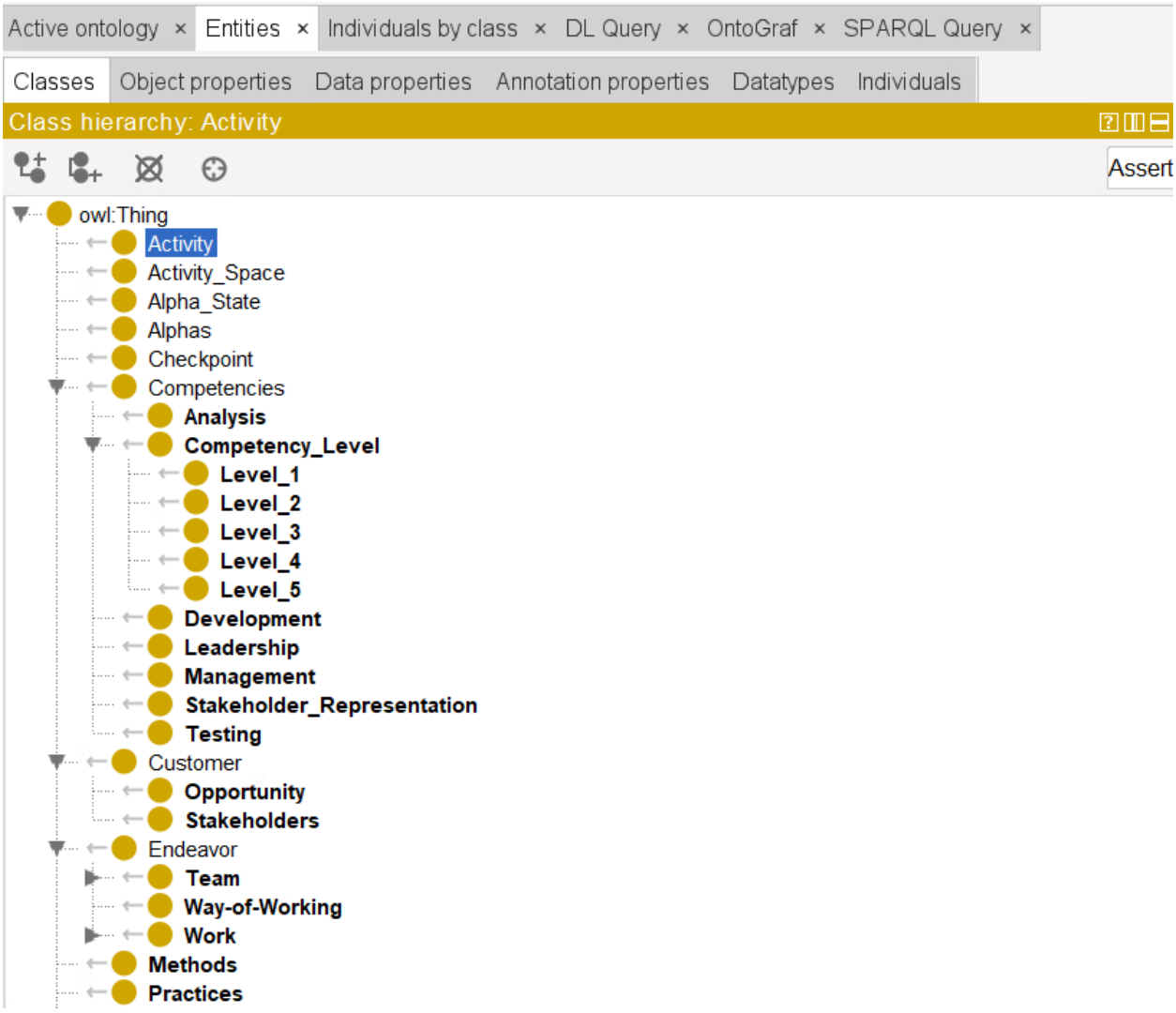


Figure 16: Essence Ontology Class and subclass.

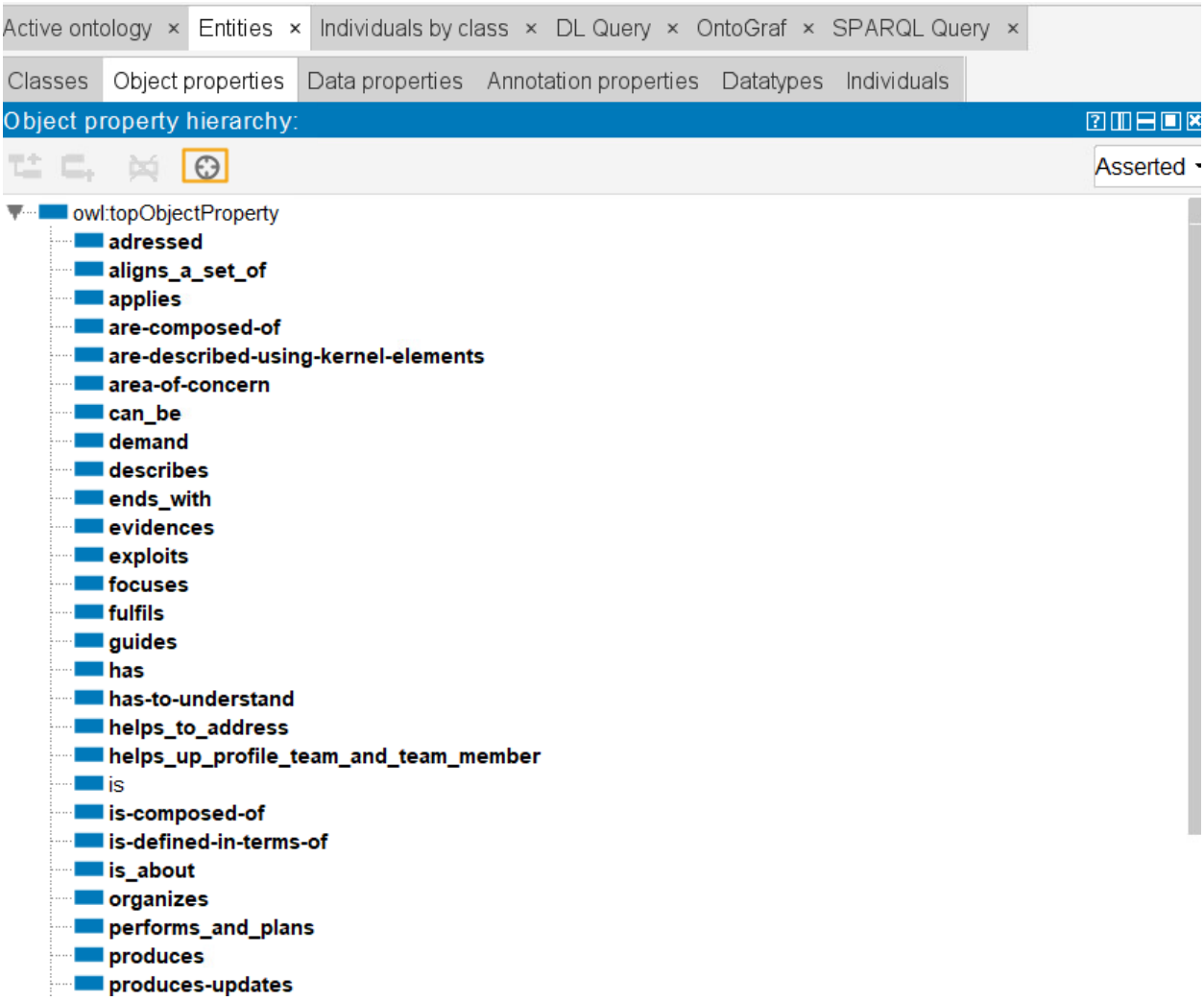


Figure 17: Essence Ontology Object Property.

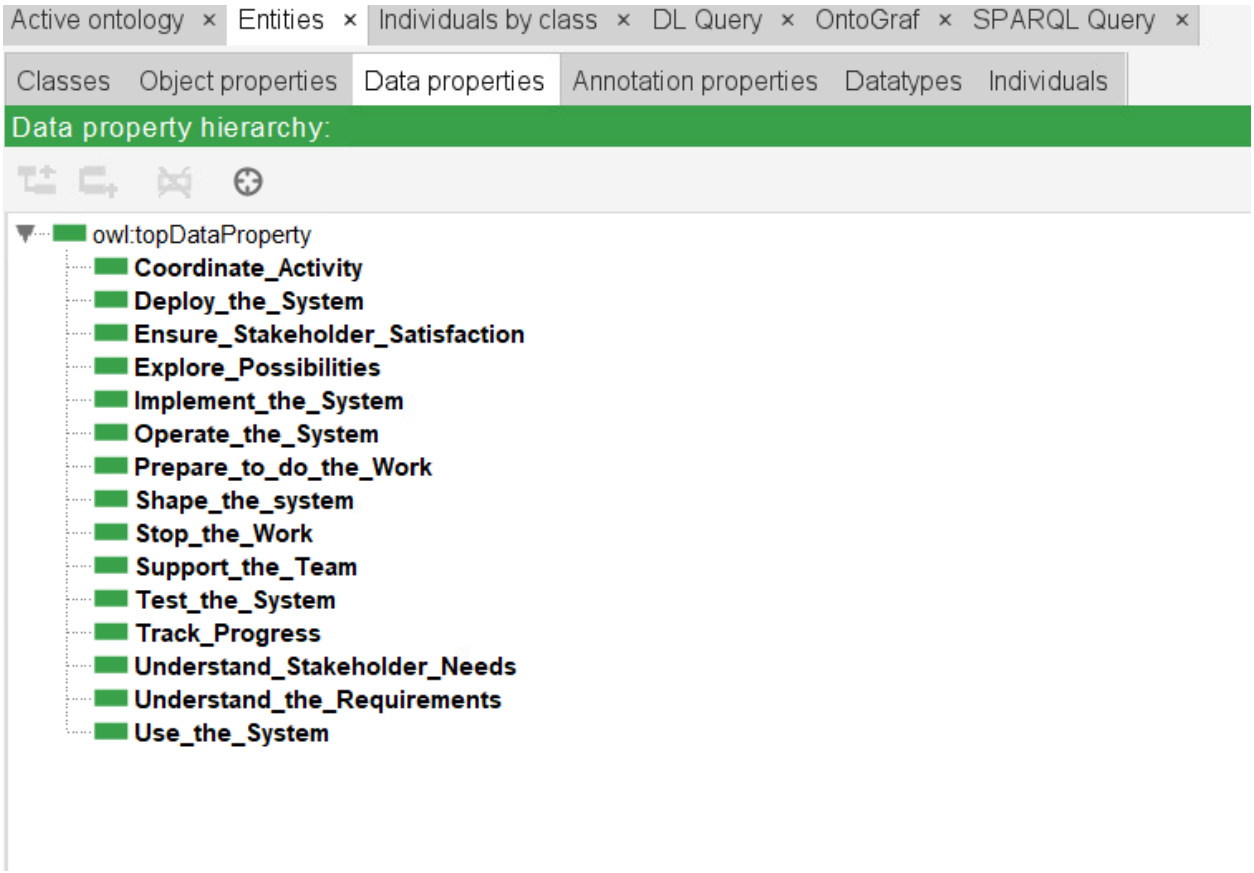


Figure 18: Essence Ontology Data Property

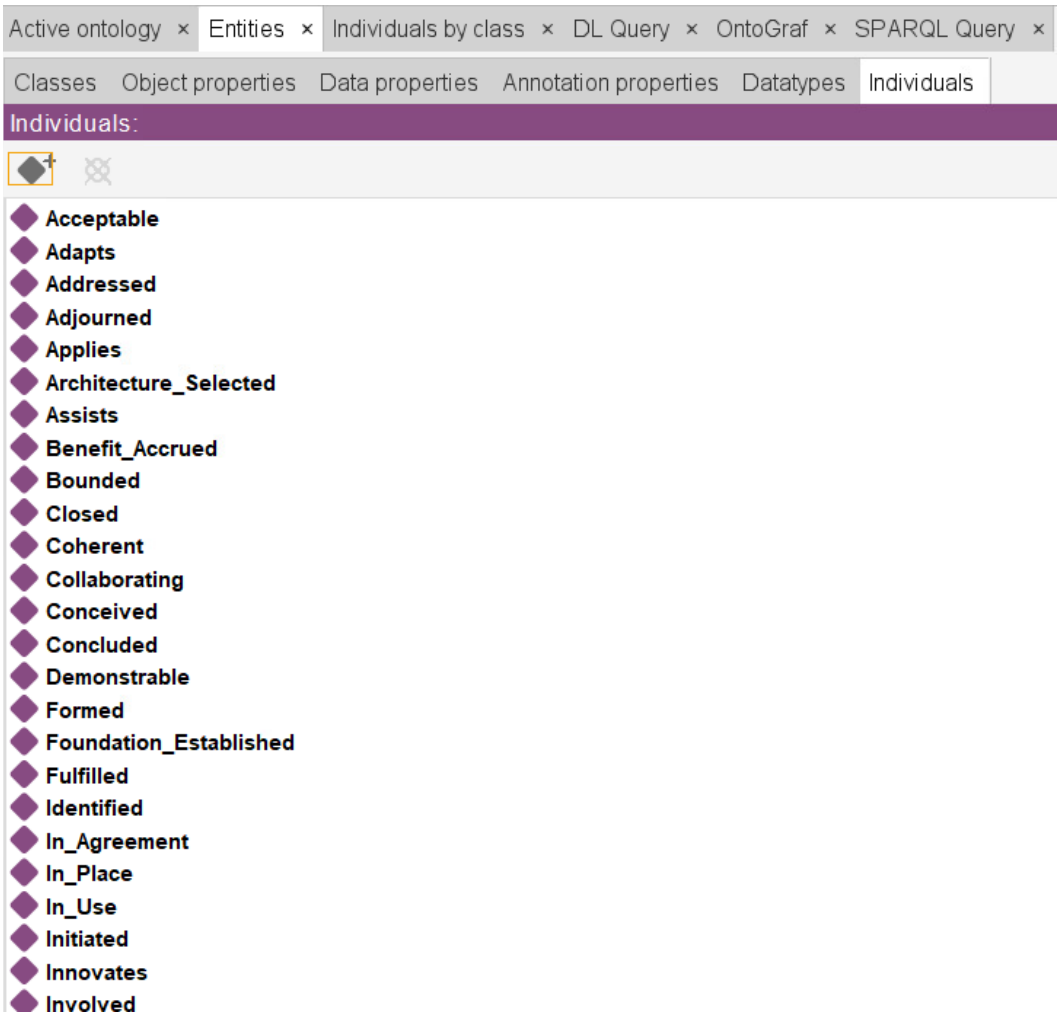


Figure 19: Essence Ontology Individuals

3.13 PM² Ontology

The development of an ontology based on PM², the Project Management Methodology developed by the European Commission, holds significant importance in the realm of project management, especially within organizations that aim for enhanced consistency, efficiency, and interoperability in their project execution processes. Ontologies, in essence, serve as structured frameworks that facilitate a shared understanding of a specific domain by defining the relationships between concepts within that domain. In the context of PM², an ontology would encapsulate the methodology's processes, artifacts, roles, and guidelines, thus promoting a standardized approach to project management across various projects and organizations.

Firstly, an ontology grounded in PM² would ensure uniformity in terminology and conceptual understanding across different stakeholders. This is particularly crucial in large, multi-national projects where diverse teams might have varying interpretations of project management terms and practices. By providing a clear and consistent conceptual framework, the PM² ontology would mitigate misunderstandings and enhance communication efficiency among project teams, leading to smoother collaboration and project execution.

Moreover, the integration of PM² into an ontology could significantly enhance decision-making processes. With a well-defined structure that interrelates all aspects of the PM² methodology, project managers and team members can more easily access and utilize critical information. This structured information retrieval supports informed decision-making, reducing the risk of errors and improving the overall quality of project outcomes.

In addition, an ontology based on PM² could facilitate the automation of project management tasks. By employing semantic technologies, it becomes possible to develop intelligent systems that can reason about the project data, automate routine tasks, and provide proactive recommendations. This level of automation not only improves efficiency but also allows project managers to focus on more strategic aspects of their projects.

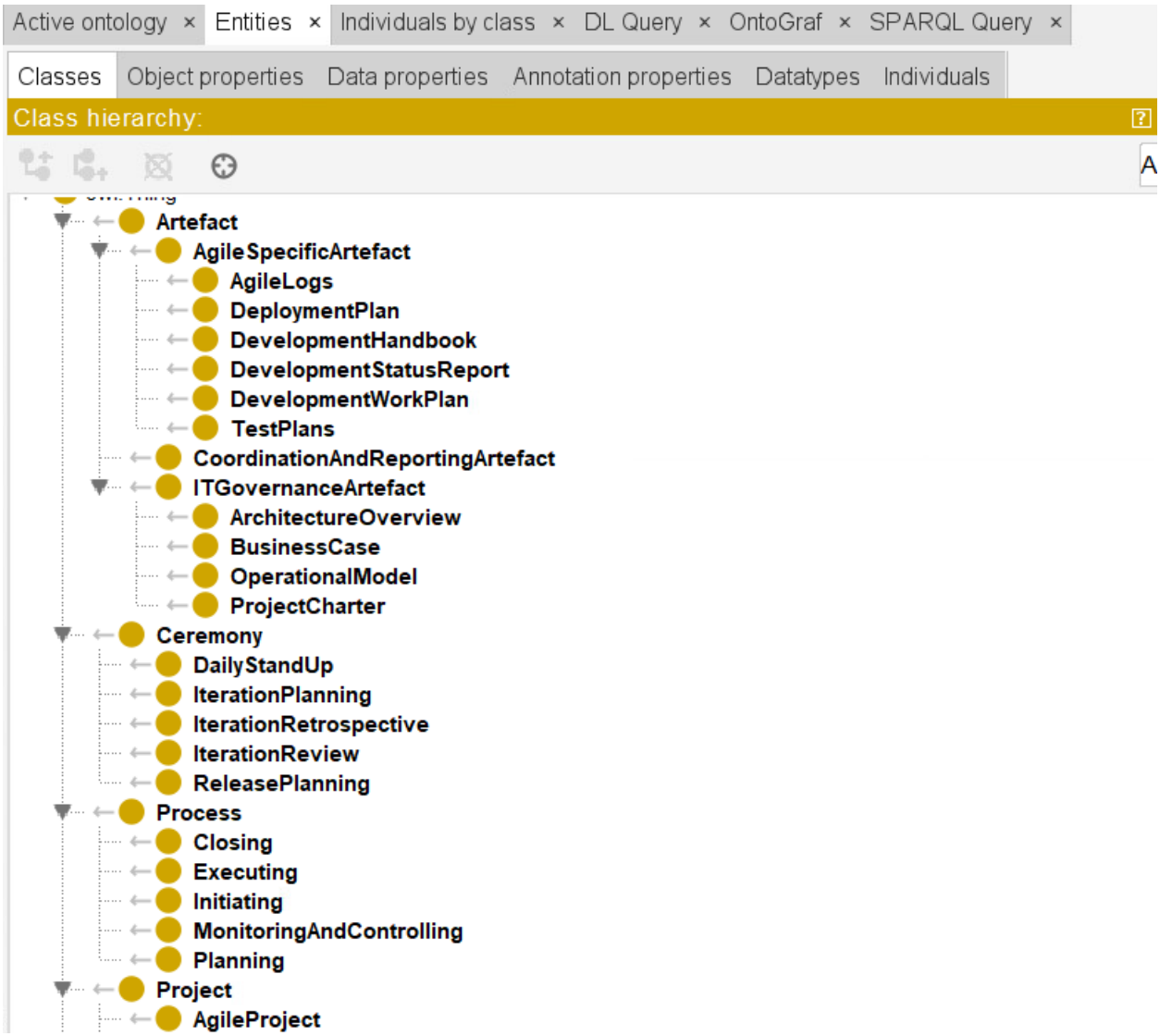


Figure 20: PM² Ontology Class and subclass.

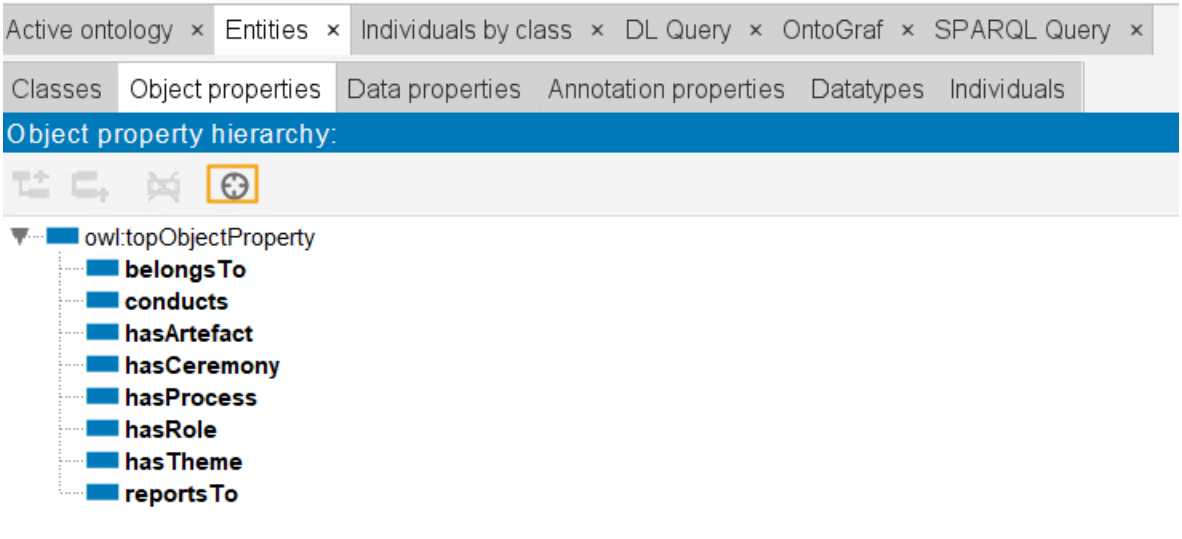


Figure 21: PM² Ontology Object Property

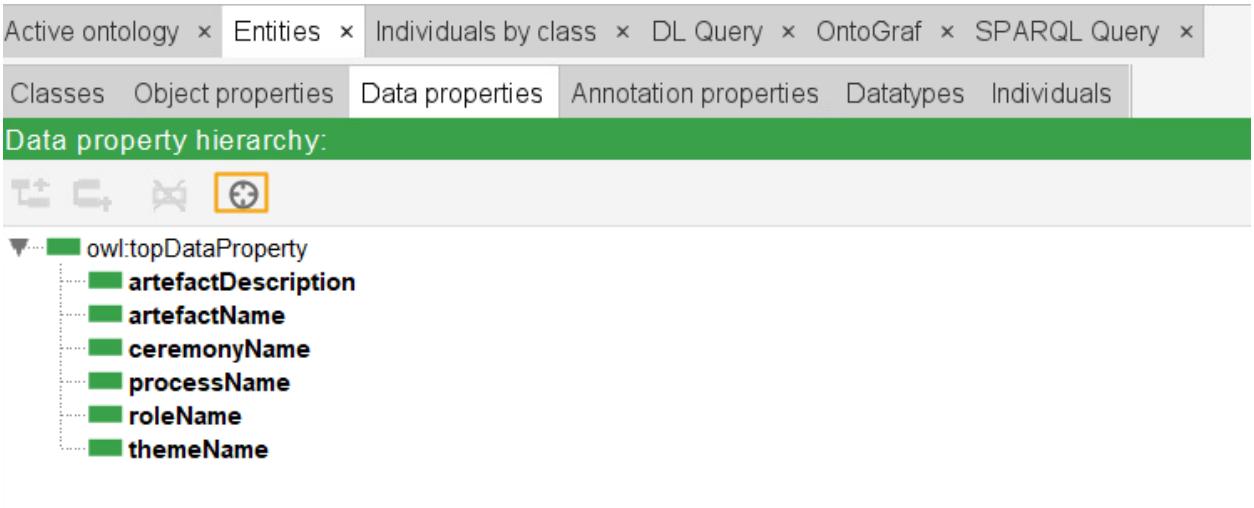


Figure 22: PM² Ontology Data Property

3.14 Incorporation of IT PM Requirements, Roles, and Processes

To incorporate IT Project Management (PM) requirements, roles, and processes into an ontology, a structured approach is required. This approach will include defining concepts, classes, object properties, and their descriptions. The following comprehensive process outlines these steps, along with examples of concepts, classes, and properties.

3.14.1 Key Concepts

- Project Management: The overall discipline of managing projects.
- Requirement Management: Processes to capture, analyze, and manage requirements.
- Roles and Responsibilities: Various roles involved in the project and their responsibilities.
- Processes and Activities: Steps and procedures involved in managing projects.

3.14.2 Classes

- Project
- Requirement
- Role
- Process
- Activity
- Artefact
- Stakeholder

3.14.3 Define Object Properties

- hasRequirement: Links a project to its requirements.
- hasRole: Links a project to the roles involved.
- performsActivity: Links a role to an activity they perform.
- usesArtefact: Links a process to the artefacts used.

- involvesStakeholder: Links a project to the stakeholders involved.

3.14.4 Create Classes and Define Properties

3.14.4.1 Project Class

- Description: A temporary endeavor undertaken to create a unique product, service, or result.
- Properties:
  - hasRequirement: Requirement
  - hasRole: Role
  - involvesStakeholder: Stakeholder

3.14.4.2 Requirement Class

- Description: A capability that a product or service must have to satisfy a stakeholder's need.
- Properties:
  - belongsToProject: Project
  - hasPriority: String
  - isValidatedBy: Role

3.14.4.3 Role Class

- Description: A set of responsibilities assigned to an individual or group within the project.
- Properties:
  - performsActivity: Activity
  - isPartOf: Stakeholder

3.14.4.4 Process Class

- Description: A set of interrelated activities that transform inputs into outputs.
- Properties:
  - usesArtefact: Artefact

- isPerformedBy: Role

3.14.4.5 Activity Class

- Description: A distinct, scheduled portion of work performed during the course of a project.
- Properties:
  - isPartOfProcess: Process
  - isPerformedBy: Role

3.14.4.6 Artefact Class

- Description: Tangible outputs produced and used during a project.
- Properties:
  - isUsedIn: Process
  - belongsToProject: Project

3.14.4.7 Stakeholder Class

- Description: An individual or group with an interest in the project's outcome.
- Properties:
  - hasRole: Role
  - isInvolvedIn: Project

3.15 Integration of Knowledge Graph

Integrating a Knowledge Graph (KG) into Essence and Project Management (PM) ontologies is a comprehensive and meticulous process that demands a well-defined approach to ensure semantic interoperability and enhanced project management insights. This process begins with defining the objectives and scope of the integration. Clear objectives are essential to guide the integration process, which might include goals such as improving data interoperability, enhancing semantic search capabilities, or supporting advanced analytics in project management contexts (Studer et al., 1998).

The next step involves selecting and analyzing existing ontologies. It is crucial to review the Essence and PM ontologies to understand their current structure, including their classes, properties, and instances. This review helps identify any gaps or areas where the KG can add significant value. For instance, examining the Essence ontology might reveal a need for more detailed representations of project phases or team roles, while the PM ontology might benefit from additional contextual information about methodologies and best practices (Hogan et al., 2020).

Selecting or constructing the appropriate knowledge graph is the subsequent step. If a suitable existing KG aligns with the identified gaps and goals, such as DBpedia or Wikidata, it can be integrated. However, if no existing KG meets the specific needs, constructing a custom KG tailored to the Essence and PM ontologies is necessary. This construction involves gathering relevant data sources, defining the schema, and ensuring that the KG is rich in context and accurately represents the domain knowledge required (Euzenat & Shvaiko, 2013).

The process of ontology alignment and mapping is critical to ensure that the integration is seamless. This step involves aligning the concepts in the KG with those in the Essence and PM ontologies. Matching classes, properties, and instances across the ontologies and the KG requires defining mapping rules and equivalence relations. These mappings ensure that entities in the KG correctly correspond to entities in the Essence and PM ontologies, facilitating coherent and meaningful integration (Wache et al., 2001).

Schema integration follows, where the Essence and PM ontologies are extended to incorporate new concepts, properties, and relationships from the KG. Schema matching techniques are employed to ensure that the structure of the KG integrates seamlessly with the existing ontologies. This step might involve extending the Essence ontology to include new project management methodologies represented in the KG or enhancing the PM ontology with detailed descriptions of team roles and responsibilities sourced from the KG (Studer et al., 1998).

Data integration and population are the next crucial steps. Data from the KG needs to be transformed into a format compatible with the Essence and PM ontologies, typically RDF or OWL format. This transformation ensures that the data can be seamlessly integrated and queried within

the ontology framework. Once transformed, the extended ontologies are populated with instances from the KG, ensuring that the data is consistent and accurately reflects the domain knowledge (Hogan et al., 2020).

Consistency checking and validation are essential to maintain the integrity of the integrated ontology. Reasoning tools are used to check for inconsistencies within the integrated ontology, ensuring that the relationships and properties are logically coherent. Validation against predefined criteria ensures that the integrated ontology meets the objectives and scope defined at the outset. This step might involve validating that the new concepts introduced from the KG align with the existing project management frameworks and methodologies in the Essence and PM ontologies (Euzenat & Shvaiko, 2013).

Following consistency checking, ontology evaluation and refinement are necessary to ensure the integrated ontology is fit for purpose. Evaluation metrics such as coverage, coherence, and usability are employed to assess the integrated ontology. Based on the evaluation results, refinements are made to address any identified issues or areas for improvement. This iterative process ensures that the integrated ontology is robust and effectively enhances the semantic richness of the Essence and PM frameworks (Wache et al., 2001).

Implementation and use case testing are critical to validate the practical applicability of the integrated ontology. Implementing the integrated ontology in a real-world scenario or a prototype helps test its functionality and performance. Use case testing with specific project management scenarios ensures that the integrated ontology delivers the expected benefits and enhances decision-making processes. For instance, testing might involve using the integrated ontology to support automated project team composition or to improve semantic search capabilities within a project management tool (Studer et al., 1998).

Documentation and dissemination of the integration process are crucial for transparency and future collaboration. Detailed documentation of the integration process, mapping rules, and any modifications made to the ontologies ensures that the work can be understood and replicated by others. Sharing the integrated ontology with the broader community allows for feedback and

potential collaboration, fostering further improvements and innovations in the field (Hogan et al., 2020).

Finally, ongoing maintenance and updates are necessary to ensure that the integrated ontology remains relevant and up-to-date. Regular updates to reflect new knowledge and changes in the underlying Essence and PM ontologies or the KG are essential. Implementing version control helps track changes and manage different versions of the integrated ontology, ensuring that the ontology evolves in line with advancements in project management practices and knowledge graph technologies (Euzenat & Shvaiko, 2013).

3.16 Extension of Essence SE Standard with PM Components

The Essence framework, formally known as the "Kernel and Language for Software Engineering Methods (Essence) Version 1.2," offers a standardized, practice-independent structure for software engineering methods (Jacobson et al., 2014). This framework encapsulates essential elements necessary for any software engineering endeavor through its primary components: the kernel, the language, and practices. The kernel includes fundamental elements such as "Alpha," "Activity Space," and "Competency," which serve as the foundation for any software development process, while the language component offers the means to describe methods and practices in a standardized way, facilitating communication and understanding across diverse teams and projects.

However, to fully address the complexities of project management (PM), it becomes imperative to integrate additional PM components into the Essence framework. Project management encompasses the management of scope, time, cost, quality, human resources, communication, risk, and procurement within projects, which are crucial for ensuring project success as highlighted by the PM^2. Integrating these elements into the Essence framework not only broadens its applicability but also enhances its robustness, providing a more comprehensive approach to managing software projects.

A thorough review of existing literature reveals various attempts to marry project management practices with software engineering frameworks. (Petersen & Wohlin, 2009) discuss the alignment of agile methodologies with traditional project management practices, showcasing both the benefits and challenges of such integration. Similarly, (Kuhrmann et al., 2018) emphasize the necessity for hybrid approaches that blend agile and traditional PM elements to address the dynamic needs of software projects. These insights underscore the need for a structured approach to extend the Essence framework by incorporating project management components systematically.

To begin this extension, it is essential first to conduct a detailed comparative analysis of the Essence framework and established PM standards, particularly PM^2. PM^2 outlines ten knowledge areas: integration, scope, schedule, cost, quality, resource, communication, risk, procurement, and stakeholder management. By mapping these areas to the Essence kernel elements, we can identify the gaps and opportunities for enhancement. This comparative analysis lays the groundwork for a systematic integration process, ensuring that the extended framework comprehensively covers all critical aspects of project management.

The primary goal of extending the Essence framework is to create a seamless integration of software engineering practices with robust project management methodologies. This involves defining clear objectives for the extension process, such as ensuring that the new components are well-aligned with the existing kernel elements, maintaining the practice-independence of the framework, and enhancing its ability to manage complex software projects effectively. Achieving these objectives requires a detailed plan that outlines specific steps for integrating each PM component into the Essence framework.

One of the first steps in this plan is to develop detailed descriptions and models for each PM component, ensuring that they are compatible with the Essence kernel elements. This involves creating new "Alphas" and "Activity Spaces" that correspond to PM^2's knowledge areas, such as developing an "Alpha" for project scope that includes defining project deliverables, work breakdown structure, and scope validation processes. Similarly, new "Activity Spaces" can be

created for schedule management, cost estimation, quality assurance, resource allocation, risk assessment, and stakeholder communication.

These new elements must then be rigorously tested and validated through case studies and practical applications. This involves applying the extended Essence framework to real-world projects and assessing its effectiveness in managing project scope, time, cost, quality, and other critical aspects. Such validation is crucial to ensure that the new PM components integrate seamlessly with the existing framework and provide tangible benefits in real-world project management scenarios.

Moreover, it is important to gather feedback from practitioners and stakeholders who utilize the extended framework in their projects. Their insights and experiences can provide valuable feedback for further refining and optimizing the new PM components, ensuring that they meet the practical needs of project managers and teams.

In addition to developing and validating new PM components, it is essential to provide comprehensive documentation and training materials for users of the extended Essence framework. This includes detailed guidelines on how to implement and utilize the new PM elements, case studies that illustrate best practices, and training programs to educate project managers and teams on effectively leveraging the extended framework.

The extension process also requires continuous monitoring and improvement to adapt to the evolving needs of project management and software engineering. This involves establishing a feedback loop where users can provide ongoing feedback on the framework's effectiveness, and updates can be made to address new challenges and opportunities in project management.

By incorporating PM components into the Essence framework, we not only enhance its robustness but also provide a more holistic approach to managing software projects. This extension allows project managers to leverage a standardized, practice-independent framework that integrates the best practices of both software engineering and project management, ultimately leading to more successful project outcomes.

Extending the Essence framework to include project management components requires a deep understanding of both domains and a careful mapping of their respective elements. For example, the integration knowledge area in PM^2 can be aligned with the Alpha "Work" in the Essence framework, which encompasses the progress and completion of work items. This alignment allows for a cohesive understanding of how project integration is managed, from defining the project charter to developing the project management plan and directing and managing project work.

Similarly, scope management in PM2 can be mapped to the Alpha "Requirements" in the Essence framework. This involves detailed processes such as collecting requirements, defining scope, creating the work breakdown structure, and validating and controlling scope. By integrating these processes into the Essence framework, we provide a comprehensive view of how project scope is managed from inception to completion.

Schedule management, another critical knowledge area in PM^2, can be integrated into the Essence framework by developing new Activity Spaces focused on time management. This includes defining activities, sequencing them, estimating durations, developing the schedule, and controlling it. These Activity Spaces ensure that project managers have a clear roadmap for managing project timelines within the Essence framework.

Cost management, which involves estimating, budgeting, and controlling costs, can be incorporated by creating new Alphas and Activity Spaces that address financial planning and monitoring. This integration ensures that project managers can effectively manage project budgets and financial performance within the Essence framework.

Quality management, encompassing quality planning, assurance, and control, can be aligned with the Competency element in the Essence framework. This involves defining quality standards, conducting quality assurance activities, and implementing quality control measures. By integrating these processes, the Essence framework ensures that quality is maintained throughout the project lifecycle.

Resource management, which involves planning, acquiring, developing, and managing project teams, can be incorporated into the Essence framework by creating new Alphas for team composition and resource allocation. This integration provides a comprehensive view of how human and physical resources are managed within the project.

Communication management, critical for ensuring effective information flow among project stakeholders, can be mapped to new Activity Spaces focused on stakeholder engagement and communication planning. This includes identifying stakeholders, planning communications, managing stakeholder expectations, and ensuring that information is distributed effectively.

Risk management, which involves identifying, analyzing, and responding to project risks, can be integrated by developing new Alphas and Activity Spaces focused on risk planning, assessment, and mitigation. This integration ensures that project managers can proactively manage potential risks within the Essence framework.

Procurement management, which involves acquiring goods and services from external sources, can be mapped to new Activity Spaces focused on procurement planning, execution, and monitoring. This integration ensures that project procurement processes are effectively managed within the Essence framework.

Stakeholder management, which involves identifying and engaging project stakeholders, can be incorporated by developing new Alphas and Activity Spaces focused on stakeholder analysis, engagement planning, and management. This integration ensures that stakeholder expectations are managed effectively within the project.

The process of extending the Essence framework also involves creating comprehensive guidelines and best practices for integrating project management components. This includes developing detailed documentation on how to implement and utilize the new PM elements, case studies that illustrate successful integrations, and training programs to educate project managers and teams on leveraging the extended framework effectively.

In addition to documentation and training, it is essential to establish a robust support system for users of the extended Essence framework. This includes providing access to online resources, forums, and communities where users can share experiences, ask questions, and seek guidance. Establishing a feedback loop where users can provide ongoing feedback on the framework's effectiveness is also crucial. This feedback can be used to make continuous improvements and updates to the framework, ensuring that it remains relevant and effective in managing modern project challenges.

To validate the extended Essence framework, it is important to conduct pilot projects and case studies that apply the framework to real-world scenarios. These projects should be carefully selected to represent a diverse range of industries and project types, ensuring that the framework's applicability and effectiveness are thoroughly tested. The results of these case studies can provide valuable insights into the framework's strengths and areas for improvement, guiding future refinements and enhancements.

By incorporating PM components into the Essence framework, we create a comprehensive tool that addresses the full spectrum of project management and software engineering needs. This integration not only enhances the framework's robustness but also provides a unified approach to managing projects, ensuring that all critical aspects of project management are covered.

