Modelling Assignment:

Ontology-Based Job Matching System

MSc AI, Module 4 Knowledge Representation and Reasoning

Pavlos Papachristos

Pp24589

Contents

Modelling Assignment:

Ontology-Based Job Matching System with Evaluation and Implementation Guidance

1. Introduction

This work attempts to document and evaluate a knowledge-based system developed using Protégé in order to match job seekers with suitable job opportunities.

The model developed applies knowledge representation and reasoning methods, based in OWL ontologies that align individual qualifications and preferences with employment requirements.

This analysis includes the business context, ontology modelling decisions, practical testing via DL and SPARQL queries, and recommended visualisation components such as OntoGraf diagrams and a decision tree. It concludes with implementation steps for final submission and highlights strengths and improvement areas.

2. Business Context and Justification of Approach

One known issue with the traditional keyword-based job platforms is that often they fail to capture the semantic context and in this case the candidate’s qualifications links to the job's requirements.

This project adopts an ontology-driven method to model the job recruitment domain using semantics. Using OWL ontologies enables logic-based reasoning, such as inferring that a job seeker with e.g. JavaSkill and a Master’s Degree may qualify for a Software Developer role. This enhances the precision and personalization of job matching services.

3. Ontology Design and Modelling

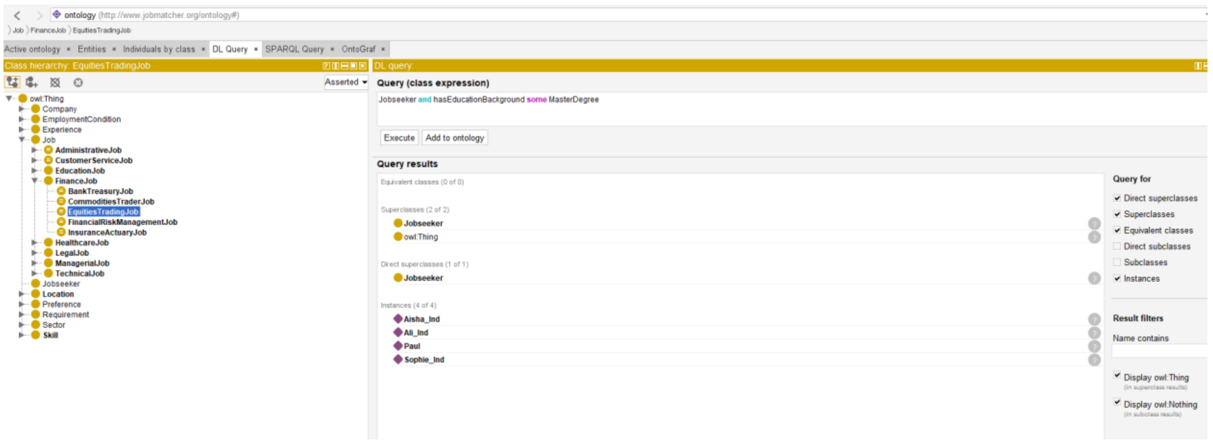
The core classes created include JobSeeker, Job, Skill, EducationExperience, WorkExperience, Condition, and Preference. Object properties such as hasSkill, hasEducationBackground, and requiresSkill define semantic relationships. The hierarchy is structured so that MasterDegree is a subclass of EducationExperience, and individuals are correctly typed to support reasoning.

3.1 Testing and Reasoning Outputs

The following DL and SPARQL queries used to test the logical consistency and the data integrity in order to identify job seekers with: a) certain education qualifications, b) computing skills, c) previous work experience, d) work requirements (job permit) and e) location preferences, f) suitable to the role:

a) Find job seekers with a Master's Degree:

Jobseeker and hasEducationBackground some MasterDegree



ontology (http://www.jobmatcher.org/ontology#)
Job 〉 FinanceJob 〉 EquitiesTradingJob
Active ontology
Entities
Individuals by class
DL Query
SPARQL Query
OntoGraf
Class hierarchy: EquitiesTradingJob
Asserted
DL query:
Execute
Add to ontology

```
owl:Thing
  Company
  EmploymentCondition
  Experience
  Job
    AdministrativeJob
    CustomerServiceJob
    EducationJob
    FinanceJob
      BankTreasuryJob
      CommoditiesTraderJob
      EquitiesTradingJob
      FinancialRiskManagementJob
      InsuranceActuaryJob
    HealthcareJob
    LegalJob
    ManagerialJob
    TechnicalJob
  Jobseeker
  Location
  Preference
  Requirement
  Sector
  Skill
```

```
Query (class expression)
Jobseeker and hasEducationBackground some MasterDegree

Query results
Equivalent classes (0 of 0)

Superclasses (2 of 2)
  Jobseeker
  owl:Thing

Direct superclasses (1 of 1)
  Jobseeker

Instances (4 of 4)
  Aisha_Ind
  Ali_Ind
  Paul
  Sophie_Ind
```

