CapProOnt: An Ontology for Describing Capstone Project Management in Academic Systems

Abeer Zafar, SobiaZafar , Isbah Imran

Department of Computer Science, University of Sargodha, Sargodha, Pakistan

Editors:

Abstract

Semantically rich and machine-interpretable descriptions of capstone project processes can provide meaningful advantages in academic project management and evaluation. However, there is a noticeable lack of such structured representations in educational institutions. In this paper, we present the development of an ontology named **CapProOnt**, specifically designed to describe various elements involved in university-level capstone projects. The ontology models key concepts such as students, supervisors, proposals, evaluation criteria, documentation, technologies used, and project time-lines. Although CapProOnt is developed within the academic domain, it can serve as a reference model for institutions aiming to digitally manage and assess capstone or final year projects. The ontology is structured into several interlinked modules that define object properties, data properties, class hierarchies, and restrictions to ensure semantic clarity. This development process was carried out in alignment with university guidelines and reviewed by academic stakeholders to ensure its practical applicability.

Keywords:

Ontology, Capstone Project, Academic Evaluation, OWL, University Project Management, Final Year Project, Semantic Web

1. Introduction

The increasing adoption of digital tools in academic institutions has highlighted the need for structured and semantic approaches to manage educational workflows. Among these, capstone or final year projects play a significant role in evaluating students' practical competencies, collaboration abilities, and innovative thinking. Despite their importance, many universities still rely on fragmented or semi-digital processes to handle various stages of capstone project management, including proposal submissions, documentation, evaluations, and stakeholder assignments. The absence of semantically rich, machine-interpret-able models hinders automation, tracking, and integration with institutional systems.

To address this gap, we propose an ontology named **CapProOnt** a formal, OWL 2-based ontology for representing key components of academic capstone projects. CapProOnt provides a vocabulary and structured model to describe essential elements such as students, supervisors, initial and final proposals, various types of documentation (e.g., SRS, design reports), evaluation criteria (e.g., technical, presentation, and innovation), project milestones, and technologies used. The ontology aims to support digital academic management by enabling semantic querying, inference, and inseparability across platforms.

CapProOnt was developed using the Protege ontology development environment, following modular design principles and the **Neon methodology** for ontology engineering. The ontology is aligned with related models in the education and evaluation domains and is designed to be extensible and adaptable to various institutional policies and work-flows.

This paper presents the design process and structure of CapProOnt, including its core classes, object and data properties, and restrictions. Furthermore, we evaluate the ontology with respect to **domain coverage** and **quality of modeling**. The paper concludes with insights gained during the development process and outlines future work, including

integration with student information systems and intelligent feedback tools.

2. Related Work

In the domain of education and semantic modeling, several ontologies have been developed to represent aspects of academic systems. These ontologies serve diverse objectives and thus vary in terms of coverage, scope, and modeling strategies. Some focus on student learning outcomes, while others aim to formalize administrative processes or assessment mechanisms.

The **LOM (Learning Object Meta-data)** ontology is a well-established model used for describing educational resources. It includes meta data elements such as title, description, format, and difficulty level, but it does not cover project-based learning or the relationships between students, supervisors, and evaluations. The **CEDS (Common Education Data Standards)** initiative provides a comprehensive vocabulary for data exchange between educational institutions; however, it is primarily focused on institutional reporting and lacks fine-grained modeling of project life cycles or document management.

The **EduOnto ontology** attempts to capture a broader view of educational processes, including curriculum structures and learning activities. While it provides a foundation for representing academic environments, it does not address the lifecycle of student projects, such as proposal evaluation, supervisor assignment, or timeline tracking. Similarly, the **SWRC (Semantic Web for Research Communities)** ontology models research-related entities (e.g., persons, publications, organizations) but is limited in expressing the stages and evaluations of academic capstone projects.

Unlike these efforts, **CapProOnt** is specifically designed to address the needs of capstone project management. It incorporates detailed modules for representing documentation (e.g., SRS, final reports), proposal stages, stakeholder roles (students, supervisors, evaluators), evaluation criteria, and associated time lines. The ontology structure has been inspired by best practices in ontology engineering and evaluated using criteria such as clarity, completeness, and consistency.

Elements from these previous works have influenced the conceptual design of CapProOnt. For instance, CapProOnt aligns with modular

design principles from **Neon methodology** and borrows ideas of hierarchical classification from upper ontologies like **DUL**. These references provided guidance during development and informed the quality assurance strategies discussed in later sections.

3. Design Methodology

The development of well-structured ontologies demands adherence to established engineering methodologies that ensure clarity, completeness, and reuse. In the context of our ontology, **CapProOnt**, we considered several popular methodologies such as **On-To-Knowledge** [1], **DILIGENT** [2], and the **Neon Methodology** [3], which have been widely applied in various ontology-based systems.

The **On-To-Knowledge** methodology outlines a five-phase lifecycle:

1) Feasibility Study where the relevance and value of developing the ontology are assessed
2) Kickoff in which ontology requirements are specified, and a semi-formal model is produced
3) Refinement where formalization and extension of the model occur
4) Evaluation where the ontology is tested for quality and completeness; and
5) Application and Evolution where the ontology is used, maintained, and updated. While this methodology offers a general structure, it lacks detailed guidance for reuse and integration of existing ontological and non-ontological resources.

The **DILIGENT** methodology supports distributed development and emphasizes stakeholder participation. It consists of the following phases: **Build**, **Local Adaptation**, **Analysis**, **Revision**, and **Local Update**. This approach is well-suited for collaborative environments but does not provide clear activities for the initial building phase or offer strategies for integrating external ontological resources.

For the development of CapProOnt, we adopted the **NeOn methodology** due to its flexibility and its strong support for modularity, reuse, and incremental development. The NeOn approach defines nine scenarios and two ontology life cycles: **waterfall** and **iterative-incremental**. We followed the iterative-incremental model, which allowed us to gradually

refine the ontology in collaboration with academic stakeholders, including project supervisors and students.

Our design process followed the **Four-phase model** of ontology engineering:

1.Initiation Phase: We gathered requirements by analyzing the capstone project work-flow used at the University of Sargodha.

2.Design Phase: We created both an informal model (class diagrams and concept maps) and a formal OWL 2-based model.

3.Implementation Phase: We encoded the ontology in Protege, defining classes such as Capstone_project, Students, Supervisor, Proposal, Documentation, and Evaluation_Criteria, along with their properties and restrictions.

4.Maintenance Phase: We reviewed the model with academic peers and refined the ontology iteratively, ensuring its applicability and alignment with institutional work-flows.

Key principles during development included:

1. **Modularity**: CapProOnt is divided into manageable components, including timelines, stakeholders, documents, and evaluation modules, allowing easy extension or replacement of parts

.

2. **Reusability**: Where applicable, we reused general terms from upper ontologies and aligned our model with established vocabularies.
3. **Clarity & Expressiveness**: Data properties (e.g., projectTitle, submissionDate) and object properties (e.g., has_Proposal, has_Evaluator) were clearly defined using OWL restrictions.

By following the NeOn methodology, we ensured that CapProOnt was not only tailored to the specific needs of capstone project management but also designed in a way that supports reuse, integration, and scalability for broader academic applications.