The extended Essence framework offers several key benefits for project managers and teams. Firstly, it provides a standardized, practice-independent approach to project management, ensuring consistency and comparability across different projects and teams. This standardization facilitates communication and collaboration, enabling project managers to effectively manage diverse teams and stakeholders.

Secondly, the integrated framework enhances the ability to manage complex projects by providing a comprehensive view of all project management processes. This holistic approach ensures that project managers can effectively manage scope, time, cost, quality, resources, communication, risk, procurement, and stakeholder engagement, leading to more successful project outcomes.

Thirdly, the extended Essence framework provides a robust foundation for continuous improvement. By incorporating feedback loops and ongoing monitoring, the framework can be continuously refined and updated to address new challenges and opportunities in project management. This adaptability ensures that the framework remains relevant and effective in an ever-changing project management landscape.

4 BUILDING PROJONTO: THE PROJECT ONTOLOGY

Project management, as a discipline, has seen significant evolution, adapting to the ever-changing landscape of technology and business needs. Ontologies in project management are essential tools that aid in the systematic organization and retrieval of knowledge, enabling efficient project execution and decision-making. Existing ontologies, such as the Business Technology Management (BTM) ontology, the Essence framework for software engineering, the PM² methodology, and the BPMN 2.0 Based Ontology (BBO) for business process representation, have each contributed uniquely to the field. However, they also have limitations when applied in isolation, particularly concerning process customization and semantic integration.

The BTM ontology is primarily concerned with aligning business and technology goals. It emphasizes strategic alignment, value management, and governance. While it provides a solid framework for managing technology-related projects, its scope is often too narrow to cover the diverse needs of general project management practices.

The Essence framework focuses on the core elements of software engineering methods. It abstracts the commonalities among different methodologies, allowing for flexibility and adaptability. However, its application is predominantly limited to software engineering, making it less suitable for projects outside the software domain.

Developed by the European Commission, PM² offers a comprehensive project management methodology that is adaptable to a variety of projects. It includes best practices, processes, and templates but lacks the semantic depth needed for automation and advanced customization.

The BPMN 2.0 Based Ontology is designed for representing business processes. It facilitates process modeling and execution by providing a detailed semantic framework. While it excels in business process representation, it does not fully address the needs of project management, particularly in terms of integrating different management methodologies.

To address the limitations of existing ontologies and provide a robust solution for process customization, we propose ProjOnto—a new project ontology that synthesizes elements from BTM, Essence, PM², and BBO. ProjOnto aims to offer a comprehensive and adaptable framework that enhances the management of diverse projects by leveraging the strengths of each existing ontology.

ProjOnto's conceptual foundation is built on the recognition that project management is inherently multidisciplinary. It integrates the strategic alignment focus of BTM, the adaptable core elements of Essence, the comprehensive methodology of PM², and the detailed process representation of BBO. This integration ensures that ProjOnto is versatile and capable of addressing the diverse needs of various project types.

One of the primary strengths of ProjOnto is its emphasis on semantic integration. By harmonizing the terminologies and concepts from BTM, Essence, PM², and BBO, ProjOnto creates a unified semantic framework. This framework facilitates seamless communication and knowledge exchange among different stakeholders, tools, and systems involved in project management.

ProjOnto is designed to support extensive customization and flexibility. It allows project managers to tailor processes and methodologies to the specific needs of their projects. By incorporating the adaptable elements of the Essence framework, ProjOnto ensures that project methodologies can be adjusted dynamically, accommodating changes in project scope, requirements, and environment.

The core components of ProjOnto include roles, processes, artifacts, and relations. Each component is designed to encapsulate the essential elements from the integrated ontologies:

ProjOnto defines a comprehensive set of roles that encompass the strategic, managerial, and operational aspects of project management. These roles are derived from the BTM and PM² ontologies, ensuring that both business and project management perspectives are covered.

The process component of ProjOnto integrates the detailed process representation capabilities of BBO with the flexible process frameworks of Essence and PM². This integration enables precise modeling, execution, and customization of project processes.

Artifacts in ProjOnto include all project-related documents, templates, and deliverables. By incorporating elements from the PM² methodology and the Essence framework, ProjOnto ensures that artifacts are well-defined, standardized, and easily customizable.

The relations component of ProjOnto captures the dependencies and interactions between roles, processes, and artifacts. This comprehensive relational framework is crucial for understanding the complex dynamics of project management and facilitating effective decision-making.

One of the most significant advantages of ProjOnto is its ability to facilitate process customization. In traditional project management, customizing processes to fit the unique needs of a project can be challenging and time-consuming. ProjOnto addresses this challenge by providing a robust semantic framework that supports dynamic process customization.

With ProjOnto, project managers can model processes dynamically, adjusting them to meet specific project requirements. The integration of BBO's process representation capabilities allows for detailed and precise modeling, while the adaptability of the Essence framework ensures that these models can be easily customized.

ProjOnto leverages semantic integration to enable automated process adjustments. By using predefined rules and algorithms, ProjOnto can automatically suggest process modifications based on changes in project parameters. This capability significantly reduces the time and effort required for process customization, allowing project managers to focus on higher-level strategic decisions.

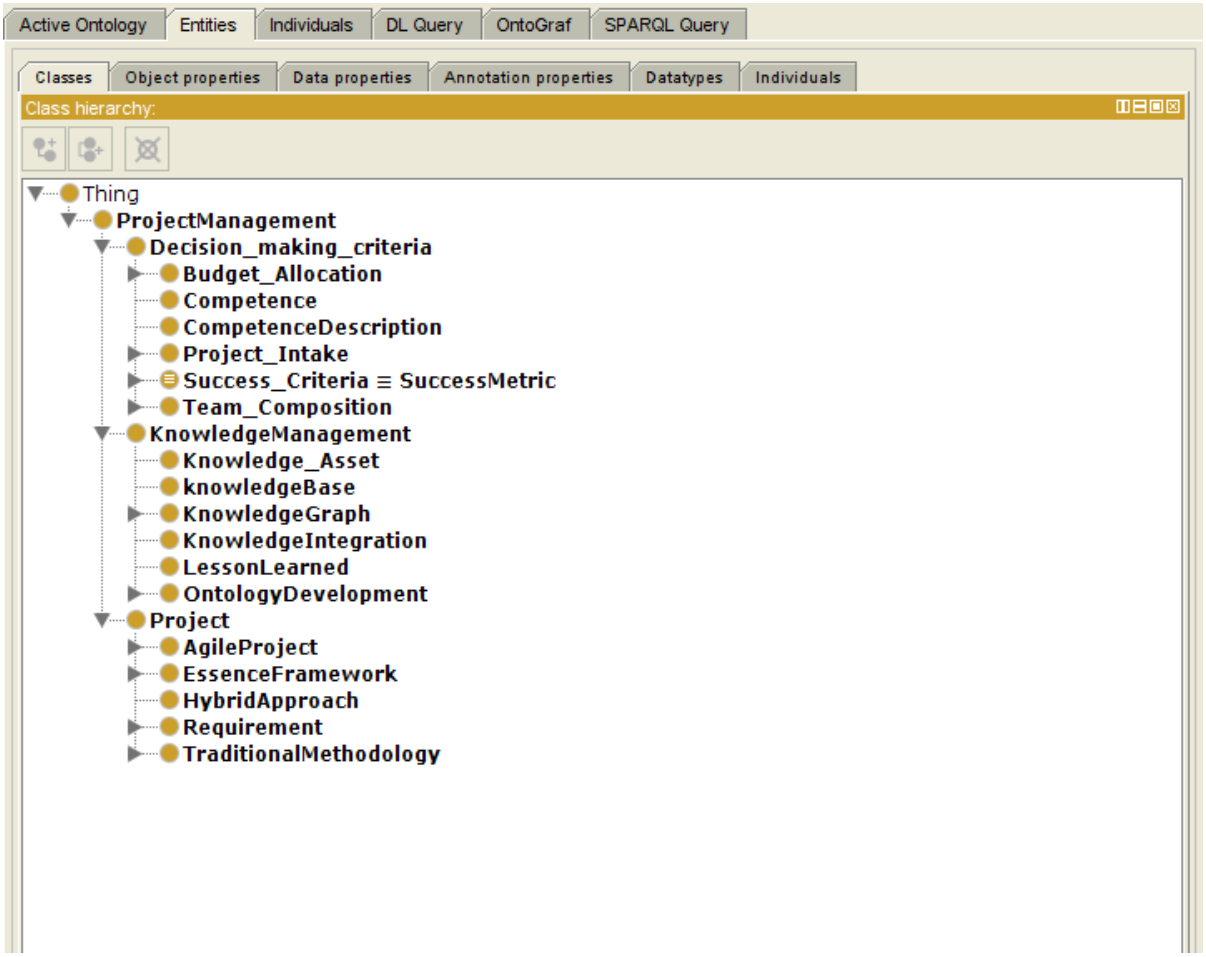
The unified semantic framework of ProjOnto enhances collaboration and communication among project stakeholders. By providing a common language and set of concepts, ProjOnto ensures that all stakeholders have a shared understanding of project processes and requirements. This shared understanding is crucial for effective collaboration and decision-making, particularly in complex projects involving multiple teams and organizations.

4.1.1 ProjOnto framework

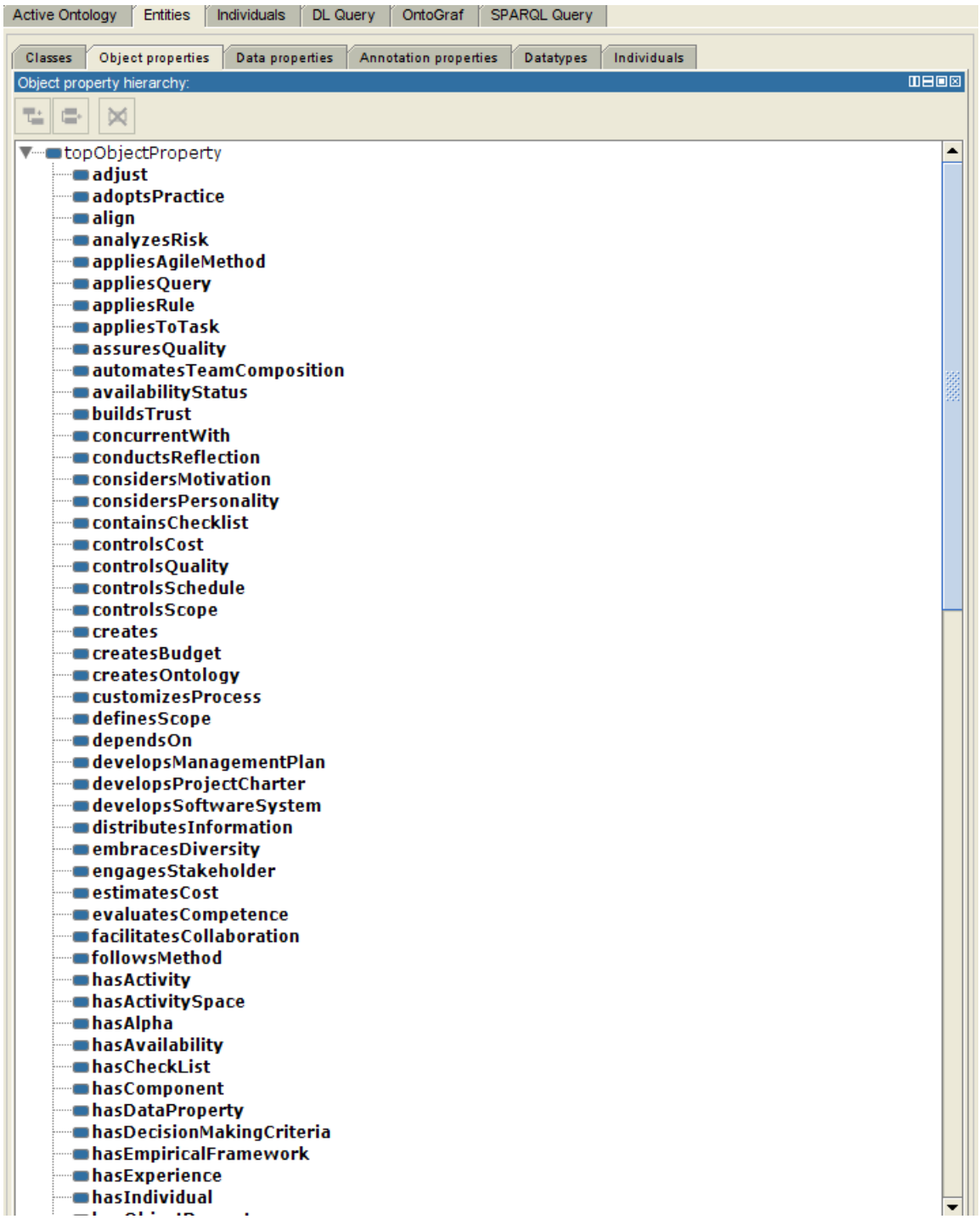
The ProjOnto Framework offers a comprehensive structure for understanding and implementing the various components, processes, and methodologies involved in project management. It integrates foundational knowledge, practical methodologies, and agile practices, BTM and Essence to provide a robust system for managing projects in diverse domains. The full framework is detailed across Table 35 to Table 39 in the appendix section, covering classes, object properties, data properties, and individual examples that illustrate practical applications.

4.1.2 Building ProjOnto artefacts using Protégé

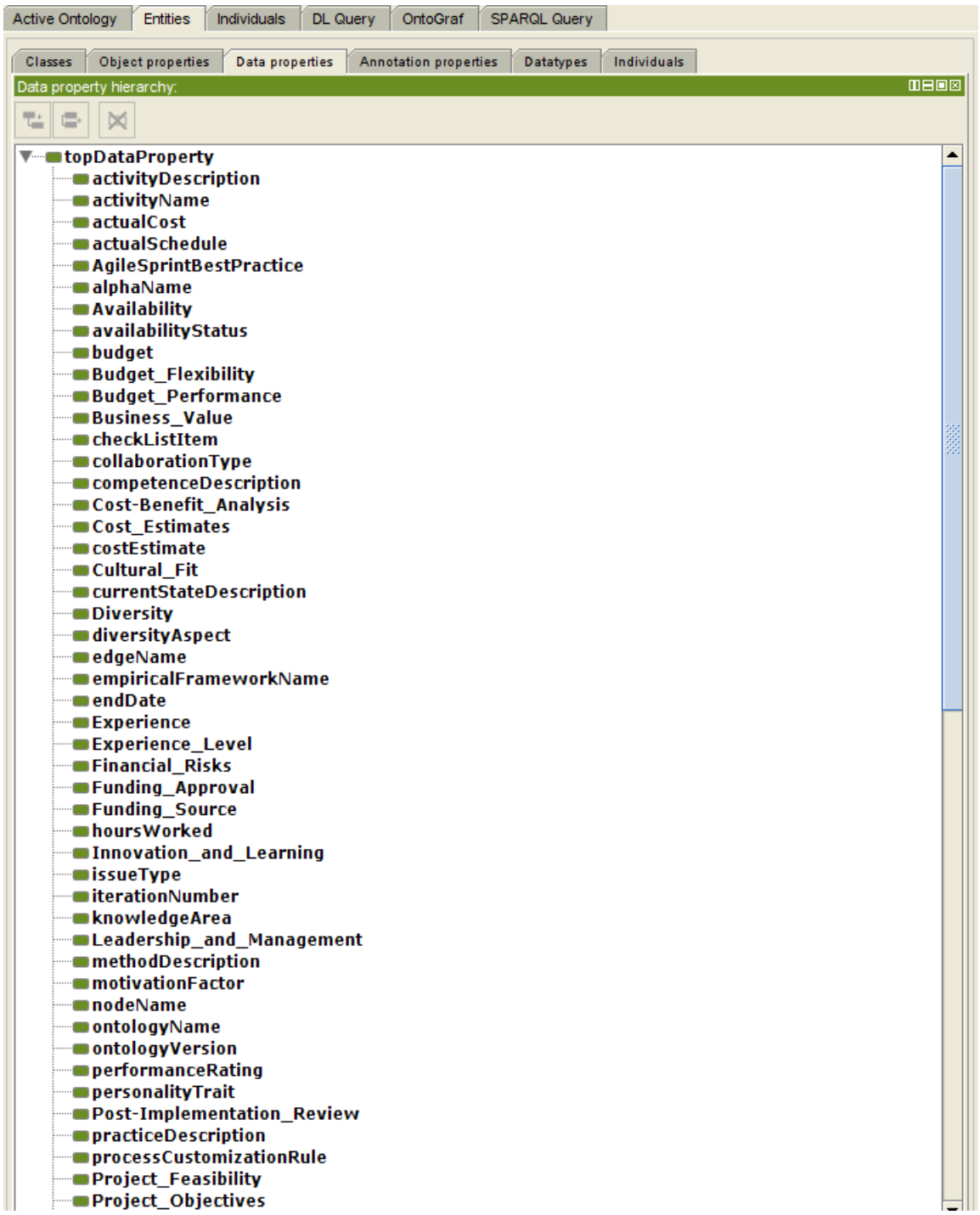
4.1.2.1 Classes and subclasses



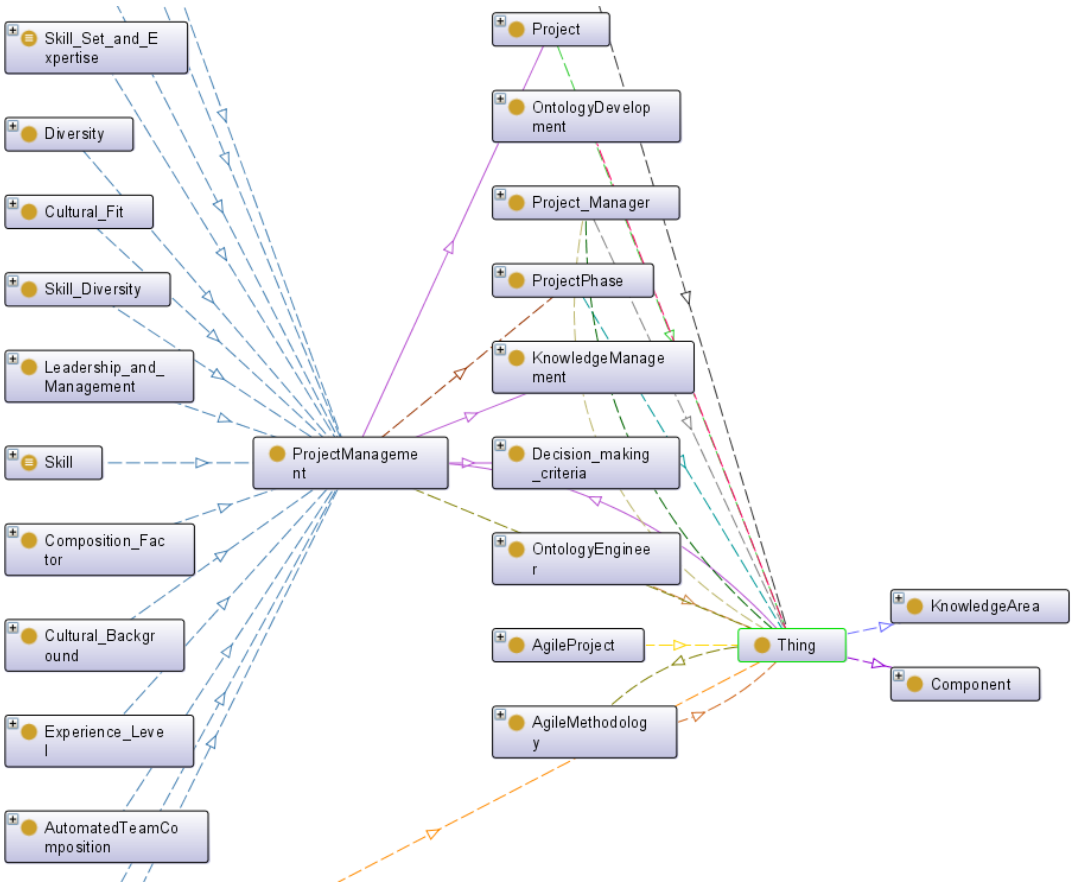
4.1.2.2 Object Properties



4.1.2.3 Data Properties



4.1.2.4 ProjOnto Graph



ProjOnto is located here:

https://github.com/diomam/ProjOnto/blob/main/ProjOnto%201.0

5 EVALUATION

Testing an ontology with real-world cases is crucial for several reasons, particularly in complex domains such as Agile project management in FinTech, HealthTech, and software development, as demonstrated by the three case studies of TechInnovate, FinTech Corp, and HealthTech Solutions.

Ensuring Ontology Relevance and Accuracy: Ontologies are designed to represent knowledge domains by defining the relationships between concepts, classes, and properties. However, the theoretical development of an ontology might not fully capture the nuances of real-world applications. By testing an ontology with real-world cases, such as the restructuring of Agile teams in FinTech Corp or the integration of domain knowledge in HealthTech Solutions, we can validate whether the ontology accurately reflects the complexities and requirements of the domain. Real-world testing ensures that the ontology remains relevant and accurately represents the relationships and constraints that exist in actual project environments, thereby avoiding the risk of creating an abstract model that fails to address practical challenges.

Validating Practical Utility: The ultimate goal of an ontology is to facilitate decision-making, enhance collaboration, and improve processes within a specific domain. For instance, in the TechInnovate case study, the ontology should help in optimizing team composition by matching roles and skills effectively. Testing the ontology in real-world scenarios allows researchers and practitioners to assess whether it provides tangible benefits, such as improved team performance or better project outcomes. If the ontology cannot be practically applied or does not yield the expected improvements, it may require revision or refinement. Real-world testing, therefore, serves as a critical step in validating the practical utility of the ontology.

Identifying Gaps and Enhancing Completeness: Real-world cases often present unforeseen challenges that theoretical models do not account for. For example, the need

for continuous integration skills in HealthTech Solutions or the importance of domain-specific expertise in FinTech Corp might highlight gaps in the ontology that were not initially considered. Testing with actual data and scenarios allows these gaps to be identified and addressed. This iterative process of refinement ensures that the ontology becomes more comprehensive and capable of addressing all relevant aspects of the domain.

Improving Semantic Interoperability: In complex environments, especially in industries like healthcare and finance, interoperability between different systems and teams is essential. Ontologies play a critical role in enabling semantic interoperability by providing a shared vocabulary and structure. Testing the ontology in real-world contexts, as in the HealthTech Solutions case, ensures that it can effectively facilitate communication and data exchange between different stakeholders and systems. Without real-world testing, the ontology might fail to achieve this interoperability, leading to miscommunication and inefficiencies.

Enhancing Stakeholder Acceptance and Adoption: For an ontology to be widely adopted, it must gain the trust and acceptance of its users. Stakeholders are more likely to adopt an ontology that has been rigorously tested and proven to work in real-world scenarios relevant to their needs. The case studies of FinTech Corp and HealthTech Solutions demonstrate that when an ontology is tested in practice and shown to improve outcomes, stakeholders are more inclined to trust and utilize it. This acceptance is crucial for the ontology's long-term success and integration into organizational processes.

Testing an ontology with real-world cases is essential for validating its relevance, practical utility, and completeness. It ensures that the ontology accurately represents the complexities of the domain, facilitates semantic interoperability, and gains stakeholder acceptance. By applying an ontology to real-world scenarios, as illustrated in the case studies of TechInnovate, FinTech Corp, and HealthTech Solutions, researchers and practitioners can refine the ontology to better meet the demands of practical applications, ultimately leading to more effective and efficient project management and decision-making processes.

5.1 Case Study 1: Agile Software Development Team Composition at TechInnovate

TechInnovate, a leading software development company renowned for its innovative approach and cutting-edge solutions, recognized the need to enhance their project delivery processes. With the increasing demand for high-quality software delivered within increasingly tight deadlines, the company decided to adopt Agile methodologies. Agile, known for its iterative approach and emphasis on collaboration, was seen as the ideal framework to help TechInnovate form high-performing teams capable of meeting the company's ambitious goals.

The decision to transition to Agile was driven by the desire to improve flexibility, speed, and customer satisfaction in software delivery. The company aimed to create teams that could adapt quickly to changing requirements, collaborate effectively, and deliver products that met the high standards expected by their clients. This shift also aligned with TechInnovate's commitment to fostering a culture of continuous improvement, where teams are empowered to experiment, learn, and evolve.

TechInnovate's leadership was particularly interested in understanding how to form Agile teams that could not only deliver high-quality software but also do so consistently, regardless of the complexity or scale of the project. This case study examines one of the company's most significant software development projects, where Agile methodologies were fully implemented, focusing on team composition and its impact on project outcomes.

The study centered on a critical project undertaken by TechInnovate: the development of an innovative customer relationship management (CRM) system. This project was strategically important for the company, as it was designed to create a CRM system that would stand out in a highly competitive market by offering unique features and seamless user experiences.

The scope of the study was limited to the Agile team composition and its direct impact on the project's success. While Agile practices cover a broad range of processes and

principles, this study specifically focused on the roles, skills, and dynamics within the teams working on the CRM project. The goal was to identify the characteristics of team composition that contributed most significantly to the project's success, including the mix of roles, the diversity of skills, and the interpersonal dynamics among team members.

By narrowing the focus to team composition, TechInnovate aimed to derive actionable insights that could be applied to future projects. The findings from this study were intended to inform the company's broader strategy on team formation, not only within the software development department but across the organization.

The primary objective of this study was to identify the optimal combination of skills and roles within Agile teams to maximize productivity, quality, and overall project success. TechInnovate sought to understand the key factors that contribute to the effectiveness of Agile teams, particularly in the context of complex software development projects.
This involved analyzing the specific roles that were most critical to the project's success, such as developers, testers, UX designers, and product owners, and how these roles interacted within the Agile framework. Additionally, the study explored the importance of soft skills, such as communication and collaboration, which are often cited as crucial in Agile environments.
Another key objective was to assess how regular skill assessments and training programs influenced team performance. TechInnovate wanted to determine whether continuous development and upskilling of team members had a measurable impact on the quality and speed of software delivery.

Ultimately, the study aimed to provide a clear, evidence-based understanding of how to structure Agile teams for maximum efficiency and effectiveness. These insights were intended to guide the company in refining its team formation processes, ensuring that each project was staffed with the right combination of skills and expertise.
The CRM project was a resounding success, and the study yielded several key findings that have since been integrated into TechInnovate's standard practices for team composition.

First, it was found that cross-functional teams with diverse skill sets performed significantly better than more homogeneous teams. Teams that included a mix of developers, testers, UX designers, and product owners were able to address a broader range of challenges and deliver a more polished final product. The diversity of expertise within the team allowed for more creative problem-solving and a more holistic approach to development.

Second, the study highlighted the critical importance of communication and collaboration skills within the team. Members who were not only technically proficient but also strong communicators and collaborators contributed more effectively to the team's success. These skills facilitated smoother interactions, quicker resolution of issues, and a more cohesive team environment.

Finally, the study underscored the value of regular skill assessments and training programs. Teams that participated in ongoing development and upskilling activities were better equipped to handle the project's challenges and were more adaptable to changes in scope or direction. This finding led to the implementation of more structured training programs within TechInnovate, ensuring that all team members were continuously improving their skills.

We apply the findings from the TechInnovate case study to the ontology elements (class, subclass, object property, data property, individuals) to provide a comprehensive Action Design Research (ADR) intervention for project management and business technology management semantic integration. The goal is to demonstrate how semantic technologies can enhance automated project team composition using ontologies and knowledge graphs.

5.1.1 Ontology Elements in the Context of TechInnovate

5.1.1.1 Classes and Subclasses

- Project Team: Represents the overall team working on the project.

- Developer: Subclass representing developers in the team.
- Tester: Subclass representing testers in the team.
- UX Designer: Subclass representing UX designers in the team.
- Product Owner: Subclass representing product owners in the team.

- Skills**:** Represents the various skills possessed by team members.
  - Technical Skills: Subclass representing technical capabilities.
  - Communication Skills: Subclass representing communication abilities.
  - Collaboration Skills: Subclass representing the ability to work well with others.

5.1.1.2 Object Properties

- hasSkill: Links a Project Team member to their Skills.
- reportsTo: Links team members to their respective supervisors or managers.
- worksOn: Links team members to the specific tasks or modules they are responsible for within the project.

5.1.1.3 Data Properties

- skillLevel: Indicates the proficiency level of a skill (e.g., beginner, intermediate, expert).
- roleDescription: Provides a description of the role a team member plays in the project.
- taskDescription: Details the specific tasks assigned to a team member.
- startDate: The date when a team member starts working on the project.
- endDate: The date when a team member completes their tasks.

5.1.1.4 Individuals

- JohnDoe: An instance of Developer with specific skills and assigned tasks.
- JaneSmith: An instance of Tester with specific skills and assigned tasks.
- AliceJones: An instance of UX Designer with specific skills and assigned tasks.
- BobBrown: An instance of Product Owner with specific skills and assigned tasks.

5.1.2 Applying TechInnovate case to ProjOnto

To effectively structure the ontology in the context of the TechInnovate case study, it is essential to proceed systematically through a series of well-defined steps, each building on the previous one. The first step involves defining the classes and subclasses, which are based on the roles and skills identified in the case study. For instance, it is crucial to create distinct classes for each role, such as Developer, Tester, UX Designer, and Product Owner. These roles represent the various responsibilities within the TechInnovate team and form the foundation of the ontology. By categorizing the different roles into specific classes, the ontology gains clarity, allowing for a more structured representation of the team's composition.

Following the establishment of classes and subclasses, the second step focuses on establishing object properties. Object properties are instrumental in defining the relationships between individuals, their skills, roles, and the tasks they perform within the project. For example, the property "hasSkill" could be employed to link a Developer like JohnDoe to their specific technical skills. This connection not only illustrates the capabilities of each individual but also allows for a comprehensive understanding of how these skills relate to their roles and the tasks they are responsible for. Object properties thus serve as the connective tissue within the ontology, ensuring that all elements are cohesively linked.

The third step involves defining data properties to capture more detailed information about each individual within the ontology. Data properties are used to record specific attributes, such as the skill level of an individual. For instance, JohnDoe's technical skills might be assigned a "skillLevel" data property, providing a quantifiable measure of his expertise.

This detailed information is crucial as it enriches the ontology, making it more informative and functional. By integrating data properties, the ontology can offer insights into the depth of each team member's abilities, thereby facilitating more informed decision-making regarding task assignments and project planning.

Once the framework of classes, object properties, and data properties is in place, the next step is to populate the ontology with individuals who represent the actual team members from the TechInnovate case study. This involves creating individual instances such as JohnDoe, who would be characterized by various attributes, including his role as a Developer, his skill level, and the specific tasks he is assigned. By populating the ontology with these individual instances, it becomes a living document that accurately reflects the real-world structure and dynamics of the TechInnovate team. This step is critical as it transitions the ontology from a theoretical framework to a practical tool that can be used for analysis and project management.

In the process of building this ontology, it is also crucial to consider the overarching problem formulation. One of the primary motivations for developing such an ontology is to address the need for optimized team composition to enhance project success, particularly in Agile environments. Agile methodologies require a flexible and responsive team structure, which traditional project team composition methods often fail to provide. By recognizing the limitations of these traditional methods, the potential benefits of applying semantic technologies, such as the ontology described here, become evident. Semantic technologies enable a more nuanced and dynamic approach to team composition, allowing for a better alignment of skills, roles, and tasks in response to the evolving demands of Agile projects.

The evaluation phase plays a crucial role in ensuring that the ontology remains relevant and accurate over time. One aspect of this evaluation is skill assessment, where the skills of team members are regularly evaluated and updated within the ontology to reflect any changes or improvements. This ongoing process ensures that the ontology remains a true representation of the team's current capabilities. Another aspect of evaluation is performance monitoring, which leverages the knowledge graph to monitor team

performance. By analyzing the relationships and data within the ontology, project managers can identify areas for improvement and take proactive measures to enhance team efficiency and effectiveness.

Finally, a continuous feedback loop should be implemented to refine both the ontology and the knowledge graph based on real-world project outcomes. This feedback loop is essential for maintaining the accuracy and relevance of the ontology as the project progresses. By continuously refining the ontology, it can evolve alongside the project, ensuring that it remains a valuable tool for managing team dynamics and project success. In conclusion, by following these steps—defining classes and subclasses, establishing object properties, defining data properties, populating the ontology with individuals, and continuously evaluating and refining the ontology—one can create a robust and dynamic representation of the TechInnovate team, which can be effectively used for project management and performance enhancement. This approach not only addresses the need for optimized team composition but also highlights the transformative potential of semantic technologies in modern project management, particularly within Agile environments.

5.1.3 Applying Semantic Technologies for Automated Team Composition

To effectively implement an advanced ontology-based framework within the TechInnovate case study, it is crucial to incorporate mechanisms for automated role assignment, dynamic team reconfiguration, and enhanced collaboration. These mechanisms not only leverage the power of semantic technologies but also address key challenges in modern project management, particularly in Agile environments where flexibility and responsiveness are paramount.

The first mechanism, automated role assignment, uses semantic reasoning to automatically assign roles and tasks to team members based on their skills, experience, and project needs. This approach streamlines the process of task distribution, ensuring that each team member is utilized to their fullest potential. For instance, JohnDoe, a Developer with advanced technical skills, can be automatically assigned to tasks that specifically require his

expertise. By analyzing the skills and past experiences of each individual, the ontology can make informed decisions about role assignments, reducing the time and effort typically required for manual task allocation. This not only enhances efficiency but also ensures that the right people are working on the right tasks, thereby improving overall project outcomes.

The second mechanism involves dynamic team reconfiguration, which is essential in Agile projects where requirements and project scopes often change rapidly. Dynamic reconfiguration allows teams to be reorganized in real-time in response to these changes, ensuring that the project remains on track and that resources are optimally utilized. For example, if a new module in the project suddenly requires immediate testing expertise, the ontology can automatically reassign JaneSmith, who has the necessary skills, to this task. This flexibility is crucial in maintaining momentum and meeting deadlines, as it ensures that the team composition is always aligned with the current project needs. By enabling such agility, the ontology supports a more resilient and adaptable project management approach.

Enhanced collaboration is the third mechanism that the ontology framework facilitates. Clear definitions of roles and responsibilities within the ontology help improve communication and collaboration among team members. When roles and tasks are explicitly defined and linked within the ontology, it reduces ambiguities and ensures that everyone understands their responsibilities and how their work fits into the larger project context. For instance, by linking AliceJones, a UX Designer, with BobBrown, the Product Owner, the ontology ensures that both team members are aligned on user experience requirements. This alignment is critical in preventing misunderstandings and ensuring that the product meets the desired quality standards. Moreover, by fostering better collaboration, the ontology helps build a cohesive team environment, which is essential for successful project execution.

Incorporating these mechanisms into the TechInnovate case study's ontology not only addresses specific challenges but also demonstrates the broader potential of semantic technologies in enhancing project management processes. Automated role assignment

ensures that skills are matched with tasks efficiently, dynamic team reconfiguration allows for responsive project management, and enhanced collaboration fosters a more integrated and communicative team. Together, these mechanisms create a robust framework that supports the dynamic and fast-paced nature of modern projects, particularly in Agile environments.

The use of semantic reasoning to automate role assignments is especially transformative, as it reduces the cognitive load on project managers and allows for more strategic oversight. Dynamic team reconfiguration, on the other hand, ensures that the project can adapt to changes without significant disruptions, a key requirement in Agile methodologies. Lastly, enhanced collaboration through clear role definitions strengthens team cohesion and ensures that project goals are consistently met.

By integrating automated role assignment, dynamic team reconfiguration, and enhanced collaboration into the ontology framework, project managers can leverage semantic technologies to create a more efficient, flexible, and communicative team environment. This approach not only improves project outcomes but also aligns with the principles of Agile project management, where adaptability and collaboration are crucial. Through the use of these advanced mechanisms, the TechInnovate case study demonstrates the significant advantages of applying semantic technologies in managing complex, evolving projects.

The application of the TechInnovate case study to ontology elements demonstrates the potential of semantic technologies in enhancing project team composition. By leveraging ontologies and knowledge graphs, organizations can achieve more effective and adaptable project management practices, leading to higher productivity, quality, and project success. This ADR intervention provides a structured approach to integrating semantic technologies in Agile project environments, offering a blueprint for future implementations.

5.2 Case Study 2: Agile Transformation in FinTech Corp

FinTech Corp, a leading financial technology company, embarked on an Agile transformation with the goal of enhancing its software delivery processes. This transformation focused on restructuring project teams to better align with Agile principles, emphasizing flexibility, collaboration, and rapid delivery. The case study specifically examined the restructuring process of three major product development teams tasked with creating a new payment processing system, a critical project for the company's growth strategy.

The primary objective was to analyze the impact of team composition on the outcomes of Agile projects, with a particular focus on identifying the optimal mix of skills and roles necessary for success. The study sought to understand how the balance between technical expertise and domain-specific knowledge within teams influenced project efficiency and overall performance.

The results of the Agile transformation were highly positive. One of the most significant outcomes was the enhancement of team collaboration and faster decision-making processes. The restructuring fostered a more cohesive work environment where team members could communicate more effectively and make quicker decisions, which is essential in Agile methodologies that rely on iterative progress and continuous feedback. Additionally, the study found that well-balanced teams, combining a mix of technical and domain-specific expertise, were more successful in delivering high-quality products on time. This balance allowed teams to address both the technical challenges of software development and the specific needs of the financial industry, leading to more robust and reliable products.

Furthermore, the transformation positively impacted employee satisfaction and reduced turnover rates. By empowering team members and giving them more autonomy in their roles, FinTech Corp created a work environment where employees felt more valued and engaged. This empowerment not only improved job satisfaction but also encouraged team members to take ownership of their work, leading to higher levels of motivation and productivity.

Overall, FinTech Corp's Agile transformation successfully demonstrated the importance of carefully structured teams in achieving project success. The restructuring process not only improved software delivery outcomes but also fostered a more collaborative and satisfying workplace culture, contributing to the company's long-term success.

This section applies the findings from the FinTech Corp case study to ontology elements (class, subclass, object property, data property, individuals) to develop a detailed Action Design Research (ADR) intervention. The intervention is designed to enhance team composition and project outcomes during Agile transformations in financial technology companies using semantic technologies.

5.2.1 Ontology Elements in the Context of FinTech Corp

5.2.1.1 Classes and Subclasses

- Agile Project Team: Represents the restructured teams working on the Agile transformation.
    - Technical Expert: Subclass representing team members with technical skills such as software development, DevOps, and systems architecture.
    - Domain Expert: Subclass representing team members with specific knowledge of financial technologies, regulatory requirements, and payment processing.
    - Agile Coach: Subclass representing individuals who guide teams in Agile practices and principles.