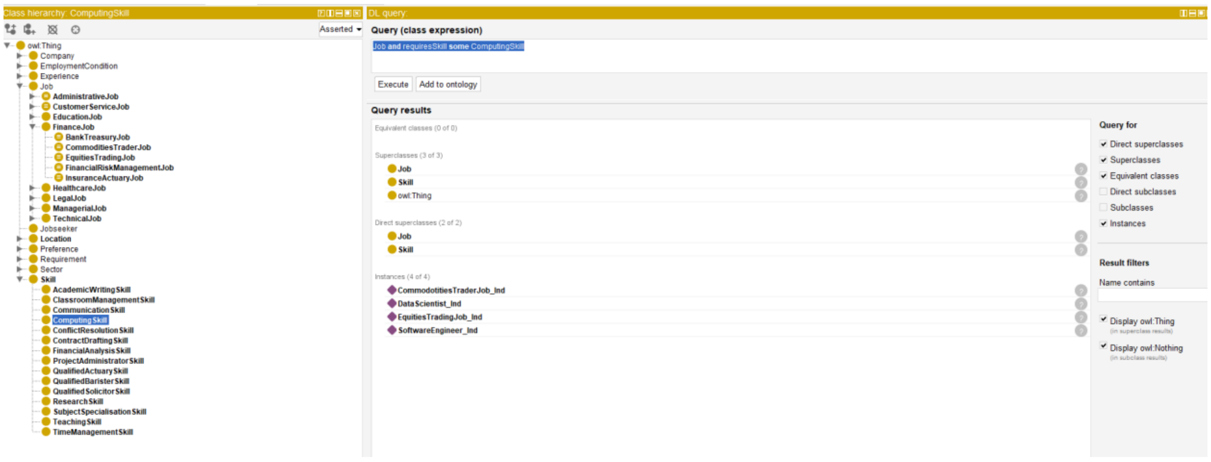
Query for
- ✓ Direct superclasses
- ✓ Superclasses
- ✓ Equivalent classes
- ☐ Direct subclasses
- ☐ Subclasses
- ✓ Instances

Result filters
Name contains

- ✓ Display owl:Thing (in superclass results)
- ✓ Display owl:Nothing (in subclass results)

b) Find jobs that require Computing skills:

Job and requiresSkill some ComputingSkill



Class hierarchy: ComputingSkill

Asserted

```
owl:Thing
  Company
  EmploymentCondition
  Experience
  Job
    AdministrativeJob
    CustomerServiceJob
    EducationJob
    FinanceJob
      BankTreasuryJob
      CommoditiesTraderJob
      EquitiesTradingJob
      FinancialRiskManagementJob
      InsuranceActuaryJob
    HealthcareJob
    LegalJob
    ManagerialJob
    TechnicalJob
  Jobseeker
  Location
  Preference
  Requirement
  Sector
  Skill
    AcademicWritingSkill
    ClassroomManagementSkill
    CommunicationSkill
    ComputingSkill
    ConflictResolutionSkill
    ContractDraftingSkill
    FinancialAnalysisSkill
    ProjectAdministratorSkill
    QualifiedActuarySkill
    QualifiedBaristerSkill
    QualifiedSolicitorSkill
    ResearchSkill
    SubjectSpecialisationSkill
    TeachingSkill
    TimeManagementSkill
```

DL query:

Query (class expression)

```
Job and requiresSkill some ComputingSkill
```

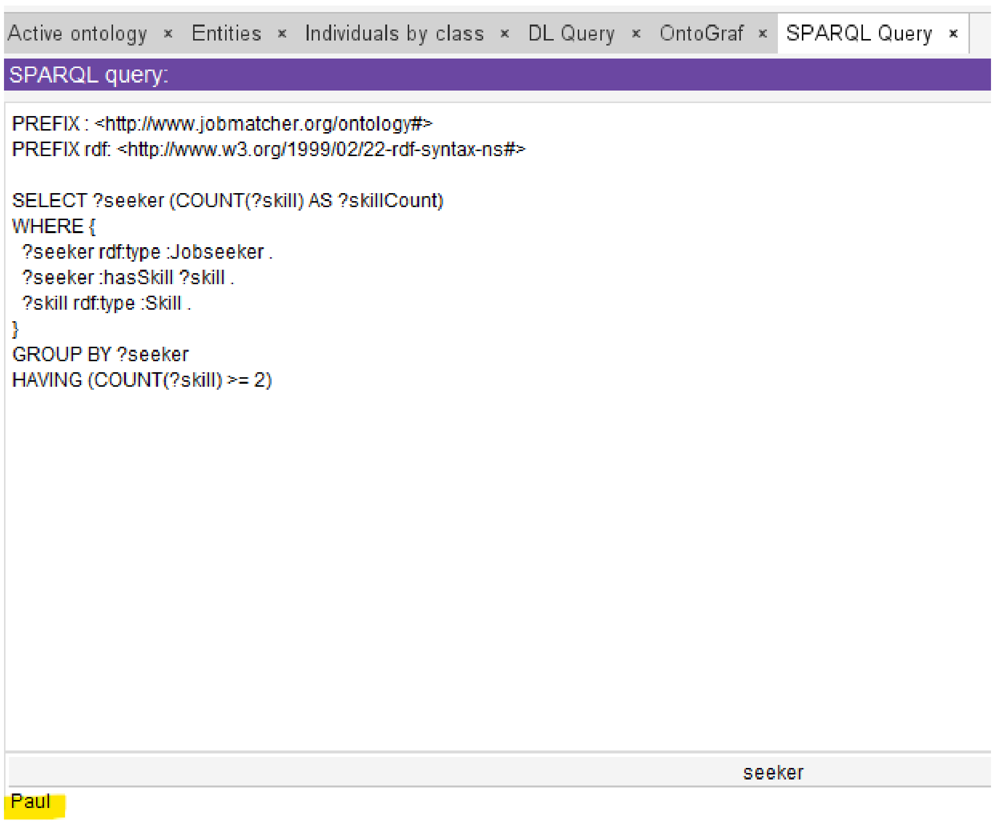
Query results

```
Equivalent classes (0 of 0)

Superclasses (3 of 3)
  Job
  Skill
  owl:Thing

Direct superclasses (2 of 2)
  Job
  Skill

Instances (4 of 4)
  CommodotitiesTraderJob_Ind
  DataScientist_Ind
  EquitiesTradingJob_Ind
  SoftwareEngineer_Ind
```

Query for: ✓ Direct superclasses, ✓ Superclasses, ✓ Equivalent classes, ☐ Direct subclasses, ☐ Subclasses, ✓ Instances

Result filters: Name contains; ✓ Display owl:Thing (in superclass results); ✓ Display owl:Nothing (in subclass results)

c) Find candidates with more than one skill using the SPARQL:



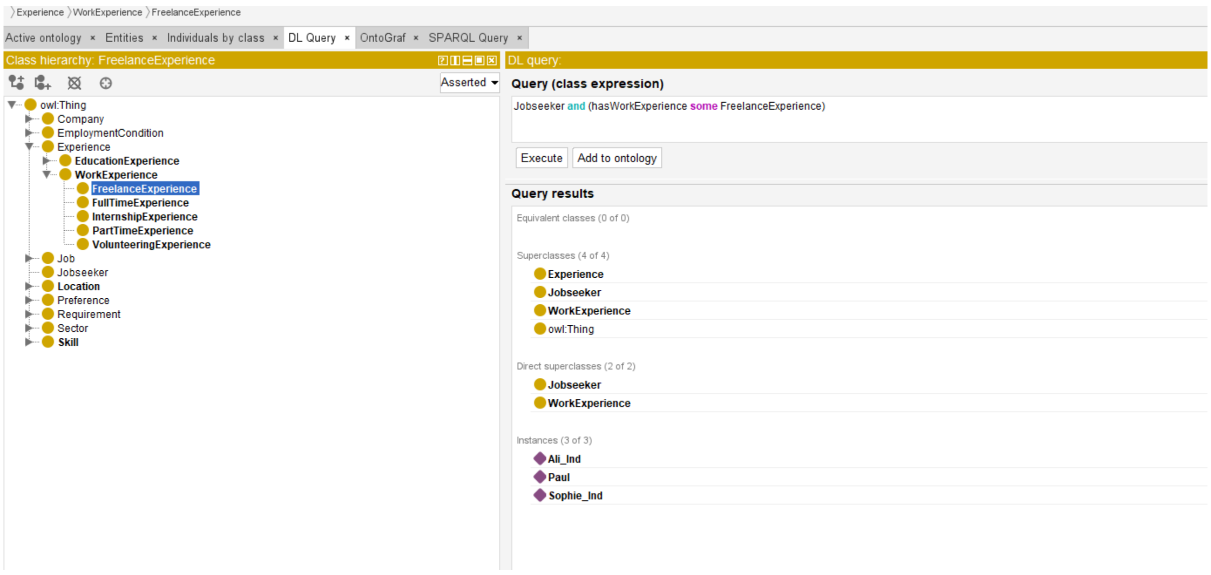
SPARQL query:

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

SELECT ?seeker (COUNT(?skill) AS ?skillCount)
WHERE {
  ?seeker rdf:type :Jobseeker .
  ?seeker :hasSkill ?skill .
  ?skill rdf:type :Skill .
}
GROUP BY ?seeker
HAVING (COUNT(?skill) >= 2)
```

```
                                        seeker
Paul
```

d) Candidates with Internship experience:

Jobseeker and hasWorkExperience some InternshipExperience



Experience 〉WorkExperience 〉FreelanceExperience
Active ontology × Entities × Individuals by class × DL Query × OntoGraf × SPARQL Query ×
Class hierarchy: FreelanceExperience
Asserted
DL query:
Execute
Add to ontology

```
owl:Thing
  Company
  EmploymentCondition
  Experience
    EducationExperience
    WorkExperience
      FreelanceExperience
      FullTimeExperience
      InternshipExperience
      PartTimeExperience
      VolunteeringExperience
  Job
  Jobseeker
  Location
  Preference
  Requirement
  Sector
  Skill
```

Query (class expression)

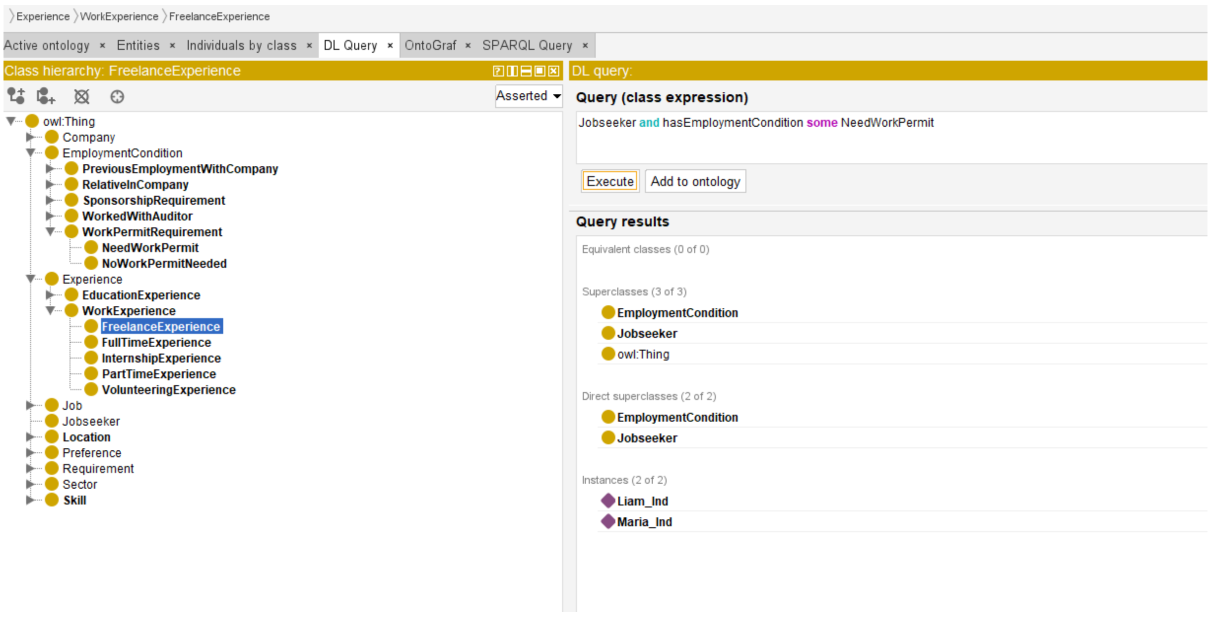
Jobseeker and (hasWorkExperience some FreelanceExperience)

Query results

