Journal of Computer and Knowledge Engineering, Vol. , No.. 2025.
DOI:

Providing an Ontology for Extracting Functional and non-Functional Software Requirements of an Online Sales System*

Houbakht Attaran[1], Esmaeil Kheirkhah[2], Hassan Shakeri[3]

***ABSTRACT*- Incomplete and costly requirements extraction, especially for complex systems such as online sales applications within the smart city ecosystem, remains a major challenge in software engineering. Requirements engineering in smart city software is often considered the "missing link" in development methodologies. While ontologies have been proposed to address this, issues with relationship definition and non-functional requirement (NFR) extraction persist. This paper presents a comprehensive ontology for an online sales system as a case study. We detail a structured methodology for ontology development and demonstrate its efficacy. The results, compared with data from four commercial software producers, show that the ontology extracts functional and non-functional requirements with high coverage, leading to significant savings in extraction time, project implementation, costs, and reduction of late-stage changes.**

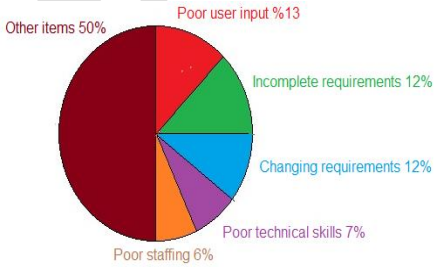
***Index Terms*- Development Methodology, Ontology, Requirement Engineering, Smart city, Software Development Lifecycle.**

INTRODUCTION

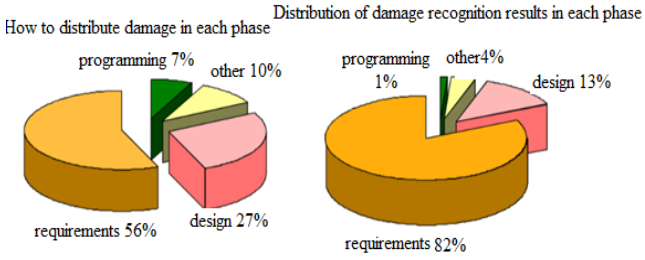
The timely and accurate extraction of both functional and non-functional requirements is a persistent challenge in software development, particularly for large-scale applications like those in a smart city. A smart city, built upon the Internet of Things (IoT), requires a comprehensive model to describe its entities, features, and their interactions. An ontological description can identify software classes, their relationships, properties, and interactions, forming a structural basis for determining comprehensive software requirements.

Software designed for a smart city integrates human, physical, and digital systems. However, existing integration methods often result in isolated "island" applications that lack interoperability, suffer from data redundancy, and face potential conflicts. This fragmentation makes identifying common requirements difficult. The lack of clarity in required features, services, and classes challenges software architectures and methodologies, often leading to project failures.

Various factors contribute to the failure of projects based on information technology system. Graph1 shows these factors and the extent of each role and Graph 2 identifies main sources of failures in projects.



Graph 1: The impact of various factors on the failure of IT based projects (CHAOS 2021)



Graph 2: The main sources of errors in projects (CHAOS 2021)

As evident from these analyses, a significant percentage of IT project failures stem from weaknesses in requirements identification and management. Extracting requirements is time-consuming, costly, and often incomplete at a project's outset, especially for non-functional requirements (NFRs). The core problems include:

1. Time-consuming requirements extraction.
2. Incompleteness of extracted requirements.
3. High cost of requirements engineering.

* Manuscript received Revised, accepted .
[1] Department of Computer Engineering, Ma. C., Islamic Azad University, Mashhad, Iran, Email: Houbakht.Attaran@iau.ac.ir
[2] Corresponding author: Department of Computer Engineering, Ma. C., Islamic Azad University, Mashhad, Iran, Email: esmaeil.kheirkhah@iau.ac.ir
[3] Department of Computer Engineering, Ma. C., Islamic Azad University, Mashhad, Iran, Email: hassanshakeri@iau.ac.ir

4. The particular challenge of identifying NFRs early.

To address these issues, this paper proposes and evaluates the use of ontology. Ontologies play a fundamental role in the standard description and sharing of domain knowledge [1]. The primary challenges in designing a requirements extraction ontology lie in determining classes, their relationships, properties, and methods. This article contributes a novel ontology for an online sales system, detailing its development methodology and evaluating its performance against industry practices. The results indicate substantial improvements in requirement coverage, especially for NFRs, and significant reductions in time, cost, and project churn.

RESEARCH BACKGROUND

This section reviews literature in three focused areas: ontologies in requirements engineering, ontologies in e-commerce/smart cities, and foundational smart city frameworks. The concept of a "smart city" has evolved over decades, with definitions varying based on community needs and technological context [2]. Our previous works [3] [4] provide an ontology-based platform for determining high-level software classes in a smart city, which serves as the foundation for this study.

Requirements engineering is widely recognized as a critical challenge, often termed the "missing link" in software development methodologies, especially for large projects [5]. Agile methodologies, while popular, also face significant requirements engineering challenges [6].

Ontologies have been applied to structure knowledge in specific domains. In education, ontologies like OntoGamif [7] and instructional design frameworks [8] model domain concepts. In healthcare, ontologies integrate data from heterogeneous devices [9] or model smart healthcare systems (SHCO) [10]. For intelligent transportation, ontology-based architectures describe system components [11]. In urban planning, ontologies aid building energy modeling [13] and social theory analysis [12].

Closer to our focus, ontologies are increasingly used in systems engineering and requirements management. Recent work includes ontologies for AI-powered system requirements (AI-REM) [14], ontology-based explainable AI in manufacturing [15], and ontology embedding techniques [16]. Other studies explore ontology-based search systems [17], knowledge graphs for infrastructure resilience [18], and advanced ontology modeling frameworks [19] [20]. Research also addresses transforming safety requirements into ontological knowledge [21], creating digital threads for systems engineering [22], model-based safety analysis [23], and functional analysis [24]. Large-scale spatial knowledge graphs also rely on robust ontologies [25].