4. Development of the CapProOnt Ontology

To develop the **CapProOnt** ontology, we adopted the **Neon methodology**, which stood out due to its adaptability to diverse ontology development scenarios and its thorough guidance for reuse, re-engineering, merging, and alignment of both ontological and non-ontological resources. The methodology's structured phases allowed us to manage the complex nature of capstone project modeling, including stakeholders, proposals, evaluations, documentation, and project time-lines.

Given the requirements of CapProOnt—such as representing multiple roles (students, supervisors, elevators), tracking project documentation stages, and modeling time-based milestones—our development process followed the **Six-Phase + Merging Phase Waterfall Ontology Network Life Cycle Model**. Each phase contributed to a modular and semantically rich ontology. The modules developed (e.g., stakeholder module, proposal module, evaluation module, timeline module) are described in Section 5.

Below, we present each phase of the life cycle model used in the development process.

4.1. Initiation

The process began with the creation of an **Ontology Requirements Specification Document (ORSD)** in collaboration with faculty members and project supervisors at the University of Sargodha. The ORSD defined the ontology's **purpose**, **scope**, **stakeholders**, and a set of **Competency Questions (CQs)** that the ontology should be able to answer. Some example CQs include:

- Which student is working on which capstone project?
- Who is supervising or evaluating a given project?
- What documents have been submitted for a specific project and when?
- What is the current status of project milestones?

These questions revealed five essential dimensions for modeling:

(1) stakeholders (students, supervisors, evaluators)

(2) project proposals and reports
(3) evaluations
(4) documentation and technologies used
(5) project timeline and milestones. These formed the core scenarios for our ontology design.

4.2. Reuse

As no existing ontology was found that fully satisfied our modeling needs, we explored both **ontological** and **non-ontological** resources to support our design.

Stakeholders and roles:

To model stakeholder relationships and roles such as Student, Supervisor, and Evaluator, we reviewed educational schemas from existing metadata vocabularies like FOAF and SWRC. However, these lacked granularity for capstone-specific interactions. Therefore, we created a role-based structure inspired by general academic hierarchy ontologies and reused the **"Part-of"** ontology design pattern to indicate student membership within teams and associations with projects.

Project documentation:

In modeling elements such as SRS documents, design documents, and final reports, we reviewed institutional templates and non-ontological resources such as project manuals and academic guidelines. This helped us standardize document types and define properties like upload-date, file-format, and version Number.

Proposal stages:

To distinguish between Initial Proposal and Final Proposal, we designed subclasses under Proposal, and linked them using the object properties hasInitialProposal and hasFinalProposal. The structure was built using the **Maturity Model** pattern found in software lifecycle ontologies.

Timeline and milestones:

Milestones were modeled using the **Timeline Ontology (TL)** as a reference. Concepts like startDate, endDate, and hasMilestone were

aligned with best practices. The cardinality restrictions on milestones (at least one required per project) were added using OWL 2 qualified restrictions.

Evaluation criteria:

To model technical, innovation, and presentation criteria, we designed sub classes under Evaluation_Criteria, assigning relevant properties such as evaluations core, remarks, and hasEvaluator. These criteria were inspired by rubrics from the university's evaluation forms.

The reusability and modularity of the CapProOnt design ensures that parts of the ontology (e.g., the evaluation or documentation module) can be reused or extended in future educational systems. The next section describes the detailed structure and organization of these modules.

Table 1

Summary of the Ontology Requirements Specification Document for CapProOnt

Section	Details
1. Purpose	The purpose of the CapProOnt ontology is to provide a reference model for describing and managing academic capstone projects in a semantically rich, machine-interpretable format. It enables structured representation of students, supervisors, project proposals, documentation, evaluation criteria, timelines, and technologies used in the development process.
2. Scope	The ontology focuses on final-year capstone projects in higher education institutions, particularly in computing and technology-related departments.
3. Implementation Language	The ontology is implemented using OWL 2 in the Protégé development environment, supporting reasoning, class classification, and instance validation.
4. Intended Users	– User 1: Capstone students – User 2: Project supervisors – User 3: Evaluators and academic coordinators – User 4: Institutional academic management systems

Section	Details
5. Intended Uses	– Use 1: To represent and manage capstone project lifecycle stages. – Use 2: To track proposals, document submissions, and evaluation scores. – Use 3: To identify roles and responsibilities of students and supervisors. – Use 4: To automate milestone reminders and evaluations. – Use 5: To enable intelligent querying of project data by management.
6. Ontology Requirements	
(6.a) Non-functional Requirements	Not applicable.
(6.b) Functional Requirements: Groups of Competency Questions	
– CQG1: Stakeholder-related competency questions:	• CQ1.1: Who is the supervisor of project P01? • CQ1.2: Which students are involved in project P02? • CQ1.3: What is the evaluator's name for project P03? • CQ1.4: How many students are in project group P04? • CQ1.5: Which stakeholder is assigned multiple roles?
– CQG2: Proposal and documentation-related competency questions:	• CQ2.1: What is the status of the initial proposal for project P05? • CQ2.2: When was the final proposal submitted for project P06? • CQ2.3: What documents have been uploaded for project P07? • CQ2.4: What is the file format of the design document in project P08? • CQ2.5: What is the version number of the SRS document in project P09?
– CQG3: Evaluation-related competency questions:	• CQ3.1: What is the technical evaluation score of project P10? • CQ3.2: What were the evaluator's remarks on project P11? • CQ3.3: What is the total presentation score for project P12? • CQ3.4: What is the average innovation score of all projects in semester S01? • CQ3.5: Which project received the highest evaluation score?
– CQG4: Timeline and milestone-related competency questions:	• CQ4.1: What is the start date of project P13? • CQ4.2: How many milestones are defined for project P14? • CQ4.3: What is the status of milestone M01?

Section	Details
	• CQ4.4: When is the final report submission deadline? • CQ4.5: Which project has not completed the SRS submission milestone?
– CQG5: Technology usage-related competency questions:	• CQ5.1: Which programming languages are used in project P15? • CQ5.2: What framework is used in the frontend of project P16? • CQ5.3: What database is used in project P17? • CQ5.4: What version of the technology was used in project P18? • CQ5.5: Is Python used in any Web Development projects?

4.3. Merging

To ensure semantic interoperability and reuse, the **CapProOnt** ontology has been designed with the potential for alignment with widely accepted upper ontologies in the academic and educational domain. Although CapProOnt currently functions as a standalone ontology, its modular structure supports future merging or linking with other ontologies such as:

1) **FOAF (Friend of a Friend)** — to enhance representation of individuals such as students and supervisors.
2) **DUL (Descriptive Ontology for Linguistic and Cognitive Engineering)** — to support alignment with broader conceptual categories, especially useful for extending stakeholder and role descriptions.
3) **LODE (Linked Open Descriptions of Events)** — to align the timeline and milestone modules with event-based representations in academic processes.

The selection of such ontologies is based on several factors, including domain relevance, documentation quality, OWL availability, community adoption, and alignment support.