```
Equivalent classes (0 of 0)

Superclasses (4 of 4)
  Experience
  Jobseeker
  WorkExperience
  owl:Thing

Direct superclasses (2 of 2)
  Jobseeker
  WorkExperience

Instances (3 of 3)
  Ali_Ind
  Paul
  Sophie_Ind
```

e) Candidates needing a work permit:

Jobseeker and hasEmploymentCondition some NeedWorkPermit



Experience 〉WorkExperience 〉FreelanceExperience
Active ontology × Entities × Individuals by class × DL Query × OntoGraf × SPARQL Query ×
Class hierarchy: FreelanceExperience
Asserted
DL query:
Execute
Add to ontology

```
owl:Thing
  Company
  EmploymentCondition
    PreviousEmploymentWithCompany
    RelativeInCompany
    SponsorshipRequirement
    WorkedWithAuditor
    WorkPermitRequirement
      NeedWorkPermit
      NoWorkPermitNeeded
  Experience
    EducationExperience
    WorkExperience
      FreelanceExperience
      FullTimeExperience
      InternshipExperience
      PartTimeExperience
      VolunteeringExperience
  Job
  Jobseeker
  Location
  Preference
  Requirement
  Sector
  Skill
```

Query (class expression)

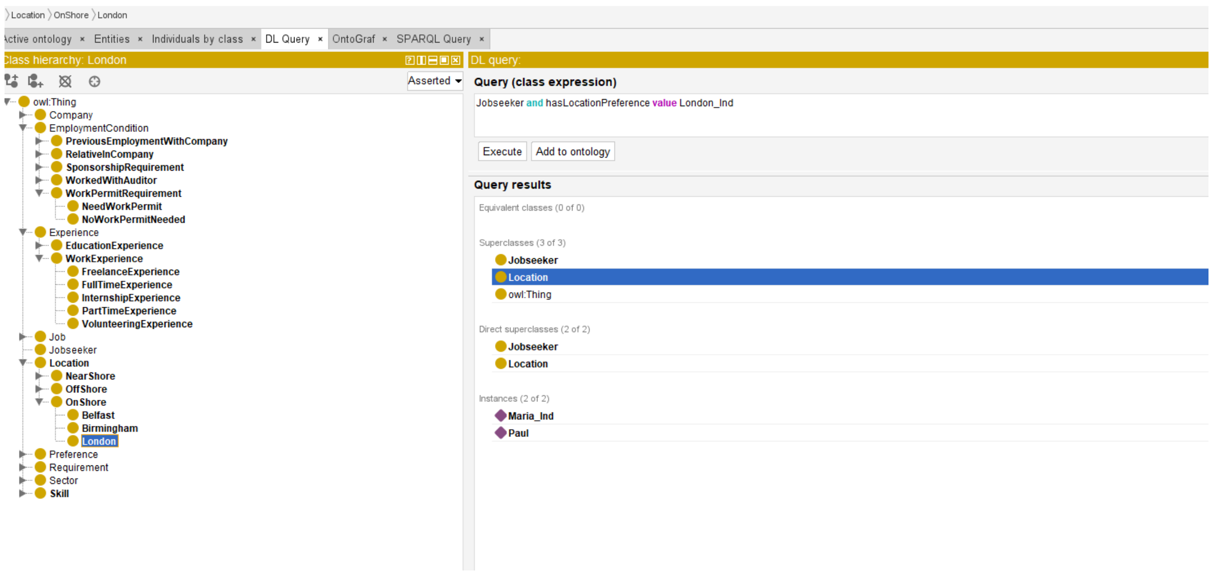
Jobseeker and hasEmploymentCondition some NeedWorkPermit

Query results

```
Equivalent classes (0 of 0)

Superclasses (3 of 3)
  EmploymentCondition
  Jobseeker
  owl:Thing

Direct superclasses (2 of 2)
  EmploymentCondition
  Jobseeker

Instances (2 of 2)
  Liam_Ind
  Maria_Ind
```

f) Candidates preferring London:

Jobseeker and hasLocationPreference value London_Ind


Location 〉 OnShore 〉 London
Active ontology × Entities × Individuals by class × DL Query × OntoGraf × SPARQL Query ×
Class hierarchy: London
Asserted
DL query:
Execute
Add to ontology