Despite these advances, a review of the literature reveals a gap: there is a lack of a detailed, practical ontology specifically designed to systematically extract both functional and non-functional software requirements for a concrete system within the smart city domain. This work aims to fill that gap.

METHODOLOGY

This section details the structured, multi-phase methodology employed to develop and evaluate the requirements extraction ontology.

A. Ontology Development Process

The ontology was developed using an iterative, expert-driven approach:

1. Class Identification: Core classes (e.g., Administrator, Customer, Online-Shop, Goods) were identified through analysis of standard online sales system features and consultations with three domain experts each having over 15 years of experience in e-commerce software development.

2. Hierarchy and Relationship Definition: The class hierarchy and relationships (e.g., is-a, has-a) were defined. The high-level smart city ontology from our prior works [3] [4] provided the foundational superclass (Store) for the Online-Shop class, ensuring alignment with a broader smart city context.

3. Property and Constraint Specification: For each class, Object Properties (relationships between instances) and Data Properties (attributes like hasName, hasPrice) were added. Crucially, properties to capture NFRs were explicitly defined, such as hasMaxResponseTime (Data Property), requiresAuthentication (Object Property constraint on an action), or hasAvailabilityRequirement.

4. Tool and Implementation: The ontology was formally constructed using Protégé (version 5.5.0), enabling precise definition in OWL. This allowed for logical consistency checking and the use of reasoning.

B. Requirements Extraction Mechanism

Requirements were extracted programmatically:

1. The populated ontology was exported as an RDF graph.

2. Custom SPARQL queries were designed to traverse the graph. Different query patterns targeted different requirement types (see Table 1).

3. Functional requirements were primarily derived from queries about class methods, actions (canPerform), and sequences of operations.

4. Non-functional requirements were derived from queries about Data Properties encoding quality attributes (e.g., hasMaxResponseTime < 2s) and conditional Object Properties (e.g., performAction X only if requiresAuthentication is true).

C. Evaluation and Comparison Method

To validate the ontology's effectiveness, a comparative analysis was conducted:

- Data Source: Historical project data was collected from four software companies (labeled C1-C4) specializing in online sales systems. The data included

final requirement specifications, project timelines, and cost breakdowns.

- Comparison Metrics: The ontology's output was compared against the companies' final, implemented requirements on several metrics: total functional & non-functional requirements, time/cost for requirements phase, and the percentage/cost of changes due to late-discovered requirements.
- Analysis: Quantitative comparison was supplemented with qualitative analysis to interpret differences in requirement granularity and classification.

IMPLEMENTATION AND RESULTS OF THE PROPOSED ONTOLOGY

Based on the methodology, a detailed ontology for an online sales system was implemented. The overarching smart city ontology from our prior work [4] providing the context is shown in Figure 1.