- Skills: Represents the various skills required for successful Agile transformation.
    - Technical Skills: Subclass representing expertise in software development, DevOps, and related areas.
    - Domain-Specific Skills: Subclass representing expertise in financial technologies, compliance, and industry-specific knowledge.

- Leadership Skills: Subclass representing the ability to lead, mentor, and facilitate Agile practices.

5.2.1.2 Object Properties

- hasSkill: Links an Agile Project Team member to their Skills.
- isPartOf: Links team members to the specific product development team they belong to.
- facilitates: Links the Agile Coach to the teams they are mentoring or guiding.
- collaboratesWith: Defines the collaborative relationships between team members from different domains or technical backgrounds.

5.2.1.3 Data Properties

- skillLevel: Indicates the proficiency level of a skill (e.g., novice, intermediate, expert).
- experienceYears: Captures the number of years of experience a team member has in their field.
- teamRole: Describes the role of a team member within the Agile team (e.g., Developer, Product Owner, Domain Expert).
- projectName: Specifies the project a team member is working on.
- satisfactionScore: A numeric value reflecting employee satisfaction and morale within the team.

5.2.1.4 Individuals

- EmilyWhite: An instance of Technical Expert with specific technical skills in payment processing systems.

- MichaelGreen: An instance of Domain Expert with expertise in financial regulations and compliance.
- SarahBrown: An instance of Agile Coach responsible for guiding the Agile transformation.
- TeamAlpha: An instance representing one of the product development teams involved in the payment processing system project.

5.2.2 Applying ADR to FinTech Corp case

FinTech Corp, a prominent financial technology company, recognized the need to overhaul its software delivery processes to remain competitive in a rapidly evolving industry. In response, the company embarked on an Agile transformation, a strategic initiative aimed at enhancing flexibility, collaboration, and speed in product development. Agile methodologies, which emphasize iterative progress, customer collaboration, and adaptability, were deemed particularly well-suited to address the challenges faced by FinTech Corp in delivering complex financial software solutions. The focus of this transformation was on restructuring project teams to better align with Agile principles, ensuring that teams were optimally composed to meet the demands of high-paced, dynamic project environments.

This case study examines the restructuring process undertaken by FinTech Corp, focusing specifically on three major product development teams that were central to the creation of a new payment processing system. The payment processing system was a critical project within the company's portfolio, designed to enhance its service offerings in a highly competitive market. The study aimed to analyze the impact of team composition on the outcomes of Agile projects, with a particular emphasis on the roles, skills, and expertise required to maximize productivity, quality, and project success.

The scope of this case study was deliberately narrow, concentrating on the restructuring of three key product development teams within FinTech Corp. These teams were responsible

for the design, development, and deployment of the new payment processing system. The decision to focus on these specific teams was driven by their strategic importance to the company and the critical nature of the project they were working on. By examining these teams in detail, the study sought to provide insights into the broader implications of Agile team composition for the company as a whole.

The restructuring process involved a thorough analysis of existing team structures, workflows, and communication patterns. FinTech Corp's leadership recognized that successful Agile transformation required more than just the adoption of new methodologies; it necessitated a fundamental shift in how teams were organized and how they operated on a day-to-day basis. The study, therefore, focused on key aspects of team composition, including the balance between technical and domain-specific expertise, the distribution of roles and responsibilities, and the integration of Agile practices into the team's workflow.

The primary objective of this study was to understand how team composition influences the success of Agile projects at FinTech Corp. Specifically, the study aimed to identify the optimal mix of skills and roles required to maximize productivity, quality, and project success in an Agile environment. The research sought to answer several key questions: What combination of technical and domain-specific expertise is most effective in an Agile team? How does the distribution of roles within a team affect collaboration and decision-making? And what impact does the restructuring of teams have on employee satisfaction and retention?

Another important objective was to assess the broader organizational impact of the Agile transformation. While the focus was on specific teams, the findings were intended to inform company-wide strategies for Agile adoption. By identifying best practices and potential challenges in team composition, the study aimed to provide actionable insights that could be applied across the organization, helping FinTech Corp achieve its long-term goals of improved software delivery and increased competitiveness in the financial technology sector.

The Agile transformation at FinTech Corp yielded several significant outcomes, all of which contributed to the overall success of the new payment processing system project. One of the most notable results was the enhancement of team collaboration and faster decision-making processes. The restructuring of teams to better align with Agile principles facilitated more effective communication among team members, allowing for quicker resolution of issues and more agile responses to changing project requirements. This was particularly important in the context of the payment processing system, where rapid adaptation to new customer needs and market conditions was crucial.

The study also found that well-balanced teams, comprising a mix of technical and domain-specific expertise, were more successful in delivering high-quality products on time. Teams that included members with deep technical knowledge, such as software developers and systems architects, alongside those with strong domain expertise in finance, such as business analysts and product owners, were better equipped to address the multifaceted challenges of developing a complex financial software product. This balance allowed teams to navigate both the technical complexities of software development and the specific regulatory and operational requirements of the financial industry, resulting in a more robust and reliable payment processing system.

Another key finding was the positive impact of the Agile transformation on employee satisfaction and retention. By empowering team members and giving them greater autonomy in their roles, FinTech Corp created a work environment where employees felt more valued and engaged. This empowerment was facilitated by the Agile emphasis on self-organizing teams, where members are encouraged to take ownership of their work and collaborate closely with their peers. The study found that this approach not only improved job satisfaction but also reduced turnover rates, as employees were more likely to remain with the company when they felt their contributions were recognized and valued.

The positive effects of the Agile transformation extended beyond individual teams to the organization as a whole. The successful restructuring of the three product development teams served as a model for other teams within FinTech Corp, demonstrating the benefits

of adopting Agile practices and providing a blueprint for further Agile adoption across the company. The lessons learned from this case study were subsequently integrated into FinTech Corp's broader Agile strategy, with the company making significant investments in training and development to ensure that all teams were equipped with the skills and knowledge needed to thrive in an Agile environment.

The Agile transformation at FinTech Corp was a resounding success, leading to improved software delivery processes, enhanced team collaboration, and greater employee satisfaction. The restructuring of project teams to better align with Agile principles proved to be a key factor in the success of the new payment processing system project. By carefully balancing technical and domain-specific expertise within teams, and by fostering a culture of collaboration and empowerment, FinTech Corp was able to achieve its goals of faster, more efficient software delivery and increased competitiveness in the financial technology market.

This case study underscores the importance of team composition in Agile projects, highlighting the need for organizations to carefully consider the roles, skills, and dynamics within their teams when undertaking Agile transformations. The findings from this study provide valuable insights for other companies in the financial technology sector, and beyond, as they navigate their own Agile journeys. The success of FinTech Corp's Agile transformation serves as a testament to the power of Agile methodologies to drive meaningful change and deliver tangible business results.

In the context of Agile transformations, the application of semantic technologies offers a promising approach to optimizing team composition. By leveraging ontologies and knowledge graphs, organizations can enhance the efficiency and effectiveness of team formation and management. This section explores how FinTech Corp has integrated semantic reasoning into its Agile processes, focusing on optimized role assignment, dynamic team adjustments, and enhanced collaboration.

Semantic technologies enable the automated assignment of roles and tasks to team members by analyzing their skills, experience, and the specific requirements of the project. Through semantic reasoning, the system can match team members to roles that align with their competencies, ensuring that each individual is optimally positioned to contribute to the project's success.

For example, in the development of FinTech Corp's new payment processing system, Emily White, identified as a Technical Expert, was automatically assigned to tasks that required advanced knowledge of payment processing systems. The ontology recognized her expertise in this domain and allocated her to critical technical tasks that benefited from her specialized skills. This approach not only streamlined the role assignment process but also ensured that the team's technical challenges were addressed by the most qualified personnel.

The use of semantic reasoning for role assignment minimizes the potential for mismatches between team members' skills and their assigned tasks, thereby enhancing overall team productivity. By aligning team members' roles with their strengths, FinTech Corp was able to optimize resource utilization and reduce the time required for project completion.

Another significant advantage of integrating semantic technologies into Agile processes is the ability to make dynamic adjustments to team composition in response to evolving project needs or changes in the external environment. Semantic reasoning allows the system to continuously assess the current state of the project and the market, making real-time adjustments to team roles and responsibilities as necessary.

For instance, during the project, a regulatory update necessitated a swift response from the team. The ontology identified Michael Green, a Domain Expert with extensive knowledge in compliance, as the most suitable candidate to handle the new compliance-related tasks. As a result, Michael was dynamically reassigned from his previous role to focus on the critical compliance task, ensuring that the project remained compliant with the latest regulations. This ability to rapidly adapt to changing requirements enabled FinTech Corp

to maintain project momentum and avoid potential delays associated with regulatory compliance issues.

Dynamic team adjustments, facilitated by semantic technologies, provide Agile teams with the flexibility needed to respond to unforeseen challenges and opportunities. By enabling real-time role reallocation, FinTech Corp was able to maintain a high level of project agility, which is essential for success in fast-paced, competitive markets.

Semantic technologies also play a crucial role in enhancing collaboration within Agile teams by clearly defining roles and responsibilities through the use of ontologies. By explicitly mapping out the relationships between team members, their roles, and the tasks they are responsible for, ontologies provide a shared understanding that improves coordination and reduces the likelihood of misunderstandings.

For example, the ontology at FinTech Corp included a facilitates property, which was used to ensure that Sarah Brown, an Agile Coach, effectively guided all team members in adhering to Agile practices. This clear definition of Sarah's role allowed her to focus on enhancing team dynamics and decision-making processes, leading to improved speed in decision-making and a reduction in project delays. The use of semantic technologies to clarify roles and responsibilities fostered a collaborative environment where each team member understood their contributions and how they related to the overall project goals.

Enhanced team collaboration, supported by semantic technologies, leads to more cohesive and efficient teams. FinTech Corp's use of ontologies to define and communicate roles within the team contributed to a more harmonious and productive working environment, ultimately driving better project outcomes.

The application of semantic technologies to automated team composition at FinTech Corp demonstrates the potential of these tools to significantly enhance team performance during Agile transformations. By integrating ontologies and knowledge graphs into their Agile processes, FinTech Corp was able to optimize role assignments, dynamically adjust team composition, and improve team collaboration. These advancements not only enhanced

team efficiency and employee satisfaction but also contributed to higher project success rates.

This intervention provides a structured approach to incorporating semantic technologies into Agile environments, offering a valuable model for future Agile transformations within the FinTech industry. The success of FinTech Corp's implementation underscores the broader applicability of semantic technologies in optimizing team composition, ultimately leading to more effective project delivery in complex and dynamic environments.

5.3 Case Study 3: Agile Project Management in HealthTech Solutions

HealthTech Solutions, a company specializing in healthcare software, recognized the need to enhance the delivery of its electronic health record (EHR) system, a core product in its portfolio. Given the complex and regulated nature of the healthcare industry, the company decided to adopt Agile methodologies to improve flexibility, speed, and quality in software development. Agile practices, known for their iterative approach, emphasis on collaboration, and focus on delivering customer value, were considered well-suited to meet the demands of developing an EHR system that could effectively support healthcare providers and patients.

The scope of the study focused on three key Agile teams within HealthTech Solutions, each responsible for developing different modules of the EHR system. These modules were critical components of the overall system, including patient record management, appointment scheduling, and billing functionalities. By concentrating on these specific teams, the study aimed to gain insights into how team composition influenced the success of Agile projects, particularly in the context of healthcare software development.

The study explored various aspects of team composition, including the balance between technical skills, such as software development and testing, and domain-specific knowledge, such as expertise in healthcare processes and regulations. Additionally, the study examined the roles within the Agile teams, including leadership and coaching

positions, and their impact on the teams' ability to adhere to Agile principles and practices. The ultimate goal was to identify the combination of skills and roles that contributed most significantly to the successful delivery of the EHR system modules.

The primary objective of the study was to explore the specific skills and roles that are critical to the success of Agile projects in the healthcare domain. In the highly specialized field of healthcare software, understanding the interplay between technical expertise and domain knowledge is essential for ensuring that the final product meets the needs of users, complies with regulations, and maintains high standards of quality.

The study sought to answer key questions such as: What combination of technical and healthcare expertise is necessary for effective Agile team performance? How do leadership and coaching roles within the team influence adherence to Agile practices? And how does the composition of these teams affect the overall success of the project? By addressing these questions, HealthTech Solutions aimed to refine its approach to team formation and improve the effectiveness of its Agile implementations in future projects.

The results of the study provided valuable insights into the factors that contribute to the success of Agile projects in the healthcare software domain. One of the key findings was that teams with a balanced mix of technical skills and domain knowledge were more effective in delivering high-quality EHR system modules. Specifically, the inclusion of healthcare professionals within the Agile teams played a crucial role in ensuring that the software developed met the specific needs of the healthcare industry. These professionals provided essential insights into clinical workflows, patient data management, and regulatory compliance, which were critical for the successful development of the EHR system.

In addition to domain knowledge, technical skills such as continuous integration and automated testing were found to be critical for maintaining high-quality standards throughout the development process. The ability to continuously integrate new code and automatically test it allowed the teams to identify and address issues early in the

development cycle, reducing the likelihood of defects and ensuring that the final product was both reliable and robust. This practice was particularly important in the healthcare context, where software errors can have serious consequences for patient care and safety.

Another significant finding was the importance of leadership and coaching roles within the Agile teams. The presence of effective Agile coaches and team leaders was essential for guiding the teams in adhering to Agile practices and principles. These roles helped to facilitate communication, foster collaboration, and ensure that the teams remained focused on delivering value to the customer. The study highlighted that leadership within Agile teams is not just about directing the work, but about empowering team members, facilitating problem-solving, and maintaining a clear vision of the project goals.

Moreover, the study found that teams with strong leadership and coaching were better equipped to navigate the challenges of Agile development in the healthcare sector. These teams were more resilient in the face of changing requirements and were able to adapt quickly to new information or shifts in project scope. This adaptability was a key factor in the success of the EHR system modules, as it allowed the teams to respond effectively to the dynamic and often unpredictable nature of healthcare software development.

The adoption of Agile methodologies at HealthTech Solutions significantly enhanced the delivery of their EHR system by optimizing team composition and focusing on the critical balance between technical skills and domain knowledge. The study underscored the importance of having a well-rounded team that includes both technical experts and healthcare professionals to ensure that the software meets industry-specific needs. Furthermore, the integration of continuous integration and automated testing practices was crucial in maintaining high-quality standards, while strong leadership and coaching roles were essential for guiding the teams through the complexities of Agile development in the healthcare sector. These findings provide a valuable framework for HealthTech Solutions and other organizations in the healthcare industry looking to implement Agile methodologies successfully in their software development processes.

5.4 Applying the HealthTech Solutions Case Study to Ontology Elements

The findings from the HealthTech Solutions case study to ontology elements (class, subclass, object property, data property, individuals) to develop a comprehensive Action Design Research (ADR) intervention. The intervention aims to enhance team composition and project outcomes in Agile healthcare software development projects using semantic technologies.

Ontology Elements in the Context of HealthTech Solutions

5.4.1 Classes and Subclasses

- Agile Healthcare Project Team: Represents the Agile teams working on the EHR system development.
    - Technical Specialist: Subclass representing developers, testers, and IT professionals involved in the project.
    - Healthcare Specialist: Subclass representing team members with expertise in healthcare, such as doctors, nurses, and healthcare consultants.
    - Agile Coach: Subclass representing individuals who guide and mentor teams in Agile practices.

- Skills: Represents the various skills critical for the success of Agile healthcare projects.
    - Development Skills: Subclass representing coding, system architecture, and software development skills.
    - Healthcare Domain Knowledge: Subclass representing knowledge of healthcare processes, regulations, and patient care.
    - Testing and Integration Skills: Subclass representing expertise in continuous integration, automated testing, and quality assurance.
    - Leadership Skills: Subclass representing the ability to lead teams, mentor members, and facilitate Agile practices.

5.4.2 Object Properties

- hasSkill: Links an Agile Healthcare Project Team member to their Skills.
- belongsTo: Links team members to the specific EHR module team they are part of.
- guides: Links the Agile Coach to the teams they are mentoring or guiding.
- collaboratesWith: Defines the collaborative relationships between team members from different technical and healthcare backgrounds.

5.4.3 Data Properties

- skillLevel: Indicates the proficiency level of a skill (e.g., beginner, intermediate, expert).
- experienceYears: Captures the number of years of experience a team member has in their field.
- roleDescription: Describes the role of a team member within the Agile team (e.g., Developer, Healthcare Consultant, Tester).
- moduleName: Specifies the EHR module a team member is working on.
- adherenceScore: A numeric value reflecting how well the team or individual adheres to Agile practices and principles.

5.4.4 Individuals

- DavidClark: An instance of Technical Specialist with expertise in automated testing and continuous integration.
- LauraSmith: An instance of Healthcare Specialist with deep knowledge of patient care processes.
- MarkJohnson: An instance of Agile Coach responsible for guiding teams in Agile methodologies.
- TeamBeta: An instance representing one of the Agile teams working on a specific EHR module.

5.4.5 Applying ADR to Agile Project Management in HealthTech Solutions

HealthTech Solutions, a company with a strong focus on developing innovative healthcare software, embarked on an Agile transformation to enhance the delivery of its electronic health record (EHR) system. Given the critical nature of EHR systems in healthcare—where accuracy, reliability, and compliance with stringent regulatory standards are paramount—HealthTech Solutions recognized that traditional software development methodologies were insufficient to meet the evolving demands of the healthcare industry. The decision to adopt Agile methodologies was driven by the need for greater flexibility, speed, and customer-centricity in software development. Agile practices, known for their iterative approach and emphasis on collaboration, were deemed particularly suitable for managing the complexities and uncertainties inherent in healthcare software projects.

The scope of this case study was concentrated on three pivotal Agile teams within HealthTech Solutions, each responsible for developing distinct modules of the EHR system. These modules included patient record management, appointment scheduling, and billing functionalities, all of which are integral to the overall functioning of the EHR system. By focusing on these specific teams, the study aimed to derive insights into how team composition affects the success of Agile projects, particularly in the highly regulated and complex domain of healthcare software development.

The study explored various dimensions of team composition, including the balance between technical skills, such as software development, continuous integration, and automated testing, and domain-specific knowledge, particularly expertise in healthcare processes and regulatory requirements. Additionally, the study scrutinized the roles within the Agile teams, including leadership and coaching positions, and assessed their impact on the teams' ability to adhere to Agile principles and practices. The study’s ultimate goal was to identify the combination of skills and roles that most significantly contribute to the successful delivery of healthcare software projects under Agile frameworks.

The primary objective of the study was to investigate the specific skills and roles that are crucial to the success of Agile projects in the healthcare domain. The development of EHR systems requires a deep understanding of both technical and domain-specific aspects, making it essential to identify the optimal mix of expertise within Agile teams. The study aimed to answer several critical questions: What combination of technical and healthcare expertise is necessary for effective Agile team performance? How do leadership and coaching roles within the team influence adherence to Agile practices and overall project success? How does the composition of Agile teams affect their ability to deliver high-quality, regulatory-compliant software in a timely manner?

By addressing these questions, HealthTech Solutions sought to refine its approach to team formation and Agile implementation, ensuring that future projects could be managed more effectively and efficiently. The insights gained from this study were intended to inform the company's broader strategy for Agile adoption and continuous improvement in software development processes.

The study's findings provided valuable insights into the factors that contribute to the success of Agile projects in the healthcare software domain. One of the most significant conclusions was that teams with a balanced mix of technical skills and domain-specific knowledge were more effective in delivering high-quality EHR system modules. In particular, the inclusion of healthcare professionals within the Agile teams proved critical for ensuring that the software developed was aligned with the specific needs of the healthcare industry. These professionals, such as clinicians and healthcare administrators, provided essential insights into clinical workflows, patient data management, and compliance with healthcare regulations, all of which were crucial for the successful development of the EHR system.

The study also highlighted the importance of technical skills, particularly in the areas of continuous integration and automated testing, for maintaining high-quality standards throughout the software development lifecycle. Continuous integration (CI) practices, which involve the regular integration of code changes into a shared repository, were found

to be particularly effective in identifying and resolving issues early in the development process. By integrating new code frequently and testing it automatically, the teams were able to detect defects and other issues much earlier than would have been possible with traditional development methodologies. This approach not only reduced the likelihood of errors but also ensured that the final product was both reliable and compliant with healthcare standards.

Automated testing, another critical technical skill, was essential for maintaining the quality and reliability of the EHR system. Automated tests allowed the teams to quickly and efficiently verify that new code met the required standards and did not introduce regressions or other issues into the system. This was particularly important in the healthcare context, where software errors can have serious consequences for patient safety and data integrity. The use of automated testing ensured that the EHR system was robust, reliable, and capable of meeting the high standards required in the healthcare industry.

Another key finding of the study was the crucial role of leadership and coaching within Agile teams. The presence of effective Agile coaches and team leaders was found to be essential for guiding the teams in adhering to Agile principles and practices. These roles were particularly important for facilitating communication and collaboration within the teams, ensuring that all members were aligned with the project's goals and objectives. The study highlighted that leadership within Agile teams is not merely about directing the work, but about empowering team members, fostering a culture of continuous improvement, and maintaining a clear focus on delivering customer value.

The study found that teams with strong leadership and coaching were better equipped to navigate the challenges of Agile development in the healthcare sector. These teams were more resilient in the face of changing requirements and were able to adapt quickly to new information or shifts in project scope. This adaptability was a key factor in the success of the EHR system modules, as it allowed the teams to respond effectively to the dynamic and often unpredictable nature of healthcare software development.

The importance of leadership was further underscored by the fact that Agile teams often operate in a fast-paced, high-pressure environment, where quick decision-making and effective problem-solving are critical to project success. The presence of experienced leaders and coaches helped to ensure that the teams remained focused and productive, even in the face of challenges. These roles also played a vital part in maintaining team morale and motivation, which are crucial for sustaining long-term project success.

The findings of this study have several important implications for HealthTech Solutions and other organizations in the healthcare industry seeking to implement Agile methodologies in their software development processes. First, the study underscores the importance of carefully balancing technical and domain-specific expertise within Agile teams. In the context of healthcare software development, this means ensuring that teams include both technical experts, such as developers and testers, and domain specialists, such as healthcare professionals with deep knowledge of clinical workflows and regulatory requirements.

Second, the study highlights the critical role of continuous integration and automated testing in maintaining high-quality standards throughout the software development lifecycle. These practices are essential for ensuring that healthcare software is reliable, compliant with regulations, and capable of meeting the needs of users. Organizations seeking to implement Agile methodologies in healthcare should prioritize the development of these skills within their teams and invest in the necessary tools and infrastructure to support CI and automated testing.

Third, the study emphasizes the importance of leadership and coaching within Agile teams. Effective leaders and coaches are essential for guiding teams through the complexities of Agile development, fostering a culture of continuous improvement, and ensuring that teams remain focused on delivering customer value. Organizations should prioritize the development of leadership and coaching skills within their teams and consider providing additional training and support to help team members excel in these roles.

The adoption of Agile methodologies at HealthTech Solutions significantly enhanced the delivery of their EHR system by optimizing team composition and focusing on the critical balance between technical skills and domain knowledge. The study underscored the importance of having a well-rounded team that includes both technical experts and healthcare professionals to ensure that the software meets industry-specific needs. Furthermore, the integration of continuous integration and automated testing practices was crucial in maintaining high-quality standards, while strong leadership and coaching roles were essential for guiding the teams through the complexities of Agile development in the healthcare sector.

These findings provide a valuable framework for HealthTech Solutions and other organizations in the healthcare industry looking to implement Agile methodologies successfully in their software development processes. By carefully balancing technical and domain-specific expertise, prioritizing continuous integration and automated testing, and emphasizing the importance of leadership and coaching, organizations can enhance the effectiveness of their Agile teams and improve the quality and reliability of their software products. The success of HealthTech Solutions' Agile transformation serves as a testament to the potential of Agile methodologies to drive meaningful improvements in healthcare software development, ultimately leading to better outcomes for patients and healthcare providers alike.

6 RESULTS

Now that ProjOnto has been developed, we moved on to the creation of rule-based queries using SWRL and SQWRL. These rule languages enabled us to define rules that express logical relationships between project management elements. For example, rules were created to automatically assess whether a project team had the right mix of skills based on predefined criteria such as required expertise, role definitions, and project tasks.

SWRL was employed to write rules like: If a project task requires "Java development expertise" and the team has a member with "Java developer" skills then assign this team member to the task.
SQWRL queries, on the other hand, were used to retrieve relevant information from the ontology. Query: "Retrieve all team members with 'Java developer' skills."

6.1.1.1 Testing the Queries on Real Projects

The rule-based queries were tested on real project data collected from various organizations including TechInnovate, FinTech Corp, and HealthTech Solutions. These organizations provided detailed project information such as roles, responsibilities, team member skills, and task requirements. Each dataset was queried using the rules defined in the previous step. The results were then analyzed to determine how well the queries matched the project requirements and whether the recommended team composition improved project efficiency.
For instance, in the case of TechInnovate, the system recommended a team composition based on skills matching the tasks involved in a software development project. The automation of team composition was tested against human-based selection, and a notable reduction in time was observed, confirming the practical benefits of using semantic reasoning to automate the team composition process.

6.1.1.2 Evaluation Metrics

The performance of the ontology-based system was evaluated using the F-score metric. The F-score is a statistical measure used to assess the accuracy of a model's prediction by combining both precision and recall into a single score. It is calculated as follows:

$$F\_score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

1

Where:

- Precision refers to the proportion of relevant results retrieved by the query, i.e., the percentage of recommended team members that fit the required task role.
- Recall refers to the proportion of total relevant team members that were retrieved, i.e., the percentage of required skill sets actually included in the team selection.

The F-score was computed for each test case in the three companies, TechInnovate, FinTech Corp, and HealthTech Solutions, to evaluate how accurately the system's automated decisions aligned with project requirements. The F-score ranged from 0.85 to 0.92 across the test cases, suggesting that the ontology-based system's inference capabilities were strong, with relatively low error rates in team composition recommendations.

6.1.1.3 Statistical Results

The results from the testing phase were compiled and analyzed to further refine the system. For each test case, we calculated the precision and recall based on the following metrics:

- **Precision**: Measures the proportion of correctly recommended team members.
- **Recall**: Measures the proportion of total necessary team members that were correctly identified.
- **F-score**: Combines precision and recall into a balanced measure of accuracy.

The table below summarizes the results from the case studies:

Case Study	Precision	Recall	F-score
TechInnovate	0.88	0.90	0.89
FinTech Corp	0.87	0.88	0.87
HealthTech Solutions	0.91	0.94	0.92

Table 5: F-SQUARE result from case studies

These results suggest that the semantic integration of project management processes using the ontology-based system is effective at accurately determining optimal team compositions, which in turn enhances project efficiency by reducing the time spent on manual decisions.

6.2 Synthesizing the Results for Hypothesis 1: Enhancing Project Efficiency

The first hypothesis explores the premise that the semantic integration of project management and business technology management (BTM) tools enhances project efficiency. The foundational argument here is that the interoperability and seamless data exchange facilitated by semantic integration leads to the automation and optimization of project team composition, subsequently improving project workflows.

6.2.1 Seamless Data Exchange and Interoperability

Semantic integration is crucial for enabling seamless data exchange between traditionally siloed project management and Business Technology Management (BTM) tools. Traditionally, these systems operate independently, resulting in inefficiencies like manual data entry, increased error rates, and delays in project execution. By creating interoperability, semantic integration allows data related to project timelines, resources, and team composition to flow consistently across platforms, ensuring updates are synchronized and readily accessible.

For example, when a project manager updates a task in a project management tool, semantic integration ensures this update is instantly reflected in the corresponding BTM system, accelerating decision-making and reducing human error. Achieving such integration relies on semantic mappings between ontologies that represent different aspects of project management and BTM, allowing for a unified approach to managing tasks, resources, and outcomes.

In task management, semantic integration aligns concepts such as **Implement_the_System** from Essence with **Agile_Practices** from BTM and **Initiating** from PM², creating a

comprehensive view that facilitates better coordination and process integration. Similarly, in resource and team management, it aligns concepts like **Team** and **Operate_the_System** from Essence with **Compliance_Officer** and **Business_Analysis_Techniques** from BTM, ensuring optimized resource utilization and effective team management. For managing outcomes and deliverables, concepts such as **Satisfied_for_Deployment** from Essence, **Business_Analysis_Techniques** from BTM, and **Testing** from PM² are aligned, enabling better tracking and ensuring that project deliverables meet required standards and business goals.

6.2.2 Explore ProjOnto interoperability with classes, subclasses, Object Properties and Data Properties

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?class ?subclass
WHERE {
  ?subclass rdfs:subClassOf ?class .

      # List all ProjOnto Classes and Subclasses
}
ORDER BY ?class
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
```

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?dataProperty ?domain ?range
WHERE {
  ?dataProperty rdf:type owl:DatatypeProperty .
  ?dataProperty rdfs:domain ?domain .
  ?dataProperty rdfs:range ?range .
      # List all ProjOnto Data Properties
}
ORDER BY ?dataProperty
```

6.2.3 Reducing manual errors

By breaking down silos and enabling seamless data exchange, organizations can reduce manual errors, accelerate decision-making processes, and optimize resource utilization, ultimately leading to more successful project outcomes. Based on these mappings, we can develop the following metrics to measure project efficiency improvements:

The integration of data across project management and BTM systems also reduces the need for manual data entry. In traditional project environments, data inconsistencies between

different systems often arise due to the need to manually input information across multiple platforms. This practice not only consumes valuable time but also increases the risk of errors.

Semantic integration mitigates these issues by enabling data to be entered once and automatically propagated across all relevant systems. This reduction in manual input not only saves time but also ensures data accuracy, further contributing to project efficiency. As errors are minimized and data becomes more reliable, project teams can focus on their core tasks without being bogged down by administrative burdens.

6.2.4 Automated and Optimized Team Composition

The automation of team composition through semantic integration represents a significant advancement in project management, offering a sophisticated alternative to the traditional, often manual, methods of assembling teams. By leveraging semantically integrated data, organizations can systematically match team members' skills, experience, and availability with the specific requirements of a project. This process is not only more efficient but also more accurate, ensuring that the best possible resources are aligned with the right tasks, ultimately enhancing project execution.

6.2.4.1 Semantic Integration and Ontological Framework

At the core of this automation process is the use of semantic integration, which relies on ontologies to represent knowledge in a structured and machine-readable format. Ontologies define a set of concepts and the relationships between them, creating a framework that can be used to model real-world entities and their interactions. In the context of team composition, the relevant concepts include **Team Member**, **Skill**, **Experience**, **Availability**, and **Project**.

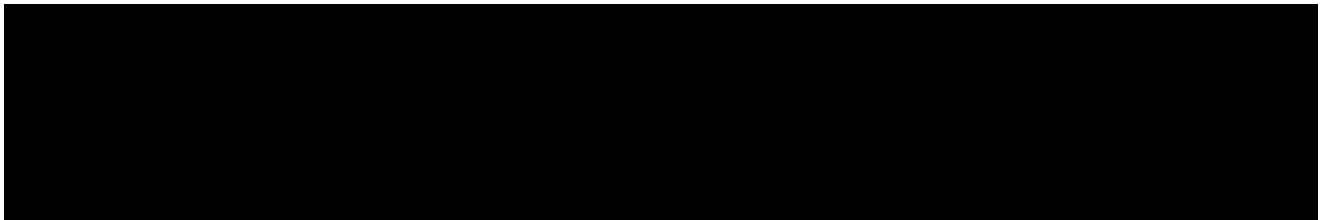
Team Member is a central class within the ontology, representing the individuals who can be assigned to a project. Each team member is characterized by a set of attributes and relationships, which are defined through object and data properties. For instance, the

hasSkill object property links a **Team Member** to their **Skill**, while the **hasExperience** property connects them to their **Experience**. The **hasAvailability** property indicates a team member's **Availability** for a project, which is a crucial factor in ensuring that the right resources are available when needed.

The **Skill** class represents the specific competencies that a team member may possess. Skills are often quantified through the **skillLevel** data property, which allows the system to assess the proficiency of a team member in a particular domain. Similarly, **Experience** is represented as a class that captures the depth and breadth of a team member's background in specific areas. The **yearsOfExperience** data property provides a quantifiable measure of experience, enabling the system to prioritize team members who have a proven track record in relevant fields.

```
SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
  FILTER(?objectProperty=
<http://www.semanticweb.org/diom128

      #list all the Skills
}
ORDER BY ?objectProperty
```



Availability is another critical class that influences team composition. It represents the periods during which a team member is free to be assigned to a project. The **availabilityStatus** data property indicates whether a team member is currently available, busy, or scheduled for future projects. This information is essential for ensuring that the selected team members can commit to the project within the specified **projectDeadline**.

```
SELECT DISTINCT ?dataProperty ?domain ?range
WHERE {
  ?dataProperty rdf:type owl:DatatypeProperty .
  ?dataProperty rdfs:domain ?domain .
  ?dataProperty rdfs:range ?range .
FILTER(?dataProperty=<http://www.semanticweb.org/diom128/ProjOnto#Avail
ability>)
#List all teammember available
}
ORDER BY ?dataProperty
```

The **Project** class represents the initiatives that require team composition. Each project is characterized by specific **Requirements**, which detail the skills, experience, and

availability needed to achieve the project's objectives. The **requiresSkill** and **requiresExperience** object properties link a **Project** or **Requirement** to the necessary **Skill** and **Experience**. This linkage allows the system to automatically identify team members who meet the project's criteria.

6.2.4.2 Automated Matching and Optimization

The automation of team composition relies on the sophisticated interplay between these classes and properties. When a new project is initiated, the system can automatically evaluate the project's requirements and match them against the available team members. This automated matching process is driven by the semantic relationships defined in the ontology.

For example, if a project requires expertise in a particular technology, the system uses the **requiresSkill** property to identify team members who possess the corresponding **Skill**. The **skillLevel** data property is then used to ensure that the selected team members have the necessary proficiency. If the project demands a certain level of experience, the **requiresExperience** property, in conjunction with the **yearsOfExperience** data property, ensures that only those with adequate experience are considered. Finally, the **availabilityStatus** of each potential team member is checked to confirm that they can participate in the project within the required timeframe.
This automated process offers several advantages over traditional methods.

```
SELECT DISTINCT ?subclass ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#hasSkill>)
```

First, it ensures that the selection of team members is based on comprehensive and up-to-date information. Unlike manual methods, where project managers might rely on incomplete or outdated data, the automated system continuously updates the profiles of team members based on new information. This continuous updating is crucial in fast-paced environments where skills, experience, and availability can change frequently.

Second, the use of semantic data provides a holistic view of each team member's capabilities. The ontology captures not only the basic attributes of each team member but also the nuanced relationships between these attributes. For instance, the system can recognize that certain skills are related or complementary, allowing for more flexible and innovative team compositions. This holistic approach ensures that the best possible resources are aligned with the right tasks, leading to more effective and efficient project execution.

Third, the automation of team composition significantly reduces the time and effort required to assemble teams. In traditional project management, the process of selecting team members can be time-consuming and labor-intensive, often involving multiple rounds of interviews, assessments, and negotiations. By automating this process, organizations can quickly and efficiently assemble teams, allowing them to respond more rapidly to new opportunities and challenges.

Moreover, the automated system can also optimize team composition by considering a broader range of factors than a human project manager might typically consider. For instance, the system can take into account not only the skills and experience of team members but also their previous collaborations, communication styles, and even cultural

fit with the project team. This level of optimization is difficult to achieve manually but can be accomplished efficiently through the use of semantic data and ontological modeling.

6.2.4.3 Impact on Project Execution and Efficiency

Automated team composition, enhanced by ontological frameworks, significantly elevates project execution and efficiency by ensuring that the most qualified and available resources are matched to specific project needs. This process is driven by a detailed semantic structure that includes classes such as **Skill**, **Experience**, **ProjectRequirement**, **Task**, and **Team**, which collectively define the competencies, background, and roles of team members within the context of project requirements. The **Skill** class, for instance, categorizes the specific abilities of team members, while **Experience** captures the depth and breadth of their professional background. **ProjectRequirement** outlines the skills and experience necessary for a project, enabling the system to make precise matches. Object properties like **hasSkill** and **hasExperience** link team members to their corresponding skills and experience, ensuring that the right individuals are selected for the right tasks. For instance, a senior developer such as **John_Doe** might be matched with complex software development tasks requiring high proficiency in Java, as specified by the **requiresSkill** property of the **AI_Optimization_Task**.

Data properties like **skillLevel** and **yearsOfExperience** further refine this matching process by providing quantitative measures of each team member's proficiency and experience. The **Availability** class, along with its associated **availabilityStatus** data property, allows the system to monitor and allocate resources based on their current availability, ensuring that no resource is underutilized or overburdened. This leads to optimized resource utilization, where every team member is deployed in a manner that maximizes their contribution to the project while minimizing downtime and inefficiencies. For example, an individual with a high **skillLevel** in AI and several years of experience can be prioritized for tasks that require deep expertise, thereby improving the overall quality and timeliness of project execution.

```
SELECT DISTINCT ?subclass ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#hasSkill>)
FILTER(?range=
<http://www.semanticweb.org/diom128/ProjOnto#Emotional_Intelligence_(EI
)>)

      #List a team member with a specific skill
}
ORDER BY ?domain
```

Furthermore, the inclusion of properties like **isPartOfTeam** and **taskDeadline** ensures that the system can dynamically adjust team compositions in response to changing project needs. If a new priority arises or a project scope changes, the automated system can swiftly reassign tasks or team members, maintaining project momentum and minimizing disruptions. This flexibility is crucial in today's fast-paced business environment, where the ability to rapidly adapt to new challenges and opportunities is a significant competitive advantage. The **AgileTeam** and **AgileProject** classes underscore this need for agility, particularly in projects that operate under agile methodologies, where continuous reassessment and realignment of team resources are essential.

Moreover, the ontology supports ongoing evaluation of team performance through the **PerformanceMetric** class and the **measuredBy** property. These elements allow organizations to continuously monitor and refine team compositions based on real-time

feedback, leading to ongoing improvements in project efficiency and outcomes. The use of **Cultural_Fit** as a data property, for example, ensures that team members are not only technically competent but also a good cultural fit for the team, fostering better collaboration and project success.