```
owl:Thing
    Company
    EmploymentCondition
        PreviousEmploymentWithCompany
        RelativeInCompany
        SponsorshipRequirement
        WorkedWithAuditor
        WorkPermitRequirement
            NeedWorkPermit
            NoWorkPermitNeeded
    Experience
        EducationExperience
        WorkExperience
            FreelanceExperience
            FullTimeExperience
            InternshipExperience
            PartTimeExperience
            VolunteeringExperience
    Job
    Jobseeker
    Location
        NearShore
        OffShore
        OnShore
            Belfast
            Birmingham
            London
    Preference
    Requirement
    Sector
    Skill
```

Query (class expression)

Jobseeker and hasLocationPreference value London_Ind

Query results

Equivalent classes (0 of 0)

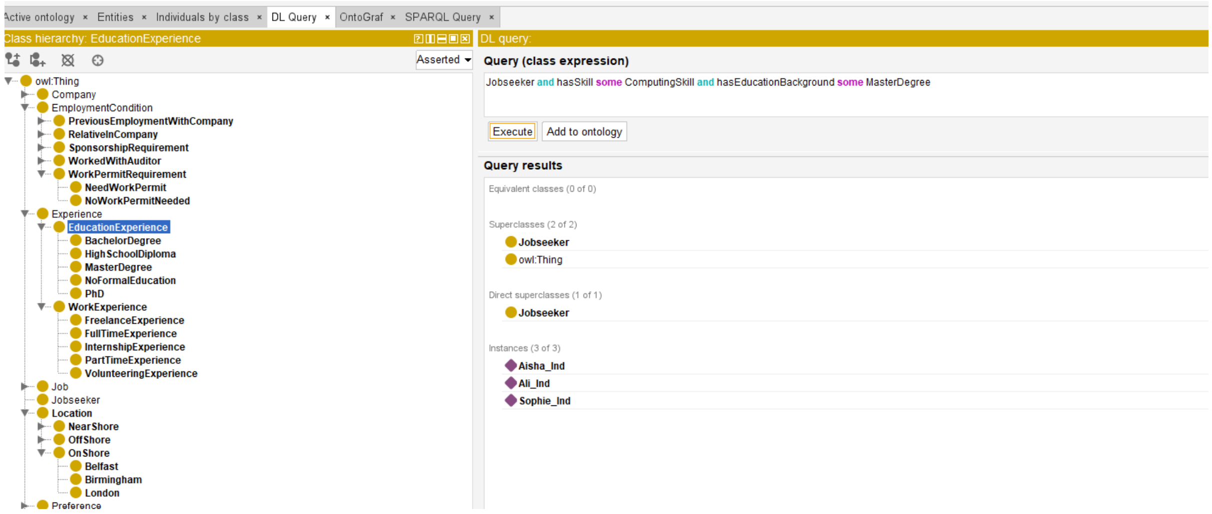
Superclasses (3 of 3)
- Jobseeker
- Location
- owl:Thing

Direct superclasses (2 of 2)
- Jobseeker
- Location

Instances (2 of 2)
- Maria_Ind
- Paul

g) Candidates eligible for a Software Engineer role:

Jobseeker and hasSkill some ComputingSkill and hasEducationBackground some MasterDegree


Active ontology × Entities × Individuals by class × DL Query × OntoGraf × SPARQL Query ×
Class hierarchy: EducationExperience
Asserted
DL query:
Execute
Add to ontology