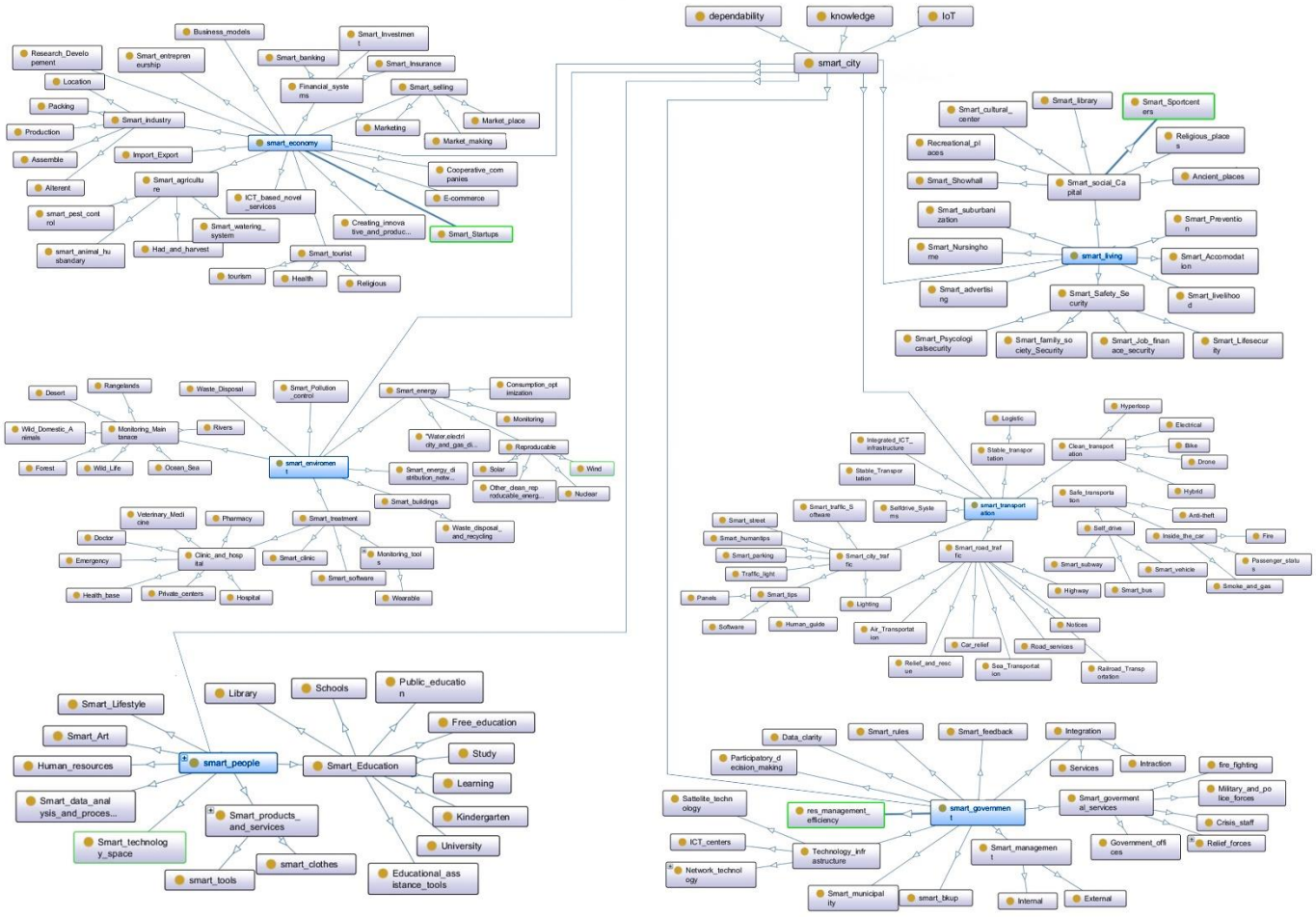


Figure 1: Proposed ontology of smart city [4]

The core ontology for the online sales system, showing main classes and relationships, is presented in Figure 2.

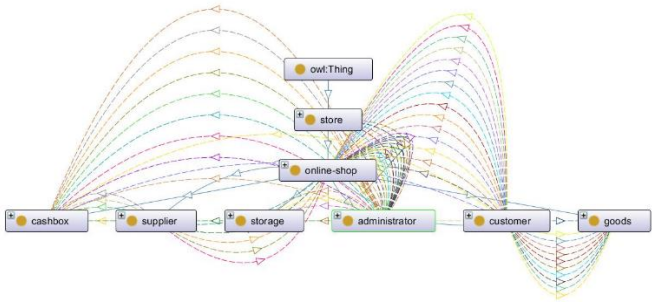


Figure 2: Proposed Ontology for Online marketing

The following figures (Fig3-Fig9) illustrate the detailed structure for key classes, showcasing properties, methods, and inheritance:

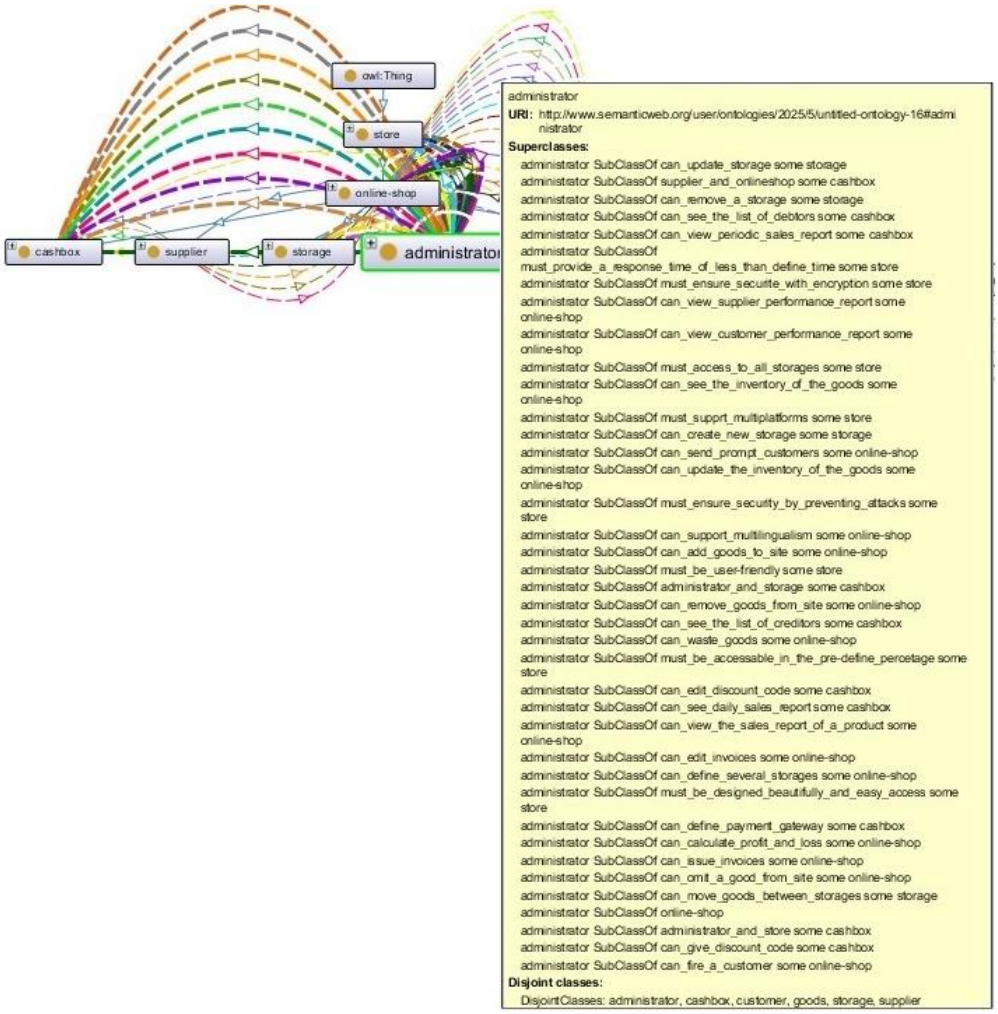


Figure 3: Methods and use cases for Administrator class

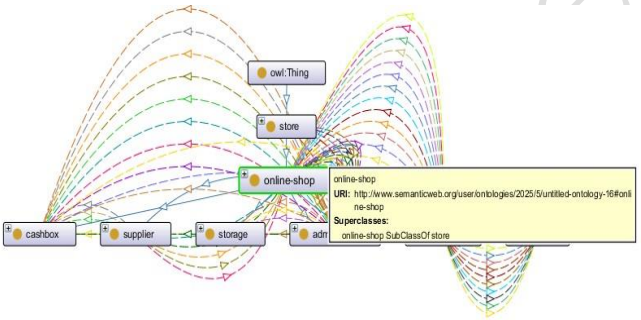


Figure 4: Online-shop class with store superclass

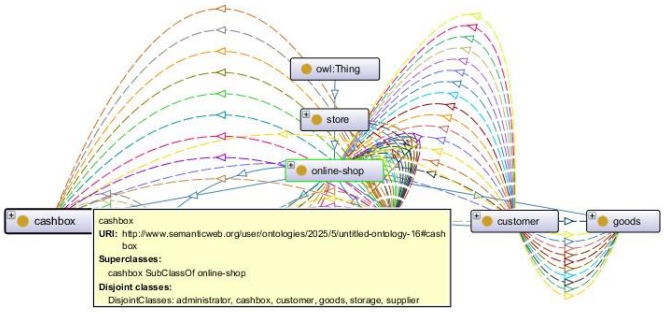


Figure 5: cashbox class with online-shop superclass and disjoint classes

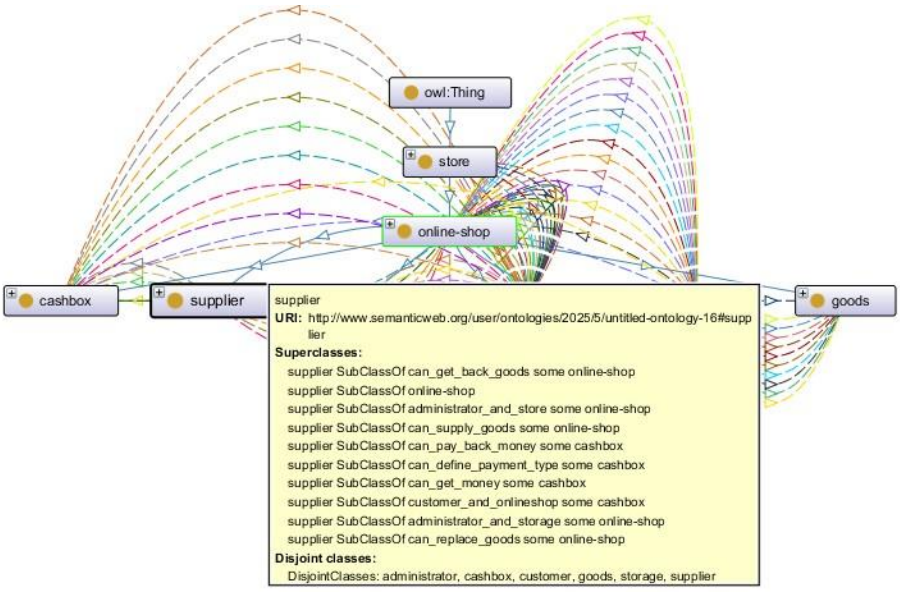


Figure 6: Supplier class with superclass, disjoint classes and methods

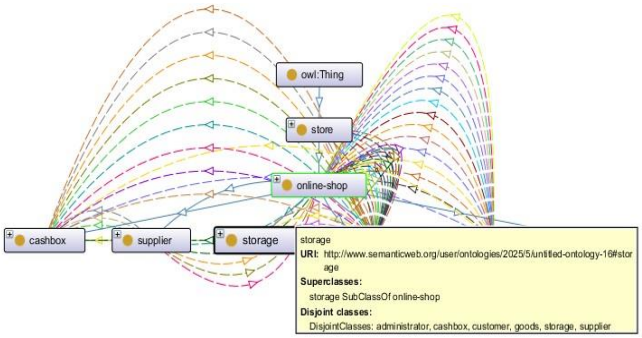


Figure 7: Storage class with superclass and disjoint classes

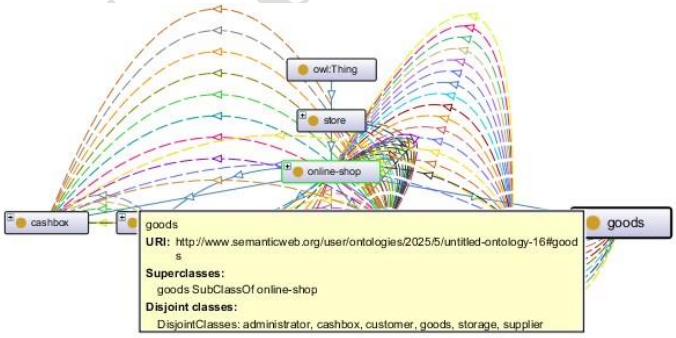


Figure 8: Goods class with superclass and disjoint classes

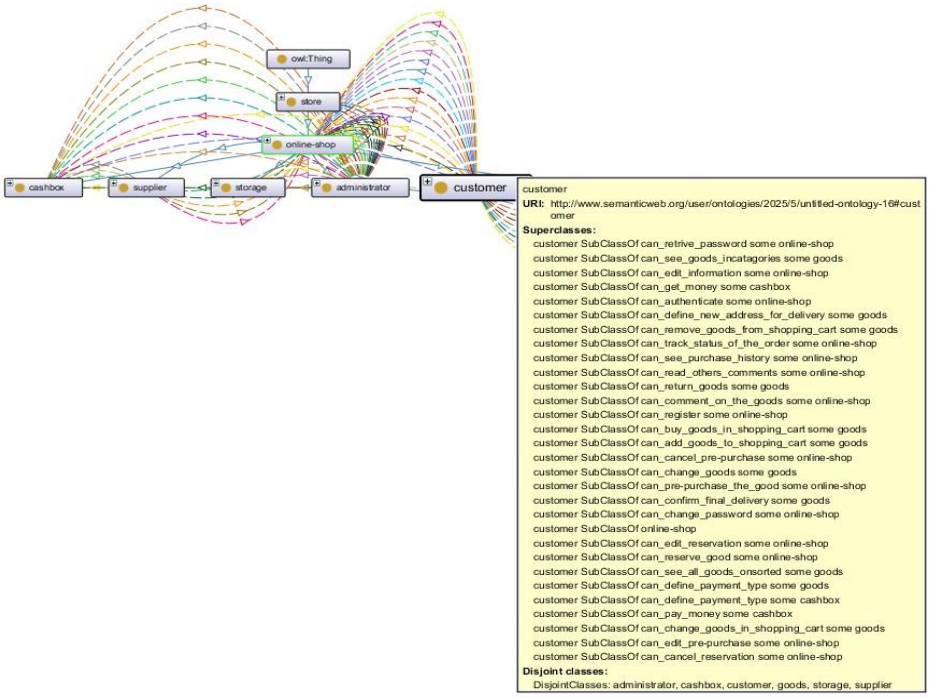


Figure 9: Customer class with superclass, disjoint classes and methods

Table 1: Examples of SPARQL Queries for Requirements Extraction.

Query Purpose	**Sample SPARQL Pattern**	**Extracted Requirement Type**
List all actions of a class	SELECT ?action WHERE { :Admin :canPerform ?action }	Functional (Admin capabilities)
Find entities with performance constraints	SELECT ?entity WHERE { ?entity :hasMaxResponseTime ?time . FILTER (?time < 2000) }	Non-Functional (Performance)
Find preconditions for an action	SELECT ?precond WHERE { :ProcessOrder :requiresAuthentication ?precond }	Non-Functional (Security)

Using the ontology and SPARQL queries, requirements were extracted. Samples of the output are shown in Figures 10 -13:

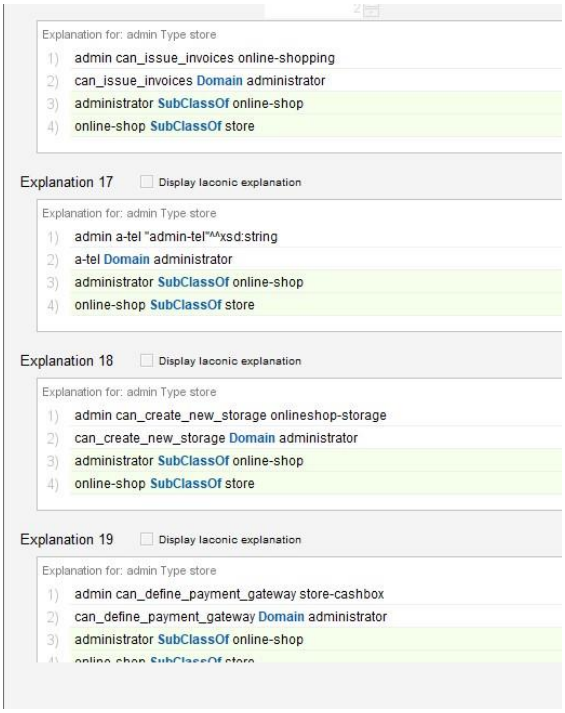


Figure 10: Part of Admin functional requirements

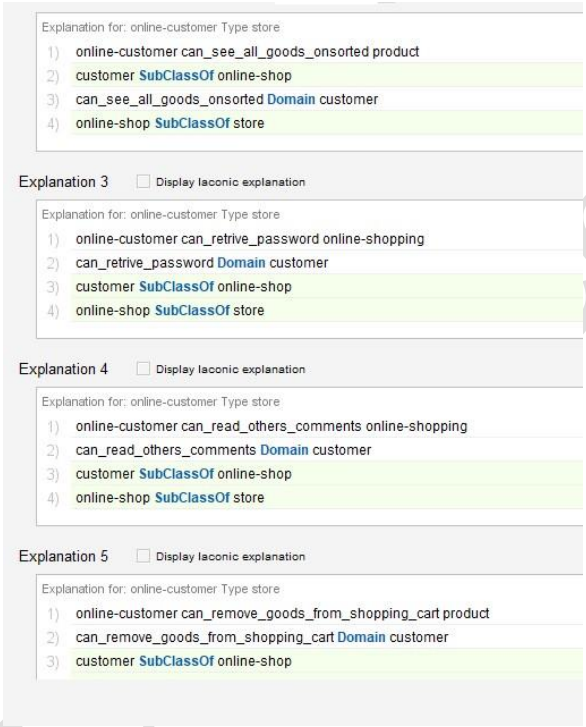


Figure 11: Part of customer functional requirements

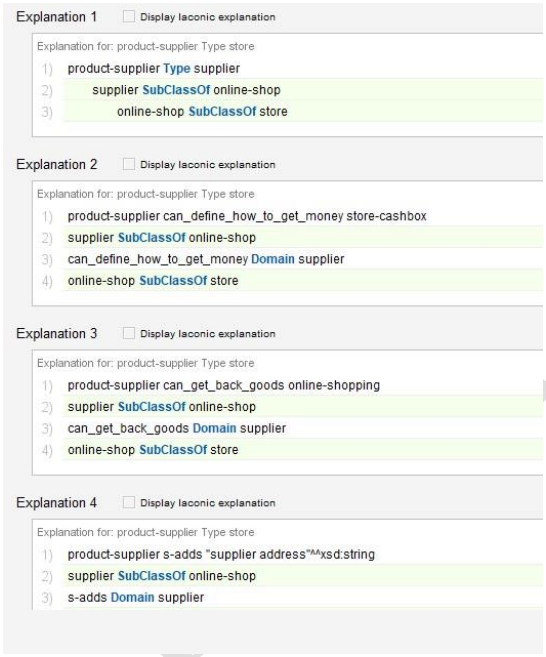


Figure 12: Part of supplier functional requirements

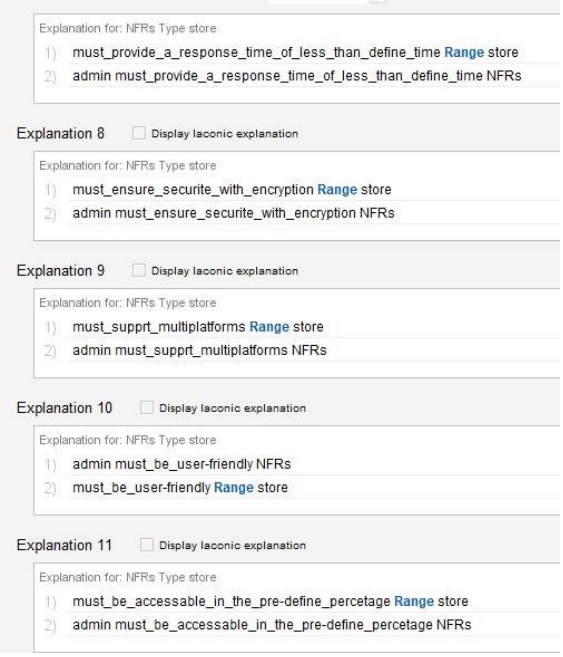


Figure 13: Part of non-functional requirements.

A SPARQL query (COUNT on owl:ObjectProperty) shown in following code, determined the ontology could yield 79 distinct requirement elements (Figure 14).

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT (COUNT (DISTINCT ?p) AS ?totalProperties)
WHERE {
?pa owl:ObjectProperty
}
```

totalProperties
"79"^^<http://www.w3.org/2001/XMLSchema#integer>

Figure 14: Number of extracted requirements

Sample instances extracted via queries like the following query, are shown in Figures 15 & 16. Also, the following code snippet was used in SPARQL to extract the requirements:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?individual ?type
WHERE {
?individual a ?type .
}
```

individual
customer_and_onlineshop
can_pre-purchase_the_good
can_get_money
can_edit_information
can_define_how_to_get_money
can_omit_a_good_from_site
can_see_goods_incatagories
customer_and_onlineshop
can_reserve_good
can_change_goods_in_shopping_cart
can_see_purchase_history
can_view_supplier_performance_report
can_pay_back_money
can_remove_a_storage
can_see_purchase_history
must_ensure_securite_with_encryption
can_change_goods_in_shopping_cart
can_see_the_list_of_debtors
can_change_goods

Figure 15: A sample of extracted requirements by SPARQL

individual
can_edit_information
can_see_goods_incatagories
can_calculate_profit_and_loss
can_change_goods_in_shopping_cart
can_get_back_goods
can_see_the_list_of_creditors
can_create_new_storage
can_send_prompt_customers
can_change_goods
can_support_multilingualism
can_reserve_good
must_be_user-friendly
can_supply_goods
can_edit_pre-purchase
customer_and_onlineshop
must_provide_a_response_time_of_less_than_define_time
can_see_daily_sales_report
must_supprt_multiplatforms
can_comment_on_the_goods
can_view_periodic_sales_report

Figure 16: Another sample of extracted requirements by SPARQL

COMPARISON AND PROOF OF EFFECTIVENESS

The ontology's performance was compared against the final project outcomes of four companies (Company1-Company4). The key comparison is between the total requirements identified by the ontology and the total requirements ultimately implemented by the companies. The evaluation metrics are defined in Table 2.

- Number of functional requirements extracted during project execution (Np)
- Total number of non-functional requirements (Nnon)
- Total number of final requirements (Ntotal)
- Total time spent on requirements extraction (Te)
- Cost spent on requirements extraction (Ce)
- Percentage of changes in design and implementation with discovery of new requirements (CHp)
- Time required to implement changes (Tch)
- Cost spent on implementing changes (Cch)

Table 2: Comparative parameter values

	Np	Nnon	Ntotal	Te	Ce	CHp	Tch	Cch
Company 1	22	3	74	103	2.5	20%	40	7
Company 2	21	5	81	98	2.2	8%	25	5
Company 3	32	4	78	84	1.8	10%	52	11
Company 4	28	4	76	100	2	10%	45	9
Ontology	0	5	79	1	0.1	0	0	0

The following charts visualize the comparison:

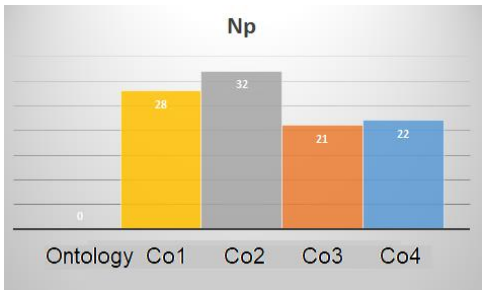


Chart 1: Number of functional requirements extracted during project execution

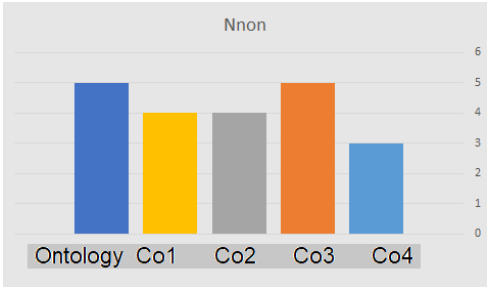


Chart 2: Total number of non-functional requirements

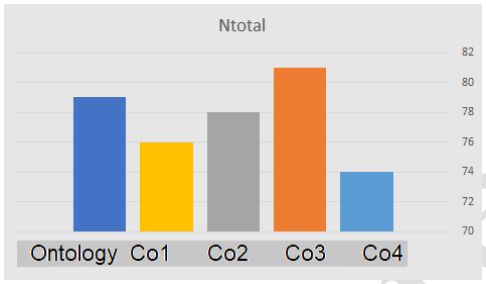


Chart 3: Comparison of Total final requirements

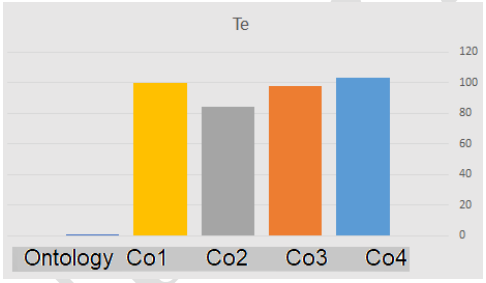


Chart 4: Total time spent extracting requirements

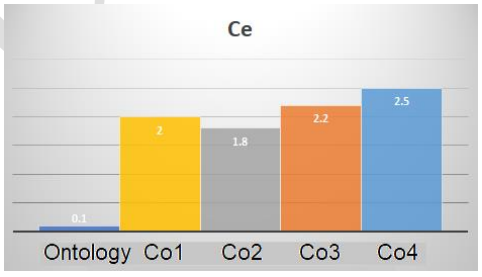


Chart 5: Cost spent on extraction requirements

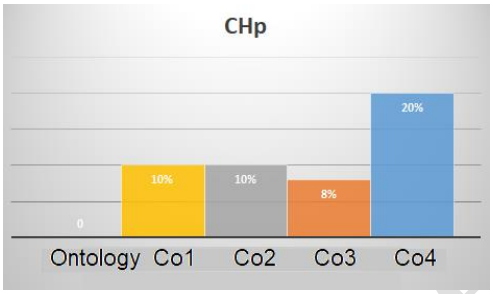


Chart 6: Percentage of changes in design and implementation due to discovery of new requirements

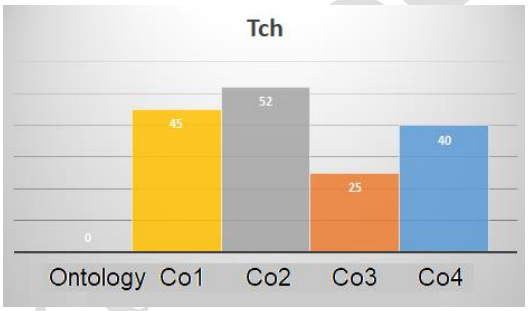


Chart 7: Time required to make changes

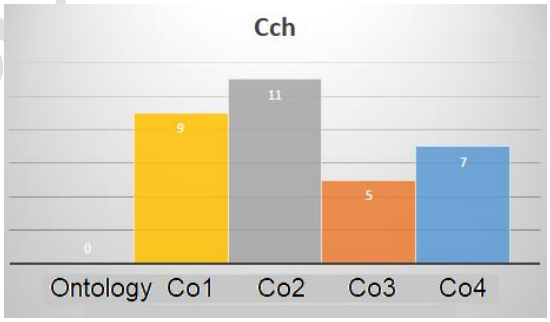


Chart 8: Cost spent on making changes

DISCUSSION

This section interprets the results, addresses the research questions, and examines the study's limitations.

A. Interpretation of Results

The comparative analysis shows that the ontology-based approach identified a comprehensive set of requirements early in the lifecycle. Key findings include:

- High Coverage: The ontology's total requirement count (79) closely aligns with or exceeds the final requirement count of the companies (range: 72-82), indicating high coverage. The slightly higher count in some cases is attributed to the ontology's more granular decomposition of some composite requirements.

- Early NFR Identification: As shown in Chart 3, the ontology successfully identified a substantial set of NFRs upfront, whereas companies discovered many of these later during development (contributing to CHp).
- Efficiency Gains: Charts 5, 6, 8, and 9 demonstrate dramatic reductions in time and cost for both the initial requirements phase and for implementing changes, directly addressing the core problems outlined in the introduction.

B. Answering the Research Questions

The study set out to address the challenges of requirements engineering for smart city software. The results confirm that:

- A well-structured ontology can systematically extract both functional and non-functional requirements.
- This method leads to more complete early requirement specifications, reducing late discoveries and their associated costs.
- The approach is practically effective, as evidenced by favorable comparisons with industrial practices.

C. Threats to Validity

- Construct Validity: The use of "number of requirements" as a primary metric may not fully capture requirement quality or priority. Future work should include qualitative assessments by practitioners.
- Internal Validity: The study relies on historical data from a limited number of companies (four). While efforts were made to verify data accuracy, potential recall bias exists. The ontology's development involved expert judgment, which introduces subjectivity, though we mitigated this through multiple experts and iteration.
- External Validity: The current evaluation is focused on the online sales domain within the smart city context. Generalizability to other smart city subsystems (e.g., smart grid, traffic management) requires further validation through additional case studies.

CONCLUSION AND FUTURE WORK

This paper presented a novel ontology-based framework for extracting software requirements, demonstrated through an online sales system case study. By detailing a rigorous methodology and providing a fair comparison with industrial data, we showed that the approach achieves high requirement coverage, particularly for non-functional requirements, while generating significant savings in time, cost, and project instability.

Future work will focus on: (1) extending the ontology to other smart city domains to test generalizability, (2) developing a semi-automated tool that integrates this ontological approach into standard requirements management platforms, and (3) conducting longitudinal studies to measure the impact on project success rates.

ACKNOWLEDGMENT

The authors of this article sincerely thank all the software companies that provided their experiences, statistics, and performance to compare and prove the effectiveness of the new model.