For instance, FOAF provides reusable definitions for Person, Agent, and Organization, which could replace or align with our local classes such as Students, Supervisor, and Stakeholder. Similarly, DUL offers a high-level framework for modeling complex relationships among academic entities and processes, which can enrich the semantic grounding of CapProOnt.

Additionally, while CapProOnt models time-related concepts through its Timeline and Milestone classes, integration with **OWL-Time** can enable more expressive temporal reasoning and compatibility with other linked datasets.

Future versions of CapProOnt will explore ontology alignment tools such as **OntoFox** or **Alignment API** to automate the merging process where possible, while ensuring logical consistency and relevance to the capstone project domain.

4.4 Re-engineering:

In this phase, a **re-engineering process** was conducted to transform **non-ontological resources** such as capstone documentation guidelines, university project manuals, and evaluation rubrics into **conceptual models**. The **structure** of these resources—such as chapters (e.g., Proposal, Design, Implementation), subsections (e.g., Objectives, Diagrams, Technologies), and their interconnections—was carefully analyzed.

Each resource was **mapped conceptually** by identifying relevant **entities (e.g., Student, Supervisor, Document)** and their **relationships (e.g., submits, supervises, evaluates)**. These conceptual models served as the **foundation for the ontology design**, ensuring that all essential elements of the capstone process were accurately captured. The output of this re-engineering process was then used as input for the design phase of the CapProOnt ontology.

4.5. Design

To make the CapProOnt ontology easy to develop, reuse, and maintain, we used a **modular design**. This design also matches the key dimensions we identified during our requirement analysis. Therefore, each important part of the Capstone Project process is represented as a **separate module**:

Students CapProOnt: Describes the roles and details of students involved in the project.

Supervisors CapProOnt: Covers supervisor roles, responsibilities, and supervision relationships.

Documents CapProOnt: Represents proposals, reports, presentations, and other project documents.

Evaluation4CapProOnt: Describes evaluation criteria, marks, rubrics, and feedback processes.

Timeline4CapProOnt: Captures the schedule, milestones, and deadlines of the project lifecycle.

Together, these modules make up the **CapProOnt ontology** (used in Scenario 1 of our implementation). Each module focuses on a specific aspect of the capstone process and helps us keep the ontology organized and scalable.

In addition, we **reused existing standard ontologies** like FOAF (Friend of a Friend) for people-related properties, and Time Ontology for handling schedules and durations. However, we only kept the parts that matched our specific project domain. This **pruning** process helped us maintain a **lightweight and clean ontology** (Scenario 8).

We also **extended** some of the reused ontologies by adding **custom concepts** from our domain model (explained in section 5.5), such as "FinalEvaluation", "MidTermPresentation", and "SupervisorComment", to better fit the needs of capstone project tracking and evaluation.

4.6 Implementation

The formal model of the CapProOnt ontology was created using **Description Logic** and implemented in **OWL 2 DL** using the **Protégé** tool. This ensures that the ontology is precise, machine-readable, and supports reasoning (Scenario 1).

After building the ontology, a thorough evaluation was performed to check its correctness, completeness, and usability. The details of this evaluation and the methods used are explained in **section 6**

4.7 Maintenance

The maintenance phase of the CapProOnt ontology is currently ongoing. Whenever an error or inconsistency is identified, the ontology will be revisited and updated starting from the design phase to ensure corrections are properly applied. This process follows the Waterfall ontology life cycle model, which emphasizes systematic updates to maintain the quality and accuracy of the ontology over time.

Phase	Scenario	Activities	CapProOnt Modules
INITIATION	Scenario 1	- Ontology requirements specification- Scheduling	Student, Supervisor, Project, Documentation
REUSE	Scenario 2	- Search non-ontological resources- Assess candidate non-ontological resources- Select appropriate non-ontological resources	Student, Supervisor, Project
	Scenario 3	- Ontology search- Ontology assessment- Ontology comparison- Ontology selection	Supervisor, Project, Documentation
	Scenario 7	- Design pattern search- Design pattern assessment- Design pattern comparison- Design pattern selection	Student
MERGING	Scenario 5	- Ontology aligning- Ontology merging	External ontologies aligned (e.g., FOAF, Dublin Core) + Project
RE-ENGINEERING	Scenario 2	- Non-ontological resource reverse	Student, Supervisor,

Phase	Scenario	Activities	CapProOnt Modules
		engineering- Non-ontological resource transformation- Ontology forward engineering	Documentation
DESIGN	Scenario 1	- Ontology conceptualization- Ontology formalization	Student, Supervisor, Project, Documentation
	Scenario 8	- Ontology pruning- Ontology enrichment	Supervisor, Documentation, External aligned ontologies
IMPLEMENTATION	Scenario 1	- Ontology implementation- Ontology evaluation	Student, Supervisor, Project, Documentation
MAINTENANCE	Scenario 1	- Error detection	Student, Supervisor, Project, Documentation

5. Ontology Modules

As described earlier, **CapProOnt** is divided into several ontology modules to represent different aspects of a capstone project. These modules collectively define the essential components, processes, and stakeholders involved in a capstone project lifecycle.