```
owl:Thing
    Company
    EmploymentCondition
        PreviousEmploymentWithCompany
        RelativeInCompany
        SponsorshipRequirement
        WorkedWithAuditor
        WorkPermitRequirement
            NeedWorkPermit
            NoWorkPermitNeeded
    Experience
        EducationExperience
            BachelorDegree
            HighSchoolDiploma
            MasterDegree
            NoFormalEducation
            PhD
        WorkExperience
            FreelanceExperience
            FullTimeExperience
            InternshipExperience
            PartTimeExperience
            VolunteeringExperience
    Job
    Jobseeker
    Location
        NearShore
        OffShore
        OnShore
            Belfast
            Birmingham
            London
    Preference
```

Query (class expression)

Jobseeker and hasSkill some ComputingSkill and hasEducationBackground some MasterDegree

Query results

Equivalent classes (0 of 0)

Superclasses (2 of 2)
- Jobseeker
- owl:Thing

Direct superclasses (1 of 1)
- Jobseeker

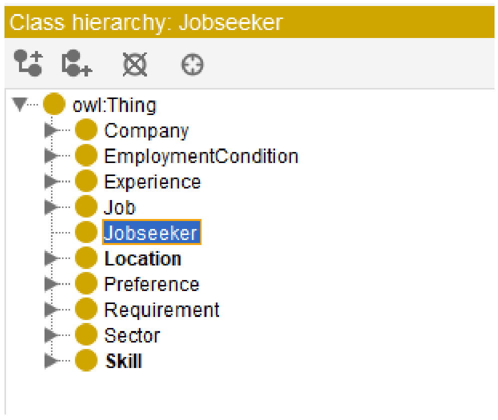
Instances (3 of 3)
- Aisha_Ind
- Ali_Ind
- Sophie_Ind

These DL queries validated the ontology integrity and support classification and filtering logic within the AI system.