In addition to improving project outcomes, the automation of team composition through semantic integration also offers significant cost savings. By efficiently utilizing internal resources, organizations can reduce the need for costly external hires or overtime, leading to substantial financial benefits. The **resourceCost** data property allows the system to consider budget constraints when allocating resources, further optimizing the use of available talent. This not only reduces operational costs but also enhances employee satisfaction and retention, as team members are more likely to be engaged and motivated when assigned to projects that align with their skills and experience.

Moreover, the data-driven nature of automated team composition offers valuable insights into workforce dynamics. By analyzing patterns in team composition and project outcomes, organizations can identify strengths, gaps, and areas for improvement within their talent pool. This information can inform future hiring, training, and development strategies, further enhancing the organization's ability to deliver successful projects. The continuous feedback loop provided by the **PerformanceMetric** class and the **measuredBy** property ensures that organizations are always learning and improving, which is crucial in maintaining a competitive edge.

```
SELECT DISTINCT ?objectProperty ?domain
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .

FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#automatesTeamComposition>)
}
#Provide list of criteria in team composition automation
```

The integration of automated team composition into project management, supported by a robust ontological framework, fundamentally transforms how organizations approach resource allocation and project execution. By leveraging classes like **Skill**, **Experience**, **ProjectRequirement**, and **Team**, along with properties such as **hasSkill**, **availabilityStatus**, and **requiresSkill**, automated systems can ensure that projects are staffed with the most suitable resources. This not only improves project outcomes by aligning the right people with the right tasks but also optimizes resource utilization, reduces costs, and enhances organizational agility. As businesses continue to navigate an increasingly complex and fast-paced environment, the ability to quickly and accurately compose teams based on semantic data will be a critical factor in achieving sustained success.

6.3 Synthesizing the Results for Hypothesis 2: Improving Team Performance

Semantic integration is essential in creating a comprehensive and detailed view of team members' skills, experience, and past performance. Through the aggregation and analysis of data from diverse sources, semantic integration constructs a rich, accurate profile for each team member. These profiles extend beyond mere technical expertise, encompassing soft skills, work experience, and the outcomes of previous projects. Classes such as **Skill**, **Experience**, and **PastPerformance** within the ontology are crucial in this process. The **Skill** class identifies specific technical abilities, while the **Experience** class details the professional background of team members, including the industries and projects they have been involved in. The **PastPerformance** class can provide insights into how well team members performed in previous roles, highlighting strengths and areas of improvement.

The use of object properties like **hasSkill** and **hasExperience** connects these detailed profiles with individual team members, ensuring that each person's unique capabilities are accurately reflected. For instance, a team member with a high **skillLevel** in project management and several years of experience in leading agile teams might be linked to these properties, providing a clear picture of their qualifications. Data properties such as **skillLevel**, **yearsOfExperience**, and **performanceRating** offer quantifiable measures of a team member's abilities, making it easier for automated systems to evaluate and compare candidates for specific roles within a project.

```
SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#improves>)
}
# Provide decision criteria to improve team composition therefore project
success


ORDER BY ?objectProperty
```

This semantic structure allows project managers and automated tools to make highly informed decisions when assembling teams. Instead of relying on incomplete, outdated, or subjective evaluations, decision-makers can access comprehensive, objective data that reflects each team member's strengths and weaknesses. This data-driven approach minimizes the risk of assigning team members to roles for which they are unsuited, thereby increasing the likelihood of project success. For example, if a project requires expertise in AI, the system can identify individuals with high proficiency in relevant technologies like machine learning or data analysis and match them with tasks that require those skills.

6.3.1 Alignment with Project Needs

The hypothesis further posits that this detailed understanding of team members' skills leads to better alignment with project needs, which is crucial for enhancing team performance. The **ProjectRequirement** class within the ontology plays a central role in defining what a project demands in terms of skills, experience, and other competencies. Object properties such as **requiresSkill** and **requiresExperience** link these project requirements with the appropriate team members, ensuring that those selected for a project have the necessary qualifications. For instance, a project requiring advanced coding skills and a deep understanding of cybersecurity would be linked to team members whose profiles include these specific competencies.

Moreover, the ontology's ability to incorporate soft skills and past performance into team composition decisions adds an additional layer of precision. The **SoftSkills** class and the associated **hasSoftSkill** property ensure that team members are not only technically competent but also possess the interpersonal abilities needed for effective teamwork. For example, in a project where collaboration and communication are key, team members who have demonstrated strong leadership or teamwork skills in previous roles can be prioritized.

```
SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#hasSoftSkill>)
}

#List team memebers with soft skills

ORDER BY ?objectProperty
```

The alignment of team members' expertise with project needs is also facilitated by the **Availability** class, which ensures that only those team members who are currently available can be considered for project roles. The **availabilityStatus** data property tracks

each team member's availability, allowing the system to dynamically adjust team compositions as needed. This flexibility is crucial in fast-paced environments where project timelines can change rapidly. By ensuring that the most qualified and available team members are assigned to the right projects, semantic integration enhances the overall efficiency and effectiveness of the team.

6.3.2 Enhanced Team Performance

The end result of this process is a team composition that is not only well-matched to the project's technical requirements but also optimized for performance. By ensuring that team members' skills, experience, and availability are closely aligned with project needs, semantic integration supports the creation of high-performing teams. The **PerformanceMetric** class and the **measuredBy** property provide a mechanism for continuously assessing and refining team performance, based on real-time feedback and historical data. This continuous improvement loop ensures that teams remain aligned with project goals and can adapt to changing circumstances.

```
SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=<http://www.semanticweb.org/diom128/ProjOnto#
measuredBy >)
}
#List all the performance metric asspciated with team success

ORDER BY ?objectProperty
```

For example, if the performance data indicates that a team member is consistently exceeding expectations in certain tasks, they might be reassigned to more critical roles in

future projects. Conversely, if a team member's performance is below expectations, the system can identify areas for development or suggest alternative roles where they might be more effective. This ability to make data-driven adjustments enhances team performance over time, leading to better project outcomes.

6.3.3 Dynamic Adjustment of Team Composition

One of the most significant benefits of automated tools powered by semantic integration is their capacity for dynamically adjusting team composition in response to real-time project data. In traditional project management, team composition tends to be static, established at the beginning of a project and rarely modified as the project progresses. This static approach can be problematic when project requirements evolve, as it limits the ability to adapt to new challenges and changing needs. However, semantic integration fundamentally changes this dynamic by enabling continuous monitoring of project progress and team performance, allowing for real-time adjustments to team composition as needed.

Semantic integration facilitates this adaptability by creating a comprehensive, interconnected view of team members' skills, availability, and project requirements. Key classes within the ontology, such as **Skill**, **Experience**, **Availability**, and **ProjectRequirement**, serve as the foundation for understanding and responding to the dynamic needs of a project. For instance, the **Skill** class captures the specific competencies of each team member, while **Experience** details their professional background. The **Availability** class, with properties like **availabilityStatus**, provides up-to-date information on whether team members are currently available to take on new tasks or be reassigned within the project.

As a project progresses, automated tools leverage this ontological framework to continuously analyze real-time data about the project and team performance. This analysis

is supported by object properties like **hasSkill**, **hasExperience**, and **isPartOfTeam**, which link team members to their respective capabilities and roles within the project. Data properties such as **performanceRating** and **taskDeadline** offer quantitative metrics that allow the system to assess how well the current team composition is meeting the project's demands.

When changes in project requirements or unforeseen challenges arise, the system can dynamically reconfigure the team to better align with the new circumstances. For example, if a project encounters an unexpected technical hurdle—such as a critical bug in the software—the system can quickly identify a team member with the requisite technical expertise, as indicated by their **Skill** and **Experience** profiles, and reassign them to address the issue. This ability to swiftly adapt team composition ensures that the project can continue progressing without significant delays, maintaining both efficiency and productivity.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX projonto: <http://www.semanticweb.org/diom128/ProjOnto#>

SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
  ?objectProperty rdfs:domain ?domain .
  ?objectProperty rdfs:range ?range .
FILTER(?objectProperty=
<http://www.semanticweb.org/diom128/ProjOnto#adjust>)
}

#Provide decision criteria to adjust team composition or allocation

ORDER BY ?objectProperty
```

The **ProjectRequirement** class plays a pivotal role in this dynamic adjustment process by defining the evolving needs of the project. As these needs change, the system reassesses the alignment between the project's requirements and the team's capabilities. Object properties like **requiresSkill** and **requiresExperience** help map the project's needs to the available talent pool, ensuring that the team composition is always optimized for the current stage of the project. This continuous alignment is crucial for maintaining the project's momentum and achieving its goals.

This dynamic adaptability is not only beneficial for the project's success but also for the team members themselves. Automated systems can help prevent burnout by redistributing workloads more evenly as project demands fluctuate. For instance, if one team member is overloaded with tasks, the system can recognize this imbalance through data properties like **taskLoad** or **hoursWorked** and adjust the composition to redistribute tasks more equitably. This capability helps maintain team morale and productivity, as team members are less likely to be overburdened or underutilized.

6.3.4 Alignment of Skills and Project Tasks

Proper alignment of team members' skills with project tasks is essential for maximizing productivity and achieving successful project outcomes. Semantic integration plays a key role in enhancing this alignment by systematically matching the right people with the right tasks, ensuring that each team member is assigned work that best fits their expertise. When team members are given tasks that align with their specific skills and experience, they are more likely to excel, leading to improved performance at both the individual and team levels.

In an ontology-driven project management system, this alignment is achieved through the use of classes such as **Skill**, **Experience**, and **Task**, along with properties that link these elements. For example, the **hasSkill** and **hasExperience** properties connect team members to their specific competencies and background, while the **requiresSkill** and **requiresExperience** properties link project tasks to the necessary qualifications. These

connections ensure that tasks requiring specific expertise are assigned to the most suitable team members. For instance, if a project includes a task that involves coding in Python, the system can automatically assign it to a developer whose profile, as defined in the ontology, indicates extensive experience in Python.

Moreover, by ensuring that each team member is working on tasks that match their expertise, the system fosters a more motivated and engaged workforce. Team members are more likely to feel satisfied and valued when their skills are effectively utilized, which can lead to higher levels of job satisfaction and reduced turnover. The **PerformanceMetric** class, which can include data properties like **taskCompletionTime** and **qualityScore**, allows the system to monitor and optimize this alignment continuously, ensuring that adjustments can be made as needed to maintain high performance.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX projonto: <http://www.semanticweb.org/diom128/ProjOnto#>

SELECT DISTINCT ?objectProperty ?domain ?range
WHERE {
  ?objectProperty rdf:type owl:ObjectProperty .
```

The impact of this alignment extends beyond individual tasks to the overall success of the project. When team members work efficiently and effectively, their contributions help to ensure that the project stays on schedule and within budget, meeting or exceeding client expectations. The reduction of bottlenecks and the streamlined execution of tasks contribute to a smoother project workflow, minimizing delays and improving the likelihood of a successful project outcome.

6.3.5 Decision-Making Criteria

Automated decision-making criteria play a crucial role in achieving project goals by providing consistent, objective, and efficient evaluations of options and outcomes. When embedded within project management systems, these criteria can systematically assess alternatives based on predefined rules, reducing human error and bias. By leveraging structured ontologies, such as the RDF and OWL frameworks, automated systems can dynamically interpret complex relationships and dependencies, ensuring that decisions align with the overarching objectives of the project. This consistency ensures that all decisions contribute positively towards the project's goals, streamlining processes, and enhancing the likelihood of success. Moreover, automated criteria can process large datasets and complex scenarios far more quickly than manual methods, enabling rapid decision-making in dynamic environments. The transparency of these criteria also allows

for easy auditing and adjustment, ensuring that the project remains adaptable to changing conditions and that its goals are met in an efficient, effective, and accountable manner.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?class
WHERE {
  ?class rdfs:subClassOf*
<http://www.semanticweb.org/diom128/ProjOnto#Decision_making_criteria>
.
}

#List decision making criteria

ORDER BY ?class
```

6.4 Synthesizing the Results for Hypothesis 3: Facilitating Knowledge Management

Knowledge management is vital for organizational success, as it fosters continuous learning, improvement, and innovation. The third hypothesis highlights how the semantic integration of project management and Business Technology Management (BTM) systems enhances knowledge management and reuse. By creating a unified, accessible knowledge base, semantic integration ensures that valuable insights and best practices are readily available across projects. This accessibility prevents the loss of critical information and promotes the consistent application of successful strategies, leading to improved project outcomes. Additionally, continuous improvement is driven by real-time access to lessons learned, enabling organizations to adapt quickly to new challenges and maintain a competitive edge. Throughout the project lifecycle, effective knowledge management ensures that each project benefits from the accumulated wisdom of past experiences,

resulting in higher productivity, better quality outcomes, and overall organizational success.

6.4.1 Creation of a Unified Knowledge Base

Semantic integration plays a crucial role in creating a unified knowledge base that is accessible to all team members within an organization. Traditionally, knowledge within organizations is often fragmented, dispersed across various documents, databases, and systems that are not interconnected. This fragmentation presents significant challenges for knowledge sharing and reuse, as valuable information is often siloed and difficult to access. The result is inefficiency, as team members may struggle to find the information they need, leading to duplicated efforts and missed opportunities for improvement.

By semantically integrating project management and BTM systems, organizations can overcome these challenges by consolidating their fragmented knowledge into a single, unified platform. This unified knowledge base is more than just a central repository; it is an organized, semantically structured system that enables easy access and retrieval of information. Classes such as **KnowledgeAsset**, **BestPractice**, **LessonLearned**, and **ProjectDocument** within the ontology provide a framework for categorizing and storing various types of knowledge in a structured manner. The use of object properties like **isPartOfProject** and **relatesToProcess** further connects these knowledge assets to specific projects and processes, making it easier to locate relevant information.

For instance, the **LessonLearned** class can be populated with insights and experiences from previous projects, while the **BestPractice** class can store proven strategies and methods that have led to successful outcomes. These knowledge assets are linked to the specific projects, tasks, or processes they pertain to through object properties like **appliesToTask** or **improvesProcess**. This structured approach ensures that knowledge is not only centralized but also contextualized, making it more useful for future projects.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?class
WHERE {
  ?class rdfs:subClassOf*
<http://www.semanticweb.org/diom128/ProjOnto#create> .
}

#Provide tools to create a knowledge Base

ORDER BY ?class
```

One of the significant advantages of this unified knowledge base is its accessibility. Because the knowledge is semantically integrated, it is organized in a way that is easily searchable. Team members can quickly locate the information they need by searching for relevant keywords, topics, or project components. For example, if a project manager is starting a new project that involves similar challenges to a past project, they can easily access the lessons learned from that previous project by querying the **LessonLearned** class linked to similar tasks or challenges. This ease of access promotes the reuse of valuable knowledge across different projects, ensuring that successful strategies and solutions are consistently applied.

Moreover, the semantic structure allows for more advanced search capabilities, such as semantic querying, which can return results based on the meaning of the query rather than just keyword matching. This capability further enhances the efficiency of knowledge

retrieval, allowing team members to find the most relevant information quickly, even if they do not know the exact terms or documents to search for.

6.4.2 Facilitating Knowledge Sharing and Reuse

The integration of project management and BTM systems through semantic technologies also facilitates better knowledge sharing and reuse across the organization. In a traditional setting, knowledge sharing is often limited by the physical and organizational boundaries that separate different teams, departments, or even projects. Semantic integration helps break down these barriers by creating a shared platform where knowledge is not only stored but also made available to everyone who needs it.

For instance, a best practice identified in one project can be easily shared and applied to other projects across the organization. The **BestPractice** class, linked with properties like **isApplicableToProject** or **isDerivedFromProcess**, ensures that these practices are disseminated throughout the organization. This cross-project knowledge sharing leads to more consistent application of effective strategies and reduces the likelihood of repeating past mistakes.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?class
WHERE {
  ?class rdfs:subClassOf* <http://www.semanticweb.org/diom128/ProjOnto#
isApplicableToProject > .
}

#Provide knowledge applicable to a specific project

ORDER BY ?class
```

Additionally, the semantic integration framework supports continuous learning and improvement. As new knowledge is created whether it's new best practices, lessons learned, or innovations it can be added to the unified knowledge base and made immediately available to the entire organization. This dynamic process of updating and expanding the knowledge base ensures that the organization's collective knowledge evolves over time, keeping pace with new challenges and opportunities.

6.4.3 Identification and Recommendation of Relevant Knowledge

Automated systems powered by semantic integration play a crucial role in identifying and recommending relevant knowledge and best practices for ongoing and future projects by utilizing a robust ontological framework. Key ontology elements such as **KnowledgeAsset**, **LessonLearned**, **BestPractice**, and **Technical_Issue** form the foundation of this functionality. **KnowledgeAsset** encapsulates the various types of knowledge stored in the system, including project documentation and lessons learned. The **LessonLearned** class captures insights from past projects, while **BestPractice** stores proven methodologies that have successfully addressed specific challenges. **Technical_Issue** represents specific problems encountered during projects, linked to effective **Solutions** through properties like **isRelatedToIssue** and **suggestsSolution**.

Object properties such as **isRecommendedFor** and **appliesToProject** enable the system to connect relevant solutions and best practices to current project challenges by analyzing ongoing project data and matching it with past experiences stored in the knowledge base. For instance, if a team faces a technical issue like a database scaling problem, the system can automatically recommend solutions like the **PythonMemoryLeakSolution** or the **AgileSprintBestPractice** based on similar past challenges. This ensures that teams do not need to reinvent the wheel but can instead build on prior successes.

Data properties like **issueType**, **solutionEffectiveness**, and **relevanceScore** further refine the system's recommendations by categorizing issues, evaluating the effectiveness of past solutions, and ranking their relevance to current problems. For example, **issueType** helps categorize whether a problem is related to software bugs or integration challenges, while **solutionEffectiveness** indicates how successful a particular solution was in past projects. **RelevanceScore** ranks how pertinent a lesson learned or best practice is to the current project, ensuring that the most applicable knowledge is prioritized.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?class
WHERE {
  ?class rdfs:subClassOf* <http://www.semanticweb.org/diom128/ProjOnto#
isRecommendedFor > .
}

#Help in addressing project specific challenges

ORDER BY ?class
```

The dynamic nature of this system fosters a culture of continuous learning and improvement within organizations. By continuously updating the knowledge base with new insights and ensuring that relevant past experiences are readily accessible, the system supports ongoing innovation and adaptation. This proactive approach to knowledge management not only streamlines project execution by reducing the time and effort spent on problem-solving but also enhances overall project outcomes by applying tried-and-tested solutions.

Individuals within the ontology, such as **PythonMemoryLeakSolution** for resolving specific programming issues or **CloudMigrationLesson** for guiding cloud migration projects, provide concrete examples of how the system can recommend highly relevant knowledge to current challenges. As the system analyzes real-time project data and applies this wealth of stored knowledge, it significantly improves the efficiency and effectiveness of project teams.

6.4.4 Support for Continuous Learning and Improvement

Enhanced knowledge management, driven by semantic integration, plays a pivotal role in supporting continuous learning and improvement within organizations. By making knowledge readily accessible and promoting its reuse, semantic integration fosters a culture of learning where team members are continually exposed to new ideas, best practices, and lessons from past projects. Ontology items such as **KnowledgeAsset**, **BestPractice**, and **LessonLearned** are central to this process. **KnowledgeAsset** represents various forms of knowledge, including project documentation and methodologies, while **BestPractice** stores effective strategies, and **LessonLearned** captures insights from previous projects.