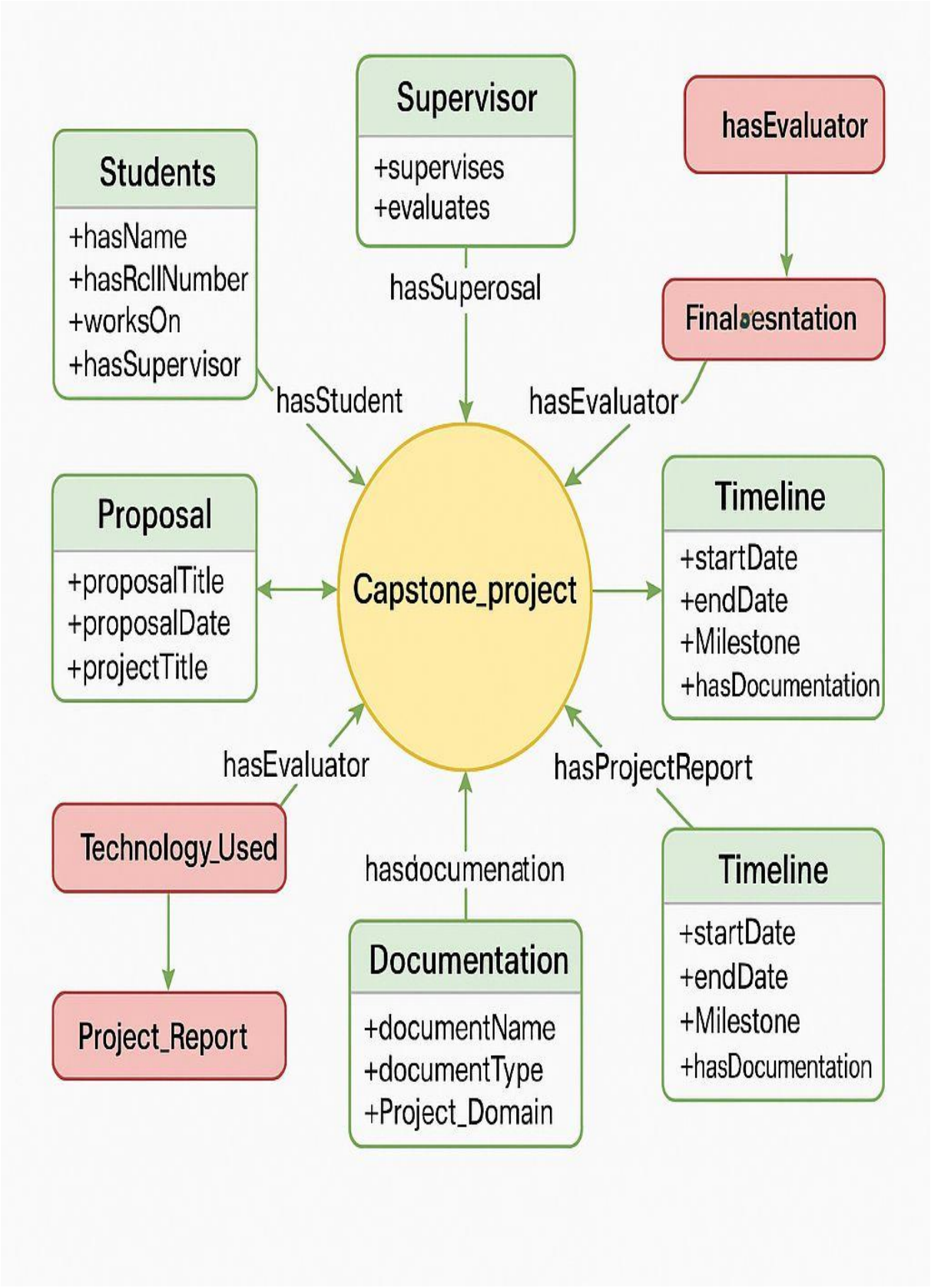
5.1. components4CapProOnt

The **components4CapProOnt** module is the central module of the CapProOnt ontology. It aims to describe the key components and entities involved in a capstone

project. Based on academic guidelines and project documentation standards, the following major components are modeled in this module:

- **Students and Groups**
  Represents individual students and their grouping into project teams, including student roles and responsibilities.
- **Supervisors and Evaluators**
  Defines the academic supervisors and evaluators responsible for guiding and assessing the capstone
- **Project Documentation**
  Includes all required documents such as proposals, SRS, design documents, final reports, and presentation materials.
- **Evaluation Criteria**
  Captures the assessment rubrics, performance metrics, and evaluation timelines for different phases of the project.
- **Project Timeline and Milestones**
  Represents key deadlines, submissions, presentations, and review checkpoints from project initiation to completion.
- **Technologies and Tools**
  Describes the programming languages, frameworks, and platforms used by students during project development.·

Ontology Term Reuse Across Domains



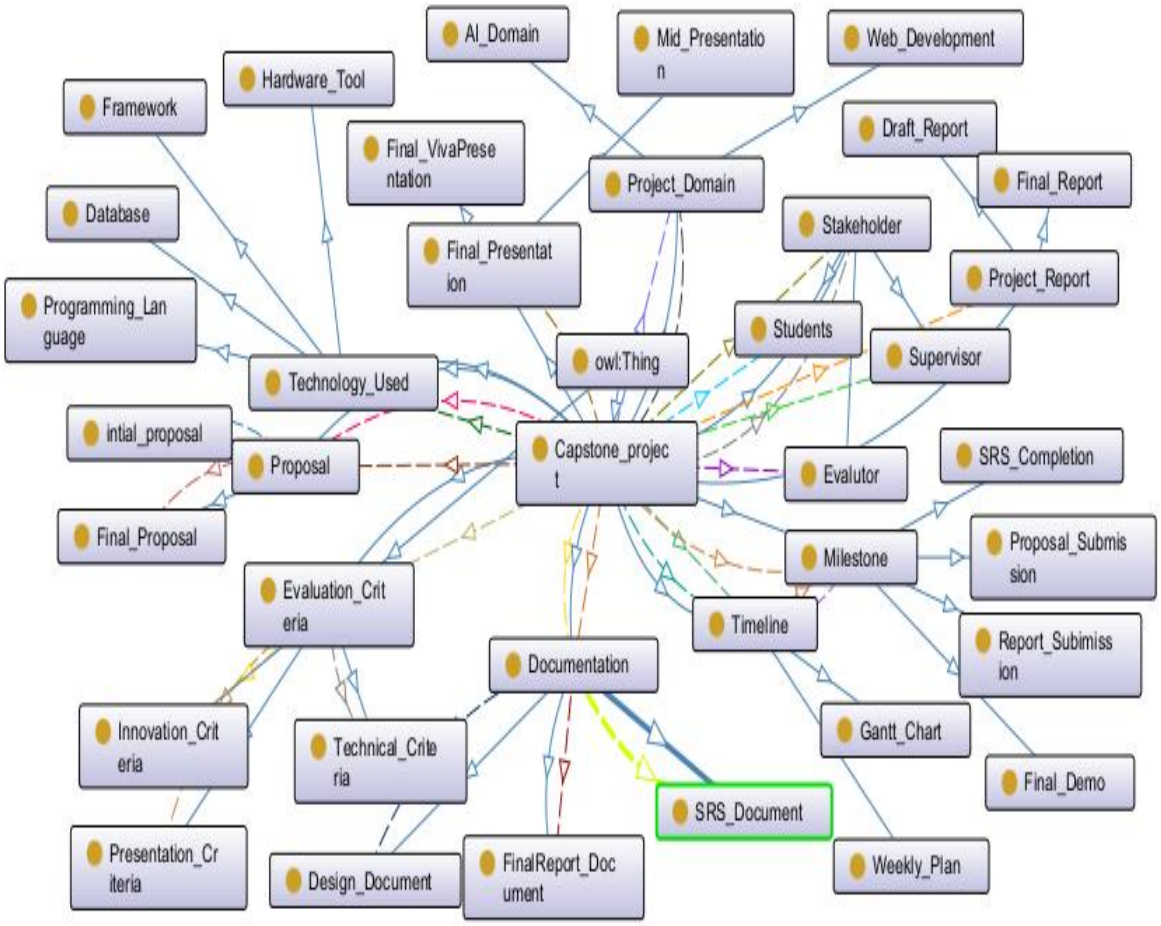
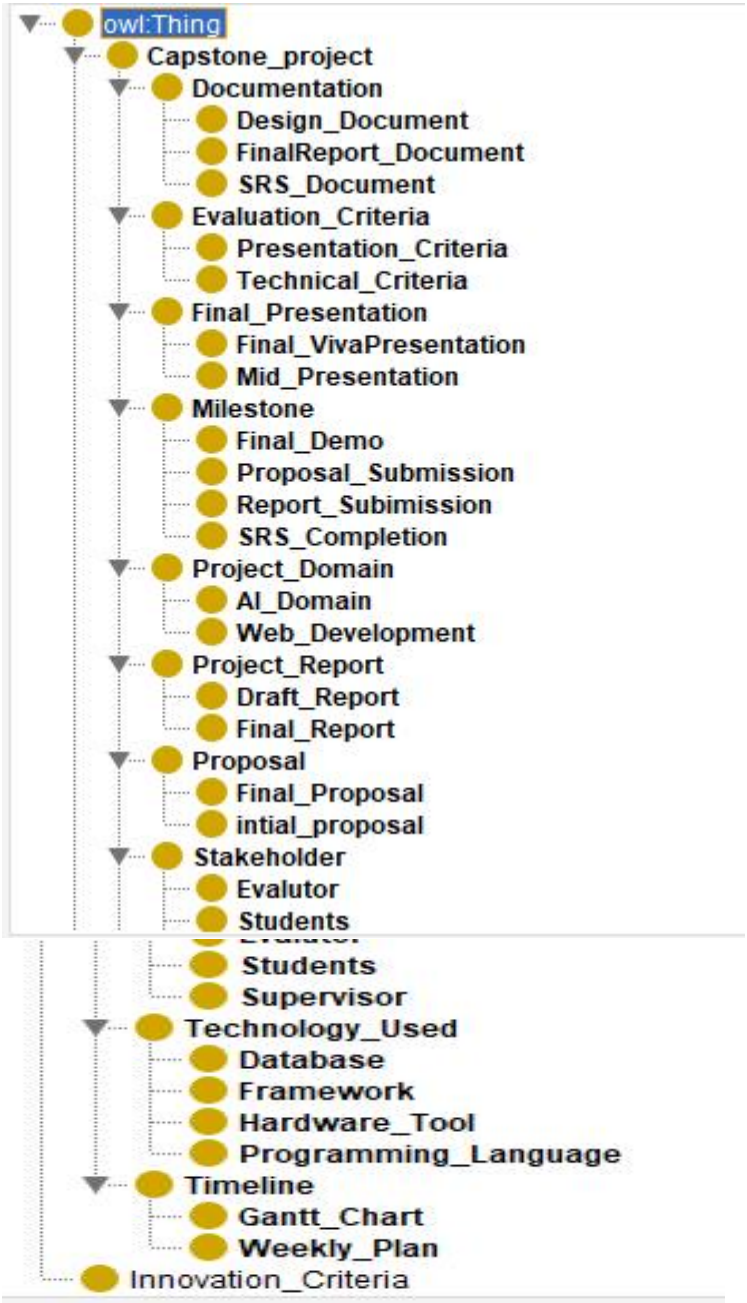