3.2 Ontology Structures Visualisation

Class Hierarchy

The Class hierarchy is outlined at its high level in the following graph:

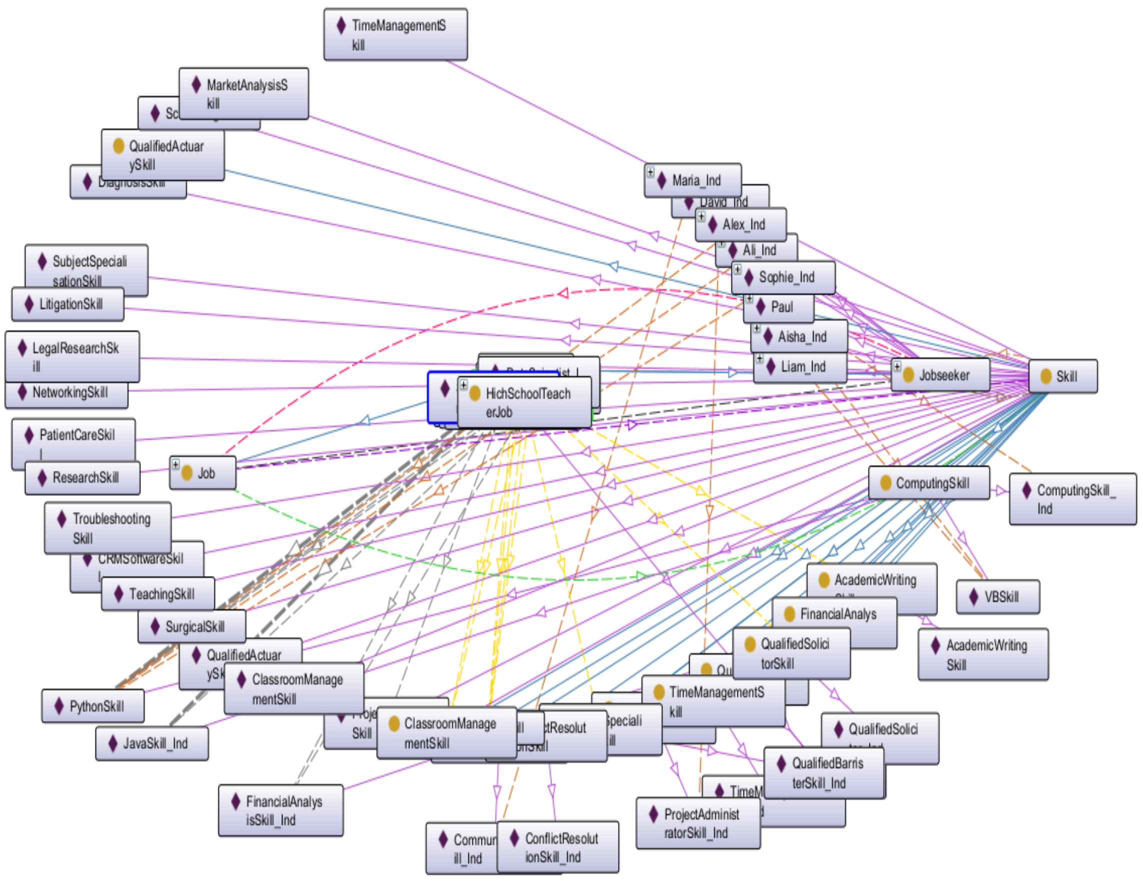


Class hierarchy: Jobseeker

- owl:Thing
  - Company
  - EmploymentCondition
  - Experience
  - Job
  - Jobseeker
  - **Location**
  - Preference
  - Requirement
  - Sector
  - **Skill**

Figure 1: Class hierarchy: Jobseeker

The below graphs is a sample of the developed Class hierarchies the full list of Classes can be retrieved by running the attached Protégé project file in **Appendix.**



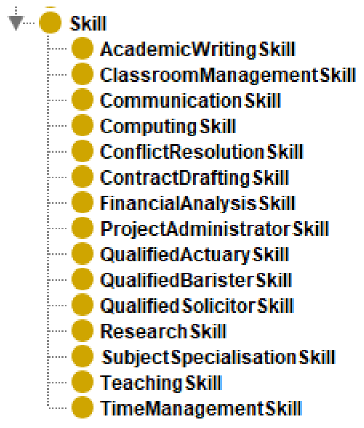
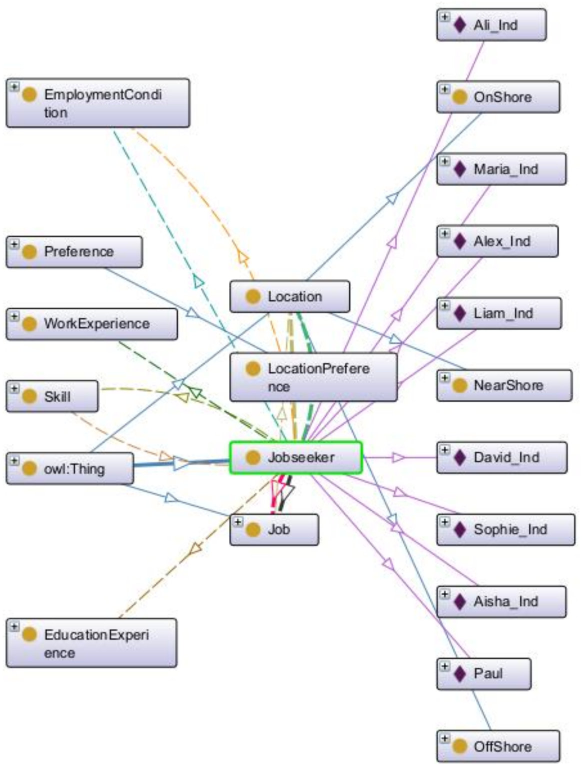

- Skill
  - AcademicWritingSkill
  - ClassroomManagementSkill
  - CommunicationSkill
  - ComputingSkill
  - ConflictResolutionSkill
  - ContractDraftingSkill
  - FinancialAnalysisSkill
  - ProjectAdministratorSkill
  - QualifiedActuarySkill
  - QualifiedBaristerSkill
  - QualifiedSolicitorSkill
  - ResearchSkill
  - SubjectSpecialisationSkill
  - TeachingSkill
  - TimeManagementSkill

Figure 2a, 2b: Class Skill Hierarchy Structure graphs



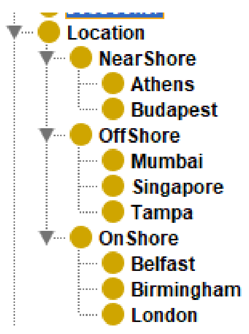
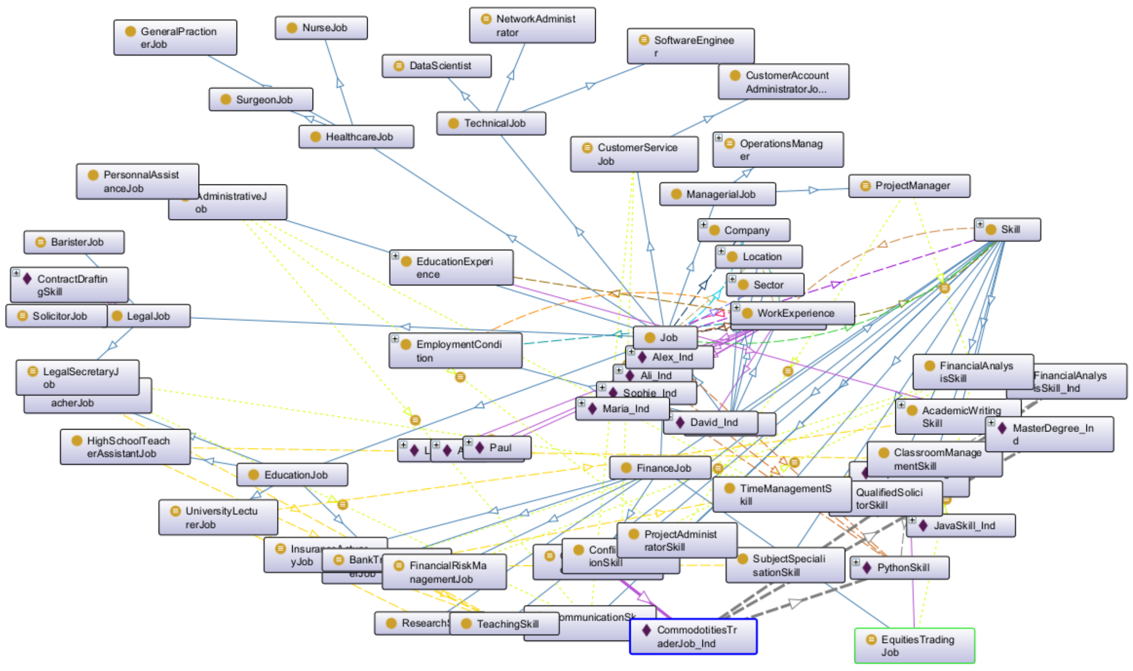
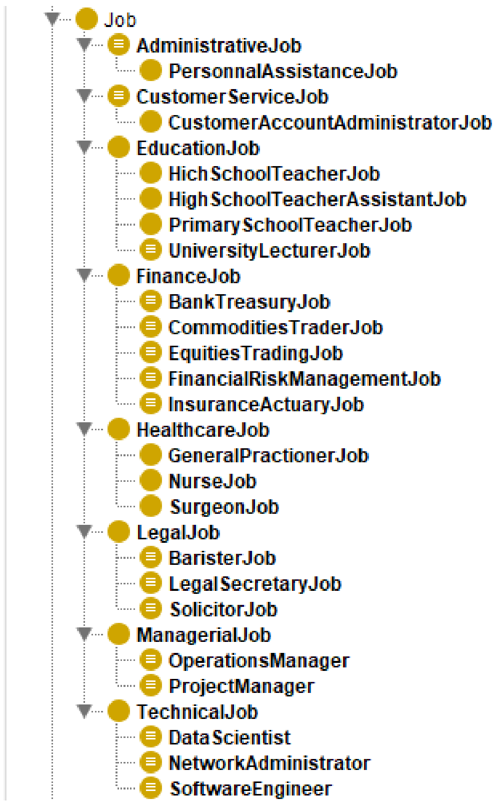


Figure 3a, 3b: Class Location Hierarchy Structure graphs