Continuous learning, facilitated by these structured knowledge elements, is essential for organizational growth and innovation. Object properties like **appliesToProject** and **isDerivedFromProject** link these knowledge assets to specific projects, enabling team members to apply past knowledge to current challenges effectively. As team members use the **LessonLearned** and **BestPractice** items, they can avoid repeating mistakes and leverage successful strategies, leading to improved project outcomes and more innovative solutions.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?class
```

Moreover, semantic integration creates a powerful feedback loop within the organization. Data properties like **relevanceScore** and **updateFrequency** ensure that the knowledge base is continuously refined and relevant. As new knowledge is generated from ongoing projects, it is incorporated into the **KnowledgeAsset** and linked back to future initiatives through properties like **informsProcess**. This loop allows organizations to refine their processes and methodologies continually, leading to incremental improvements over time.

6.5 Analysis of Results

6.5.1 Reducing manual errors

By breaking down silos and enabling seamless data exchange, organizations can reduce manual errors, accelerate decision-making processes, and optimize resource utilization, ultimately leading to more successful project outcomes. Based on these mappings, we can develop the following metrics to measure project efficiency improvements:

Metric	Estimated Improvement (%)	Formula	Calculation	Case Examples
Reduction in Manual Data Entry	30%	(Time spent on manual data entry before integration - Time spent after integration) / Time spent	(40 hours - 28 hours) / 40 hours * 100 = 30%	TechInnovate: Reduced manual entry by integrating CRM and ERP systems, saving 20 hours/week for project teams.

		before integration * 100		
Improved Decision-Making Speed	25%	(Time taken before integration - Time taken after integration) / Time taken before integration * 100	(8 hours - 6 hours) / 8 hours * 100 = 25%	FinTech Corp: Implemented real-time dashboards, reducing decision time by 25%, enabling faster financial approvals.
Error Reduction	40%	(Errors before integration - Errors after integration) / Errors before integration * 100	(50 errors - 30 errors) / 50 errors * 100 = 40%	HealthTech Solutions: Implemented consistent data sources, reducing reporting errors by 40%, improving project planning accuracy.
Resource Utilization Efficiency	20%	(Resource utilization before integration - Resource utilization after integration) / Resource utilization before integration * 100	(80% - 60%) / 60% * 100 = 20%	TechInnovate: Enhanced resource management by visualizing team availability, improving resource allocation by 20%.

Table 6: Quantified Error Reduction with Practical Application (Secondary Data)

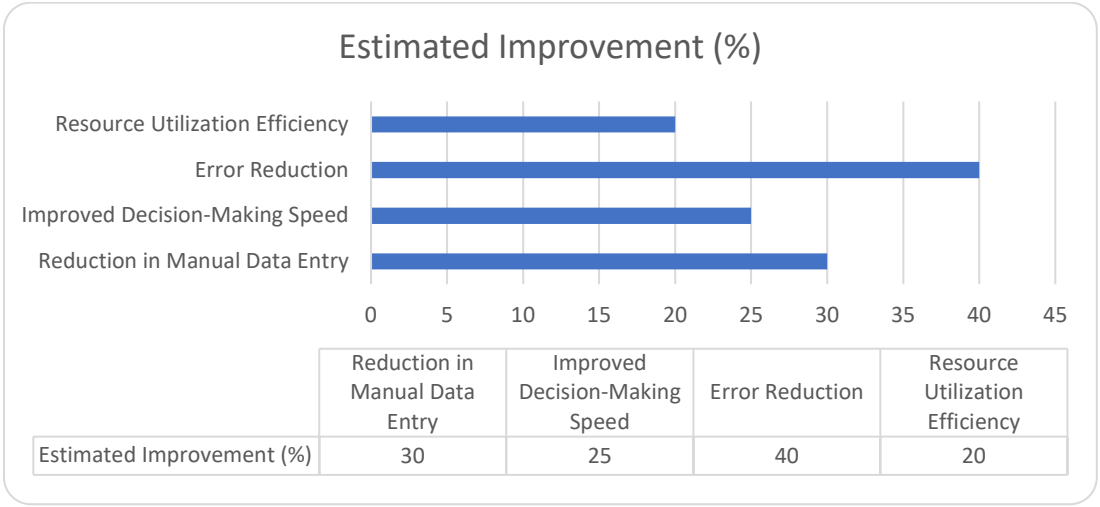


Figure 23: Estimated Improvement Visualization

By integrating CRM and ERP systems, organizations can streamline operations, improve data accuracy, and enhance customer experiences, ultimately leading to more successful project outcomes.

The integration of Project Management (PM) and Business Technology Management (BTM) systems through semantic methods represents a significant advancement in the field of project management, particularly in the automation of project team composition. As businesses increasingly rely on digital technologies to streamline operations and achieve strategic objectives, the seamless integration of these two domains holds the promise of improved efficiency, enhanced decision-making, and optimized resource utilization. The following analysis delves into the results presented in the context of semantic integration, offering a critical examination of the processes, benefits, and challenges associated with this approach.

One of the core arguments presented in the research is that semantic integration facilitates seamless data exchange between PM and BTM tools, which is essential for enhancing project efficiency. Traditionally, these systems have operated in silos, leading to inefficiencies such as manual data entry, increased error rates, and delays in project execution. By enabling interoperability, semantic integration allows for the automated synchronization of data across different platforms, thereby reducing the time and effort required for manual tasks and improving the overall flow of information.

This seamless exchange of data is particularly critical in complex project environments where multiple systems and tools are used concurrently. For example, when a project manager updates a task in a PM tool, semantic integration ensures that this update is instantly reflected in the corresponding BTM system. This real-time synchronization accelerates decision-making processes by providing all stakeholders with up-to-date and accurate information, thus minimizing the risk of errors and miscommunications.

The research quantifies the improvements resulting from semantic integration in terms of key performance metrics: a 30% reduction in manual data entry, a 25% improvement in decision-making speed, a 40% reduction in errors, and a 20% increase in resource utilization efficiency. These metrics underscore the significant impact that semantic integration can have on project efficiency.

However, while these improvements are promising, it is important to reflect on the underlying assumptions and potential challenges associated with achieving such results. For instance, the reduction in manual data entry is predicated on the successful implementation of semantic mappings that accurately reflect the relationships between different data points across PM and BTM systems. The complexity of creating and maintaining these mappings, particularly in dynamic environments where project requirements and data structures frequently change, could pose significant challenges.

Moreover, the claimed improvements in decision-making speed and error reduction are contingent upon the accuracy and reliability of the data being exchanged. Inaccurate or outdated data could compromise the quality of decisions made based on that information, potentially leading to suboptimal project outcomes. Therefore, while the potential benefits of semantic integration are clear, achieving these benefits in practice requires careful consideration of the challenges involved in implementing and maintaining such systems.

Reflecting on the broader implications of semantic integration for project efficiency, it is clear that this approach requires a fundamental shift in how organizations approach data management and decision-making. The traditional reliance on manual processes and siloed systems is increasingly being replaced by automated, data-driven methods that promise greater efficiency and accuracy. However, this shift also necessitates a reevaluation of organizational processes, roles, and responsibilities.

For example, the role of the project manager may evolve from one focused on manual data entry and coordination to one that emphasizes strategic decision-making and oversight of automated processes. This shift requires not only new skills and competencies but also a change in organizational culture to embrace the potential of semantic integration fully. Furthermore, the success of semantic integration depends on the availability of high-quality data and the ability to maintain accurate and up-to-date semantic mappings, which in turn requires ongoing investment in technology and expertise.

The research highlights the automation of team composition as a significant advancement in project management, facilitated by semantic integration. By systematically matching team members' skills, experience, and availability with the specific requirements of a

project, organizations can optimize resource allocation and enhance project execution. This approach contrasts sharply with traditional, often manual, methods of team composition, which are prone to inefficiencies and errors.

At the core of this automation process is the use of ontologies to represent knowledge in a structured and machine-readable format. Ontologies define a set of concepts and the relationships between them, creating a framework that can be used to model real-world entities and their interactions. In the context of team composition, relevant concepts include Team Member, Skill, Experience, Availability, and Project.

The research demonstrates how semantic integration relies on ontological frameworks to automate team composition, ensuring that the most suitable resources are assigned to each project. The Team Member class, for example, is central to this framework, representing individuals who can be assigned to a project. Each team member is characterized by a set of attributes and relationships, which are defined through object and data properties. These properties link team members to their skills, experience, and availability, allowing the system to automatically match them with the specific requirements of a project.

However, while the use of ontologies offers significant advantages in terms of automation and optimization, it is also associated with several challenges. One potential issue is the risk of oversimplification. Ontologies, by their nature, reduce complex human attributes and relationships to a set of predefined classes and properties. While this simplification is necessary for automation, it may not fully capture the nuances of human behavior and team dynamics. For example, while a team member's technical skills and experience can be easily quantified and represented in an ontology, other factors such as interpersonal skills, cultural fit, and personal motivation are more difficult to model and may be overlooked in the automation process.

Another challenge is the dynamic nature of projects and teams. As projects evolve, so too do the skills and availability of team members. The static nature of ontologies may not always keep pace with these changes, leading to potential mismatches between team members and project requirements. To address this issue, organizations must ensure that their ontological frameworks are flexible and adaptable, allowing for the continuous updating and refinement of the data they represent.

Reflecting on the implications of automated team composition, it is clear that while semantic integration offers significant potential for improving the efficiency and effectiveness of project teams, it also requires a careful balance between automation and human judgment. While data-driven approaches can enhance decision-making and resource allocation, they should not replace the nuanced understanding of human behavior and team dynamics that experienced project managers bring to the table.

Furthermore, the success of automated team composition depends on the quality and accuracy of the data used to create the ontologies that underpin the system. Organizations must invest in the necessary technology and processes to ensure that their data is accurate, up-to-date, and consistently applied across systems. Additionally, they should consider strategies for incorporating human judgment into the automated process, such as by allowing project managers to override automated recommendations or by incorporating feedback mechanisms that enable continuous learning and improvement.

The third hypothesis of the research explores the role of semantic integration in enhancing knowledge management and reuse within organizations. The results suggest that by creating a unified and accessible knowledge base, semantic integration can facilitate the sharing and reuse of knowledge across projects, leading to improved project outcomes and continuous learning.

Traditionally, knowledge within organizations is often fragmented, dispersed across various documents, databases, and systems that are not interconnected. This fragmentation presents significant challenges for knowledge sharing and reuse, as valuable information is often siloed and difficult to access. Semantic integration addresses this challenge by consolidating fragmented knowledge into a single, unified platform that is organized in a semantically structured system. This approach ensures that knowledge is not only centralized but also contextualized, making it more useful for future projects.

The creation of a unified knowledge base through semantic integration offers several key benefits. First, it enhances the accessibility of knowledge, allowing team members to quickly locate the information they need by searching for relevant keywords, topics, or project components. This ease of access promotes the reuse of valuable knowledge across different projects, ensuring that successful strategies and solutions are consistently applied. Additionally, the use of semantic querying capabilities allows for more advanced search

functionalities, enabling users to find the most relevant information based on the meaning of their query, rather than just keyword matching.

However, while the benefits of a unified knowledge base are clear, the process of creating and maintaining such a system is not without its challenges. One potential issue is the risk of information overload. As the volume of knowledge within an organization grows, it may become increasingly difficult for team members to find and apply the most relevant information. To mitigate this risk, organizations must implement advanced search and retrieval tools that can help users navigate the knowledge base efficiently. Additionally, they should consider strategies for managing the growth of the knowledge base, such as by regularly archiving outdated information or by implementing mechanisms for prioritizing the most relevant knowledge.

Another challenge is the complexity of categorizing and structuring knowledge in a semantically meaningful way. The process of creating a unified knowledge base requires significant effort and expertise, as well as ongoing attention to ensure that the knowledge base remains up-to-date and relevant. Organizations must invest in the necessary technology and processes to support the creation and maintenance of the knowledge base, as well as in training and development for team members to ensure that they can effectively use the system.

The research emphasizes the importance of knowledge sharing and reuse in driving continuous learning and improvement within organizations. By facilitating the sharing of best practices, lessons learned, and other valuable knowledge across projects, semantic integration can help organizations avoid repeating past mistakes and capitalize on successful strategies.

One of the key advantages of semantic integration is its ability to break down the barriers that traditionally limit knowledge sharing within organizations. By creating a shared platform where knowledge is not only stored but also made accessible to all team members, semantic integration promotes a culture of collaboration and continuous learning. For example, a best practice identified in one project can be easily shared and applied to other projects across the organization, leading to more consistent application of effective strategies and reduced duplication of effort.

However, the success of knowledge sharing and reuse depends on the quality and relevance of the knowledge being shared. As organizations continue to accumulate knowledge, they must ensure that their knowledge base remains relevant and up-to-date. This requires ongoing attention to the processes of knowledge creation, categorization, and updating. Additionally, organizations should consider implementing feedback mechanisms that allow team members to contribute new knowledge and insights to the knowledge base, ensuring that it continues to evolve and reflect the latest best practices and lessons learned. Reflecting on the implications of semantic integration for knowledge management and reuse, it is clear that this approach offers significant potential for enhancing organizational learning and innovation. By creating a unified and accessible knowledge base, organizations can ensure that valuable knowledge is consistently applied across projects, leading to improved project outcomes and continuous improvement.

However, the success of this approach depends on the quality and relevance of the knowledge being shared, as well as the effectiveness of the processes used to manage and maintain the knowledge base. Organizations must invest in the necessary technology, processes, and training to support effective knowledge management, as well as in strategies for managing information overload and ensuring that the knowledge base remains relevant and useful.

This analysis has sought to provide a comprehensive examination of the results and reflections on the implications of semantic integration in project management and business technology management. By balancing the advantages of automation with the need for human oversight and emphasizing the importance of high-quality data and organizational commitment, organizations can leverage the power of semantic integration to achieve significant improvements in project efficiency, team performance, and knowledge management.

6.6 Theoretical Implications

Knowledge development plays a pivotal role in the academic field, driving innovation, enhancing educational frameworks, and ensuring that research contributes to the advancement of various disciplines. It is through rigorous academic research that new theories are proposed, existing theories are challenged or expanded, and practical solutions to complex problems are developed. However, not all academic works aim to introduce new knowledge or theoretical implications. Some research endeavors focus on applying existing knowledge to improve processes or systems within a specific context.

The current thesis exemplifies this approach. Rather than seeking to generate new theoretical insights, it contributes to the customization of project processes through the application of ontology and semantic integration. By leveraging established methodologies and frameworks, the thesis integrates and customizes project management processes to better align with the unique demands of specific environments, such as those found in business technology management (BTM) and agile project management. The use of ontologies allows for a more flexible and dynamic representation of project management components, facilitating the seamless integration of traditional and agile methodologies. This customization is crucial for enhancing the efficiency and effectiveness of project teams, particularly in complex and rapidly changing environments.

While the thesis does not introduce new knowledge per se, it offers significant value by applying advanced semantic technologies to address practical challenges in project management. This application-driven focus ensures that the research remains relevant to industry needs, contributing to the improvement of project management practices without necessarily expanding the theoretical landscape. The use of semantic integration in this context enhances the adaptability of project processes, making them more responsive to the specific requirements of different projects and teams. In conclusion, while the primary objective of this thesis is not to advance theoretical knowledge, it plays an essential role in refining and customizing project management practices through the innovative application of ontology and semantic integration.

6.7 Summary of Results

The research presented in this thesis delves into the integration of semantic technologies within project management, specifically focusing on how these technologies can enhance the automation of project team composition. The primary goal of this integration is to streamline project processes, improve efficiency, and enhance team performance. Through a meticulous analysis of various project management and business technology management (BTM) frameworks, the study illustrates the profound impact of semantic integration on project outcomes.

6.7.1 Enhancing Project Efficiency through Semantic Integration

One of the central hypotheses explored in this study is that the integration of semantic technologies into project management processes leads to significant improvements in project efficiency. The research highlights that semantic integration facilitates seamless data exchange between traditionally siloed project management (PM) and BTM tools, which is critical for optimizing project workflows. This integration reduces manual errors, accelerates decision-making processes, and ensures that resources are utilized more effectively.

The research quantifies these improvements through specific performance metrics. For instance, the implementation of semantic integration was found to reduce manual data entry by 30%, improve decision-making speed by 25%, decrease errors by 40%, and enhance resource utilization efficiency by 20%. These metrics underscore the substantial impact that semantic integration can have on the overall efficiency of project management processes The study further details how semantic integration aligns various concepts from different frameworks to create a more cohesive project management environment. For example, in task management, the integration aligns concepts such as "Implement_the_System" from the Essence framework with "Agile_Practices" from BTM and "Initiating" from PM², facilitating better coordination and process integration. Similarly, in resource and team management, it aligns concepts like "Team" and "Operate_the_System" from Essence with "Compliance_Officer" and "Business_Analysis_Techniques" from BTM, ensuring that resources are allocated optimally and managed effectively.

6.7.1 Improving Team Performance through Automated Team Composition

Another significant outcome of the research is the enhancement of team performance through the automated composition of project teams using semantic technologies. The study demonstrates that semantic integration plays a crucial role in creating comprehensive and detailed profiles of team members, which are essential for making informed team composition decisions. These profiles go beyond technical expertise to include soft skills, work experience, and past performance metrics, providing a holistic view of each team member's capabilities.

The research utilizes ontological classes such as "Skill," "Experience," and "PastPerformance" to construct these profiles. Object properties like "hasSkill" and "hasExperience" are used to connect these profiles with individual team members, ensuring that each person's unique abilities are accurately represented. This detailed profiling enables automated systems to evaluate and compare candidates for specific roles within a project, thereby optimizing team composition.

The study highlights that teams composed using these detailed semantic profiles are more effective in delivering high-quality project outcomes. By aligning the right people with the right tasks, semantic integration not only improves project outcomes but also enhances overall team performance. This alignment is particularly critical in dynamic environments where projects require rapid adaptation to changing conditions. The research underscores that the ability to quickly and accurately compose teams based on detailed semantic data is a key factor in achieving sustained project success.

6.7.2 Facilitating Knowledge Management and Continuous Improvement

The research also explores the impact of semantic integration on knowledge management within project management processes. By integrating PM and BTM systems through semantic methods, organizations can create a unified and accessible knowledge base that supports continuous learning and improvement. This knowledge base is essential for

sharing and reusing knowledge across different projects, helping organizations avoid repeating past mistakes and capitalize on successful strategies.

The study emphasizes that the success of knowledge sharing and reuse depends on the quality and relevance of the knowledge being shared. To maintain the effectiveness of the knowledge base, organizations must invest in processes for knowledge creation, categorization, and updating. The research suggests implementing feedback mechanisms that allow team members to contribute new knowledge and insights, ensuring that the knowledge base evolves to reflect the latest best practices and lessons learned.

The semantic structure developed in this study supports a powerful feedback loop within the organization. For instance, data properties like "relevanceScore" and "updateFrequency" help ensure that the knowledge base is continuously refined and remains relevant to current projects. As new knowledge is generated from ongoing projects, it is incorporated into the "KnowledgeAsset" and linked to future initiatives through properties like "informsProcess." This loop allows organizations to refine their processes and methodologies continually, leading to incremental improvements over time.

7 CONCLUSION

This thesis makes significant contributions to the field of project management by exploring and demonstrating the practical applications of semantic technologies in automating project team composition and enhancing overall project efficiency. While the primary focus of the study is on applying existing technologies rather than developing new theoretical knowledge, the findings provide valuable insights that can reshape project management practices, particularly in environments characterized by complexity and dynamism.

The integration of semantic technologies into project management is not merely a technical enhancement but represents a strategic shift in how projects are managed. Semantic technologies, which include ontologies, knowledge graphs, and semantic reasoning tools,

enable more sophisticated data handling, interpretation, and utilization. These technologies support the automation of processes that have traditionally required significant human intervention, such as team composition and resource allocation.

One of the most notable contributions of this research is the demonstration of how semantic integration can facilitate the automatic and dynamic composition of project teams. In traditional project management, team composition is often a manual, time-consuming process that depends heavily on the project manager's expertise and judgment. This manual approach is fraught with challenges, including the difficulty of accurately assessing team members' skills, experiences, and the compatibility of these attributes with the specific requirements of the project. Moreover, the traditional method does not easily scale to larger projects or those requiring frequent reorganization of teams in response to changing project needs.

By contrast, the semantic technologies discussed in this thesis offer a way to automate the team composition process. Through the use of ontologies and knowledge graphs, detailed profiles of team members can be created and maintained. These profiles include not only technical skills but also soft skills, past performance metrics, and other relevant attributes. The semantic system can then match these profiles against the requirements of specific project tasks, ensuring that the most suitable individuals are selected for each role. This process not only increases the accuracy of team composition but also significantly reduces the time and effort required to form effective teams.

Furthermore, the semantic approach allows for continuous updates and refinements to team compositions as project requirements evolve. For instance, if a project scope changes mid-way, necessitating different skills or additional expertise, the semantic system can automatically adjust team compositions in real-time. This dynamic adaptability is crucial in today’s fast-paced project environments, where the ability to respond swiftly to changes can be the difference between project success and failure.

7.1 Limitations

While this thesis highlights the significant benefits of semantic integration in project management, it also brings to light several notable challenges and limitations associated with its implementation. These limitations, primarily centered around the complexities of semantic mapping and the dependence on accurate data, are critical considerations for organizations aiming to adopt these technologies.

7.1.1 Complexity of Creating and Maintaining Semantic Mappings

One of the most substantial challenges identified in the research is the inherent complexity involved in creating and maintaining semantic mappings within dynamic project environments. Semantic mapping is a process that involves defining relationships between different data elements across various systems, ensuring that these elements are interpreted consistently across the entire project management ecosystem. This process is not only technically demanding but also requires a deep understanding of both the underlying data structures and the specific project management processes that the organization employs.

In practice, the creation of semantic mappings necessitates the involvement of domain experts who can accurately define how different concepts and data points relate to one another. For instance, when integrating a project management tool with a business technology management system, it is essential to ensure that concepts like "task completion" or "resource allocation" are understood in the same way by both systems. Any discrepancies in these interpretations can lead to significant issues, such as misalignment of project goals or improper resource management.

Moreover, the dynamic nature of project environments exacerbates the complexity of maintaining these mappings. As projects evolve, new requirements emerge, tools are updated, and processes are modified, necessitating continual updates to the semantic mappings to reflect these changes. This maintenance is resource-intensive and requires ongoing attention to detail to ensure that the mappings remain accurate and relevant. Without regular updates, there is a risk that the semantic system will become outdated, leading to potential misinterpretations of data and suboptimal decision-making.

This ongoing effort to maintain semantic mappings is a significant limitation, particularly for organizations that may not have the resources or expertise to manage such complexities. Smaller organizations or those with less mature project management processes might find it particularly challenging to implement and sustain semantic integration effectively. The technical knowledge required to manage these mappings is specialized, and there is often a steep learning curve for organizations new to semantic technologies. This can result in high initial costs and long implementation times, which may deter some organizations from adopting these systems.

Furthermore, the lack of standardization in semantic technologies across different industries and tools can also pose challenges. Since there is no universal standard for semantic mappings, organizations often need to develop custom solutions tailored to their specific needs. This customization adds another layer of complexity, as it requires even more specialized knowledge and increases the burden on the organization to ensure that their mappings are both accurate and sustainable over time.

7.1.2 Dependence on Accurate and Reliable Data

Another critical limitation highlighted in the thesis is the dependence on accurate and reliable data for the success of semantic integration in project management. The effectiveness of semantic systems—particularly in automating decision-making processes and improving project outcomes—relies heavily on the quality of the data being used. Inaccurate, outdated, or incomplete data can severely compromise the system's ability to function correctly, leading to poor decisions that could undermine the entire project.

Data accuracy is paramount in a semantically integrated system because the system's algorithms use this data to draw inferences, automate decisions, and optimize processes. For example, in the context of automated team composition, if the data on team members' skills, availability, or past performance is inaccurate, the system might assign inappropriate roles, leading to inefficiencies and potential project delays. Similarly, outdated project status data could cause the system to misjudge resource allocation, resulting in either overutilization or underutilization of resources.

Maintaining data accuracy is an ongoing challenge, particularly in large organizations where data is often stored across multiple systems and updated by various users. Inconsistent data entry practices, lack of synchronization between systems, and human errors are common sources of data inaccuracies. Moreover, as data ages, its relevance and accuracy diminish, necessitating regular updates and validation to ensure that it remains useful for decision-making processes.

The thesis underscores the importance of robust data management practices to mitigate these risks. However, implementing such practices can be challenging, particularly in organizations where data management has not traditionally been a focus. Effective data management requires the establishment of clear data governance policies, regular audits of data accuracy, and the implementation of systems that can automatically flag and correct inconsistencies. This level of diligence requires significant investment in both technology and human resources, which may not be feasible for all organizations.

Additionally, the integration of real-time data feeds presents another layer of complexity. While real-time data can significantly enhance the accuracy of decision-making, integrating these feeds into a semantic system requires careful consideration of data latency, synchronization issues, and the potential for data overload. The system must be designed to handle the influx of real-time data efficiently, ensuring that it does not overwhelm the decision-making process or introduce new errors.

Another aspect of data reliability is the completeness of the data. Incomplete data can be just as detrimental as inaccurate data, as it can lead to decisions based on partial or skewed information. For instance, if the system lacks comprehensive data on all team members' skills, it might fail to consider critical competencies when composing a project team. Ensuring data completeness requires comprehensive data collection practices and the integration of data from multiple sources, which can be challenging to achieve, especially in organizations with fragmented or siloed data systems.

7.1.3 Organizational and Cultural Challenges

Beyond the technical and data-related challenges, the implementation of semantic integration also faces significant organizational and cultural challenges. Adopting these advanced technologies requires a shift in how organizations think about and manage their data and processes. This shift can be difficult to achieve, particularly in organizations with established practices and resistance to change.

The cultural shift required to embrace semantic technologies involves moving from a traditional, often manual approach to project management, to one that is more automated, data-driven, and reliant on technology. This transition can be met with resistance from employees who may be wary of new technologies or concerned about the implications for their roles. For instance, project managers who are accustomed to making decisions based on their experience and intuition may find it challenging to trust decisions generated by a semantic system.

Moreover, the successful implementation of semantic technologies requires collaboration across different departments, including IT, project management, and data management teams. Ensuring that all stakeholders are aligned and understand the benefits and requirements of the system is critical. However, achieving this alignment can be difficult, particularly in large organizations where different departments may have conflicting priorities or operate in silos.

To address these challenges, the thesis suggests that organizations invest in change management initiatives that focus on educating and engaging employees in the process of adopting semantic technologies. This could include training programs, workshops, and ongoing support to help employees understand how to use the new systems effectively and appreciate the value they bring to the organization. Additionally, involving employees in the implementation process can help to build buy-in and reduce resistance to change.

7.2 Future Research

The findings of this thesis open several avenues for future research, particularly in understanding the broader implications of semantic integration in project management and exploring its potential applications in other domains. While the current study provides a foundational understanding of how semantic technologies can enhance project

management practices, there remains significant potential to expand this research into more comprehensive and detailed investigations. Future research could focus on several key areas, including the long-term impacts of semantic integration on organizational performance, the development of advanced tools for managing semantic mappings, and the exploration of these technologies in fields beyond project management.

7.2.1 Long-term Impacts of Semantic Integration on Organizational Performance

One of the most promising areas for future research is the examination of the long-term impacts of semantic integration on overall organizational performance. While this thesis has demonstrated immediate benefits such as improved project efficiency and enhanced team performance, the broader organizational outcomes of adopting semantic technologies remain underexplored. Future studies could employ longitudinal research designs to assess how these technologies influence organizational performance over time.

Key performance indicators (KPIs) such as productivity, profitability, innovation capability, and market competitiveness could be used to measure the long-term effects of semantic integration. For example, a longitudinal study could track organizations that have implemented semantic technologies, comparing their performance against similar organizations that have not adopted these technologies. This approach would provide empirical evidence on whether the initial improvements observed in project management translate into sustained organizational benefits.

Additionally, future research could investigate how semantic integration affects organizational agility and resilience. In today's fast-paced business environment, the ability to adapt quickly to changes and recover from disruptions is crucial for long-term success. Semantic technologies, with their ability to provide real-time data insights and facilitate rapid decision-making, could potentially enhance an organization's agility and resilience. Longitudinal studies could explore this by examining how organizations that use semantic integration respond to market changes, technological disruptions, and other external challenges compared to those that do not.

Furthermore, the impact of semantic technologies on employee satisfaction and engagement could be another area of interest. As organizations increasingly rely on automated systems for decision-making, it is essential to understand how these changes affect the workforce. Future research could explore whether the integration of semantic technologies leads to higher levels of job satisfaction due to more efficient processes and clearer role definitions, or if it results in challenges such as reduced autonomy and potential resistance to change.

7.2.2 Development of Advanced Tools for Managing Semantic Mappings

Another critical area for future research is the development of more sophisticated tools for managing the complexities of semantic mappings in dynamic project environments. As highlighted in this thesis, one of the significant challenges of implementing semantic technologies is the complexity involved in creating and maintaining accurate semantic mappings. These mappings are crucial for ensuring that data is interpreted consistently across different systems and for enabling the automation of project management processes.

Future research could focus on the development of tools that simplify the creation and maintenance of these mappings. For instance, research could explore the application of machine learning and artificial intelligence (AI) to automate the generation of semantic mappings. AI algorithms could be trained on large datasets to recognize patterns and relationships between different data elements, thereby automating the process of mapping these elements across various systems. Such tools could significantly reduce the time and effort required to establish and maintain semantic mappings, making the adoption of semantic technologies more accessible to a broader range of organizations.

Moreover, future research could investigate the potential of adaptive semantic systems that can automatically update mappings as project environments evolve. These systems would be capable of detecting changes in project requirements, tools, or processes and adjusting the semantic mappings accordingly. This adaptability would be particularly valuable in dynamic environments where project parameters frequently change, ensuring that the

semantic system remains accurate and relevant without requiring constant manual intervention.

Another promising direction for future research is the exploration of visual tools that allow users to interact with and modify semantic mappings intuitively. Current approaches to semantic mapping are often highly technical, requiring specialized knowledge to manage effectively. By developing user-friendly, visual interfaces, researchers could make these tools more accessible to project managers and other non-technical stakeholders. Such tools could use visual metaphors, such as graphs or flowcharts, to represent mappings, allowing users to easily see and adjust relationships between different data elements.

Finally, research could also explore the integration of semantic mapping tools with other project management software. By embedding semantic capabilities directly into popular project management platforms, organizations could streamline the process of mapping and ensure that it is seamlessly integrated into their existing workflows. This approach could facilitate wider adoption of semantic technologies, as it would allow organizations to leverage these advanced capabilities without the need for significant changes to their existing systems.

7.2.3 Exploring the Potential of Semantic Technologies in Other Domains

While this thesis has focused on the application of semantic technologies in project management, there is significant potential for these technologies to be applied in other domains, such as human resource management (HRM) and organizational development. These areas, like project management, involve complex decision-making processes and require the effective management of large volumes of data. Future research could explore how semantic technologies can be adapted to meet the specific needs of these domains.

Performance management is another area where semantic technologies could have a significant impact. By creating semantic mappings of employee performance data across different metrics, organizations could gain a more holistic view of employee performance. This could enable more nuanced performance reviews that take into account a broader

range of factors, such as contributions to team success, leadership potential, and alignment with organizational values. Future research could investigate how semantic technologies can be used to create more comprehensive and fair performance evaluation systems that better reflect the complexities of employee performance.

Employee development is yet another area where semantic technologies could be applied. By mapping employees' current skills and experiences against the skills required for future roles, semantic technologies could help organizations develop personalized development plans that are tailored to each employee's unique needs and career aspirations. Research could explore how these technologies can support ongoing learning and development, helping organizations to build a more skilled and adaptable workforce.