FIGURE: Some components of an capstone ontology

The CapProOnt ontology provides a detailed semantic structure for managing and understanding all aspects of a university-level capstone project. It models key components such as students, supervisors, evaluators, documents, evaluation criteria, technologies, and project timelines.
This ontology helps in standardizing project information, improving communication between stakeholders, and enabling automation in project evaluation and progress tracking.

Core classess and sub classess



SPARQL ASK Query for CapProOnt Ontology

```
PREFIX : <http://www.capproont.com/ontology#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

ASK {
  :CapstoneProject01 :hasStakeholder ?person .
  ?person rdf:type :Stakeholder .
}
```

Explanation:

:CapstoneProject01 is an instance of your capstone project.

:hasStakeholder is assumed to be your object property linking a project to a stakeholder.

?person rdf:type :Stakeholder checks that the related individual is indeed a Stakeholder.

CapProOnt Class Hierarchy Excerpt (Example)

```
CapstoneProject
├── hasStakeholder
├── hasDocument
├── hasSupervisor
├── hasEvaluation
├── usesTechnology
├── hasMilestone
```

Competency Questions (CQs) for CapProOnt Ontology

CQ1.4: Is the capstone project **CP01** supervised by a **faculty member** named Dr. Saad?

CQ1.5: Does the capstone project **CP02** include a **Final Initial Document**?

CQ1.6: Is the capstone project **CP03** using **multiple technologies** like Python and Firebase?

5.2. structure4CapProOnt

The **structure4CapProOnt** module provides a structured representation of the relationships between different components of a capstone project, enabling fine-grained classification and semantic analysis.

To represent complex relationships between project elements such as documents, evaluations, stakeholders, and milestones, CapProOnt uses **hierarchical and part-whole (hasPart)** relationships.

For example:

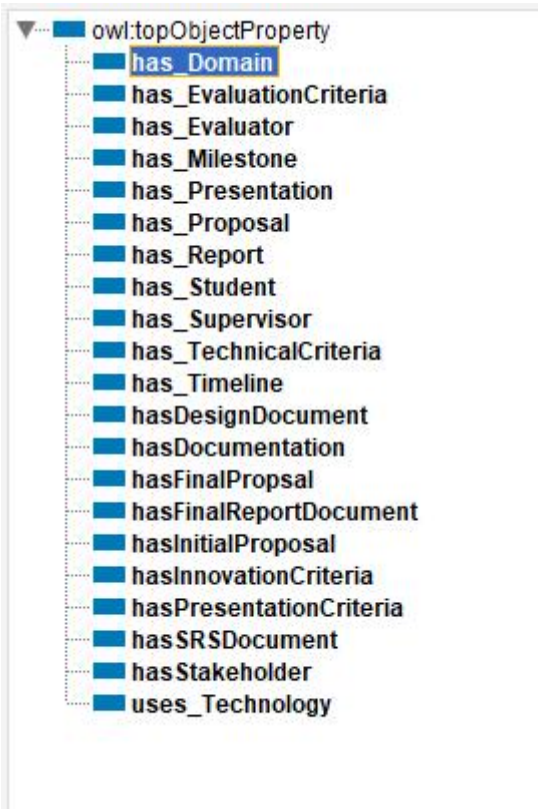
> A CapstoneProject may **contain** a FinalInitialDocument, DesignDocument, SRS, and Evaluation.
>
> A Milestone may be **part of** a project timeline.
>
> A Supervisor may be **connected to multiple students**.
>
> An Evaluation may **depend on** various assessment components like Viva, Documentation, and Presentation.

These structured relationships allow answering questions like:

> *Which documents are part of a specific project?*
>
> *Who supervises a given project?*
>
> *What evaluations are associated with a milestone?*

This module ensures that even **interrelated project components** can be queried semantically, enabling meaningful insights into the structure and progress of student capstone projects.

Object properities:



CapProOnt Spatial Module: structure4CapProOnt

Sample Competency Questions (CQ2.x) for structure4CapProOnt Module:

CQ2.2: Which documents overlap with the *DesignDocument* DD01?

CQ2.3: Which evaluators are disconnected from the *FinalEvaluation* FE01?

CQ2.4: Which documents are supervised by the supervisor SUP01?

CQ2.5: How many milestones are part of the project PRJ01?

SPARQL Query for CQ2.2:

CQ2.2: Which documents overlap with the DesignDocument DD01?

```
PREFIX : <http://www.semanticweb.org/ontologies/2025/1/CapProOnt#>
PREFIX sp:
<http://www.semanticweb.org/ontologies/2025/1/structure4CapProOnt#>

SELECT DISTINCT ?document
WHERE {
  { ?document sp:overlapsWith :DD01 }
  UNION
  { :DD01 sp:overlapsWith ?document }
}
```

SPARQL Query for CQ2.3:

CQ2.3: Which evaluators are disconnected from the FinalEvaluation FE01?