REFERENCES

[1] Chen, Zuohai, et al. "Ontology Construction of City Hotline Service for Urban Grassroots Governance." *2022 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)* . IEEE, 2022

[2] Benkhaled, Sihem, et al. "An ontology-based contextual approach for cross-domain applications in internet of things." *Informatica* 46.5 (2022).

[3] Houbakht Attaran, Nahid Kheibari, and Davoud Bahrepour. "Toward integrated smart city: A new model for implementation and design challenges." *GeoJournal* 87, no. Suppl 4 (2022): 511-526.

[4] Houbakht Attaran, Esmaeil Kheirkhah, Hassan Shakeri. " Providing an Ontology-based Framework to Determine the High-Level Software Classes of a Smart City." Creative City Design (CCD), VOL 7, No 3, 2024, DOI: 10.71619/crcd-2024-8176

[5] Houbakht Attaran, Esmaeil Kheirkhah, " Review and comparison of software development methodologies." 15th International Conference on Science and Technology Advances, Mashhad ,https://civilica.com/doc/1608987

[6] Houbakht Attaran, Esmaeil Kheirkhah, Hassan Shakeri. " The Challenges of Requirements Engineering in the Development of Smart City Software Using Agile Methodologies: A Systematic Literature Review." Journal of Information Technologies in Engineering Design, Issue 1, VOL 17 (2024).

[7] Bouzidi, R., De Nicola, A., Nader, F. & Chalal, R. (2019). OntoGamif: A modular ontology for integrated gamification. Applied Ontology, 14(3), 215–249. doi:10.3233/AO-190212.

[8] Chimalakonda, S. & Nori, K. (2020). An ontology based modeling framework for design of educational technologies. Smart Learning Environments, 7(28), 1–24.

[9] Peng, C. & Goswami, P. (2019). Meaningful integration of data from heterogeneous health services and home environment based on ontology. Sensors, 19(8), 1747.

[10] Tiwari, S. & Abraham, A. (2020). Semantic assessment of smart healthcare ontology. International Journal of Web Information Systems, 16(4), 475–491. doi:10.1108/IJWIS-05-2020-0027.

[11] Fernandez, Susel, et al. "Ontology-based architecture for intelligent transportation systems using a traffic sensor network." *Sensors* 16.8 (2016): 1287.

[12] Epstein B. *Social Ontology* . Cambridge University Press; 2025.

[13] Ma, Rui, Qi Li, Botao Zhang, Hao Huang, and Chendi Yang. "An ontology-driven method for urban building energy modeling." *Sustainable Cities and Society* 106 (2024): 105394.

[14] Sadovski, Eran, Itzhak Aviv, and Irit Hadar. "Towards a Comprehensive Ontology for Requirements Engineering for AI-Powered Systems." In *International Working Conference on Requirements Engineering: Foundation for Software Quality* , pp. 219-230. Cham: Springer Nature Switzerland, 2024.

[15] Naqvi, Muhammad Raza, Linda Elmhadhbi, Arkopaul Sarkar, Bernard Archimede, and Mohamed Hedi Karray. "Survey on ontology-based explainable AI in manufacturing." *Journal of Intelligent Manufacturing* 35, no. 8 (2024): 3605-3627.

[16] Chen, Jiaoyan, Olga Mashkova, Fernando Zhapa-Camacho, Robert Hoehndorf, Yuan He, and Ian Horrocks. "Ontology embedding: a survey of methods, applications and resources." *IEEE Transactions on Knowledge and Data Engineering* (2025).

[17] Huettemann, Sebastian, Roland M. Mueller, and Barbara Dinter. "Designing ontology-based search systems for research articles." *International Journal of Information Management* 83 (2025): 102901.

[18] Gan, Bin-Lin, Dong-Mei Zhang, Zhong-Kai Huang, Fei-Yu Zheng, Rui Zhu, and Wei Zhang. "Ontology-driven knowledge graph for decision-making in resilience enhancement of underground structures: Framework and application." *Tunneling and Underground Space Technology* 163 (2025): 106739.

[19] Kim, Siku, Yee Yeng Liau, and Kwangyeol Ryu. "Ontology Modeling Using Fractal and Fuzzy Concepts to Optimize Metadata Management." *Applied Sciences* 15, no. 13 (2025): 7193.

[20] Bettaz, Mohamed, and Mourad Maouche. "Towards an Ontological-based CIM Modeling Framework for IoT Applications." *Informatica* 48, no. 4 (2025).

[21] Wu, Zhijiang, Mengyao Liu, and Guofeng Ma. "A Semi-Automatic Ontology Development Framework for Knowledge Transformation of Construction Safety Requirements." *Buildings* 15, no. 4 (2025): 569.

[22] Gautier, Raphael H., Noe Lepez Da Silva Duarte, Adam Baker, and Dimitri Mavris. "Formulation of an Ontology-Based Digital Thread for Analysis and Decision Support Activities." In *AIAA SCITECH 2025 Forum* , p. 1092. 2025.

[23] Mokos, Konstantinos, Panagiotis Katsaros, and Preben Bohn. "Model-based safety analysis of requirement specifications." *Journal of Systems and Software* 219 (2025): 112231.

[24] Johnson, Hamilton E., Mayuranath SureshKumar, Hanumanthrao Kannan, and L. Dale Thomas. "Ontology-Based Functional Analysis of an Aerospace System." In *AIAA SCITECH 2025 Forum* , p. 1212. 2025.

[25] Shimizu, Cogan, Shirly Stephen, Adrita Barua, Ling Cai, Antrea Christou, Kitty Currier, Abhilekha Dalal et al. "The KnowWhereGraph ontology." *Journal of Web Semantics* (2024): 100842.