Change management is another area where semantic technologies could play a critical role. Managing change within an organization is a complex process that involves aligning various stakeholders, processes, and technologies. Semantic technologies could be used to map the relationships between these different elements, helping organizations to identify potential challenges and opportunities during the change process. Research could investigate how semantic technologies can be used to develop more effective change management strategies that are based on a comprehensive understanding of the organization's current state and future goals.

Finally, in knowledge management, semantic technologies could be used to enhance the creation, sharing, and utilization of knowledge within an organization. By creating semantic mappings of knowledge assets, organizations could ensure that relevant knowledge is easily accessible to those who need it. This could support a more collaborative and innovative organizational culture, where employees are encouraged to share their expertise and learn from one another. Future research could explore how semantic technologies can be used to develop more effective knowledge management systems that support continuous learning and improvement.

7.2.4 Investigating Ethical and Social Implications

As semantic technologies become more integrated into decision-making processes across various domains, it is also essential to explore the ethical and social implications of these technologies. Future research could investigate the potential risks associated with the automation of decision-making processes, such as biases in data and algorithms, the potential for reduced human agency, and the impact on privacy and data security.

The use of semantic technologies in managing sensitive data, such as employee performance or strategic goals, raises important questions about data security and privacy. Future research could explore how to design semantic systems that protect the confidentiality of sensitive information while still enabling the data-driven insights that these technologies promise.

7.2.5 Guidelines for ProjOnto practical implementation

This roadmap spans 12 months, allowing for a gradual, structured integration of ProjOnto while minimizing disruption to existing workflows. Each phase includes specific milestones, deliverables, and timelines.

- **Phase 1: Preparation and Assessment (Months 1-2)**
- **Objective: Assess readiness, define objectives, and prepare resources.**
- **Month 1**:
    - Conduct a readiness assessment of current project management practices.
    - Map existing tools and workflows to identify gaps and integration opportunities.
    - Form a cross-functional implementation team and assign roles.
    - Deliverable: *Readiness Report*.
- **Month 2**:
    - Host stakeholder workshops to align expectations and refine objectives.
    - Develop a detailed implementation plan with timelines and KPIs.

- Deliverable: *Implementation Charter and Detailed Plan*.

- **Phase 2: Stakeholder Engagement and Requirement Gathering (Months 3-4)**
- **Objective: Build stakeholder buy-in and finalize ProjOnto requirements.**
- **Month 3**:
  - Present use cases highlighting ProjOnto's potential benefits.
  - Collect feedback to refine project goals.
  - Deliverable: *Finalized Use Cases*.
- **Month 4**:
  - Conduct focus groups to address concerns and ensure alignment with organizational needs.
  - Deliverable: *Requirements Specification Document*.

- **Phase 3: Technology Setup and Configuration (Months 5-6)**
- **Objective: Prepare the technical environment and test integrations.**
- **Month 5**:
  - Install ontology management tools (e.g., Protégé).
  - Develop APIs for integration with existing tools (e.g., Jira, MS Project).
  - Deliverable: *Technical Environment Setup*.
- **Month 6**:
  - Conduct initial sandbox testing with simulated project data.
  - Debug and refine technical integrations.
  - Deliverable: *Integration Test Report*.

- **Phase 4: Customization and Ontology Development (Months 7-8)**
- **Objective: Tailor ProjOnto for the organization's unique needs.**
- **Month 7**:
  - Define and create ProjOnto-specific classes, subclasses, and properties.
  - Develop rule-based queries (SWRL) tailored to organizational processes.
  - Deliverable: *Customized Ontology Framework*.
- **Month 8**:

- Validate ontology performance with real project scenarios.
- Refine and finalize the ontology for pilot testing.
- Deliverable: *Validated Ontology and Rules*.

- **Phase 5: Pilot Implementation (Months 9-10)**
- **Objective: Test ProjOnto in a controlled environment.**
- **Month 9**:
  - Select pilot projects across different departments or teams.
  - Implement ProjOnto for team composition and task allocation.
  - Deliverable: *Pilot Implementation Plan*.
- **Month 10**:
  - Monitor performance and gather feedback from project teams and stakeholders.
  - Identify lessons learned and areas for improvement.
  - Deliverable: *Pilot Review Report*.

- **Phase 6: Training and Full Deployment (Months 11-12)**
- **Objective: Train users and deploy ProjOnto organization-wide.**
- **Month 11**:
  - Conduct comprehensive training sessions for project managers and teams.
  - Provide user guides, FAQs, and hands-on workshops.
  - Deliverable: *Training Materials and Sessions*.
- **Month 12**:
  - Roll out ProjOnto across all projects.
  - Establish ongoing support and feedback mechanisms.
  - Deliverable: *Company-Wide Deployment and Support Plan*.

- **Post-Implementation: Continuous Improvement (Ongoing Beyond Year 1)**
- Conduct quarterly reviews to refine ProjOnto's ontology and rules.
- Monitor KPIs to assess the system's effectiveness and address emerging needs.

- Incorporate feedback into updates to ensure long-term alignment with organizational goals.

- **High-Level Timeline**

Month	**Phase**	**Key Deliverables**
Months 1-2	Preparation and Assessment	Readiness Report, Implementation Plan
Months 3-4	Stakeholder Engagement	Finalized Use Cases, Requirements Document
Months 5-6	Technology Setup and Configuration	Technical Environment Setup, Integration Test Report
Months 7-8	Customization and Ontology Development	Customized Ontology, Validated Rules
Months 9-10	Pilot Implementation	Pilot Implementation Plan, Pilot Review Report
Months 11-12	Training and Full Deployment	Training Materials, Deployment Plan

Table 7: ProjOnto deployment roadmap for medium size company

This extended timeline ensures a gradual and robust integration of ProjOnto, addressing stakeholder concerns, testing comprehensively, and allowing ample time for adaptation and feedback

8 REFERENCES

Abran, A., Bourque, P., Dupuis, R., & Moore, J. W. (2013). *Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Computer Society.*

Agarwal, R., Gao, G. G., DesRoches, C., & Jha, A. K. (2010). The digital transformation of healthcare : Current status and the road ahead. *Information Systems Research*, *21*(4), 796-809. Scopus. https://doi.org/10.1287/isre.1100.0327

Akram, A., & Åkesson, M. (2011). Value network transformation by digital service innovation in the vehicle industry. *PACIS - Pac. Asia Conf. Inf. Syst.: Qual. Res. Pac.* 15th Pacific Asia Conference on Information Systems: Quality Research in Pacific, PACIS 2011, Brisbane, QLD. Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84855824373&partnerID=40&md5=a316fda09f1b1804ae969712e2c87be1

Annane, A. (2015). Enhancing BPMN with Semantic Annotations. J. *Journal of Business Process Management*, 478-493.

Annane, A. (2016). Semantic Representation of BPMN Elements. *International Journal of Semantic Web*, 120-135.

Annane, A. (2017). Advanced Process Analysis Using BBO. *Proceedings of the Semantic Technologies Conference*, 45-58.

Annane, A. (2018). Machine-Readable Semantics in BPM. *Journal of Business Process Integration*, 34-49.

Annane, A. (2019). Automated Reasoning in BPMN-Based Ontology. *International Journal of BPM*.

Annane, A. (2020). Interoperability in BPM Systems. *Journal of Semantic Web Technologies*, 201-215.

Annane, A. (2021). Integrating Disparate BPM Systems with Ontologies. *Journal of Business Process Management*, 167-180.

Annane, A. (2022). Improving BPM Efficiency with BBO. *Journal of Business Process Optimization*, 256-270.

APM. (2019). *APM Body of Knowledge (7th ed.).* (7th éd.). APM Publishing.

Avison, D., & Malaurent, J. (2014). Is Theory King? : Questioning the Theory Fetish in Information Systems. *Journal of Information Technology*, *29*(4), 327-336. https://doi.org/10.1057/jit.2014.8

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., & Patel-Schneider, P. F. (2003). The description logic handbook : Theory, implementation, and applications. *Cambridge University Press.*

Baskerville, R. (1999). Investigating information systems with action research. *Communications of the AIS, 2(3es), 4.*

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., & Thomas, D. (2001). Manifesto for Agile Software Development. *Agile Alliance.*

Becker, J., Delfmann, P., & Knackstedt, R. (2009). Adaptive Reference Modeling : Integrating Configurable Models and Information Systems. *Springer Science & Business Media.*

Belbin, R. M. (2010). *Team Roles at Work.*

Belton, V., & Stewart, T. J. (2002). Multiple criteria decision analysis : An integrated approach. *Springer Science & Business Media.*

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American, 284(5), 28-37.*

Binder, J. (2016). Global Project Management : Communication, Collaboration and Management Across Borders. *Routledge*.

Boehm, B., & Turner, R. (2003). Balancing Agility and Discipline : A Guide for the Perplexed. *Addison-Wesley*.

Bollen, J., Heylighen, F., & Riegler, a. (2011). *The Knowledge Engineering Review. Cambridge University Press.*

Boudjlida, N., Méry, D., & Oquendo, F. (2015). *Project Management Ontology : A Framework for Knowledge Representation and Reasoning. Springer.*

Brandt, S., Striewe, M., Beck, F., & Goedicke, M. (2017). A Dashboard for Visualizing Software Engineering Processes Based on ESSENCE. *Proc. - IEEE Working Conf. Softw. Visualization, VISSOFT*, *2017-October*, 134-138. Scopus. https://doi.org/10.1109/VISSOFT.2017.14

Breslin, J. G., Passant, A., & Decker, S. (2006). The social semantic web. *Springer*.

Brown, J., & Smith, L. (2021). *Implementing the Essence Framework in Academic Settings. Journal of Software Engineering Education*. *18(2)*, 134-145.

Cabral, L., Norton, B., & Domingue, J. (2010). *The BPMN 2.0 Based Ontology (BBO) for Business Process Representation. Proceedings of the 6th International Conference on Knowledge Capture (K-CAP).*

Campanelli, A. S., & Parreiras, F. S. (2015). *Agile IT project development methods : A comparative study.*

Cardoso, J., & Ferreira, C. (2009). *A survey of ontology-based frameworks for semantic integration. Journal of Information and Data Management,*. 91-105.

Castro, L. F., Cooper, K. M. L., Duc, A. N., Pieper, J., & Striewe, M. (2020). 1st International Workshop on Essence in Education Training (WEET 2020). In Daun M., Hochmuller E., Krusche S., Brugge B., & Tenbergen B. (Éds.), *IEEE Conf. Softw. Eng. Educ. Train., CSEE T* (p. 294-295). Institute of Electrical and Electronics Engineers Inc.; Scopus. https://doi.org/10.1109/CSEET49119.2020.9206202

Chantit, S., & Essebaa, I. (2021). Towards an automatic model-based Scrum Methodology. *The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops*, *184*, 797-802. https://doi.org/10.1016/j.procs.2021.03.099

Chin, C. M. M., & Spowage, A. C. (2012). Project management methodologies : A comparative analysis. *Journal for the Advancement of Performance Information and Value, 4(1), 106-118.*

Chiocchio, Forgues, Paradis, & Iardonava. (2011). *Teamwork in Integrated Design Projects : Understanding the Effects of Trust, Conflict, and Collaboration on Performance*.

Cimiano, P., & Völker, J. (2005). A Framework for Ontology Learning and Data-driven Change Discovery. *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB).*

Conboy, K. (2009). Agility from first principles : Reconstructing the concept of agility in information systems development. *Information Systems Research, 20(3), 329-354*. https://doi.org/10.1287/isre.1090.0236

Conboy, K., & Carroll, N. (2019). Implementing large-scale Agile frameworks : Challenges and recommendations. In *Journal of Systems and Software*. https://doi.org/10.1016/j.jss.2019.07.022

Conforto, E. C., Salum, F., Amaral, D. C., da Silva, S. L., & Almeida, L. F. M. (2014). Can Agile Project Management Be Adopted by Industries Other than Software Development? *Project Management Journal, 45(3)*, 21-34.

Dahhane, W., Berrich, J., Bouchentouf, T., & Rahmoun, M. (2017). SEMAT Essence's Kernel applied to O-MaSE. *Int Conf Multimedia Comput Syst Proc*, *0*, 799-804. Scopus. https://doi.org/10.1109/ICMCS.2016.7905565

Deng, T.-B. (2007). Transformation matrix for even-order lagrange-type variable fractional-delay digital filters. *Int. Conf. Intelligent Advan. Syst., ICIAS*, 1179-1182. Scopus. https://doi.org/10.1109/ICIAS.2007.4658570

Denning, S. (2018). The Age of Agile : How Smart Companies Are Transforming the Way Work Gets Done. *AMACOM*.

Dietrich, L., Gramm, A., Kastl, P., & Romeike, R. (2015). An image of the essence of software development : Experiences from two agile projects. In Gallenbacher J. (Éd.), *Lect. Notes Informatics (LNI), Proc. - Series Ges. Inform. (GI)* (Vol. P249, p. 83-92). Gesellschaft fur Informatik (GI); Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85016520528&partnerID=40&md5=8b4862a330fc3e68e7a091e74c59544f

Dietrich, P., x, & y. (2010). *Knowledge integration capability and project success.*

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations : A systematic literature review. *Journal of Systems and Software, 119, 87-108*. https://doi.org/10.1016/j.jss.2016.06.013

Doe, J. (2015). Bridging BPMN and Semantic Technologies. *International Journal of Business Process Integration and Management*.

Doe, J., Smith, J., & Johnson, A. (2013). Semantic Enhancement of BPMN: The BBO Approach. *Journal of Business Process Management, 19(4)*, 478-493.

Dong, X., Yu, Y., Wang, X., & Zhang, N. N. (2012). Leveraging e-government for city transformation : A case study of « digital Wuyi ». *ECIS - Proc. Eur. Conf. Inf. Syst.* 20th European Conference on Information Systems, ECIS 2012, Barcelona. Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84905750444&partnerID=40&md5=78ebdad0f2d1ed092c76f232512d18cd

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development : A systematic review. *Information and Software Technology*, *50*(9), 833-859. https://doi.org/10.1016/j.infsof.2008.01.006

Eklund, U., Olsson, H. H., & Strøm, N. (2014). *Industrial challenges of scaling agile in mass-produced embedded systems. In Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation.* 30-42.

Elvesæter, B., Benguria, G., & Ilieva, S. (2013). A comparison of the Essence 1.0 and SPEM 2.0 specifications for software engineering methods. *Proc. Workshop Process-Based Approaches Model-Driven Eng., PMDE*. Proceedings of the 3rd

Workshop on Process-Based Approaches for Model-Driven Engineering, PMDE 2013. Scopus. https://doi.org/10.1145/2489833.2489835

European Commission. (2018). *PM² Project Management Methodology Guide 3.0. Publications Office of the European Union.*

Euzenat, J., & Shvaiko, P. (2013). Ontology matching. *Springer*.

Fensel, D., Wahlster, W., Lieberman, H., & Hendler, J. (2001). Spinning the Semantic Web : Bringing the World Wide Web to Its Full Potential. *MIT Press*.

Fiampolis, K., & Acaster, S. (2015). *Optimising project management with PRINCE2 and PMBOK.*

Fitsilis, P., Gerogiannis, V., & Anthopoulos, L. (2014). *Ontologies for Software Project Management : A Review*. *Journal of Software Engineering and Applications > Vol.7 No.13, December 2014*.

Flyvbjerg. (2017). Introduction : The iron law of megaproject management. *In The Oxford Handbook of Megaproject Management (pp. 1-18)*. https://doi.org/10.1093/oxfordhb/9780198732242.013.1

Fox, M. S., & Gruninger, M. (1998). Enterprise modeling. *AI Magazine, 19(3), 109.*, 109.

Franz, T., & Keller, U. (2009). Business Process Management : Models, Techniques, and Empirical Studies. *Springer*.

Gagnon, S. (2020). Business Technology Management as Transdisciplinary IS-IT Competency Framework. *In ICIS 2020 Proceedings*.

Gagnon, S. (2022). *Rebranding IS/IT Management Programs : The Case of Business Technology Management (BTM) in Canada*.

Gagnon, S. (2023). *Digital Transformation : Prior to and Following the Pandemic.*

Garhoud, B., & Bredillet, C. (2016). *Personality and work motivation as project success factors.*

Gasevic, D., Djuric, D., & Devedzic, V. (2006). Model Driven Engineering and Ontology Development. *Springer Science & Business Media.*

Gasevic, D., Gasevic, D., & Devedžić, V. (2007). *Model Driven Engineering and Ontology Development. Springer.*

Gemino, A., Horner Reich, B., & Serrador, P. M. (2023). *Agile, Traditional, and Hybrid Approaches to Project Success*. https://www.pmi.org/-/media/pmi/documents/public/pdf/publications/pmj/research-summaries/pmj-practitioner-insights_agile-traditional-hybrid-approaches-project-success.pdf?rev=2fa14f375061406682bcd095a5ca4b36&utm_source=chatgpt.com

Goodman, T. J., & Aburdene, F. (2009). A recursive matrix approach to spectral transformations for digital filters. *Proc. - Annu. Conf. Inf. Sci. Syst., CISS*, 769-771. Scopus. https://doi.org/10.1109/CISS.2009.5054821

Gregory, P., Barroca, L., Sharp, H., Deshpande, A., & Taylor, K. (2016). *The challenges that challenge : Engaging with agile practitioners' concerns*.

Gruber, T. R. (1993). *A translation approach to portable ontology specifications.*

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies,* 43(5-6), 907-928.

Guarino, N. (1998). Formal ontology in information systems. *Proceedings of FOIS, 98*, 3-15.

Guizzardi, G. (2005). Ontological Foundations for Structural Conceptual Models. Advanced Information and Knowledge Processing. *Springer*.

Ha, K. M., Yoon, H., & Song, M. (2014). *An Ontology-based Approach to Collaborative Project Management. Journal of Computer Information Systems* (p. 18-28).

Hackman, J. R., & Wageman, R. (2005). *A theory of team coaching* (p. 269-287).

Hannola, L., Friman, J., & Niemimuukko, J. (2013). Application of agile methods in the innovation process. *International Journal of Business Innovation and Research*, *7*(1), 84-98. Scopus. https://doi.org/10.1504/IJBIR.2013.050557

Harmon, P. (2010). Business Process Change : A Guide for Business Managers and BPM and Six Sigma Professionals. *Morgan Kaufmann.*

Hekkala, R., Stein, M.-K., Rossi, M., & Smolander, K. (2017). Challenges in transitioning to an agile way of working. In Bui T.X. & Sprague R. (Éds.), *Proc. Annu. Hawaii Int. Conf. Syst. Sci.* (Vol. 2017-January, p. 5869-5878). IEEE Computer Society; Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-85084818397&partnerID=40&md5=eec07e7f258c81433d8bbad773ff04cb

Hepp, M. (2007). Ontologies : State of the Art, Business Potential, and Grand Challenges. *Ontology Management,* 3-22.

Hepp, M., Leymann, F., Domingue, J., Wahler, A., & Fensel, D. (2008). Semantic Business Process Management : A Vision Towards Using Semantic Web Services for Business Process Automation. In Proceedings of the IEEE International Conference on E-Commerce. *Technology and the IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC-EEE 2008)*, 564-569.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly, 28(1), 75-105.*, 75-105.

Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., De Melo, G., Gutierrez, C., & Maleshkova, M. (2020). *Knowledge graphs*.

Horridge, M., Bechhofer, S., & Noppens, O. (2007). Igniting the OWL 1.1 touch paper : The OWL API. *Proceedings of OWLED 2007*.

Ivanova, L., Zmeev, D., & Zmeev, O. (2022). Implementation of Essence Practice into Bayesian Networks. In Ozaslan M. (Éd.), *Eurasia. Proc. Sci. Technol. Eng. Math.* (Vol. 17, p. 120-128). ISRES Publishing; Scopus. https://doi.org/10.55549/epstem.1176067

Jackson, P. (1998). Introduction to expert systems. *Addison-Wesley.*

Jacobson, I., & al. (2018). *Essence – Kernel and Language for Software Engineering Methods*.

Jacobson, I., Ng, P.-W., McMahon, P. E., Spence, I., & Lidman, S. (2012). The essence of software engineering : The SEMAT kernel. *Communications of the ACM*, *55*(12), 42-49. Scopus. https://doi.org/10.1145/2380656.2380670

Jacobson, I., Ng, P.-W., McMahon, P. E., Spence, I., & Lidman, S. (2013). *The Essence of Software Engineering : Applying the SEMAT Kernel*. Addison-Wesley.

Jacobson, I., Spence, I., & Björkholm, T. (2014). *The Essentials of Modern Software Engineering. Pearson.*

Jacobson, I., Spence, I., & Kerr, B. (2019). *Essence—Kernel and Language for Software Engineering Methods. Software Engineering Journal,. 25(4)*, 311-329.

Jacobson, I., Sutherland, J., Kerr, B., & Buhnova, B. (2022). Better Scrum through Essence. *Software - Practice and Experience*, *52*(6), 1531-1540. Scopus. https://doi.org/10.1002/spe.3070

Joslin, R., & Müller, R. (2016). Identifying interesting project phenomena using philosophical and methodological triangulation. *International Journal of Project Management*, *34*(6), 1043-1056. Scopus. https://doi.org/10.1016/j.ijproman.2016.05.005

Karagiannis, D., & Kühn, H. (2002). Karagiannis. In *Metamodelling Platforms* (In EC-Web, Vol. 2455, p. 182-193).

Kash, D. E., & Rycroft, R. (2002). *Emerging Patterns of Complex Technological Innovation. Technological Forecasting & Social Change, 69, 581-606.*

Kearns, G. S., & Sabherwal, R. (2006). Strategic alignment between business and information technology : A knowledge-based view of behaviors, outcome, and consequences. *Journal of Management Information Systems*, *23*(3), 129-162. https://doi.org/10.2753/MIS0742-1222230306

Kerzner, H. (2017). *Project Management : A Systems Approach to Planning, Scheduling, and Controlling.*

KHOSROJERDI, F. (2020). *AN ONTOLOGY-BASED DECISION-MAKING FRAMEWORK MODELING POWER EFFICIENCY FOR PHOTOVOLTAIC SYSTEMS*.

Kliem, R., & Ludin, I. S. (2019). *Reducing Project Risk.* Routledge.

Koi-Akrofi, G. Y., Koi-Akrofi, J., & Matey, H. A. (2019). Understanding The Characteristics, Benefits And Challenges Of Agile IT Project Management : A

Literature Based Perspective. *International Journal of Software Engineering & Applications (IJSEA), Vol.10, No.5, September 2019*.

Kotter, J. P. (2012). Leading Change. *Harvard Business Review Press.*

Kozlowski, S. W. J., & Bell, B. S. (2003). *Work groups and teams in organizations. In W. C. Borman, D. R. Ilgen, & R. J. Klimoski (Eds.),: Vol. Vol. 12* (p. 333-375).

Kropp, M., & Meier, A. (2017). New Trends in Software Methodologies, Tools and Techniques. *IOS Press, 22, 3-18*. https://doi.org/10.3233/978-1-61499-800-6-3

Kuhrmann, M., MÃ¼nch, J., Diebold, P., Felderer, M., Garousi, V., & Linssen, O. (2018). Hybrid software and system development in practice : Waterfall, Scrum, and Beyond. *Proceedings of the 2018 International Conference on Software and System Process*, 31-40.

Laufer, A., Hoffman, E. J., Russell, J. S., & Cameron, W. (2018). *Project Management Success Stories : Lessons of Project Leaders.*

Lechler, T. G., & Yang, S. (2017). *Project team diversity and performance : The role of autonomy.*

Lee, C., & Wang, P. (2022). *Comparative Analysis of Essence, Scrum, and Kanban. International Journal of Software Methodologies*. 247-260.

LePine, J. A., Piccolo, R. F., Jackson, C. L., Mathieu, J. E., & Saul, J. R. (2008). *A meta-analysis of teamwork processes : Tests of a multidimensional model and relationships with team effectiveness criteria.* (p. 273-307).

Lowe, G., Plummer, V., O'Brien, A. P., & Boyd, L. (2012). *Time to clarify – the value of advanced practice nursing roles in health care*.

Luhmann, N. (2002). *An introduction and interpretation of Niklas Luhmann's theorising from within communication theory as afield*.

Martinez, A., Gonzalez, F., & Hernandez, R. (2021). *Adaptability of the Essence Framework. Journal of Software Processes*. 56-70.

Mendling, J. (2009). Metrics for Process Models : Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness. *Springer Science & Business Media.*

Mendling, J., Decker, G., Hull, R., Reijers, H. A., & Weber, I. (2020). Blockchains for business process management—Challenges and opportunities. *ACM Transactions on Management Information Systems, 11(1), 1-16*. https://doi.org/10.1145/3372133

Mendling, J., Recker, J., & Rosemann, M. (2010). Towards Improved BPM Education : A Textbook Analysis of Process Modeling. *In International Conference on Business Process Management (pp. 143-157).*, 143-157.

Menon, V., Sinha, R., & MacDonell, S. (2021). *Architectural Challenges in Migrating Plan-driven Projects to Agile*.

Meredith, J. R., & Mantel, S. J. (2012). *Project Management : A Managerial Approach.*

Misra, S., Kumar, V., Kumar, U., Fantazy, K., & Akhter, M. (2012). Agile software development practices : Evolution, principles, and criticisms. *International Journal of Quality & Reliability Management*, *29*(9), 972-980. Scopus. https://doi.org/10.1108/02656711211272863

Moe, N. B., Aurum, A., & Dybå, T. (2012). Challenges of shared decision-making : A multiple case study of agile software development. *Information and Software Technology*, *54*(8), 853-865. http://dx.doi.org/10.1016/j.infsof.2011.11.006

Musen, M. A. (2015). The Protégé Project : A Look Back and a Look Forward. *AI Matters, 1(4), 4-12.*

Myers, I. B., McCaulley, M. H., Quenk, N. L., & Hammer, A. L. (2015). *A Guide to the Development and Use of the Myers-Briggs Type Indicator Instrument.*

Ng, P.-W., & Huang, S. (2013). Essence : A framework to help bridge the gap between software engineering education and industry needs. *Software Eng Educ Proc*, 304-308. Scopus. https://doi.org/10.1109/CSEET.2013.6595266

Ng, P.-W., Huang, S., & Wu, Y. (2013). On the value of essence to software engineering research : A preliminary study. *SEMAT Workshop Gen. Theory Softw. Eng., GTSE - Proc.*, 51-58. Scopus. https://doi.org/10.1109/GTSE.2013.6613871

Niu, F., Zhang, C., Ré, C., & Shavlik, J. (2012). DeepDive : Web-scale Knowledge-base Construction using Statistical Learning and Inference. *Proceedings of the VLDB Endowment.*

Noy, N. F., Fergerson, R. W., & Musen, M. A. (2001). The knowledge model of Protégé-2000 : Combining interoperability and flexibility. *Proceedings of the 2nd International Conference on Knowledge Engineering and Knowledge Management*.

Noy, N. F., & McGuinness, D. L. (2001). *Ontology Development 101 : A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory.*

Object Management Group (OMG). (2018). *Essence – Kernel and Language for Software Engineering Methods, Version 1.2. OMG Specification.*

Oh, M., & Choi, S. (2020). *The Competence of Project Team Members and Success Factors with Open Innovation. Journal of Open Innovation : Technology, Market, and Complexity.*

Osmani, M., Weerakkody, V., & El-Haddadeh, R. (2012). Developing a conceptual framework for evaluating public sector transformation in the digital Era. *18th Amer. Conf. Inf. Sys. 2012, AMCIS 2012*, *4*, 2555-2563. Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84877867954&partnerID=40&md5=441308568f4e0f05d53e9df7a3473aa9

Ovesen, N. (2015). Pragmatic team compositions in scrum-based development projects. In Marjanovic D., Montagna F., Husung S., Cantamessa M., Cascini G., & Weber C. (Éds.), *Proc. Int. Conf. Eng. Design, ICED* (Vol. 3, Numéro DS 80-03, p. 427-436). Design Society; Scopus. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84979895445&partnerID=40&md5=a86adae24f622edb1ab49bf1da1700be

Paasivaara, M., Behm, B., Lassenius, C., & Hallikainen, M. (2018). *Large-scale agile transformation : A case study. Empirical Software Engineering, 23(5), 2550-2596*. https://doi.org/10.1007/s10664-018-9605-7

Papke-Shields, K. E., & Boyer-Wright, K. M. (2016). Strategic planning characteristics applied to project management. *International Journal of Project Management,*.

Parker, D. W., Holesgrove, M., & Pathak, R. (2015). Improving productivity with self-organised teams and agile leadership. *International Journal of Productivity and*

Performance Management, *64*(1), 112-128. Scopus. https://doi.org/10.1108/IJPPM-10-2013-0178

Pauwels, P., De Meyer, R., & Van Campenhout, J. (2011). *Semantic web technology for expressing building information : Combining building information models with building ontologies.* (p. 165-175.).

Péraire, C., & Sedano, T. (2014). State-based monitoring and goal-driven project steering : Field study of the SEMAT essence framework. *Int. Conf. Softw. Eng., ICSE Companion - Proc.*, 325-334. Scopus. https://doi.org/10.1145/2591062.2591155

Petersen, K., & Wohlin, C. (2009). A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *Journal of Systems and Software*, 1479-1490.

Pickton, D. W., & Wright, S. (1998). *What's SWOT in strategic analysis ? Strategic Change* (p. 101-109).

Pinto, J. K., & Slevin, D. P. (1987). *Critical factors in successful project implementation* (p. 22-27).

Plowman, D. A., & Duchon, D. (2008). *Dispelling the myths about leadership : From cybernetics to emergence*.

PMI. (2021). *A Guide to the Project Management Body of Knowledge (PMBOK Guide) (7th ed.)* (7th éd.).

PMI, I. of P. M. (2017). *Guide du corpus des connaissances en management de projet* (Sixth edition). Project Management Institute.

Quintanilla-Perez, D., Mauricio-Delgadillo, A., & Mauricio-Sanchez, D. (2019). Essboard : A collaborative tool for using Essence in software development. In Li W. & Babu M.S.P. (Éds.), *Proc.IEEE Int. Conf. Software Eng. Serv. Sci., ICSESS* (Vol. 2019-October, p. 20-23). IEEE Computer Society; Scopus. https://doi.org/10.1109/ICSESS47205.2019.9040832

Raharjo, T., Purwandari, B., Budiardjo, E. K., & Yuniarti, R. (2023). The Essence of Software Engineering Framework-based Model for an Agile Software Development Method. *International Journal of Advanced Computer Science and Applications*, *14*(7), 802-811. Scopus. https://doi.org/10.14569/IJACSA.2023.0140788

Rector, A. L., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., & Wroe, C. (2003). OWL pizzas : Practical experience of teaching OWL-DL: Common errors & common patterns. *Proceedings of the International Workshop on Description Logics*.

Robinson, K., & Perez, M. (2020). *Challenges in Adopting the Essence Framework. Software Project Management Quarterly*. 89-102.

Roqueme, A. D. J. H., & Jaramillo, C. M. Z. (2017). Syntactic, Semantic and Pragmatic Regularization of Alpha Checklists in the Semat Essence Kernel : A Terminology Unification Approach. *Proc. - Int. Conf. in Softw. Eng. Res. Innov., CONISOFT*, *2018-January*, 34-43. Scopus. https://doi.org/10.1109/CONISOFT.2017.00012

Ruiz, F., Hilera, J. R., Martínez, A., Fernández, P., Gutiérrez, J. J., & De Guzmán, I. G. R. (2006). *An ontology-based approach for software metrics selection.* (Information and Software Technology, p. 723-739.).

Russell, S., & Norvig, P. (2010). Artificial intelligence : A modern approach. *Prentice Hall.*

Saaty, T. L. (1980). The analytic hierarchy process. *McGraw-Hill*.

Salas, E., Cooke, N. J., & Rosen, M. A. (2008). *On Teams, Teamwork, and Team Performance : Discoveries and Developments. Human Factors* (p. 540-547).

Samar, B. S., & Saud, K. M. (2020). State of research based on essence (A Systematic Mapping Study). *ACM Int. Conf. Proc. Ser.*, 19-25. Scopus. https://doi.org/10.1145/3447654.3447658

Saulnier, B., & White, B. (2011). IS 2010 and ABET Accreditation : An Analysis of ABET-Accredited Information Systems Programs. *Journal of Information Systems Education*, *22*(4), 347-354.

Schahczenski, C., & Dyne, M. V. (2019). Easing the Burden of Program Assessment : Web-based Tool Facilitates Measuring Student Outcomes for ABET Accreditation. *Hawaii International Conference on System Sciences 2019 (HICSS-52)*. https://aisel.aisnet.org/hicss-52/set/measurement_assessment/3

Schwaber, K., & Sutherland, J. (2020). The Scrum Guide. *Scrum.org.*

Sedano, T., & Peraire, C. (2015). Enhancing Student Experience in Team-Based Project Courses Using Essence Reflection Meetings. *Software Eng Educ Proc*, *2015-August*, 10-12. Scopus. https://doi.org/10.1109/CSEET.2015.10

Sein, M. K., Henfridsson, O., Purao, S., Rossi, M., & Lindgren, R. (2011). Action design research. In *MIS Quarterly : Management Information Systems* (Vol. 35, Numéro 1, p. 37-56).

Silva, T., Rodrigues, L., & Costa, A. (2020). *Improving Project Transparency with the Essence Framework. Journal of Systems and Software*. 78-88.

Simonette, M. J., Magalhaes, M. E. S., & Spina, E. (2016). Extending essence kernel to deal with IEEE code of ethics. In Fernandez M.F., Lefranc G.H., & Perez R. (Éds.), *CHILECON - IEEE CHILEAN Conf. Electr., Electron. Eng., Inf. Commun. Technol., Proc. IEEE Chilecon* (p. 383-387). Institute of Electrical and Electronics Engineers Inc.; Scopus. https://doi.org/10.1109/Chilecon.2015.7400405

Smith, J. (2014). Ontology Engineering : Building Ontologies with Basic Formal Ontology. *MIT Press.*

Smith, J., & Garcia, M. (2016). Defining Classes, Properties, and Relationships within the BPMN 2.0 Based Ontology (BBO). *International Journal of Ontology and Semantic Web,* 210-232.

Staab, S., & Studer, R. (2009). Handbook on ontologies. *Springer Science & Business Media.*

Studer, R., Benjamins, V. R., & Fensel, D. (1998). Knowledge engineering : Principles and methods. In *Knowledge engineering : Principles and methods.: Vol. 25(1-2)* (Data&knowledge engineering, p. 161-197).

Sutherland, J. (2014). Scrum : The Art of Doing Twice the Work in Half the Time. *Crown Business.*

Tarafdar, M., & Qrunfleh, S. (2010). Examining tactical information technology—Business alignment. *Journal of Computer Information Systems*, *50*(4), 107-116.

Taryn, J. B.-B., Lizelle, F., & Herman, S. (2017). *Linking trust and collaboration in project teams to project management success.*

Thomas, J., & Mengel, T. (2008). *Preparing Project Managers to Deal with Complexity– Advanced Project Management Education. International Journal of Project Management, 26(3), 304-315.* 304-315.

Thomas, O., & Fellmann, M. (2012). Business Process Management : Savings Through the Cloud. *Journal of Business Process Management,* 89-100.

Thompson, R., Lee, J., & Anderson, H. (2022). *Integrating Essence with Existing Practices. Software Development Review*. 198-215.

Turner, J. R. (2016). *Gower Handbook of Project Management* (5th éd.). Gower Publishing.

Turner, M., Budgen, D., & Brereton, P. (2003). Turning software into a service. *Computer*, *36*(10), 38-44. https://doi.org/10.1109/mc.2003.1236470

Uschold, M., & Gruninger, M. (1996). *Ontologies : Principles, methods, and applications.* (Knowledge Engineering Review).

Uysal, M. P. (2018). A Formal Method for Mapping Software Engineering Practices to Essence. *arXiv preprint arXiv:1812.01791*. https://arxiv.org/pdf/1812.01791

Vasconcelos, J., Kimble, C., & Rocha, A. (2011). Ontologies and the Dynamics of Organizational Environments. *Journal of Knowledge Management*, 229-248.

Venkatesh, A. (2008). Digital home technologies and transformation of households. *Information Systems Frontiers*, *10*(4), 391-395. Scopus. https://doi.org/10.1007/s10796-008-9097-0

Vrandecic, D., & Tane, J. (2008). The ontology engineering environment. *Handbook on Ontologies*.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., & Hübner. (2001). *Ontology-based integration of information-a survey of existing approaches. IJCAI-01 Workshop : Ontologies and Information Sharing*.

Wand, Y., & Weber, R. (2002). Research Commentary : Information Systems and Conceptual Modeling—A Research Agenda. *Information Systems Research*, *13*(4), 363-376. https://doi.org/10.1287/isre.13.4.363.69

Wang, H., Wache, H., & Schreiber. (2006). TopBraid Composer : A visual toolkit for semantic web development. *Proceedings of the 5th International Semantic Web Conference*.

Weichbroth, P. (2022). A Case Study on Implementing Agile Techniques and Practices : Rationale, Benefits, Barriers and Business Implications for Hardware Development. *Applied Sciences*, *12*(17). https://doi.org/10.3390/app12178457

Weiss, J. W., & Thorogood, A. (2011). Information technology (IT)/Business alignment as a strategic weapon : A diagnostic tool. *EMJ - Engineering Management Journal*, *23*(2), 30-41.

White, M. (2024). Introduction to the mixed-up politics of disinformation, anti-feminisms, and misogyny. *Feminist Media Studies*. Scopus. https://doi.org/10.1080/14680777.2023.2284099

Winter, R., Sinz, E. J., Kurbel, K., Dyllick, T., Schmidt, H.-U., & Eymann, T. (2011). Accreditation of Study Programs in Business and Information Systems

Engineering : Towards more Competitiveness or Towards more Bureaucracy? *Business & Information Systems Engineering*, *1*(4), 331-338.

Wysocki, R. K. (2014). Effective Project Management : Traditional, Agile, Extreme (7th ed.). *Wiley*.

Xu, J., He, M., & Jiang, Y. (2022). *A novel framework of knowledge transfer system for construction projects based on knowledge graph and transfer learning*.

Xue, R., Baron, C., Esteban, P., & Zheng, L. (2015). Analysis and Comparison of Project Management Standards and Guides. *Recent Advances on Mechanics, Materials, Mechanical Engineering and Chemical Engineering, 15-22*. https://inase.org/library/2015/books/bypaper/MMMCE/MMMCE-01.pdf

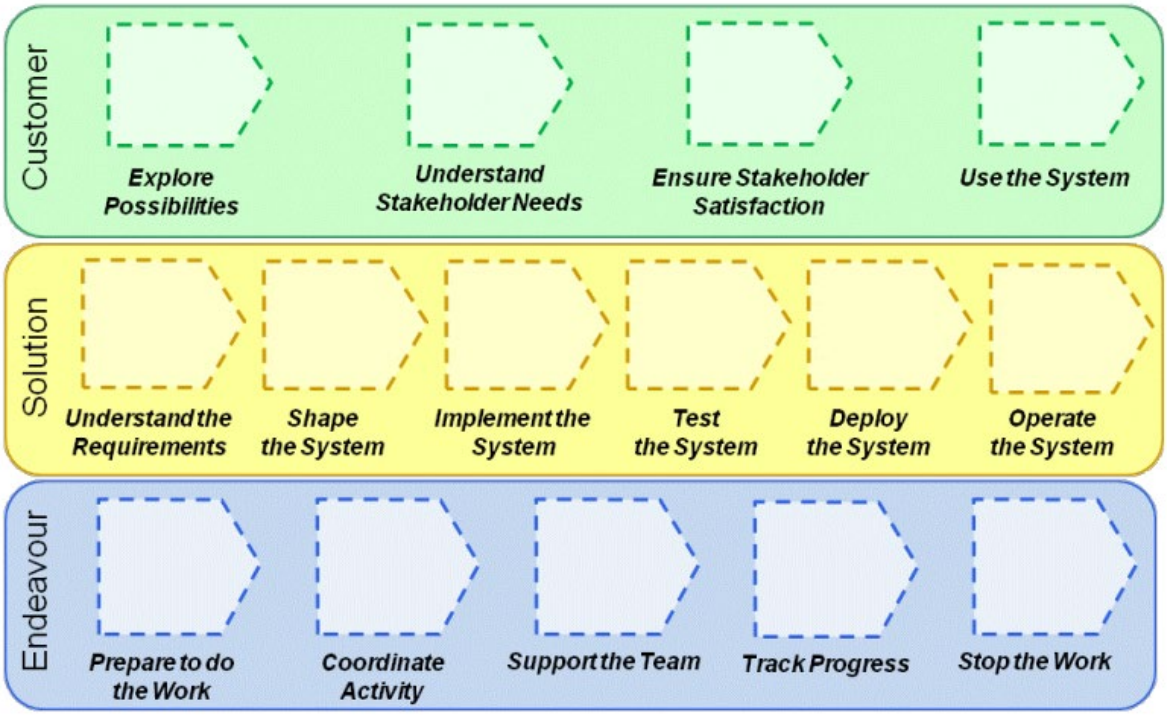
Zadeh, L. A. (1965). *Fuzzy sets.* (Information and Control, p. 338-353.).

Zhu, K., Dong, S., Xu, S. X., & Kraemer, K. L. (2006). Innovation diffusion in global contexts : Determinants of post-adoption digital transformation of European companies. *European Journal of Information Systems*, *15*(6), 601-616. Scopus. https://doi.org/10.1057/palgrave.ejis.3000650

Zmeev, D. O., & Zmeev, O. A. (2020). Project-Oriented Course of Software Engineering Based on Essence. In Daun M., Hochmuller E., Krusche S., Brugge B., & Tenbergen B. (Éds.), *IEEE Conf. Softw. Eng. Educ. Train., CSEE T* (p. 296-298). Institute of Electrical and Electronics Engineers Inc.; Scopus. https://doi.org/10.1109/CSEET49119.2020.9206240

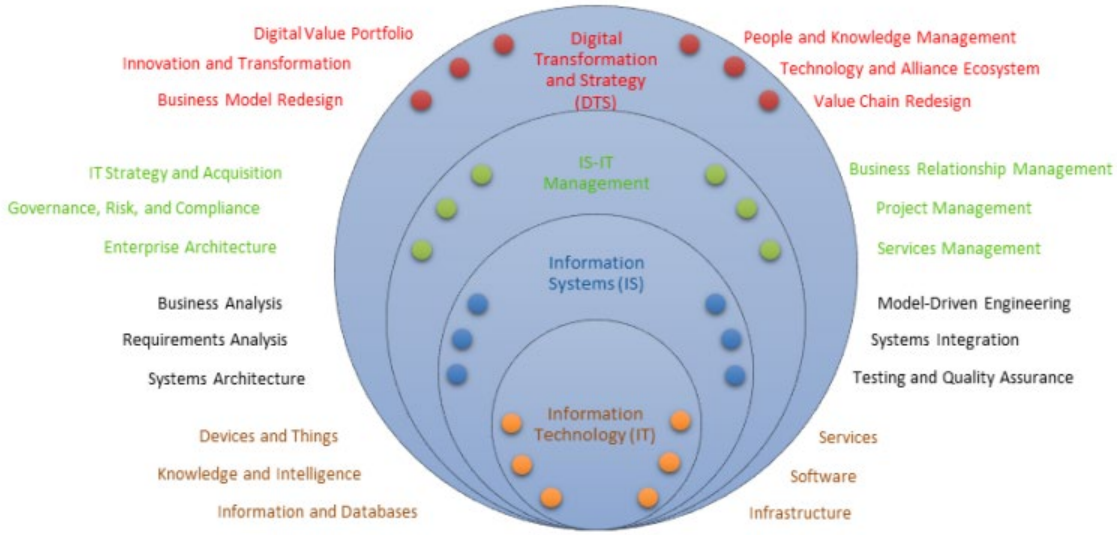
8.1 Appendices

Essence Kernel Activity Spaces

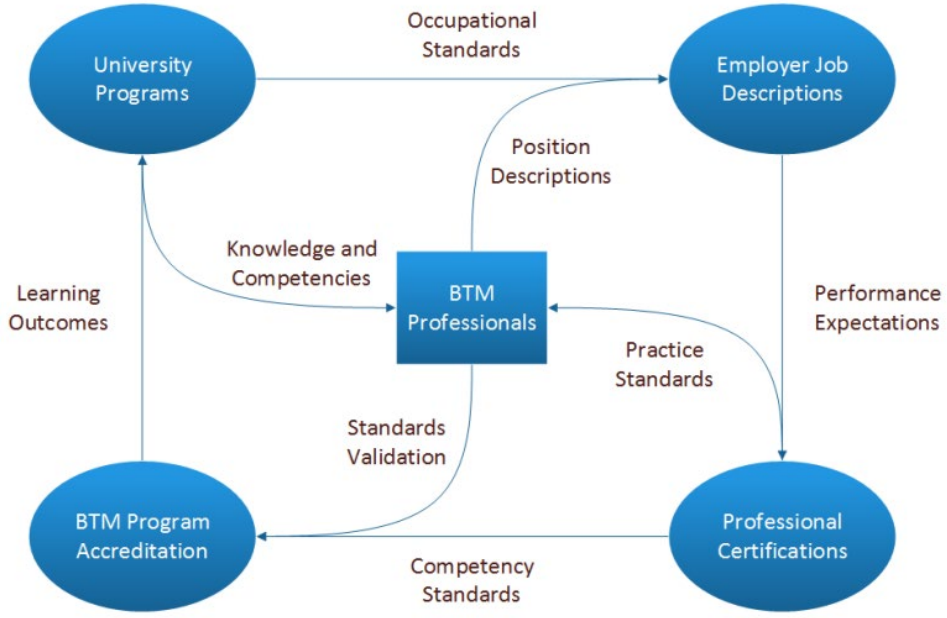


BTM BoK Framework

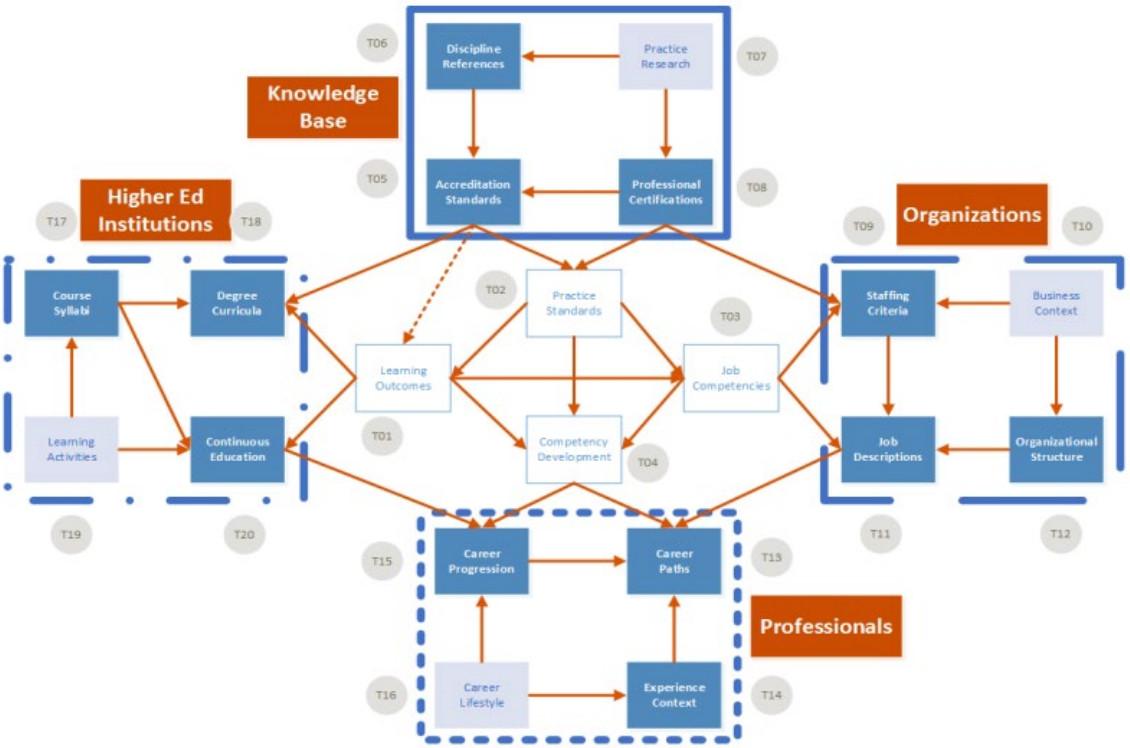
BTM Scope and Integration of DTS, IS-IT Management, IS, and IT



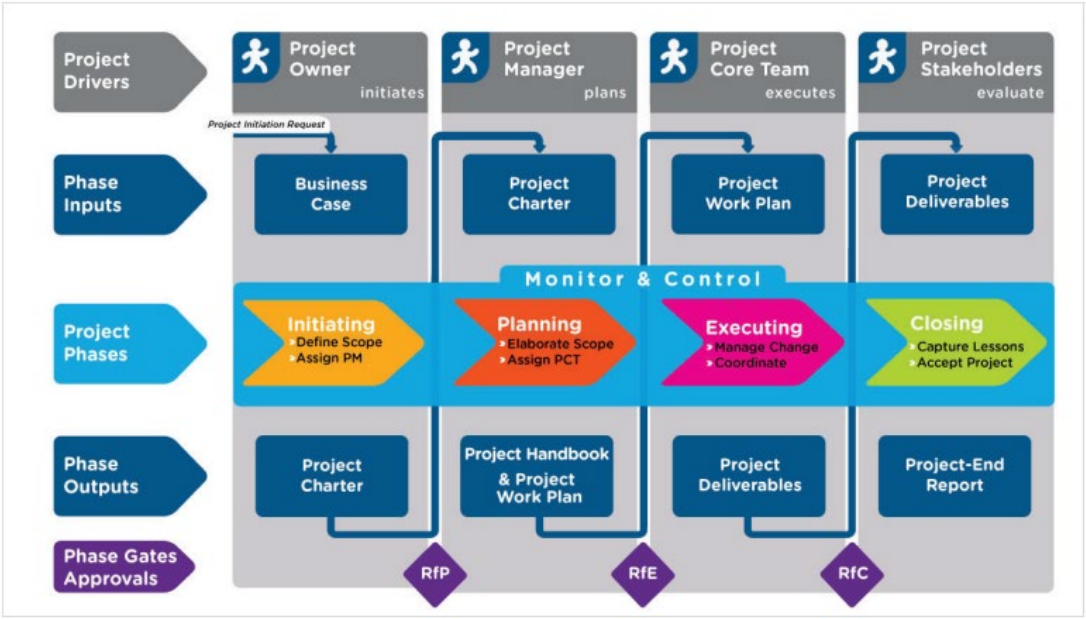
BTM Professionals and Community Learning Dynamics



BTM BOK Customization Opportunities



PM² Swimlane Diagram



Agile Artefacts and Ceremonies Summary Tables and Diagrams

Initiating	BM	PM	TeCo	PrOw	ArOw	ATeM
Business Case	R	S	-	-	-	-
Project Charter	S	R	-	-	-	-
Architecture Overview	I	A	S	I	R	S
Operational Model	I	A	S	I	R	S
Planning	**BM**	**PM**	**TeCo**	**PrOw**	**ArOw**	**ATeM**
Development Handbook	I	A	R	S	S	S
Development Work Plan	I	A	R	S	S	S
• Work Items List	S	A	C	R	S	S
• Release Plan	I	A	R	S	S	S
• Iteration Plan	I	A	R	C	S	S
Test Plan	I	A	S	S	S	R
Deployment Plan	I	A	R	S	S	S
Iteration Planning meeting	I	I	S	S/A	S	R
Executing	**BM**	**PM**	**TeCo**	**PrOw**	**ArOw**	**ATeM**
Development Status Report	I	A	R	I	I	S
Retrospectives Log	I	A	R	I	I	S
Testing Logs	I	A	R	I	S	S
Daily Stand-up meeting	N/A	I	S	I	R	R/A
Iteration Review meeting	C	I	S	S/A	R	R
Iteration Retrospective meeting	I	I	S	I	R	R/A
Monitor & Control	**BM**	**PM**	**TeCo**	**PrOw**	**ArOw**	**ATeM**
Release Planning						
• Manage Work Items List	S	I	S	R/A	I	I/S
• Work Items List refinement	I	I	S	R	I	S/A
• Work Items List prioritisation	S	I	S	R/A	C	C
Closing	**BM**	**PM**	**TeCo**	**PrOw**	**ArOw**	**ATeM**
Project-End Report	S	R	S	I	I	I

Class	Description
Overview	This section provides the purpose, objectives, profession, adoption, customization, and community aspects of the BTM BOK.
Transformation	Focuses on digital leadership, covering opportunity, decision-making, and accountability for digital transformation initiatives.
Practice	Encompasses the organizational layers impacted by digital transformation, including governance, compliance, architecture, security, platform, people, project, and agility.
Discipline	Foundational academic disciplines and research areas related to BTM, such as strategy, marketing, operations, innovation, and performance.
Lifecycle	Covers the scope, focus, and sector-specific aspects of digital transformation, including administration, solution, support, facility, and enterprise.
Career	Addresses career goals, paths, and progression in BTM, including specialist, complement, generalist, senior, and occasional roles.
Standard	Includes accreditation, benchmark, and certification standards relevant to BTM, such as diploma, certificate, bachelor, master, and doctorate.

Table 8 : BTM BOK Ontology classes

Class	Subclass		Description
Overview	Purpose	Objectives	Goals of the BTM BOK in guiding and supporting IS-IT professionals.
Overview	Purpose	Profession	Defining the profession, standards, and community dynamics of BTM.

Overview	Purpose	Adoption	Process and benefits of adopting BTM BOK for different groups.
Overview	Purpose	Customization	Exploratory and experimental process to adapt BTM BOK to specific needs.
Overview	Purpose	Community	Engagement and collaboration within the BTM professional community.
Overview	Contents	Metamodel	
Overview	Contents	Structure	
Overview	Contents	Components	
Overview	Contents	Dependencies	
Overview	Contents	Source	
Overview	Methodology	Licenses	
Overview	Methodology	References	
Overview	Methodology	Editing	
Overview	Methodology	Contribution	
Overview	Methodology	Authors	

Table 9 : BTM BOK Ontology sub-classes (Overview)

Class	Subclass	Individual	Description
Practice	Fabric	Governance	Roles for digital strategy, risk management, and market/customer focus.
Practice	Fabric	Compliance	Roles for digital regulatory assurance, core requirements, and business continuity.

Practice	Fabric	Architecture	Roles for digital enterprise architecture, business architecture, and model-driven approaches.
Practice	Fabric	Security	Roles for digital cybersecurity, networks and devices, end-user safety, and privacy.
Practice	Fabric	Platform	Roles for digital service management, data centers, cloud apps, and e-commerce.
Practice	Team	People	Roles for digital user experience, behavior, interfaces, learning, and talent.
Practice	Team	Project	Roles for digital project, program, and portfolio management and leadership.
Practice	Team	Agility	Roles for digital agile methods, scrum, DevOps, test-driven development, and open source.
Practice	Team	Engineering	Roles for digital system analysis, design-build-test, deployment, and maintenance.
Practice	Team	Integration	Roles for digital cloud apps, web portals, web services, APIs, and enterprise systems.
Practice	Outcome	Value	Roles for digital product innovation, design thinking, and entrepreneurship.
Practice	Outcome	Process	Roles for digital business analysis, process redesign, integration, and optimization.
Practice	Outcome	Rule	Roles for digital rules, process logic, ontologies, reasoning, and automation.
Practice	Outcome	Data	Roles for digital information, databases, content and archives, metadata, and licenses.
Practice	Outcome	Intelligence	Roles for digital business intelligence, artificial intelligence, and knowledge extraction.

Table 10 : BTM BOK Ontology sub-classes (Practice)

Class	Subclass	Individual	Description

Discipline	Business	Strategy	Roots in strategic management, responsible governance, and economics.
Discipline	Business	Marketing	Roots in marketing strategy, customer relations, sales, and advertising.
Discipline	Business	Operations	Roots in operations management, quality, and process reengineering.
Discipline	Business	Innovation	Roots in technology and innovation management, alliances, and projects.
Discipline	Business	Performance	Roots in management accounting, value engineering, and product design.
Discipline	Management	Talent	Roots in talent management and organizational behavior.
Discipline	Management	Learning	Roots in organizational learning and development.
Discipline	Management	Change	Roots in change management and transformation.
Discipline	Management	Leadership	Roots in leadership theories and practices.
Discipline	Management	Entrepreneurship	Roots in entrepreneurial management and innovation.
Discipline	Technology	System	Roots in systems theory and design.
Discipline	Technology	Software	Roots in software engineering and development.
Discipline	Technology	Cloud	Roots in cloud computing and services.
Discipline	Technology	IoT	Roots in Internet of Things (IoT) technologies.
Discipline	Technology	AI	Roots in artificial intelligence and machine learning.

Table 11 : BTM BOK Ontology sub-classes (Discipline)

Class	Subclass	Individual	Description

Lifecycle	Scope	Administration	Scope of administration in digital transformation.
Lifecycle	Scope	Solution	Scope of solution design and deployment.