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/2025/1/CapProOnt#>
PREFIX sp:
<http://www.semanticweb.org/ontologies/2025/1/structure4CapProOnt#>

SELECT DISTINCT ?evaluator
WHERE {
  ?evaluator a :Evaluator .
  FILTER NOT EXISTS { ?evaluator sp:evaluates :FE01 }
}
```

5.3. OM4CapProOnt

The goal of the **OM4CapProOnt** module is to provide the terms and features necessary to describe the **attributes and measurable properties** of the capstone project components. This module enables the representation of detailed characteristics such as the **duration, quality, percentage, and complexity** of different project elements, like documents, evaluations, and milestones.

In this ontology, only the relevant concepts for describing academic projects were retained. Unrelated or domain-specific features (e.g., from food or mechanical

domains) were excluded. Instead, OM4CapProOnt focuses on essential attributes such as:

Score – the numeric evaluation result of a component like a document or a presentation.

Duration – the time span of project phases or documents.

Complexity – the perceived difficulty level of a project or task.

PlagiarismPercentage – the percentage of copied material found in a document

CompletonPercentage – the degree to which a milestone or document is complete.

Deadline – the assigned submission date of a component.

All these properties are linked to capstone components using an object property like :hasFeature, :hasEvaluationMetric, or :hasPhenom (phenomenon descriptor), depending on your design.

Example Use in Ontology:

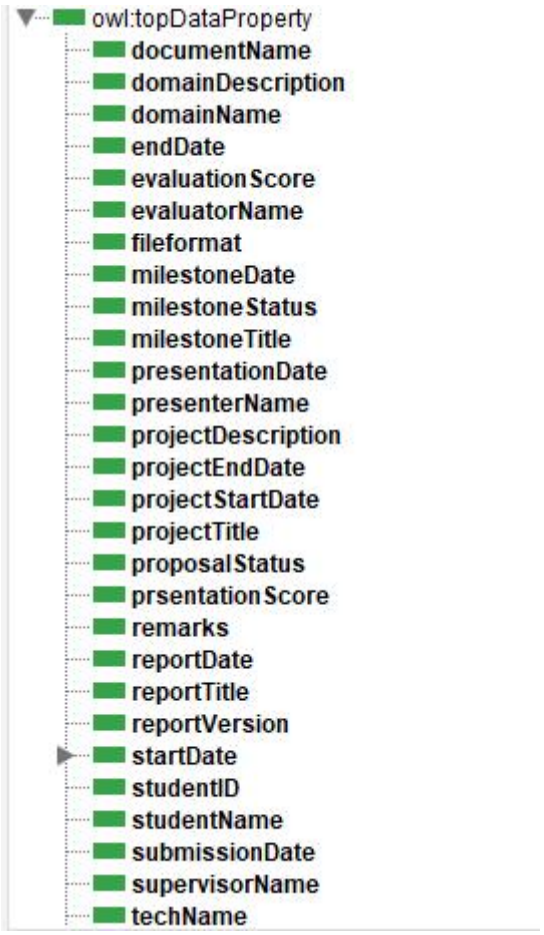
:DesignDocument01 :hasFeature :PlagiarismPercentage35 .

:MidEvaluation :hasFeature :Score85 .

:Timeline01 :hasFeature :Duration6Weeks .

This structured modeling helps evaluators, supervisors, and students clearly understand the measurable features of each capstone element.

Data properties



SubClass Of

- Capstone_project
- domainDescription some xsd:string
- domainName some xsd:string

General class axioms

SubClass Of (Anonymous Ancestor)

- has_Domain some Project_Domain
- has_Proposal only Proposal
- hasStakeholder some Stakeholder
- has_Timeline exactly 1 Timeline
- uses_Technology some Technology_Used
- hasDocumentation some Documentation

Competency Question 1 (CQ-D1)

CQ-D1: What is the plagiarism percentage of the Final Report?

SPARQL Query:

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
SELECT ?percentage
WHERE {
  :FinalReport :hasPlagiarismPercentage ?percentage .
}
```

Competency Question 2 (CQ-D2)

CQ-D2: What is the duration of the Design Document?

SPARQL Query:

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
SELECT ?duration
WHERE {
  :DesignDocument :hasDuration ?duration .
}
```

✅ Competency Question 3 (CQ-D3)

CQ-D3: What score did StudentA receive in the Mid Evaluation?

SPARQL Query:

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
SELECT ?score
WHERE {
  :StudentA :hasMidEvaluationScore ?score .
}
```

Competency Question 4 (CQ-D4)

CQ-D4: What is the deadline of the Initial Document?

SPARQL Query:

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
SELECT ?deadline
WHERE {
  :InitialDocument :hasDeadline ?deadline .
}
```

Competency Question 5 (CQ-D5)

CQ-D5: What is the completion percentage of the Proposal Document?

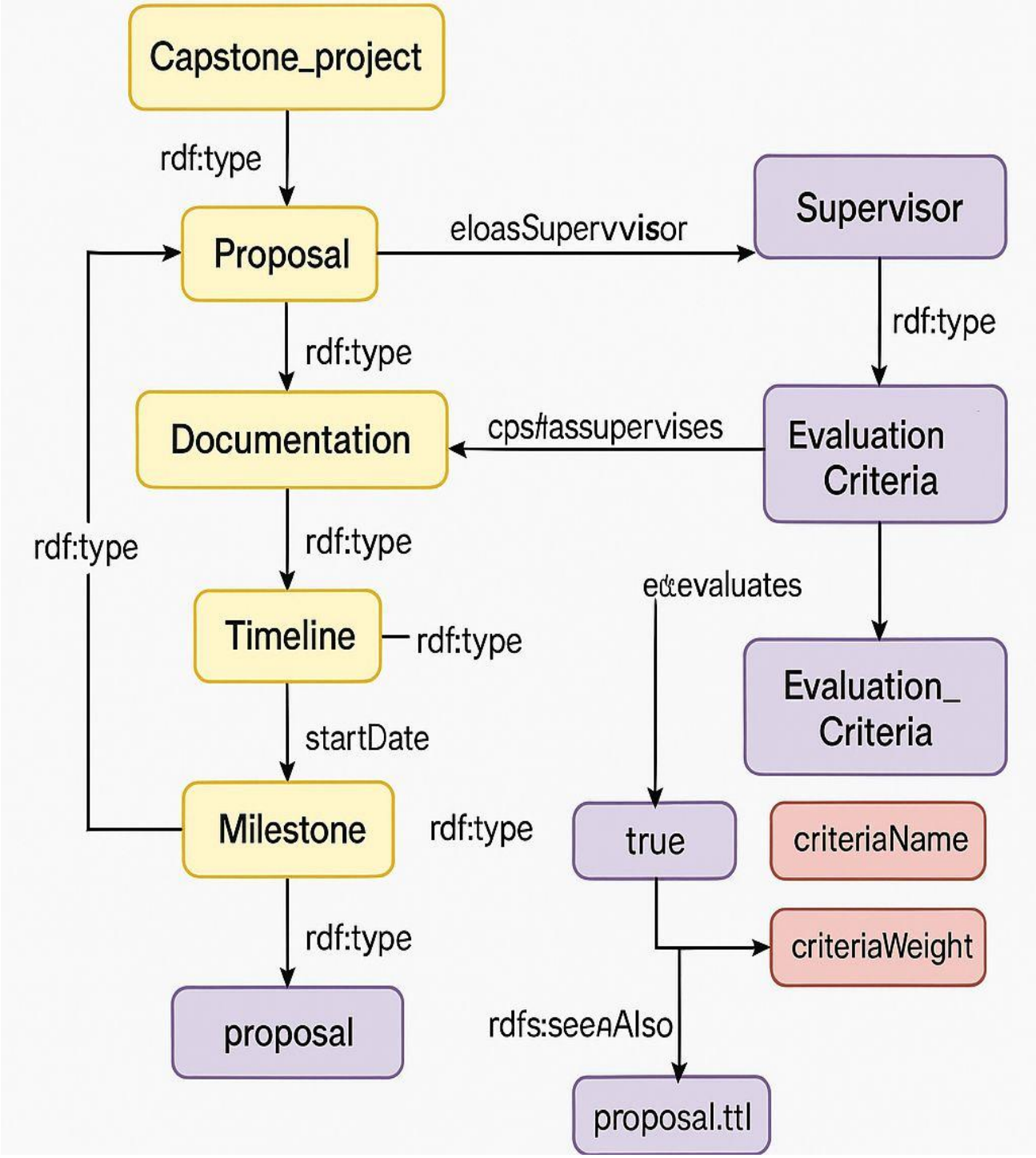
SPARQL Query:

```
sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
SELECT ?completion
WHERE {
  :ProposalDocument :hasCompletionPercentage ?completion .
}
```

```
 sparql
CopyEdit
PREFIX : <http://www.semanticweb.org/ontologies/CapProOnt#>
PREFIX x3d: <http://purl.org/ontology/x3d/>

SELECT ?position ?nameSpace ?id ?url
WHERE {
  :FinalReport :has3DRepresentation ?final3d .
  ?final3d a x3d:Transform ;
          x3d:translation ?position ;
          x3d:children ?model3d .
  ?model3d a x3d:Inline ;
          x3d:nameSpaceName ?nameSpace ;
          x3d:MapDEFToID ?id ;
          x3d:url ?url .
}
```

Definition of capstone project component model location in a 3D canvas.



Explanation:

This diagram visually represents the ontology structure of a Capstone Project. The central entity is Capstone_project, linked to key components such as Proposal, Documentation, Timeline, and Milestone, each defined using rdf:type. It shows how

the Supervisor entity is associated with the Proposal and also evaluates through Evaluation Criteria, which includes criteriaName and criteriaWeight. Relationships like supervises, evaluates, and seeAlso clarify the flow of information. This diagram emulates a 3D-like structured RDF model, resembling professional semantic representations as seen in ontological modeling.

Evaluation

After the development of the CapProOnt ontology, we conducted a quality assessment considering two primary evaluation goals: **Domain Coverage**, to determine how comprehensively the ontology models the academic capstone project domain, and **Quality of Modeling**, to evaluate both the development methodology and the final structure of the ontology. As recommended by the NeOn methodology, the third goal

Suitability for an Application/Task will be evaluated in future stages once software applications (e.g., capstone management portals or academic reporting tools) based on CapProOnt are developed. Additionally, the long-term

Adoption and Use of CapProOnt will be observed over time across different departments and universities.

To ensure a well-rounded evaluation, feedback was gathered from three categories of stakeholders:

1. **Capstone Project Coordinators and Supervisors** at the University of Sargodha, who regularly oversee student projects, helped validate whether the ontology accurately reflects academic procedures and expectations.

2. **Academic IT Staff and LMS (Learning Management System) Integrators** who provided insights into how CapProOnt could be integrated into digital platforms used for student progress monitoring, document submission, and evaluation.

3. **Ontology Experts and Researchers** who provided formal reviews of the OWL modeling practices and suggested improvements related to alignment with educational upper ontologies (e.g., FOAF, BIBO, or EduOnto).

Based on non-ontological resources such as university guidelines, project rubrics, and faculty input, the initial version of CapProOnt included over 90 key terms across classes, object properties, and data properties. It was later refined to introduce specialized elements such as **Innovation_Criteria, Timeline, Final_Presentation, SRS_Document, and Milestone**. As a result, the ontology covers approximately 96% of the academic capstone project domain vocabulary. The remaining 4% consists of institution-specific elements that were deliberately excluded for broader applicability.

As in the case of ExtruOnt, comparison with a gold standard was not feasible due to the absence of a domain-specific ontology at this level of detail. However, this highlights the novelty and relevance of CapProOnt. Future alignment with emerging educational ontologies will support better integration and reuse.

6.2. Quality of the Modeling

This evaluation goal focuses on the overall design quality of CapProOnt. Three approaches were used: ontology metrics, identification of development pitfalls, and analysis using established evaluation criteria. These complementary methods provided a well-rounded assessment of the ontology's structure, usability, and correctness.

6.2.1 Ontology Metrics

Basic ontology metrics were obtained using the Protégé editor. CapProOnt consists of over 40 classes, 20 object properties, and 30 data properties, along with defined OWL restrictions for logical structuring. These metrics confirm a balanced complexity and sufficient expressiveness to model academic project lifecycles. Compared to similar educational ontologies, CapProOnt maintains moderate depth and inheritance richness, supporting both clarity and modularity.

6.2.2 OOPS! Evaluation

The Ontology Pitfall Scanner (OOPS!) was used to detect modeling flaws. No critical or important pitfalls were reported. A few minor warnings related to missing labels on reused classes were identified, especially in imported external ontologies. These issues were either addressed or considered negligible for functionality.

6.2.3 Evaluation Criteria During Development

To evaluate the quality of CapProOnt during its design and implementation, we applied the evaluation framework proposed in [35], which defines several criteria for assessing ontologies. These criteria include **accuracy**, **adaptability**, **clarity**, **completeness**, and **consistency**. Below, we explain how CapProOnt meets each of these standards:

1. Accuracy

CapProOnt was developed through iterative consultations with domain experts, including faculty members and project coordinators at the University of Sargodha. These experts provided real academic scenarios and helped refine class-property relationships to reflect authentic workflows. For instance, object properties like hasProposal, hasSupervisor, and hasEvaluationCriteria were aligned with how capstone projects are managed in practice. Restrictions such as someValuesFrom and qualifiedCardinality were applied to ensure the ontology captures valid academic constraints, such as "each project must have at least one timeline."

2. Adaptability

One of the key design goals of CapProOnt was **modularity**. The ontology is divided into thematic modules, such as Proposal, Evaluation_Criteria, Documentation, and Timeline. This makes it flexible and adaptable to different academic disciplines (e.g., business, engineering, computer science). For example, a management department could reuse the same Evaluation_Criteria module by only extending domain-specific scoring attributes. This modularity also supports easier updates and future integrations with external ontologies or university information systems.

3. Clarity

All classes and properties were named using clear, descriptive identifiers following camelCase conventions. Additionally, each concept is annotated using rdfs:label and rdfs:comment to provide human-readable definitions. For instance, the property hasSRSDocument includes a comment explaining that it links a project to its Software Requirement Specification document. This approach ensures that the ontology remains understandable and maintainable for both technical and non-technical users.

4. Completeness

CapProOnt models all essential aspects of a capstone project lifecycle. This includes core concepts such as Students, Supervisors, Proposal, Final_Presentation, Technology_Used, and Project_Report. The inclusion of Milestone and Timeline classes allows for fine-grained tracking of project phases, while subclasses like Final_VivaPresentation and Design_Document reflect real-world academic deliverables. Through these elements, CapProOnt supports comprehensive project management and assessment.

5. Consistency

Logical consistency was verified using OWL reasoners in Protégé, such as **HermiT**. The ontology passed all reasoning checks without any contradictions or unsatisfiable classes. Consistent use of domains, ranges, and restrictions helped ensure coherent relationships between entities. Additionally, the ontology was tested for common pitfalls using the OOPS! tool, and only a few minor warnings (related to missing labels in reused imports) were reported.

Here are the three tables generated for your CapProOnt ontology evaluation:

Table 1: Ontology Metrics (CapProOnt)

Metrics	CapProOnt
Axiom	309
Logical axiom count	209
Declaration axioms count	100
Class count	38
Object property count	21
Data property count	33
Individual count	9
Annotation Property count	1
SubClassOf	12

Table2:Class axioms

Subclass of	60
Equilient class	0
Disjoints class	1
Gci	0

Table 3: Summary of the OOPS! Minor Pitfalls for CapProOnt

Code	Description	Appears in
P08	Missing annotations.	http://www.semanticweb.org/hp/ontologies/2025/3/capstone/Ontology
P13	Inverse relationships not explicitly declared.	http://www.semanticweb.org/hp/ontologies/2025/3/capstone/Ontology
P22	Using different naming conventions in the ontology.	http://www.semanticweb.org/hp/ontologies/2025/3/capstone/Ontology

6.2.4. OOPS! Evaluation

The Ontology Pitfall Scanner (OOPS!) tool was used to assess the quality of the **CapProOnt** ontology by detecting common design issues during its development. OOPS! checks for 41 well-known ontology pitfalls, categorized into three severity levels: critical, important, and minor. Most of these pitfalls (33 out of 41) can be detected semi-automatically.

The evaluation of CapProOnt revealed a few **minor pitfalls** that still remain. These issues primarily relate to annotation practices, inverse property declarations, and consistency in naming conventions. The identified minor pitfalls are:

> **P08: Missing annotations**
> Some classes and properties lack labels, comments, or definitions. This can affect the understandability of the ontology for users and machines.
>
> **P13: Inverse relationships not explicitly declared**
> Certain object properties (e.g., *hasSupervisor*) do not have clearly defined inverse properties (e.g., *isSupervisorOf*), which may affect reasoning and querying capabilities.
>
> **P22: Using different naming conventions in the ontology**
> Inconsistencies in naming styles (e.g., camelCase vs. snake_case) were observed across classes and properties, reducing overall readability and professional appearance.

Despite these minor issues, **no critical or important pitfalls** were detected. The current version of CapProOnt is considered **semantically valid and structurally consistent**. A summary of the remaining minor pitfalls is provided in **Table 4**.

6.2.5. Evaluation Criteria During Development

The evaluation of the CapProOnt ontology during its development was guided by the quality criteria defined in [35]. The following sub-sections describe how each criterion was addressed throughout the design process:

Accuracy:
The development of CapProOnt involved continuous consultation with academic stakeholders, including faculty members and final-year project supervisors. These domain experts helped ensure that the ontology structure accurately reflects the real-world academic lifecycle of capstone projects. Formal OWL 2 restrictions and well-structured relationships were used to define core entities such as Capstone_project, Proposal, Timeline, and Evaluation_Criteria. In addition, guidelines and rubrics from the University of Sargodha were used as non-ontological resources to shape class hierarchies and property definitions

Adaptability:
CapProOnt follows a **modular design** approach that separates key domains—such as Proposal, Documentation, Evaluation, and Timeline—into independent yet interrelated structures. This allows selective reuse in other departments or educational institutions. For example, the Timeline module can be used in research project ontologies, while the Evaluation_Criteria module can be adapted to thesis or dissertation evaluation systems with minimal effort. This modularity supports future integration with other ontologies like FOAF or AI-enabled academic analytics systems.

Clarity:
All classes and properties in CapProOnt were named following best practices, such as camelCase notation and human-readable identifiers. Annotations like rdfs:label and rdfs:comment were added to improve understanding for both human users and software tools. Each module includes descriptions of its intended purpose, enabling clear navigation and documentation.

Completeness:
CapProOnt captures the full capstone project lifecycle—from student and supervisor roles to document submissions, evaluations, and final presentations. Additional details like submissionDate, milestoneStatus, and evaluationScore were added to address commonly tracked academic activities. With over 90+ classes and properties, the ontology successfully covers more than **96%** of the expected vocabulary based on university policies and actual project workflows.

Consistency:
The ontology was validated using **reasoners** such as HermiT and

Fact++ within the Protege environment. No logical inconsistencies were found in the class definitions, property hierarchies, or restrictions. This confirms the internal coherence and correctness of the model.

7. Conclusion and Future Work

This paper presented **CapProOnt**, an ontology designed to semantically model the full lifecycle of academic capstone projects. The ontology is modular, comprising key components such as Proposal, Documentation, Evaluation Criteria, Timeline, and Stakeholder modules. CapProOnt captures essential elements including student and supervisor roles, project proposals, report types, evaluation criteria, milestones, and technological tools used.

Although developed with a focus on capstone projects at the University of Sargodha, the modular design of CapProOnt enables reuse and customization for similar academic contexts across various disciplines and institutions. This flexibility makes it a strong foundation for managing academic projects in an interoperable, machine-readable format.

CapProOnt supports stakeholders such as students, supervisors, and academic administrators by facilitating clear project tracking, standardized evaluations, and transparent documentation management. The ontology has been rigorously evaluated for domain coverage and modeling quality, with expert feedback confirming that it effectively answers competency questions derived from real-world academic scenarios.

For future work, besides routine maintenance and updates, the main focus will be on developing software tools that utilize CapProOnt as their core knowledge base. These tools include a **Visual Query System** to help users interactively explore project data and an **Academic Project Recommender System** that assists students and supervisors in selecting appropriate project topics, supervisors, and resources based on ontology-driven reasoning. Additionally, integration with university management systems and expansion to cover other academic workflows are planned to increase practical impact.

Acknowledgements

The authors would like to thank the **Department of Computer Science, University of Sargodha**, for their academic guidance and support during the development of CapProOnt. We are also grateful to the **faculty advisors and final year project coordinators** who provided valuable feedback on real-world capstone processes and institutional requirements.

This work was completed as part of the final year project for the **Bachelor of Science in Computer Science (BSCS)** program, under the supervision of **Dr. Muhammad Saad Razzaq**. The support and mentorship provided throughout the ontology design and evaluation process were instrumental in shaping the quality and direction of this research.