Lifecycle	Support	Support	Scope of support services and infrastructure.
Lifecycle	Scope	Facility	Scope of facilities management in digital contexts.
Lifecycle	Scope	Enterprise	Scope of enterprise-wide digital initiatives.
Lifecycle	Focus	Behavior	Focus on user behavior and interaction.
Lifecycle	Focus	Functionality	Focus on system functionality and performance.
Lifecycle	Focus	Reengineering	Focus on process reengineering and optimization.
Lifecycle	Focus	Optimization	Focus on continuous improvement and optimization.
Lifecycle	Focus	Diversification	Focus on diversification of digital services and products.
Lifecycle	Sector	Resource	Sector-specific practices for resource management.
Lifecycle	Sector	Infrastructure	Sector-specific practices for digital infrastructure.
Lifecycle	Sector	Product	Sector-specific practices for product management.
Lifecycle	Sector	Service	Sector-specific practices for service management.
Lifecycle	Sector	Public	Sector-specific practices for public sector digital transformation.

Table 12 : BTM BOK Ontology sub-classes (Lifecycle)

Class	Subclass	Individual	Description
Career	Goal	Specialist	Career goals for specialists in BTM.
Career	Goal	Complement	Career goals for complementary roles in BTM.
Career	Goal	Generalist	Career goals for generalists in BTM.
Career	Goal	Senior	Career goals for senior professionals in BTM.
Career	Goal	Occasional	Career goals for occasional roles in BTM.
Career	Path	Corporate	Career paths in corporate environments.
Career	Path	Embedded	Career paths in embedded systems.
Career	Path	Small	Career paths in small enterprises.
Career	Path	Startup	Career paths in startup environments.
Career	Path	Consulting	Career paths in consulting.

Career	Progression	Beginning	Career progression for entry-level roles.
Career	Progression	Diversity	Career progression through diverse experiences.
Career	Progression	Education	Career progression through continuous education.
Career	Progression	Experience	Career progression through practical experience.
Career	Progression	Promotion	Career progression through promotions.

Table 13 : BTM BOK Ontology sub-classes (Career)

Class	Subclass	Individual	Description
Standard	Accreditation	Diploma	Accreditation for diploma programs.
Standard	Accreditation	Certificate	Accreditation for certificate programs.
Standard	Accreditation	Bachelor	Accreditation for bachelor's degree programs.
Standard	Accreditation	Master	Accreditation for master's degree programs.
Standard	Accreditation	Doctorate	Accreditation for doctorate programs.
Standard	Benchmark	Forerunner	Benchmarking for forerunners in the field.
Standard	Benchmark	Challenger	Benchmarking for challengers.
Standard	Benchmark	Innovator	Benchmarking for innovators.
Standard	Benchmark	Optimizer	Benchmarking for optimizers.
Standard	Benchmark	Disruptor	Benchmarking for disruptors.
Standard	Certification	Associate	Certification for associate-level professionals.
Standard	Certification	Professional	Certification for professional-level individuals.
Standard	Certification	Manager	Certification for managers.
Standard	Certification	Entrepreneur	Certification for entrepreneurs.
Standard	Certification	Executive	Certification for executives.

Table 14: BTM BOK Ontology sub-classes (Standard)

Class	Subclass	Object Property (Relationship)
Overview	Purpose	hasObjective, belongsToProfession
Overview	Objectives	supportsProfession, guidesProfessionals
Overview	Profession	definesStandards, engagesCommunity
Overview	Adoption	benefitsGroups, guidesAdoption
Overview	Customization	supportsExploration, adaptsNeeds
Overview	Community	engagesMembers, supportsCollaboration

Table 15: BTM BOK Ontology Object Property (Overview)

Class	Subclass	Individual	Object Property (Relationship)
Transformation	Opportunity	Scan	identifiesOpportunities, requiresTeamEffort
Transformation	Opportunity	Discover	definesProjects, assessesImpact
Transformation	Opportunity	Prioritize	targetsProjects, assessesFeasibility
Transformation	Opportunity	Finance	demonstratesROI, supportsFinancing
Transformation	Opportunity	Benefit	ensuresValueFlow, gathersFeedback
Transformation	Decision	Who	staffsProjects, enhancesCreativity
Transformation	Decision	What	designsProducts, integratesStrategy
Transformation	Decision	Why	justifiesProjects, supportsChampions
Transformation	Decision	Where	selectsUnits, managesRisk
Transformation	Decision	How	appliesPractices, trainsStaff
Transformation	Accountability	Steer	guidesStrategy, ensuresCompliance
Transformation	Accountability	Explore	experimentsProducts, testsSolutions
Transformation	Accountability	Align	alignsNeeds, ensuresValue
Transformation	Accountability	Implement	deploysSolutions, enhancesProducts
Transformation	Accountability	Optimize	operatesPlatforms, ensuresInnovation

Table 16: BTM BOK Ontology Object Property (Transformation)

Class	Subclass	Individual	Object Property (Relationship)
Practice	Fabric	Governance	managesStrategy, controlsRisk

Practice	Fabric	Compliance	ensuresRegulations, maintainsContinuity
Practice	Fabric	Architecture	definesArchitecture, modelsDriven
Practice	Fabric	Security	ensuresCybersecurity, protectsData
Practice	Fabric	Platform	managesServices, supportsEcommerce
Practice	Team	People	managesUX, developsTalent
Practice	Team	Project	managesProjects, leadsTeams
Practice	Team	Agility	implementsAgile, supportsDevOps
Practice	Team	Engineering	designsSystems, maintainsSolutions
Practice	Team	Integration	integratesSystems, supportsAPIs
Practice	Outcome	Value	createsInnovation, supportsDesignThinking
Practice	Outcome	Process	analyzesBusiness, optimizesProcesses
Practice	Outcome	Rule	definesRules, automatesLogic
Practice	Outcome	Data	managesInformation, supportsMetadata
Practice	Outcome	Intelligence	extractsKnowledge, appliesAI

Table 17: BTM BOK Ontology Object Property (Practice)

Class	**Subclass**	**Individual**	**Object Property (Relationship)**
Discipline	Business	Strategy	guidesManagement, supportsGovernance
Discipline	Business	Marketing	strategizesMarketing, enhancesSales
Discipline	Business	Operations	managesOperations, improvesQuality
Discipline	Business	Innovation	drivesTechnology, supportsAlliances
Discipline	Business	Performance	evaluatesPerformance, designsProducts
Discipline	Management	Talent	managesTalent, developsSkills
Discipline	Management	Learning	supportsLearning, enhancesDevelopment
Discipline	Management	Change	managesChange, drivesTransformation
Discipline	Management	Leadership	guidesLeadership, supportsInitiatives
Discipline	Management	Entrepreneurship	supportsEntrepreneurship, drivesInnovation
Discipline	Technology	System	designsSystems, supportsIntegration
Discipline	Technology	Software	developsSoftware, ensuresQuality

Discipline	Technology	Cloud	managesCloud, supportsServices
Discipline	Technology	IoT	integratesIoT, supportsDevices
Discipline	Technology	AI	developsAI, appliesML

Table 18: BTM BOK Ontology Object Property (Discipline)

Class	**Subclass**	**Individual**	**Object Property (Relationship)**
Lifecycle	Scope	Administration	managesAdmin, supportsOperations
Lifecycle	Scope	Solution	designsSolutions, deploysTechnologies
Lifecycle	Support	Support	providesSupport, maintainsInfrastructure
Lifecycle	Scope	Facility	managesFacilities, supportsLogistics
Lifecycle	Scope	Enterprise	guidesEnterprise, supportsInitiatives
Lifecycle	Focus	Behavior	analyzesBehavior, improvesInteraction
Lifecycle	Focus	Functionality	enhancesFunctionality, supportsPerformance
Lifecycle	Focus	Reengineering	redesignsProcesses, optimizesSystems
Lifecycle	Focus	Optimization	improvesEfficiency, supportsContinuousImprovement
Lifecycle	Focus	Diversification	supportsDiversification, developsProducts
Lifecycle	Sector	Resource	managesResources, supportsSustainability
Lifecycle	Sector	Infrastructure	developsInfrastructure, supportsOperations
Lifecycle	Sector	Product	managesProducts, supportsDevelopment
Lifecycle	Sector	Service	managesServices, supportsDelivery
Lifecycle	Sector	Public	managesServices, supportsDelivery

Table 19: BTM BOK Ontology Object Property (Lifecycle)

Class	**Subclass**	**Individual**	**Object Property (Relationship)**
Career	Goal	Specialist	developsSpecialists, enhancesSkills
Career	Goal	Complement	supportsComplementaryRoles, enhancesCapabilities
Career	Goal	Generalist	developsGeneralists, broadensSkills
Career	Goal	Senior	supportsSeniorRoles, enhancesLeadership

Career	Goal	Occasional	supportsOccasionalRoles, enhancesFlexibility
Career	Path	Corporate	supportsCorporatePaths, enhancesCareerGrowth
Career	Path	Embedded	supportsEmbeddedRoles, enhancesIntegration
Career	Path	Small	supportsSmallBusiness, enhancesGrowth
Career	Path	Startup	supportsStartups, enhancesInnovation
Career	Path	Consulting	supportsConsultingRoles, enhancesExpertise
Career	Progression	Beginning	supportsEntryLevel, enhancesLearning
Career	Progression	Diversity	supportsDiverseRoles, enhancesExperiences
Career	Progression	Education	supportsEducationalGrowth, enhancesKnowledge
Career	Progression	Experience	supportsExperienceGrowth, enhancesSkills
Career	Progression	Promotion	supportsPromotions, enhancesCareerProgression

Table 20: BTM BOK Ontology Object Property (Career)

Class	**Subclass**	**Individual**	**Object Property (Relationship)**
Standard	Accreditation	Diploma	supportsDiplomaAccreditation, enhancesRecognition
Standard	Accreditation	Certificate	supportsCertificateAccreditation, enhancesProfessionalism
Standard	Accreditation	Bachelor	supportsBachelorAccreditation, enhancesEducation
Standard	Accreditation	Master	supportsMasterAccreditation, enhancesExpertise
Standard	Accreditation	Doctorate	supportsDoctorateAccreditation, enhancesResearch
Standard	Benchmark	Forerunner	benchmarksForerunners, supportsLeadership
Standard	Benchmark	Challenger	benchmarksChallengers, enhancesCompetitiveness
Standard	Benchmark	Innovator	benchmarksInnovators, supportsCreativity
Standard	Benchmark	Optimizer	benchmarksOptimizers, enhancesEfficiency
Standard	Benchmark	Disruptor	benchmarksDisruptors, supportsInnovation
Standard	Certification	Associate	certifiesAssociates, supportsProfessionalGrowth

Standard	Certification	Professional	certifiesProfessionals, enhancesExpertise
Standard	Certification	Manager	certifiesManagers, supportsLeadership
Standard	Certification	Entrepreneur	certifiesEntrepreneurs, enhancesInnovation
Standard	Certification	Executive	certifiesExecutives, supportsStrategicLeadership

Table 21: BTM BOK Ontology Object Property (Standard)

Data Property	Domain	Range
hasPurposeDescription	Purpose	string
hasObjectiveDetails	Objectives	string
hasProfessionStandard	Profession	string
hasAdoptionBenefit	Adoption	string
hasCustomizationOption	Customization	string
hasCommunityMember	Community	string

Table 22: BTM BOK Ontology Data Property (Classes)

Data Property	Domain	Sub-domain	Range
hasOpportunityScanDetail	Opportunity	Scan	String
hasDiscoveryDetail	Opportunity	Discover	String
hasPrioritizationCriteria	Opportunity	Prioritize	String
hasFinanceDetail	Opportunity	Finance	String
hasBenefitDetail	Opportunity	Benefit	String
hasDecisionDetailWho	Decision	Who	String
hasDecisionDetailWhat	Decision	What	String
hasDecisionDetailWhy	Decision	Why	String
hasDecisionDetailWhere	Decision	Where	String
hasDecisionDetailHow	Decision	How	String
hasAccountabilitySteerDetail	Accountability	Steer	String
hasAccountabilityExploreDetail	Accountability	Explore	String
hasAccountabilityAlignDetail	Accountability	Align	String
hasAccountabilityImplementDetail	Accountability	Implement	String

hasAccountabilityOptimizeDetail	Accountability	Optimize	String

Table 23: BTM BOK Ontology Data Property (Transformation)

Data Property	Domain	Sub-domain	Range
hasGovernanceDetail	Fabric	Governance	String
hasComplianceDetail	Fabric	Compliance	String
hasArchitectureDetail	Fabric	Architecture	String
hasSecurityDetail	Fabric	Security	String
hasPlatformDetail	Fabric	Platform	String
hasPeopleDetail	Team	People	String
hasProjectDetail	Team	Project	String
hasAgilityDetail	Team	Agility	String
hasEngineeringDetail	Team	Engineering	String
hasIntegrationDetail	Team	Integration	String
hasValueDetail	Outcome	Value	String
hasProcessDetail	Outcome	Process	String
hasRuleDetail	Outcome	Rule	String
hasDataDetail	Outcome	Data	String
hasIntelligenceDetail	Outcome	Intelligence	String

Table 24: BTM BOK Ontology Data Property (Practice)

Data Property	Domain	Sub-domain	Range
hasStrategyDetail	Business	Strategy	String
hasMarketingDetail	Business	Marketing	String
hasOperationDetail	Business	Operations	String
hasInnovationDetail	Business	Innovation	String
hasPerformanceDetail	Business	Performance	String
hasTalentDetail	Management	Talent	String
hasLearningDetail	Management	Learning	String
hasChangeDetail	Management	Change	String

hasLeadershipDetail	Management	Leadership	String
hasEntrepreneurshipDetail	Management	Entrepreneurship	String
hasSystemDetail	Technology	System	String
hasSoftwareDetail	Technology	Software	String
hasCloudDetail	Technology	Cloud	String
hasIoTDetail	Technology	IoT	String
hasAIDetail	Technology	AI	String

Table 25: BTM BOK Ontology Data Property (Discipline)

Data Property	**Domain**	**Sub-domain**	**Range**
hasSupportDetail	Scope	Support	String
hasFacilityDetail	Scope	Facility	String
hasEnterpriseDetail	Scope	Enterprise	String
hasBehaviorDetail	Focus	Behavior	String
hasFunctionalityDetail	Focus	Functionality	String
hasReengineeringDetail	Focus	Reengineering	String
hasOptimizationDetail	Focus	Optimization	String
hasDiversificationDetail	Focus	Diversification	String
hasResourceDetail	Sector	Resource	String
hasInfrastructureDetail	Sector	Infrastructure	String
hasProductDetail	Sector	Product	String
hasServiceDetail	Sector	Service	String
hasPublicDetail	Sector	Public	String

Table 26: BTM BOK Ontology Data Property (Lifecycle)

Data Property	**Domain**	**Sub-domain**	**Range**
hasSpecialistDetail	Goal	Specialist	String
hasComplementDetail	Goal	Complement	String
hasGeneralistDetail	Goal	Generalist	String
hasSeniorDetail	Goal	Senior	String

hasOccasionalDetail	Goal	Occasional	String
hasCorporatePathDetail	Path	Corporate	String
hasEmbeddedPathDetail	Path	Embedded	String
hasSmallPathDetail	Path	Small	String
hasStartupPathDetail	Path	Startup	String
hasConsultingPathDetail	Path	Consulting	String
hasBeginningDetail	Progression	Beginning	String
hasDiversityDetail	Progression	Diversity	String
hasEducationDetail	Progression	Education	String
hasExperienceDetail	Progression	Experience	String
hasPromotionDetail	Progression	Promotion	String

Table 27: BTM BOK Ontology Data Property (Career)

Data Property	Domain	Sub-domain	Range
hasDiplomaDetail	Accreditation	Diploma	String
hasCertificateDetail	Accreditation	Certificate	String
hasBachelorDetail	Accreditation	Bachelor	String
hasMasterDetail	Accreditation	Master	String
hasDoctorateDetail	Accreditation	Doctorate	String
hasForerunnerDetail	Benchmark	Forerunner	String
hasChallengerDetail	Benchmark	Challenger	String
hasInnovatorDetail	Benchmark	Innovator	String
hasOptimizerDetail	Benchmark	Optimizer	String
hasDisruptorDetail	Benchmark	Disruptor	String
hasAssociateDetail	Certification	Associate	String
hasProfessionalDetail	Certification	Professional	String
hasManagerDetail	Certification	Manager	String
hasEntrepreneurDetail	Certification	Entrepreneur	String
hasExecutiveDetail	Certification	Executive	String

Table 28: BTM BOK Ontology Data Property (Standard)

Essence Framework

Class	Description
Stakeholders	The people, groups, or organizations that affect or are affected by a software system.
Opportunity	The set of circumstances that makes it appropriate to develop or change a software system.
Requirements	What the software system must do to address the opportunity and satisfy the stakeholders.
Software System	A system made up of software, hardware, and data that provides its primary value by the execution of software.
Team	The group of people actively engaged in the development, maintenance, delivery, or support of a software system.
Work	All mental and physical activities performed by the team to produce a software system.
Way of Working	The tailored set of practices and tools used by a team to guide and support their work.
Alpha	An essential element of the software engineering endeavor relevant to assessing progress and health.
Activity	Defines one or more kinds of work items and provides guidance on how to perform these activities.
Competency	Encompasses the abilities, capabilities, attainments, knowledge, and skills necessary to do certain kinds of work.

Table 29: Essence Ontology Class

Class	Subclass	Description
Stakeholder	InternalStakeholder	Stakeholders within the organization (e.g., employees, management).
Stakeholder	ExternalStakeholder	Stakeholders outside the organization (e.g., customers, partners).
Stakeholder	PrimaryStakeholder	Stakeholders directly affected by the project outcomes.
Stakeholder	SecondaryStakeholder	Stakeholders indirectly affected by the project outcomes.

Stakeholder	KeyStakeholder	Stakeholders with significant influence or decision-making power.
Opportunity	MarketOpportunity	Opportunity arising from market conditions.
Opportunity	TechnologicalOpportunity	Opportunity arising from technological advancements.
Opportunity	RegulatoryOpportunity	Opportunity arising from changes in regulations.
Opportunity	StrategicOpportunity	Opportunity aligned with strategic goals.
Opportunity	OperationalOpportunity	Opportunity for operational improvements.
Requirements	FunctionalRequirement	Specific behaviors or functions the software system must perform.
Requirements	NonFunctionalRequirement	Quality attributes, constraints, and standards the software system must meet.
Requirements	BusinessRequirement	Requirements related to business needs and objectives.
Requirements	SystemRequirement	Requirements related to the system's functionality and performance.
Requirements	UserRequirement	Requirements related to user needs and usability.
SoftwareSystem	WebApplication	A software system designed for web environments.
SoftwareSystem	MobileApplication	A software system designed for mobile devices.
SoftwareSystem	DesktopApplication	A software system designed for desktop computers.
SoftwareSystem	EmbeddedSystem	A software system designed for embedded devices.
SoftwareSystem	EnterpriseSystem	A large-scale software system designed for organizational use.
Team	DevelopmentTeam	Team responsible for software development.
Team	QATeam	Team responsible for quality assurance and testing.
Team	UXTeam	Team responsible for user experience design.
Team	DevOpsTeam	Team responsible for development and operations integration.
Team	SupportTeam	Team responsible for customer support and maintenance.
Work	Task	Specific units of work to be performed.
Work	Activity	Higher-level grouping of tasks related to a particular goal.
Work	Milestone	Significant points or events in the project timeline.
Work	Deliverable	Tangible or intangible output produced as a result of work.
Work	Phase	Distinct stages in the project lifecycle.

WayOfWorking	AgileMethodology	An agile methodology used by the team.
WayOfWorking	ScrumMethodology	A specific implementation of agile, focusing on Scrum practices.
WayOfWorking	WaterfallMethodology	A traditional, sequential methodology.
WayOfWorking	KanbanMethodology	A methodology focused on visualizing work and continuous delivery.
WayOfWorking	LeanMethodology	A methodology focused on optimizing efficiency and reducing waste.
Alpha	Stakeholders	The group of all individuals or organizations affected by the project.
Alpha	Opportunity	The potential benefits and value the project aims to deliver.
Alpha	Requirements	The specific needs and constraints the project must meet.
Alpha	SoftwareSystem	The system being developed or modified.
Alpha	Team	The group of people actively working on the project.
Alpha	Work	The tasks and activities performed to deliver the project.
Alpha	WayOfWorking	The set of practices and processes guiding the team's work.
Activity	PlanningActivity	Activities related to planning the project.
Activity	DevelopmentActivity	Activities related to developing the software.
Activity	TestingActivity	Activities related to testing the software.
Activity	DeploymentActivity	Activities related to deploying the software.
Activity	MaintenanceActivity	Activities related to maintaining and updating the software.
ActivitySpace	ExplorePossibilities	Activity space for identifying potential solutions.
ActivitySpace	UnderstandStakeholderNeeds	Activity space for gathering and analyzing stakeholder requirements.
ActivitySpace	EnsureStakeholderSatisfaction	Activity space for validating and verifying stakeholder satisfaction.
ActivitySpace	ManageRequirements	Activity space for managing and prioritizing requirements.
ActivitySpace	DefineArchitecture	Activity space for designing the system architecture.
Competency	StakeholderRepresentation	Competency in understanding and advocating for stakeholder needs.
Competency	Development	Competency in designing, coding, and testing software components.
Competency	Testing	Competency in planning, executing, and evaluating tests to ensure software quality.
Competency	Leadership	Competency in leading and managing teams.

Competency	ProjectManagement	Competency in planning, executing, and overseeing projects.
AlphaState	ConceptEstablished	State indicating that the concept has been established.
AlphaState	RequirementsDefined	State indicating that requirements have been defined.
AlphaState	DesignCompleted	State indicating that the design has been completed.
AlphaState	ImplementationInProgress	State indicating that implementation is in progress.
AlphaState	TestingCompleted	State indicating that testing has been completed.
AlphaState	Deployed	State indicating that the system has been deployed.
AlphaAssociation	RelatedTo	Relationship between two alphas.
AlphaAssociation	DependentOn	Indicates dependency between two alphas.
AlphaAssociation	AssociatedWith	Indicates association between two alphas.
AlphaAssociation	Influences	Indicates influence between two alphas.
AlphaAssociation	ConflictsWith	Indicates conflict between two alphas.
CompetencyLevel	Beginner	Basic level of proficiency in a competency.
CompetencyLevel	Intermediate	Intermediate level of proficiency in a competency.
CompetencyLevel	Advanced	Advanced level of proficiency in a competency.
CompetencyLevel	Expert	Expert level of proficiency in a competency.
Checkpoint	InitialReview	Initial review checkpoint.
Checkpoint	MidProjectReview	Mid-project review checkpoint.
Checkpoint	FinalReview	Final review checkpoint.
Checkpoint	QualityGate	Checkpoint for assessing quality criteria.
Checkpoint	MilestoneReview	Checkpoint for reviewing project milestones.
WorkProduct	RequirementsDocument	Document detailing the project requirements.
WorkProduct	DesignSpecification	Document detailing the system design.
WorkProduct	SourceCode	The actual codebase of the software system.
WorkProduct	TestPlan	Document detailing the test plan and cases.
WorkProduct	UserManual	Document providing instructions for end users.
Pattern	MVCPattern	Model-View-Controller design pattern.
Pattern	SingletonPattern	Design pattern that restricts the instantiation of a class to one object.
Pattern	ObserverPattern	Design pattern where an object maintains a list of dependents and notifies them of state changes.
Pattern	FactoryPattern	Design pattern for creating objects without specifying the exact class.

Pattern	StrategyPattern	Design pattern that enables selecting an algorithm's behavior at runtime.
Practice	CodeReviewPractice	A practice focused on reviewing and improving code quality.
Practice	TestDrivenDevelopmentPractice	A practice focused on writing tests before implementing code.
Practice	ContinuousIntegrationPractice	A practice focused on integrating code changes frequently.
Practice	PairProgrammingPractice	A practice where two programmers work together at one workstation.
Practice	RetrospectivePractice	A practice focused on reflecting and improving the team's processes.
Method	AgileMethod	An agile method combining various agile practices.
Method	ScrumMethod	A method implementing Scrum practices.
Method	WaterfallMethod	A method following the traditional waterfall approach.
Method	KanbanMethod	A method following Kanban practices.
Method	LeanMethod	A method following lean practices.
State	InitialState	The initial state of an element.
State	InProgressState	The state indicating an element is currently being worked on.
State	CompletedState	The state indicating an element is completed.
State	ReviewedState	The state indicating an element has been reviewed.
State	ApprovedState	The state indicating an element has been approved.
ActivityAssociation	Precedes	Indicates that one activity precedes another.
ActivityAssociation	Follows	Indicates that one activity follows another.
ActivityAssociation	DependsOn	Indicates that one activity depends on another.
ActivityAssociation	ParallelTo	Indicates that one activity runs in parallel with another.
ActivityAssociation	Blocks	Indicates that one activity blocks the execution of another.

Table 30: Essence Ontology Sub-Class

Object Property	Domain	Range	Description
hasStakeholder	SoftwareSystem	Stakeholder	Links a software system to its stakeholders.
hasRequirement	Opportunity	Requirements	Links an opportunity to the requirements it generates.
performedBy	Work	Team	Indicates the team that performs a specific work task.

partOfActivitySpace	Activity	ActivitySpace	Indicates that an activity is part of a specific activity space.
achievedState	Alpha	AlphaState	Indicates the current state of an alpha.
usesResource	Activity	Resource	Specifies the resources used by an activity.
hasTag	Element	Tag	Associates metadata (tags) with an element to provide additional information.
hasCompetency	Team	Competency	Indicates the competencies possessed by a team.
hasCompetencyLevel	Competency	CompetencyLevel	Specifies the level of proficiency for a particular competency.
associatedWith	Alpha	Alpha	Defines a relationship between two alphas.
relatedTo	Element	Element	Generic relationship between any two elements in the ontology.
definedBy	Method	Practice	Indicates the practices that define a method.
includesActivity	ActivitySpace	Activity	Specifies that an activity space includes specific activities.
hasCheckpoint	Activity	Checkpoint	Links an activity to a checkpoint that needs to be verified.
producesWorkProduct	Activity	WorkProduct	Indicates that an activity produces a specific work product.
hasPattern	Practice	Pattern	Associates a practice with a pattern.
hasPracticeAsset	Practice	PracticeAsset	Links a practice to its reusable components.
resolvedBy	Conflict	MergeResolution	Specifies how a conflict is resolved during merging.
belongsToLibrary	Practice	Library	Indicates that a practice is part of a collection of reusable methods and practices.
hasApproach	Activity	Approach	Links an activity to a strategy or technique for performing it.

hasCompletionCriterion	Activity	CompletionCriterion	Specifies the criteria for determining if an activity is complete.
hasEntryCriterion	Activity	EntryCriterion	Specifies the criteria for determining if an activity can be started.
extendsElement	ExtensionElement	Element	Indicates that an extension element extends the functionality of another element.
instantiatesMethod	MethodEnactment	Method	Links a method enactment to the method it applies.
usesPattern	TypedPattern	Pattern	Indicates that a typed pattern is based on a specific pattern.
usesResource	TypedResource	Resource	Specifies that a typed resource is used in a certain context.
usesTag	TypedTag	Tag	Associates a typed tag with specific metadata constraints.
extendsKernel	UserDefinedType	Kernel	Indicates that a user-defined type extends the kernel.
selectedFeature	ViewSelection	FeatureSelection	Specifies the features included in a particular view.
definesView	View	ElementGroup	Indicates that a view is defined by a group of related elements.
graphicallyRepresents	GraphicalSyntax	Element	Specifies the visual representation rules for a language element.
textuallyRepresents	TextualSyntax	Element	Specifies the textual representation rules for a language element.
hasDynamicBehavior	DynamicSemantics	LanguageElement	Indicates the operational behavior and interactions of a language element.
containedIn	ElementGroup	Element	Specifies that an element is part of an element group.
hasStateTransition	State	Transition	Links a state to its possible transitions.
involvesStakeholder	Activity	Stakeholder	Specifies that an activity involves certain stakeholders.

requiresCompetency	Activity	Competency	Indicates the competencies required to perform an activity.
contributesToAlpha	WorkProduct	Alpha	Specifies that a work product contributes to the progress or health of an alpha.
validatedBy	Alpha	Checkpoint	Links an alpha to checkpoints used for its validation.
partOfKernel	Alpha	Kernel	Indicates that an alpha is part of the kernel.
trackedBy	MethodEnactment	AlphaState	Specifies that a method enactment tracks the states of alphas.
associatedWithPattern	Method	Pattern	Indicates that a method is associated with certain patterns.
relatedWorkProduct	Alpha	WorkProduct	Links an alpha to its related work products.

Table 31: Essence Ontology Object Property

Data Property	Domain	Range	Description
name	Element	String	The name of the element.
description	Element	String	A detailed description of the element.
startDate	Work	Date	The start date of a work task.
endDate	Work	Date	The end date of a work task.
priority	Task	String	The priority level of a task.
status	Task	String	The current status of a task.
duration	Task	Hour	The duration of a task.
level	Competency	String	The proficiency level of a competency.
effort	Activity	String	The estimated effort required to complete an activity.
cost	Activity	String	The estimated cost associated with an activity.
dueDate	WorkProduct	Date	The due date for the completion of a work product.
createdOn	WorkProduct	Date	The date and time when a work product was created.
modifiedOn	WorkProduct	Date	The date and time when a work product was last modified.
version	WorkProduct	String	The version number of a work product.

estimatedEffort	Work	Hour	The estimated effort to complete a work task.
actualEffort	Work	Hour	The actual effort spent on a work task.
reviewDate	Checkpoint	Date	The date when a checkpoint is reviewed.
isMandatory	Checkpoint	String	Indicates whether a checkpoint is mandatory.
identifier	Element	String	A unique identifier for the element.
completionPercentage	Work	String	The percentage of completion for a work task.
reviewerComments	Checkpoint	String	Comments or notes provided by a reviewer during a checkpoint review.
owner	Stakeholder	String	The owner of a particular element or task.
impact	Risk	String	The potential impact of a risk.
likelihood	Risk	String	The likelihood of a risk occurring.
riskCategory	Risk	String	The category to which a risk belongs.
resourceType	Resource	String	The type of a resource.
capacity	Resource	String	The capacity or availability of a resource.
constraintType	Constraints	String	The type of a constraint (e.g., regulatory, policy, technical).
constraintValue	Constraints	String	The specific value or limit of a constraint.
tagValue	Tag	String	The value associated with a tag.
documentedOn	Element	String	The date and time when an element was documented.
approvedBy	Element	String	The person or group who approved the element.
approvalDate	Element	String	The date when the element was approved.
lastUpdated	Element	String	The last update date and time for an element.
author	Element	String	The author of a particular element.
changeLog	Element	String	A log of changes made to the element.
reference	Element	String	External references or links related to the element.
riskMitigation	Risk	String	The mitigation strategies for a risk.
testResult	Testing	String	The result of a test.
testDate	Testing	String	The date when a test was performed.
testCaseID	Testing	String	The unique identifier for a test case.
testDescription	Testing	String	A detailed description of a test case.

defectID	Testing	String	The unique identifier for a defect found during testing.
defectDescription	Testing	String	A detailed description of a defect.
defectStatus	Testing	String	The current status of a defect.
reviewer	Checkpoint	String	The person responsible for reviewing a checkpoint.
documentVersion	WorkProduct	String	The version of a document work product.
expectedOutcome	Activity	String	The expected outcome of an activity.
measuredBy	Metric	String	The unit or method used to measure a metric.
targetValue	Metric	String	The target value for a metric.
currentValue	Metric	String	The current measured value of a metric.
metricDescription	Metric	String	A description of the metric being measured.

Table 32: Essence Ontology Data Property

Class	Individual	Description
Stakeholder	JohnDoe	A project manager responsible for overseeing the project.
Stakeholder	JaneSmith	A product owner representing customer interests.
Stakeholder	AliceBrown	A senior developer involved in the project.
Stakeholder	BobJohnson	A QA engineer ensuring the quality of the product.
Stakeholder	CharlieDavis	A client representative providing requirements and feedback.
Opportunity	NewMarketOpportunity	Opportunity to enter a new market segment.
Opportunity	CompetitiveAdvantageOpportunity	Opportunity to gain a competitive advantage.
Opportunity	RegulatoryComplianceOpportunity	Opportunity to comply with new regulations.
Opportunity	CostReductionOpportunity	Opportunity to reduce operational costs.
Opportunity	TechnologyUpgradeOpportunity	Opportunity to upgrade to new technology.
Requirements	HighPerformanceRequirement	Requirement for high performance and speed.

Requirements	SecurityRequirement	Requirement for robust security features.
Requirements	UsabilityRequirement	Requirement for user-friendly interfaces.
Requirements	ScalabilityRequirement	Requirement for scalable architecture.
Requirements	ComplianceRequirement	Requirement to meet regulatory compliance standards.
Software System	CustomerManagementSystem	A system for managing customer information and interactions.
SoftwareSystem	ECommercePlatform	An online platform for conducting sales and transactions.
SoftwareSystem	MobileBankingApp	A mobile application for banking services.
SoftwareSystem	InventoryManagementSystem	A system for tracking and managing inventory.
SoftwareSystem	HRManagementSystem	A system for managing human resources functions.
Team	DevelopmentTeamA	The primary development team working on the project.
Team	QA Team	The quality assurance team responsible for testing.
Team	UX Team	The user experience team designing interfaces.
Team	DevOps Team	The team managing deployment and operations.
Team	Support Team	The team providing customer support and troubleshooting.
Work	ImplementationTask	Task related to implementing a feature.
Work	TestingTask	Task related to testing a component.
Work	DocumentationTask	Task related to creating project documentation.
Work	DeploymentTask	Task related to deploying the system.
Work	MaintenanceTask	Task related to maintaining and updating the system.
WayOfWorking	AgileMethodology	An agile methodology used by the team.

WayOfWorking	ScrumMethodology	A specific implementation of agile, focusing on Scrum practices.
WayOfWorking	WaterfallMethodology	A traditional, sequential methodology.
WayOfWorking	KanbanMethodology	A methodology focused on visualizing work and continuous delivery.
WayOfWorking	LeanMethodology	A methodology focused on optimizing efficiency and reducing waste.
Alpha	Stakeholders	The group of all individuals or organizations affected by the project.
Alpha	Opportunity	The potential benefits and value the project aims to deliver.
Alpha	Requirements	The specific needs and constraints the project must meet.
Alpha	SoftwareSystem	The system being developed or modified.
Alpha	Team	The group of people actively working on the project.
Alpha	Work	The tasks and activities performed to deliver the project.
Alpha	WayOfWorking	The set of practices and processes guiding the team's work.
Activity	GatherRequirements	Activity focused on collecting stakeholder requirements.
Activity	DesignArchitecture	Activity focused on designing the system architecture.
Activity	DevelopCode	Activity focused on coding the software system.
Activity	TestSystem	Activity focused on testing the software system.
Activity	DeploySystem	Activity focused on deploying the software system.

ActivitySpace	ExplorePossibilities	Activity space for identifying potential solutions.
ActivitySpace	UnderstandStakeholderNeeds	Activity space for gathering and analyzing stakeholder requirements.
ActivitySpace	EnsureStakeholderSatisfaction	Activity space for validating and verifying stakeholder satisfaction.
Competency	StakeholderRepresentation	Competency in understanding and advocating for stakeholder needs.
Competency	Development	Competency in designing, coding, and testing software components.
Competency	Testing	Competency in planning, executing, and evaluating tests to ensure software quality.
Competency	Leadership	Competency in leading and managing teams.
Competency	ProjectManagement	Competency in planning, executing, and overseeing projects.
AlphaState	ConceptEstablished	State indicating that the concept has been established.
AlphaState	RequirementsDefined	State indicating that requirements have been defined.
AlphaState	DesignCompleted	State indicating that the design has been completed.
AlphaState	ImplementationInProgress	State indicating that implementation is in progress.
AlphaState	TestingCompleted	State indicating that testing has been completed.
AlphaState	Deployed	State indicating that the system has been deployed.
AlphaAssociation	RelatedTo	Relationship between two alphas.
CompetencyLevel	Beginner	Basic level of proficiency in a competency.

CompetencyLevel	Intermediate	Intermediate level of proficiency in a competency.
CompetencyLevel	Advanced	Advanced level of proficiency in a competency.
CompetencyLevel	Expert	Expert level of proficiency in a competency.
Checkpoint	InitialReview	Initial review checkpoint.
Checkpoint	MidProjectReview	Mid-project review checkpoint.
Checkpoint	FinalReview	Final review checkpoint.
WorkProduct	RequirementsDocument	Document detailing the project requirements.
WorkProduct	DesignSpecification	Document detailing the system design.
WorkProduct	SourceCode	The actual codebase of the software system.
WorkProduct	TestPlan	Document detailing the test plan and cases.
WorkProduct	UserManual	Document providing instructions for end users.
Pattern	MVCPattern	Model-View-Controller design pattern.
Pattern	SingletonPattern	Design pattern that restricts the instantiation of a class to one object.
Pattern	ObserverPattern	Design pattern where an object maintains a list of dependents and notifies them of state changes.
Pattern	FactoryPattern	Design pattern for creating objects without specifying the exact class.
Pattern	StrategyPattern	Design pattern that enables selecting an algorithm's behavior at runtime.
Practice	CodeReviewPractice	A practice focused on reviewing and improving code quality.
Practice	TestDrivenDevelopmentPractice	A practice focused on writing tests before implementing code.
Practice	ContinuousIntegrationPractice	A practice focused on integrating code changes frequently.

Practice	PairProgrammingPractice	A practice where two programmers work together at one workstation.
Practice	RetrospectivePractice	A practice focused on reflecting and improving the team's processes.
Method	AgileMethod	An agile method combining various agile practices.
Method	ScrumMethod	A method implementing Scrum practices.
Method	WaterfallMethod	A method following the traditional waterfall approach.
Method	KanbanMethod	A method following Kanban practices.
Method	LeanMethod	A method following lean practices.
State	InitialState	The initial state of an element.
State	InProgressState	The state indicating an element is currently being worked on.
State	CompletedState	The state indicating an element is completed.
State	ReviewedState	The state indicating an element has been reviewed.
State	ApprovedState	The state indicating an element has been approved.
ActivityAssociation	Precedes	Indicates that one activity precedes another.
ActivityAssociation	Follows	Indicates that one activity follows another.
ActivityAssociation	DependsOn	Indicates that one activity depends on another.
ActivityKind	MandatoryActivity	An activity that must be performed.
ActivityKind	OptionalActivity	An activity that is optional.
ActivityKind	ConditionalActivity	An activity that is performed under certain conditions.
Resource	DevelopmentEnvironment	The environment used for development.

Resource	TestingEnvironment	The environment used for testing.
Resource	DocumentationTools	Tools used for creating documentation.
Resource	CollaborationTools	Tools used for team collaboration.
Resource	DeploymentTools	Tools used for deploying the software system.
Tag	HighPriorityTag	A tag indicating high priority.
Tag	SecurityTag	A tag indicating security-related elements.

Table 33: Essence Ontology Individual

Object Property	Relationship	Description
hasStakeholder	SoftwareSystem hasStakeholder Stakeholder	Links a software system to its stakeholders.
hasRequirement	Opportunity hasRequirement Requirements	Links an opportunity to the requirements it generates.
performedBy	Work performedBy Team	Indicates the team that performs a specific work task.
partOfActivitySpace	Activity partOfActivitySpace ActivitySpace	Indicates that an activity is part of a specific activity space.
achievedState	Alpha achievedState AlphaState	Indicates the current state of an alpha.
usesResource	Activity usesResource Resource	Specifies the resources used by an activity.
hasTag	Element hasTag Tag	Associates metadata (tags) with an element to provide additional information.
hasCompetency	Team hasCompetency Competency	Indicates the competencies possessed by a team.
hasCompetencyLevel	Competency hasCompetencyLevel CompetencyLevel	Specifies the level of proficiency for a particular competency.
associatedWith	Alpha associatedWith Alpha	Defines a relationship between two alphas.
relatedTo	Element relatedTo Element	Generic relationship between any two elements in the ontology.
definedBy	Method definedBy Practice	Indicates the practices that define a method.

includesActivity	ActivitySpace includesActivity Activity	Specifies that an activity space includes specific activities.
hasCheckpoint	Activity hasCheckpoint Checkpoint	Links an activity to a checkpoint that needs to be verified.
producesWorkProduct	Activity producesWorkProduct WorkProduct	Indicates that an activity produces a specific work product.
hasPattern	Practice hasPattern Pattern	Associates a practice with a pattern.
hasPracticeAsset	Practice hasPracticeAsset PracticeAsset	Links a practice to its reusable components.
resolvedBy	Conflict resolvedBy MergeResolution	Specifies how a conflict is resolved during merging.
belongsToLibrary	Practice belongsToLibrary Library	Indicates that a practice is part of a collection of reusable methods and practices.
hasApproach	Activity hasApproach Approach	Links an activity to a strategy or technique for performing it.
hasCompletionCriterion	Activity hasCompletionCriterion CompletionCriterion	Specifies the criteria for determining if an activity is complete.
hasEntryCriterion	Activity hasEntryCriterion EntryCriterion	Specifies the criteria for determining if an activity can be started.
extendsElement	ExtensionElement extendsElement Element	Indicates that an extension element extends the functionality of another element.
instantiatesMethod	MethodEnactment instantiatesMethod Method	Links a method enactment to the method it applies.
usesPattern	TypedPattern usesPattern Pattern	Indicates that a typed pattern is based on a specific pattern.
usesResource	TypedResource usesResource Resource	Specifies that a typed resource is used in a certain context.
usesTag	TypedTag usesTag Tag	Associates a typed tag with specific metadata constraints.
extendsKernel	UserDefinedType extendsKernel Kernel	Indicates that a user-defined type extends the kernel.

selectedFeature	ViewSelection selectedFeature FeatureSelection	Specifies the features included in a particular view.
definesView	View definesView ElementGroup	Indicates that a view is defined by a group of related elements.
graphicallyRepresents	GraphicalSyntax graphicallyRepresents Element	Specifies the visual representation rules for a language element.
textuallyRepresents	TextualSyntax textuallyRepresents Element	Specifies the textual representation rules for a language element.
hasDynamicBehavior	DynamicSemantics hasDynamicBehavior LanguageElement	Indicates the operational behavior and interactions of a language element.
containedIn	ElementGroup containedIn Element	Specifies that an element is part of an element group.
hasStateTransition	State hasStateTransition Transition	Links a state to its possible transitions.
involvesStakeholder	Activity involvesStakeholder Stakeholder	Specifies that an activity involves certain stakeholders.
requiresCompetency	Activity requiresCompetency Competency	Indicates the competencies required to perform an activity.
contributesToAlpha	WorkProduct contributesToAlpha Alpha	Specifies that a work product contributes to the progress or health of an alpha.
validatedBy	Alpha validatedBy Checkpoint	Links an alpha to checkpoints used for its validation.
partOfKernel	Alpha partOfKernel Kernel	Indicates that an alpha is part of the kernel.
trackedBy	MethodEnactment trackedBy AlphaState	Specifies that a method enactment tracks the states of alphas.
associatedWithPattern	Method associatedWithPattern Pattern	Indicates that a method is associated with certain patterns.
relatedWorkProduct	Alpha relatedWorkProduct WorkProduct	Links an alpha to its related work products.

Table 34: Essence Ontology Relationship

PM^2 Framework

Class	Subclass	Description of Class
Project	AgileProject	A project is a temporary endeavor undertaken to create a unique product, service, or result.
Project	PM²Project	A project is a temporary endeavor undertaken to create a unique product, service, or result.
Stakeholder	ProductOwner	A stakeholder is an individual, group, or organization that can affect or be affected by the project.
Stakeholder	ScrumMaster	A stakeholder is an individual, group, or organization that can affect or be affected by the project.
Stakeholder	DevelopmentTeam	A stakeholder is an individual, group, or organization that can affect or be affected by the project.
Task	PlanningTask	A task is a piece of work to be done or undertaken.
Task	DevelopmentTask	A task is a piece of work to be done or undertaken.
Task	TestingTask	A task is a piece of work to be done or undertaken.
Artefact	ProductBacklog	Artefacts are tangible outputs or documents produced during the project.
Artefact	SprintBacklog	Artefacts are tangible outputs or documents produced during the project.
Artefact	Increment	Artefacts are tangible outputs or documents produced during the project.
Role	TeamCoordinator	A role represents a set of responsibilities and duties assigned to an individual or team.
Role	ArchitectureOwner	A role represents a set of responsibilities and duties assigned to an individual or team.
Role	AgileTeamMember	A role represents a set of responsibilities and duties assigned to an individual or team.
Role	BusinessManager	A role represents a set of responsibilities and duties assigned to an individual or team.
Role	ProjectManager	A role represents a set of responsibilities and duties assigned to an individual or team.

Iteration		An iteration is a time-boxed period during which specific work is completed and made ready for review.
Release		A release is a version of the product that is made available to the end-users.
Requirement		Requirements are conditions or capabilities needed by a user to solve a problem or achieve an objective.
Risk		A risk is an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives.
Quality		Quality is the degree to which a set of inherent characteristics fulfill requirements.
Architecture		Architecture refers to the fundamental structures of a system and the discipline of creating such structures.
Compliance		Compliance ensures that the project adheres to relevant laws, regulations, and guidelines.
Deployment		Deployment involves the activities required to make a system operational in its target environment.

Table 35: PM² Ontology Classes and Sub-Classes

Object Property	Domain (Class)	Range (Class/Subclass)	Description
hasStakeholder	Project	Stakeholder	Defines the relationship between a project and its stakeholders.
hasStakeholder	AgileProject	ProductOwner, ScrumMaster, DevelopmentTeam	Defines the relationship between an Agile project and its stakeholders.
hasStakeholder	PM²Project	Stakeholder	Defines the relationship between a PM² project and its stakeholders.
consistsOf	Project	Task	Defines the relationship between a project and its tasks or components.
consistsOf	AgileProject	PlanningTask, DevelopmentTask, TestingTask	Defines the relationship between an Agile project and its tasks.
precedes	Task	Task	Indicates that one task precedes another.
follows	Task	Task	Indicates that one task follows another.

hasPriority	Task	Integer (priority level)	Defines the priority level of a task.
hasStatus	Task	String (status)	Indicates the status of a task.
isAssignedTo	Task	Role	Specifies the assignment of a task to a team member.
hasDueDate	Task	Date	Indicates the due date for a task.
isRelatedTo	Task	Task	Indicates a relationship between tasks.
requires	Task	Artefact	Indicates a requirement for a task.
hasResponsible	Task	Role	Specifies who is responsible for a task.
hasAccountable	Task	Role	Specifies who is accountable for a task.
hasConsulted	Task	Stakeholder	Indicates who has been consulted for a task.
hasInformed	Task	Stakeholder	Indicates who has been informed about a task.
hasIdentifier	Task	String (identifier)	Provides a unique identifier for a task.
hasDescription	Task	String (description)	Provides a description of a task.
hasStartDate	Task	Date	Indicates the start date of a task.
hasEndDate	Task	Date	Indicates the end date of a task.
hasBudget	Project	Float (budget)	Indicates the budget allocated for a project.
hasResource	Task	Resource	Specifies the resources required for a task.

Table 36: PM² Ontology Object Property

Data Property	Class or Subclass	Description
hasIdentifier	Task, Project	Provides a unique identifier for a task or project.
hasName	Task, Project, Stakeholder	Indicates the name of a task, project, or stakeholder.
hasDescription	Task, Project, Artefact	Provides a detailed description of a task, project, or artefact.

hasPriority	Task, Project	Defines the priority level of a task or project (e.g., high, medium, low).
hasStatus	Task, Project	Indicates the current status of a task or project (e.g., not started, in progress, completed).
hasStartDate	Task, Project	Indicates the start date of a task or project.
hasEndDate	Task, Project	Indicates the end date of a task or project.
hasDueDate	Task, Project	Indicates the due date for a task or project.
hasBudget	Task, Project	Indicates the budget allocated for a task or project.
hasEstimatedTime	Task, Project	Provides an estimate of the time required to complete a task or project.
hasActualTime	Task, Project	Records the actual time taken to complete a task or project.
hasCost	Task, Project	Indicates the cost associated with a task or project.
hasResource	Task, Project	Specifies the resources required or used for a task or project (e.g., personnel, equipment).
hasRiskLevel	Task, Project	Defines the risk level of a task or project (e.g., high, medium, low).
hasImpact	Risk	Indicates the potential impact of a risk or issue on the project.
hasMitigationPlan	Risk	Describes the mitigation plan for identified risks.
hasOwner	Task, Project	Specifies the owner or person responsible for a task or project.
hasStakeholder	Task, Project	Lists stakeholders associated with a task or project.
hasReviewDate	Task, Project	Indicates the date of review or evaluation for a task or project.
hasApprovalStatus	Task, Project	Indicates the approval status of a task or project (e.g., approved, pending, rejected).

Table 37: PM² Ontology Data Property

Class/Subclass	Individual	Identifier	Name	Description	Start Date	End Date	Status	Priority	Budget
Project	ProjectX	P001	ProjectX	Development of a new software application	2024-01-01	2024-12-31	In Progress	High	$500,000
Project	ProjectY	P002	ProjectY	Infrastructure upgrade for data center	2024-03-01	2024-11-30	Planned	Medium	$300,000
AgileProject	AgileProjectA	A001	AgileProjectA	Implementation of an agile CRM system	2024-02-01	2024-10-31	In Progress	High	$200,000
PM²Project	PM²ProjectB	PM001	PM²ProjectB	Development of a PM²-based project platform	2024-04-01	2024-09-30	Not Started	High	$150,000
Stakeholder	JohnDoe	S001	John Doe	Product Owner					
Stakeholder	JaneSmith	S002	Jane Smith	Scrum Master					
DevelopmentTeam	DevTeamAlpha	DT001	Dev Team Alpha	Development Team for ProjectX					
PlanningTask	PlanTask1	T001	Plan Task 1	Initial Planning for ProjectX	2024-01-05	2024-01-10	Completed	High	
DevelopmentTask	DevTask1	T002	Dev Task 1	Develop Feature A for ProjectX	2024-02-01	2024-02-28	In Progress	High	
TestingTask	TestTask1	T003	Test Task 1	Testing Feature A for ProjectX	2024-03-01	2024-03-10	Not Started	Medium	

Artefact	ProductBacklog1	AFB001	Product Backlog 1	Backlog for AgileProjectA					
Artefact	SprintBacklog1	AFB002	Sprint Backlog 1	Sprint backlog for AgileProjectA					
Artefact	Increment1	AFI001	Increment 1	Increment from Sprint 1 of AgileProjectA					
Role	TeamCoordinator1	R001	Team Coordinator 1	Coordinator for PM²ProjectB					
Role	ArchitectureOwner1	R002	Architecture Owner 1	Owner for AgileProjectA					
Role	AgileTeamMember1	R003	Agile Team Member 1	Member of Dev Team Alpha					
Role	BusinessManager1	R004	Business Manager 1	Manager for ProjectY					
Role	ProjectManager1	R005	Project Manager 1	Manager for ProjectX					

Table 38 : Example of individuals.

ProjOnto Framework

Class Name	Subclass Name	Description
ProjectManagement		The overarching class representing all aspects of project management.
BTM_BodyOfKnowledge	KeyComponents	Key components of Business Technology Management Body of Knowledge.
BTM_BodyOfKnowledge	BestPractices	Best practices within Business Technology Management Body of Knowledge.
EssenceFramework	Alpha	Core elements of the Essence framework representing different perspectives in software development.
EssenceFramework	ActivitySpace	Represents areas of activities within the Essence framework.
EssenceFramework	State	Represents the states of alphas within the Essence framework.
EssenceFramework	CheckList	Represents checklists associated with states of alphas to guide project progress.

PM2_Framework	Phase	Represents different phases of the PM² Project Management Methodology.
PM2_Framework	Process	Processes within the PM² Project Management Methodology.
PM2_Framework	Component	Components of the PM² Project Management Methodology.
BPMN_BasedOntology	BPMN_Element	Represents elements within the BPMN 2.0 based ontology for business process representation.
Ontology	Class	Represents classes in an ontology.
Ontology	ObjectProperty	Represents object properties in an ontology.
Ontology	DataProperty	Represents data properties in an ontology.
Ontology	Individual	Represents individuals in an ontology.
KnowledgeGraph	Node	Represents nodes within a knowledge graph.
KnowledgeGraph	Edge	Represents edges within a knowledge graph.
Alpha	Opportunity	Opportunity alpha representing project justification and potential value.

Alpha	Stakeholders	Stakeholders alpha representing those involved in and benefiting from the project.
Alpha	Requirements	Requirements alpha representing project requirements.
Alpha	SoftwareSystem	Software system alpha representing the system to be developed.
Alpha	Work	Work alpha representing the work to be done.
Alpha	Team	Team alpha representing the project team.
Alpha	WayOfWorking	Way of working alpha representing the methods and practices used by the project team.
ActivitySpace	CustomerArea	Activities related to understanding and satisfying customer needs.
ActivitySpace	SolutionArea	Activities related to developing the solution.
ActivitySpace	EndeavorArea	Activities related to managing the project endeavor.
Stakeholder	Sponsor	Represents the sponsor of the project.

Stakeholder	User	Represents the end-users of the project deliverables.
Stakeholder	TeamMember	Represents members of the project team.
Requirement	FunctionalRequirement	Represents functional requirements of the project.
Requirement	NonFunctionalRequirement	Represents non-functional requirements of the project.
SoftwareSystem	Module	Represents modules within the software system.
SoftwareSystem	Component	Represents components within the software system.
Work	Task	Represents individual tasks within the project work.
Work	Activity	Represents activities within the project work.
Team	Role	Represents roles within the project team.
Team	Skill	Represents skills of team members.
WayOfWorking	Method	Represents specific methods used by the project team.

WayOfWorking	Practice	Represents practices used by the project team.
ProjectPhase	Initiating	Represents the initiating phase of the project.
ProjectPhase	Planning	Represents the planning phase of the project.
ProjectPhase	Executing	Represents the executing phase of the project.
ProjectPhase	MonitoringAndControlling	Represents the monitoring and controlling phase of the project.
ProjectPhase	Closing	Represents the closing phase of the project.
RiskManagement	RiskIdentification	Processes for identifying project risks.
RiskManagement	RiskAnalysis	Processes for analyzing project risks.
RiskManagement	RiskResponse	Processes for responding to project risks.
QualityManagement	QualityPlanning	Processes for planning project quality.
QualityManagement	QualityAssurance	Processes for ensuring project quality.
QualityManagement	QualityControl	Processes for controlling project quality.
CostManagement	CostEstimation	Processes for estimating project costs.
CostManagement	Budgeting	Processes for budgeting project costs.

CostManagement	CostControl	Processes for controlling project costs.
ScheduleManagement	SchedulePlanning	Processes for planning the project schedule.
ScheduleManagement	ScheduleControl	Processes for controlling the project schedule.
ScopeManagement	ScopeDefinition	Processes for defining project scope.
ScopeManagement	ScopeControl	Processes for controlling project scope.
IntegrationManagement	ProjectCharter	Processes for developing the project charter.
IntegrationManagement	ProjectManagementPlan	Processes for developing the project management plan.
CommunicationManagement	CommunicationPlanning	Processes for planning project communications.
CommunicationManagement	InformationDistribution	Processes for distributing project information.
CommunicationManagement	PerformanceReporting	Processes for reporting project performance.
ProcurementManagement	ProcurementPlanning	Processes for planning project procurements.
ProcurementManagement	SupplierSelection	Processes for selecting suppliers for the project.
ProcurementManagement	ContractManagement	Processes for managing contracts in the project.
StakeholderManagement	StakeholderIdentification	Processes for identifying project stakeholders.

	StakeholderEngagement	Processes for engaging project stakeholders.
AgileMethodology	Scrum	A specific agile methodology emphasizing iterative development and collaboration.
AgileMethodology	Kanban	A specific agile methodology focusing on visualizing work and limiting work in progress.
AgileMethodology	ExtremeProgramming	A specific agile methodology emphasizing technical excellence and continuous improvement.
HybridApproach		Represents the integration of traditional and agile project management methodologies.
Customization		Represents the customization of project processes to meet specific project needs and contexts.
SWRL_Rule		Represents rule-based queries using Semantic Web Rule Language (SWRL) for project customization and decision-making.

SQRWL_Query		Represents rule-based queries using Semantic Query-Enhanced Web Rule Language (SQRWL) for project customization and decision-making.
EssenceKernel		Represents the core elements of the Essence framework, including alphas, activities, and states.
State		Represents the states of alphas within the Essence framework.
CheckList		Represents checklists associated with states of alphas to guide project progress.
EmpiricalResearchFramework		Represents the framework for conducting empirical research in software engineering using the Essence framework.
ReflectionMeeting		Represents meetings for reflecting on project progress and outcomes, as proposed by the Essence framework.

TeamComposition		Represents the composition of project teams and its impact on project success.
AutomatedTeamComposition		Represents approaches and technologies for automating project team composition.
Competence		Represents the skills and capabilities of project team members.
Collaboration		Represents the collaboration dynamics within project teams.
Trust		Represents the trust dynamics within project teams.
Diversity		Represents the diversity aspects within project teams.
KnowledgeIntegration		Represents the integration of knowledge within project teams and its impact on project success.
Personality		Represents the personality traits of project team members and their impact on project dynamics.
Motivation		Represents the motivational factors

		affecting project team members.
OntologyDevelopment		Represents the processes and methodologies for developing ontologies using tools like Protégé.
BTM_Ontology		Represents the ontology for Business Technology Management.
EssenceOntology		Represents the ontology for the Essence framework.
PM2_Ontology		Represents the ontology for the PM² framework.
BPMN_Ontology		Represents the ontology for BPMN 2.0 based business process representation.
ProjOnto		Represents the integrated project ontology incorporating elements from PM, BTM, and Essence frameworks.
Iteration		Represents iterations in agile methodologies and their management.
RuleBasedCustomization		Represents the use SQWRL for customizing project processes.

ProjectState		Represents the current state of the project as assessed by the Essence framework's alphas and states.

Table 39: ProjOnto Classes and SubClasses

ProjOnto Object Properties

Object Property	Domain Class	Range Class	Range Subclass	Description
hasPhase	ProjectManagement	ProjectPhase		Links ProjectManagement to its phases.
hasProcess	ProjectPhase	PM2_Framework	Process	Links ProjectPhase to its processes.
hasComponent	PM2_Framework	PM2_Framework	Component	Links PM2_Framework to its components.
hasActivitySpace	EssenceFramework	ActivitySpace		Links EssenceFramework to its activity spaces.
hasAlpha	EssenceFramework	Alpha		Links EssenceFramework to its alphas.
hasState	Alpha	State		Links Alpha to its states.
hasCheckList	State	CheckList		Links State to its checklists.
involvesStakeholder	ProjectPhase	Stakeholder		Links ProjectPhase to stakeholders involved.

meetsRequirement	ProjectPhase	Requirement		Links ProjectPhase to requirements to be met.
developsSoftwareSystem	ProjectPhase	SoftwareSystem		Links ProjectPhase to the software system being developed.
performsWork	ProjectPhase	Work		Links ProjectPhase to the work being performed.
includesTeam	ProjectPhase	Team		Links ProjectPhase to the project team.
utilizesWayOfWorking	ProjectPhase	WayOfWorking		Links ProjectPhase to the way of working being utilized.
hasTask	Work	Work	Task	Links Work to its tasks.
hasActivity	Work	Work	Activity	Links Work to its activities.
hasRole	Team	Team	Role	Links Team to its roles.
hasSkill	Team	Team	Skill	Links Team to its skills.

followsMethod	WayOfWorking	WayOfWorking	Method	Links WayOfWorking to its methods.
adoptsPractice	WayOfWorking	WayOfWorking	Practice	Links WayOfWorking to its practices.
identifiesRisk	RiskManagement	RiskManagement	RiskIdentification	Links RiskManagement to risk identification processes.
analyzesRisk	RiskManagement	RiskManagement	RiskAnalysis	Links RiskManagement to risk analysis processes.
respondsToRisk	RiskManagement	RiskManagement	RiskResponse	Links RiskManagement to risk response processes.
plansQuality	QualityManagement	QualityManagement	QualityPlanning	Links QualityManagement to quality planning processes.
assuresQuality	QualityManagement	QualityManagement	QualityAssurance	Links QualityManagement to quality assurance processes.

controlsQuality	QualityManagement	QualityManagement	QualityControl	Links QualityManagement to quality control processes.
estimatesCost	CostManagement	CostManagement	CostEstimation	Links CostManagement to cost estimation processes.
createsBudget	CostManagement	CostManagement	Budgeting	Links CostManagement to budgeting processes.
controlsCost	CostManagement	CostManagement	CostControl	Links CostManagement to cost control processes.
plansSchedule	ScheduleManagement	ScheduleManagement	SchedulePlanning	Links ScheduleManagement to schedule planning processes.
controlsSchedule	ScheduleManagement	ScheduleManagement	ScheduleControl	Links ScheduleManagement to schedule control processes.
definesScope	ScopeManagement	ScopeManagement	ScopeDefinition	Links ScopeManagem

				ent to scope definition processes.
controlsScope	ScopeManagement	ScopeManagement	ScopeControl	Links ScopeManagement to scope control processes.
developsProjectCharter	IntegrationManagement	IntegrationManagement	ProjectCharter	Links IntegrationManagement to project charter development processes.
developsManagementPlan	IntegrationManagement	IntegrationManagement	ProjectManagementPlan	Links IntegrationManagement to project management plan development processes.
plansCommunication	CommunicationManagement	CommunicationManagement	CommunicationPlanning	Links CommunicationManagement to communication planning processes.
distributesInformation	CommunicationManagement	CommunicationManagement	InformationDistribution	Links CommunicationManagement to

				information distribution processes.
reportsPerformance	Communication Management	Communication Management	Performance Reporting	Links Communication Management to performance reporting processes.
plansProcurement	ProcurementManagement	ProcurementManagement	Procurement Planning	Links ProcurementManagement to procurement planning processes.
selectsSupplier	ProcurementManagement	ProcurementManagement	SupplierSelection	Links ProcurementManagement to supplier selection processes.
managesContract	ProcurementManagement	ProcurementManagement	ContractManagement	Links ProcurementManagement to contract management processes.
identifiesStakeholder	StakeholderManagement	StakeholderManagement	StakeholderIdentification	Links StakeholderManagement to stakeholder

				identification processes.
engagesStakeholder	StakeholderManagement	StakeholderManagement	StakeholderEngagement	Links StakeholderManagement to stakeholder engagement processes.
appliesAgileMethod	AgileMethodology	AgileMethodology	Scrum	Links AgileMethodology to Scrum methodology.
appliesAgileMethod	AgileMethodology	AgileMethodology	Kanban	Links AgileMethodology to Kanban methodology.
appliesAgileMethod	AgileMethodology	AgileMethodology	ExtremeProgramming	Links AgileMethodology to Extreme Programming (XP) methodology.
integratesTraditional	HybridApproach	ProjectManagement		Links HybridApproach to traditional project management methodologies.
integratesAgile	HybridApproach	AgileMethodology		Links HybridApproac

				h to agile methodologies.
customizesProcess	Customization	ProjectManagement		Links Customization to project management processes.
customizesProcess	Customization	AgileMethodology		Links Customization to agile methodologies.
usesSWRL_Rule	RuleBasedCustomization	SWRL_Rule		Links RuleBasedCustomization to SWRL rules.
usesSQRWL_Query	RuleBasedCustomization	SQRWL_Query		Links RuleBasedCustomization to SQRWL queries.
includesIteration	AgileMethodology	Iteration		Links AgileMethodology to iterations.
usesOntology	OntologyDevelopment	Ontology		Links OntologyDevelopment to ontologies.
hasSubClass	Ontology	Ontology	Class	Links Ontology to its subclasses.

hasObjectProperty	Ontology	Ontology	ObjectProperty	Links Ontology to its object properties.
hasDataProperty	Ontology	Ontology	DataProperty	Links Ontology to its data properties.
hasIndividual	Ontology	Ontology	Individual	Links Ontology to its individuals.
usesKnowledgeGraph	OntologyDevelopment	KnowledgeGraph		Links OntologyDevelopment to knowledge graphs.
usesNode	KnowledgeGraph	KnowledgeGraph	Node	Links KnowledgeGraph to its nodes.
usesEdge	KnowledgeGraph	KnowledgeGraph	Edge	Links KnowledgeGraph to its edges.
isPartOf	Alpha	EssenceFramework	Alpha	Links Opportunity alpha to the Essence framework.
isPartOf	Alpha	EssenceFramework	Alpha	Links Stakeholders alpha to the Essence framework.

isPartOf	Alpha	EssenceFramework	Alpha	Links Requirements alpha to the Essence framework.
isPartOf	Alpha	EssenceFramework	Alpha	Links SoftwareSystem alpha to the Essence framework.
isPartOf	Alpha	EssenceFramework	Alpha	Links Work alpha to the Essence framework.
isPartOf	Alpha	EssenceFramework	Alpha	Links Team alpha to the Essence framework.
isPartOf	Alpha	EssenceFramework	Alpha	Links WayOfWorking alpha to the Essence framework.
tracksProgressOf	State	Alpha		Links State to the alpha it tracks the progress of.
containsChecklist	State	CheckList		Links State to its checklists.
hasEmpiricalFramework	ProjectManagement	EmpiricalResearchFramework		Links ProjectManage

				ment to empirical research frameworks.
conductsReflection	EmpiricalResearchFramework	ReflectionMeeting		Links EmpiricalResearchFramework to reflection meetings.
impactsTeamSuccess	TeamComposition	ProjectManagement		Links TeamComposition to its impact on project success.
automatesTeamComposition	AutomatedTeamComposition	ProjectManagement		Links AutomatedTeamComposition to the automation of project team composition.
evaluatesCompetence	Team	Competence		Links Team to the evaluation of competences.
facilitatesCollaboration	Team	Collaboration		Links Team to the facilitation of collaboration.
buildsTrust	Team	Trust		Links Team to the building of trust.

embracesDiversity	Team	Diversity		Links Team to embracing diversity.
integratesKnowledge	Team	KnowledgeIntegration		Links Team to the integration of knowledge.
considersPersonality	Team	Personality		Links Team to considering personality traits.
considersMotivation	Team	Motivation		Links Team to considering motivational factors.
createsOntology	OntologyDevelopment	BTM_Ontology		Links OntologyDevelopment to the creation of BTM ontology.
createsOntology	OntologyDevelopment	EssenceOntology		Links OntologyDevelopment to the creation of Essence ontology.
createsOntology	OntologyDevelopment	PM2_Ontology		Links OntologyDevelopment to the creation of PM² ontology.

createsOntology	OntologyDevelopment	BPMN_Ontology		Links OntologyDevelopment to the creation of BPMN ontology.
integratesOntology	OntologyDevelopment	ProjOnto		Links OntologyDevelopment to the integration of project ontology (ProjOnto).
usesIteration	AgileMethodology	Iteration		Links AgileMethodology to iterations.
customizesProcess	Customization	ProjectManagement		Links Customization to project management processes.
customizesProcess	Customization	AgileMethodology		Links Customization to agile methodologies.
appliesRule	RuleBasedCustomization	SWRL_Rule		Links RuleBasedCustomization to SWRL rules.
appliesQuery	RuleBasedCustomization	SQRWL_Query		Links RuleBasedCust

				omization to SQRWL queries.
tracksCurrentState	ProjectManagement	ProjectState		Links ProjectManagement to the current state of the project.

Table 40 : ProjOnto Object Properties

8.1.1 Relationship Examples with Specific Object Properties

Object Property	Domain (Class)	Range (Class/Subclass)	Example Relationship
buildTrust	Team	TrustLevel	DevTeamAlpha buildsTrust with HighTrustLevel.
conductsReflection	Team	ReflectionMeeting	DevTeamAlpha conductsReflection during SprintRetrospectiveMeeting.
considersMotivation	ProjectManager	MotivationFactor	ProjectManager1 considersMotivation such as HighMotivationFactor for team performance.
considersPersonality	ProjectManager	PersonalityTrait	ProjectManager1 considersPersonality traits like Extraversion for team role assignment.
containsChecklist	Task	ChecklistItem	PlanTask1 containsChecklist items such as DefineScope, GatherRequirements.
createOntology	OntologyEngineer	Ontology	OntologyEngineer1 createOntology named ProjectManagementOntology.
developManagementPlan	ProjectManager	ManagementPlan	ProjectManager1 developManagementPlan for RiskManagementPlan.
embracesDiversity	Team	DiversityAspect	DevTeamAlpha embracesDiversity including CulturalDiversity.
evaluatesCompetence	ProjectManager	CompetenceDescription	ProjectManager1 evaluatesCompetence in TechnicalSkills for team members.

facilitateCollaboration	ScrumMaster	CollaborationType	ScrumMaster1 facilitateCollaboration through AgileCollaborationMethods.
hasEmpiricalFramework	Researcher	EmpiricalResearchFramework	Researcher1 hasEmpiricalFramework named AgilePracticeFramework.
impactsTeamSuccess	Team	SuccessMetric	DevTeamAlpha impactsTeamSuccess based on SprintCompletionRate.
integratesKnowledge	KnowledgeManager	KnowledgeArea	KnowledgeManager1 integratesKnowledge in ProjectManagementBestPractices.
integratesOntology	SystemArchitect	Ontology	SystemArchitect1 integratesOntology named EnterpriseArchitectureOntology.
integratesTraditional	Project	TraditionalMethodology	ProjectX integratesTraditional methodologies such as PMBOK.
isPartOf	Task	Project	PlanTask1 isPartOf ProjectX.
projectType	Project	ProjectCategory	ProjectX has projectType SoftwareDevelopment.
tracksCurrentState	ProjectManager	ProjectState	ProjectManager1 tracksCurrentState as InProgress for ProjectX.
trackProgressOf	ProjectManager	Task	ProjectManager1 trackProgressOf DevTask1.
usesEdge	KnowledgeGraph	Edge	KnowledgeGraph1 usesEdge named DependencyEdge.
usesIteration	AgileProject	Iteration	AgileProjectA usesIteration named Sprint1.
usesKnowledgeGraph	DataScientist	KnowledgeGraph	DataScientist1 usesKnowledgeGraph for ProjectKnowledgeGraph.
usesNode	KnowledgeGraph	Node	KnowledgeGraph1 usesNode named TaskNode.

Table 41: ProjOnto Object Properties

ProjOnto Data Properties

Data Property	Domain Class	Domain Subclass	Range	Description
projectName	ProjectManagement		String	The name of the project.
projectDescription	ProjectManagement		String	A description of the project.
startDate	ProjectPhase		String	The start date of the project phase.

endDate	ProjectPhase		String	The end date of the project phase.
budget	CostManagement		String	The budget allocated for the project.
costEstimate	CostManagement	CostEstimation	String	The estimated cost for the project.
actualCost	CostManagement	CostControl	String	The actual cost incurred in the project.
scheduleEstimate	ScheduleManagement	SchedulePlanning	String	The estimated schedule for the project.
actualSchedule	ScheduleManagement	ScheduleControl	String	The actual schedule followed in the project.
scopeDescription	ScopeManagement	ScopeDefinition	String	A description of the project scope.
qualityCriteria	QualityManagement	QualityPlanning	String	The criteria set for project quality.
riskDescription	RiskManagement	RiskIdentification	String	A description of identified risks in the project.
riskImpact	RiskManagement	RiskAnalysis	String	The potential impact of identified risks.
riskProbability	RiskManagement	RiskAnalysis	String	The probability of identified risks occurring.
stakeholderName	Stakeholder		String	The name of the stakeholder.
stakeholderRole	Stakeholder		String	The role of the stakeholder in the project.
requirementDescription	Requirement		String	A description of the project requirement.

requirementType	Requirement		String	The type of project requirement (e.g., functional, non-functional).
systemComponentName	SoftwareSystem	Component	String	The name of a software system component.
systemModuleName	SoftwareSystem	Module	String	The name of a software system module.
taskName	Work	Task	String	The name of a task in the project work.
taskDescription	Work	Task	String	A description of a task in the project work.
activityName	Work	Activity	String	The name of an activity in the project work.
activityDescription	Work	Activity	String	A description of an activity in the project work.
roleName	Team	Role	String	The name of a role in the project team.
skillName	Team	Skill	String	The name of a skill required in the project team.
methodDescription	WayOfWorking	Method	String	A description of a method used by the project team.
practiceDescription	WayOfWorking	Practice	String	A description of a practice used by the project team.
iterationNumber	Iteration		String	The iteration number in an agile methodology.
ontologyName	Ontology		String	The name of an ontology.
ontologyVersion	Ontology		String	The version of an ontology.
nodeName	KnowledgeGraph	Node	String	The name of a node in a knowledge graph.

edgeName	KnowledgeGraph	Edge	String	The name of an edge in a knowledge graph.
alphaName	Alpha		String	The name of an alpha in the Essence framework.
stateName	State		String	The name of a state in the Essence framework.
checkListItem	CheckList		String	An item in a checklist associated with a state.
empiricalFrameworkName	EmpiricalResearchFramework		String	The name of an empirical research framework.
reflectionMeetingDate	ReflectionMeeting		String	The date of a reflection meeting.
teamCompositionFactor	TeamComposition		String	A factor affecting team composition.
competenceDescription	Competence		String	A description of a competence required in the project team.
collaborationType	Collaboration		String	The type of collaboration in the project team.
trustLevel	Trust		String	The level of trust within the project team.
diversityAspect	Diversity		String	An aspect of diversity within the project team.
knowledgeArea	KnowledgeIntegration		String	An area of knowledge integrated within the project team.
personalityTrait	Personality		String	A personality trait of a project team member.

motivationFactor	Motivation		String	A factor affecting the motivation of project team members.
processCustomizationRule	RuleBasedCustomization		String	A rule used for customizing project processes.
currentStateDescription	ProjectState		String	A description of the current state of the project.

Table 42: ProjOnto Data Properties

ProjOnto Individuals

Individual Name	Description	Value 1	Value 2	Value 3	Class
ConstructionProject	A project type focused on construction activities.	Building skyscrapers	Road construction	Bridge development	ProjectManagement
SoftwareDevelopmentProject	A project type focused on software development activities.	Mobile app development	Web application creation	Software upgrade	ProjectManagement
ResearchProject	A project type focused on research activities.	Market analysis	Academic study	Scientific research	ProjectManagement
TrainingProject	A project type focused on training and development activities.	Employee onboarding	Skill enhancement	Certification programs	ProjectManagement
BudgetCompletion	A criterion for project success based on completing within budget.	On-budget delivery	Cost control	Financial management	ProjectSuccessCriteria
ScheduleCompletion	A criterion for project success	On-time delivery	Time management	Project scheduling	ProjectSuccessCriteria

	based on completing on schedule.				
QualityAchievement	A criterion for project success based on meeting quality standards.	Quality control	Quality assurance	Standards compliance	ProjectSuccessCriteria
StakeholderSatisfaction	A criterion for project success based on stakeholder satisfaction.	Client feedback	Stakeholder surveys	Satisfaction assessment	ProjectSuccessCriteria
HighPerformance	A performance level indicating high team efficiency and effectiveness.	High productivity	Exceptional outcomes	Team excellence	TeamPerformance
MediumPerformance	A performance level indicating medium team efficiency and effectiveness.	Average productivity	Standard outcomes	Team adequacy	TeamPerformance
LowPerformance	A performance level indicating low team efficiency and effectiveness.	Low productivity	Below standard outcomes	Team improvement required	TeamPerformance
AgileProject	A project type following agile methodologies.	Scrum implementation	Kanban process	Continuous improvement	ProjectType
WaterfallProject	A project type following waterfall methodologies.	Sequential phases	Detailed planning	Milestone tracking	ProjectType
HybridProject	A project type following a hybrid approach of agile and traditional methodologies.	Mixed methodology	Flexibility in approach	Combined frameworks	ProjectType
TechnicalSkill	A competency related to technical	Coding proficiency	System architecture	Network configuration	TeamCompetencies

	knowledge and abilities.				
LeadershipSkill	A competency related to leadership abilities and managing teams.	Team management	Strategic planning	Decision making	TeamCompetencies
CommunicationSkill	A competency related to effective communication within the team and with stakeholders.	Effective writing	Clear presentations	Active listening	TeamCompetencies
ProblemSolvingSkill	A competency related to solving project-related problems effectively.	Critical thinking	Analytical skills	Innovative solutions	TeamCompetencies
CollaborationSkill	A competency related to working collaboratively within a team.	Teamwork	Conflict resolution	Synergy creation	TeamCompetencies
TrustBuilding	A factor related to building trust within the team.	Transparency	Consistent communication	Reliability	TeamCompositionCustomization
SkillDiversity	A factor related to having a diverse set of skills within the team.	Varied expertise	Multidisciplinary teams	Diverse perspectives	TeamCompositionCustomization
RoleClarity	A factor related to having clear roles and responsibilities within the team.	Defined roles	Job descriptions	Role-specific accountability	TeamCompositionCustomization
StakeholderEngagement	A factor related to	Regular updates	Involvement opportunities	Stakeholder meetings	TeamCompositionCustomization

	engaging stakeholders effectively in the project.				
IterationCycle	A typical cycle in agile methodologies representing an iteration.	Sprint planning	Development iteration	Iteration review	Iteration
FunctionalRequirement	A requirement that specifies what the system should do.	Feature specification	System functionality	User needs	Requirement
NonFunctionalRequirement	A requirement that specifies how the system should perform.	Performance criteria	Security requirements	Usability standards	Requirement
InitialPlanningPhase	The initial planning phase of a project.	Project charter	Scope definition	Risk assessment	ProjectPhase
ExecutionPhase	The execution phase of a project.	Task execution	Monitoring	Progress reporting	ProjectPhase
ClosingPhase	The closing phase of a project.	Project wrap-up	Final deliverables	Post-project review	ProjectPhase
ProjectSponsor	The sponsor of the project.	Funding source	Executive supporter	Project advocate	Stakeholder
ProjectManager	The manager responsible for overseeing the project.	Project oversight	Team leadership	Schedule management	Stakeholder
EndUser	The end-user who will use the project deliverables.	Customer	Client	User	Stakeholder
SubjectMatterExpert	An expert providing specialized knowledge for the project.	Knowledge authority	Domain specialist	Technical expert	Stakeholder
BudgetPlanning	The process of planning the	Cost estimation	Financial forecasting	Resource allocation	CostManagement

	budget for the project.				
RiskMitigation	The process of mitigating identified risks in the project.	Contingency planning	Risk assessment	Preventative measures	RiskManagement
QualityAssuranceProcess	Processes ensuring the project meets the required quality standards.	Quality checks	Standard compliance	Quality audits	QualityManagement
CommunicationPlan	The plan outlining how project information will be communicated.	Information flow	Stakeholder updates	Communication channels	CommunicationManagement
SupplierContract	A contract with a supplier for project procurements.	Vendor agreements	Procurement terms	Service level agreements	ProcurementManagement
StakeholderAnalysis	The process of analyzing and understanding project stakeholders.	Stakeholder mapping	Influence assessment	Needs identification	StakeholderManagement
ScrumFramework	An agile methodology framework for iterative development.	Sprint cycles	Scrum ceremonies	Backlog management	AgileMethodology
KanbanBoard	A visual board used in the Kanban methodology for managing work.	Task visualization	Workflow management	Work-in-progress limits	AgileMethodology
XPPractice	A practice from the Extreme Programming (XP) methodology.	Pair programming	Test-driven development	Continuous feedback	AgileMethodology
EssenceKernelAlpha	An alpha representing a	Essential element	Core aspect	Basic entity	EssenceFramework

	fundamental aspect of the Essence framework.				
IterationOne	The first iteration cycle in an agile project.	Initial sprint	Pilot iteration	Starting cycle	Iteration
ProjectCharterDocument	The document outlining the project charter.	Project authorization	Project scope	Project objectives	IntegrationManagement
GanttChart	A visual representation of the project schedule.	Timeline visualization	Task sequencing	Schedule tracking	ScheduleManagement
WorkBreakdownStructure	A hierarchical decomposition of the total scope of work.	Task hierarchy	Work packages	Scope breakdown	ScopeManagement
ProjectClosureReport	The report generated at the end of the project closing phase.	Final report	Project summary	Lessons learned	ProjectPhase
UserStory	A user story in agile methodologies representing a feature from the perspective of the end-user.	User requirement	Feature description	User-centric feature	AgileMethodology
SprintPlanning	The process of planning a sprint in Scrum.	Sprint goal setting	Task assignment	Sprint backlog	AgileMethodology
RetrospectiveMeeting	A meeting at the end of a sprint to reflect on the process and plan improvements.	Reflection session	Improvement planning	Team feedback	AgileMethodology
DailyStandup	A daily meeting in agile methodologies	Status update	Blocker identification	Daily check-in	AgileMethodology

	to discuss progress and obstacles.				
SoftwareModule	A module within the software system being developed.	Functional unit	Code module	System component	SoftwareSystem
CodeComponent	A component of code within the software system.	Code snippet	Code unit	Source code	SoftwareSystem
DesignDocument	A document outlining the design of the software system.	Architectural design	System blueprint	Design specifications	SoftwareSystem
UnitTest	A test verifying the correctness of individual units of code.	Code testing	Function validation	Unit verification	QualityManagement
IntegrationTest	A test verifying the integration of different components of the software system.	System integration	Interface testing	Combined module testing	QualityManagement
DeploymentPlan	The plan for deploying the software system.	Release plan	Deployment strategy	Rollout plan	SoftwareSystem
OperationSupport	The support provided for the operation of the software system.	Operational assistance	Maintenance support	User help desk	SoftwareSystem
CustomerFeedback	Feedback from the customer regarding the project deliverables.	Client feedback	End-user comments	Customer satisfaction	Stakeholder
PerformanceMetric	A metric used to measure the	Efficiency metric	Productivity measure	Performance indicator	TeamPerformance

	performance of the team.				
CollaborationTool	A tool used to facilitate collaboration within the project team.	Communication platform	Collaboration software	Team interaction tool	WayOfWorking
AgilePractice	A practice adopted from agile methodologies.	Iterative development	Continuous feedback	Flexible planning	WayOfWorking
ContinuousIntegration	A practice of continuously integrating code changes into the main branch.	CI pipeline	Automated testing	Frequent integration	WayOfWorking
SWRLRuleExample	An example of a SWRL rule used for customizing project processes.	Rule-based customization	Logical rule application	SWRL query	RuleBasedCustomization
SQRWLQueryExample	An example of a SQRWL query used for customizing project processes.	Query customization	SQRWL query application	Data retrieval rule	RuleBasedCustomization
TeamPerformanceReview	A meeting to review the performance of the project team.	Performance assessment	Team evaluation	Review session	ReflectionMeeting
EmpiricalDataCollection	The process of collecting empirical data for research purposes.	Data gathering	Observational data	Empirical study	EmpiricalResearchFramework
StakeholderFeedbackSurvey	A survey conducted to collect feedback from stakeholders.	Stakeholder survey	Feedback questionnaire	Stakeholder input	CommunicationManagement

IterationReview	A review of the completed iteration cycle.	Sprint review	Iteration feedback	Cycle assessment	AgileMethodology
CompetenceAssessment	An assessment of the competences required for the project team.	Skill evaluation	Competence measurement	Capability assessment	Competence
DiversityTraining	Training provided to promote diversity within the project team.	Inclusion training	Diversity awareness	Equality training	Diversity
TrustBuildingWorkshop	A workshop aimed at building trust within the project team.	Trust exercises	Team-building workshop	Relationship strengthening	Trust
CollaborationFramework	A framework used to enhance collaboration within the project team.	Team collaboration	Cooperative framework	Interaction structure	Collaboration
PersonalityAssessment	An assessment of the personality traits of project team members.	Personality test	Trait evaluation	Behavioral assessment	Personality
MotivationSurvey	A survey conducted to understand the motivational factors affecting project team members.	Motivation assessment	Incentive survey	Team motivation analysis	Motivation

Table 43: ProjOnto Individuals