```
Job
    AdministrativeJob
        PersonnalAssistanceJob
    CustomerServiceJob
        CustomerAccountAdministratorJob
    EducationJob
        HichSchoolTeacherJob
        HighSchoolTeacherAssistantJob
        PrimarySchoolTeacherJob
        UniversityLecturerJob
    FinanceJob
        BankTreasuryJob
        CommoditiesTraderJob
        EquitiesTradingJob
        FinancialRiskManagementJob
        InsuranceActuaryJob
    HealthcareJob
        GeneralPractionerJob
        NurseJob
        SurgeonJob
    LegalJob
        BaristerJob
        LegalSecretaryJob
        SolicitorJob
    ManagerialJob
        OperationsManager
        ProjectManager
    TechnicalJob
        DataScientist
        NetworkAdministrator
        SoftwareEngineer
```

Figure 4a and 4b: Class Job Hierarchy Structure graphs

Individual Network

The attached OntoGraf visualization presents a detailed semantic network of the individual Paul, illustrating his relationships within the ontology. Paul is modelled as an instance of the Jobseeker class and is connected to various individuals via object property assertions.

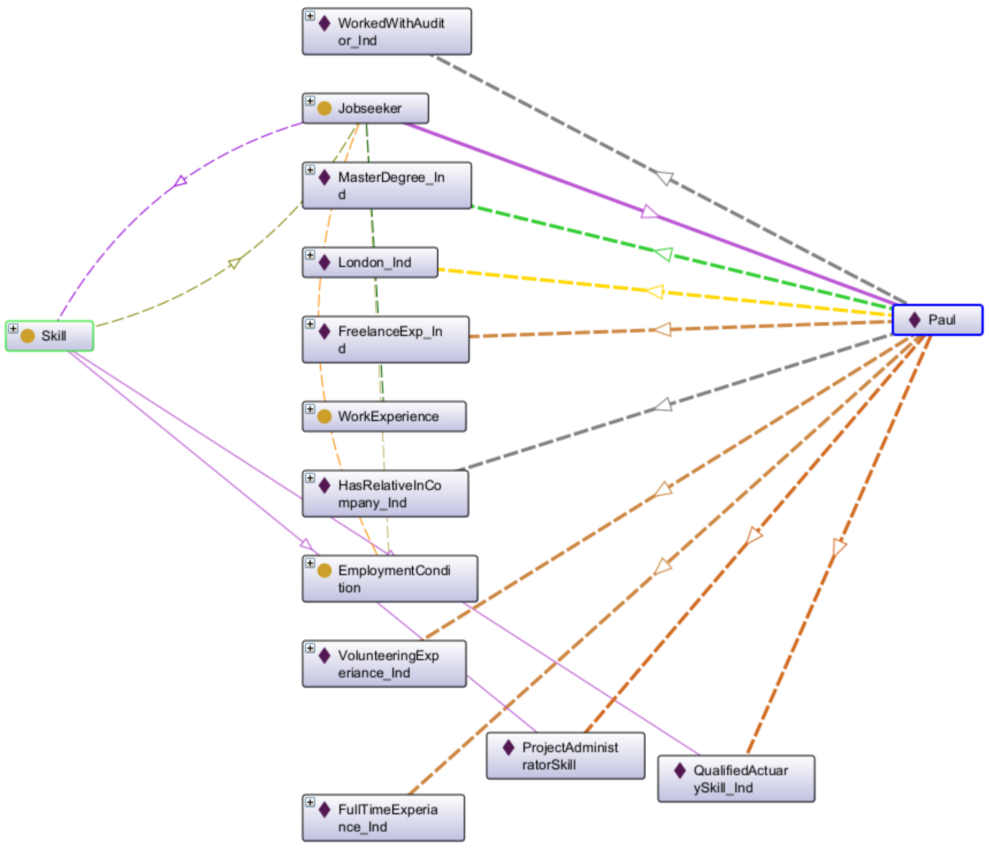


Figure 5: Individual Network – case study Paul

The Educational Background for Paul is connected to the individual MasterDegree_Ind via the property hasEducationBackground.

That indicates that he holds a Master's degree which is needed for roles where an advanced degree is expected.

Paul is also associated with multiple types of work experience through the hasWorkExperience property:

• FreelanceExp_Ind

• VolunteeringExperiance_Ind

• FullTimeExperiance_Ind

These links to Paul's professional profile provide evidence for reasoning tasks involving job suitability, eligibility, or experience diversity.

Through the hasSkill property Paul's proficiency in several areas can be detected and allow the ontology to support queries related to qualified jobseekers or those possessing domain-specific skills.

In terms of preference and conditions, Paul is linked to London_Ind via hasLocationPreference, indicating his preferred job location. Paul's individual network is illustrated via the HasRelativeInCompany_Ind (suggesting internal referral potential) and enables to position him as a well-qualified candidate within the knowledge base, supporting tasks like job matching and automated reasoning.

3.3 Decision Tree for Job Recommendation Logic

Below is a decision tree representing logical inference for a Personal Assistance role.

This decision tree models the logic required to determine whether a job seeker is eligible for a PersonalAssistanceJob.

The structure incorporates key semantic checks for education, skill, experience, and location, aligned with the object properties and classes defined in the ontology.

Step	Condition (Ontology-based Check)	Outcome / Next Step
1	Individual is typed as Jobseeker	Yes → Step 2 No → Not eligible
2	hasEducationBackground some BachelorDegree	Yes → Step 3 No → Not eligible
3	hasSkill some CommunicationSkill	Yes → Step 4 No → Not eligible
4	hasSkill some OrganisationSkill	Yes → Step 5 No → Not eligible
5	hasWorkExperience some FullTimeExperience	Yes → Step 6 No → Not eligible
6	hasLocationPreference London	Yes → Step 7 No → Not eligible
7	All required conditions met	Recommend: **PersonalAssistanceJob**

Figure 6: Decision Tree representing semantic rule logic for job matching Personal Assistance.

4. Evaluation and Improvements

During the development and validation of the reasoning structures within Protégé, several ontology modelling issues encountered which have initially reduced the effectiveness of the automated reasoning.

One of the challenges that have been met, was the incorrectly assigned types to individuals. This is a modelling error that violates the fundamental OWL 2 DL constraints.

In particular, the individual MasterDegree_Ind was erroneously assigned with multiple rdf:types across mutually disjoint classes, including Job, Skill, and EducationExperience. In OWL ontologies, such dis-jointed axioms were placed to apply the logic that individuals cannot simultaneously belong to conceptually incompatible classes (W3C, 2012). This incorrectly assigned multi-typing led the reasoner (HermiT) to infer that MasterDegree_Ind belonged to owl:Nothing. This is representing a logical contradiction that prevented any valid DL query results involving that entity.

This issue is emblematic of broader challenges in ontology engineering: while OWL provides expressive modelling capabilities, it also demands semantic rigour.

As Gruber (1993) and Noy and McGuinness (2001) emphasise, the utility of an ontology depends on its formal correctness and internal consistency.

In this case, the faulty axioms prevented the classification and the knowledge retrieval, as essential individuals like Paul or Sophie_Ind could no longer be inferred as valid instances in queries such as JobSeeker and hasEducationBackground some MasterDegree.

A systematic reassessment of individuals was conducted in order to address this issue. All the entities were reviewed for their semantic alignment with their intended class and all conflicting rdf:types were removed.
The MasterDegree_Ind was reclassified solely as an instance of EducationExperience, consistent with the class hierarchy and ontological scope. Corrections were also made to all other individuals with similar errors and they have only linked to conceptually appropriate classes.

Furthermore, assertions involving object properties like hasEducationBackground, hasSkill, and hasWorkExperience were verified for type compatibility, ensuring their ranges and domains corresponded to the correct subject-object class pairs.

With these adjustments applied, the integrity of the ontology's logic has been restored and enabled DL queries like:

JobSeeker and hasEducationBackground some MasterDegree

to return accurate and meaningful results.

The results have correctly identified Paul and other individuals with the required educational backgrounds.

The defined object property axioms were also required attention. Properties such as hasSkill, initially have lacked precise domain and range declarations. This had led to ambiguous reasoning outcomes where (for example) a Job could be incorrectly asserted to possess a Skill, ignoring the intended property use like requiresSkill.

These modelling inconsistencies have introduced a semantic 'drift' that has lead to reasoning failures and misleading inferences (Horridge et al., 2004).

To resolve this, all core object properties were explicitly defined: hasSkill now links JobSeeker to Skill, requiresSkill connects to the Job to Skill and the hasEducationBackground links JobSeeker to EducationExperience.

These modelling inconsistencies have introduced a semantic 'drift' that has lead to reasoning failures and misleading inferences (Horridge et al., 2004).

To resolve this, all core object properties were explicitly defined: hasSkill now links JobSeeker to Skill, requiresSkill connects to the Job to Skill and the hasEducationBackground links JobSeeker to EducationExperience.

These corrections have aligned the type assignments to the data and have been verified via DL and SPARQL queries that have produced meaningful inferences.

This refinement process has improved the structural robustness and inferential power of the ontology. This is achieved by eliminating contradictions and clarifying axiomatic constraints.

The system can now support the classification, the pattern matching and the use of queries answering across multiple dimensions (including candidate skills, education, work conditions and preferences). The system also facilitates more complex logical queries, such as filtering candidates by minimum skill count or location preferences, demonstrating OWL's capacity for expressive rule-based matchmaking (Motik et al., 2012). These improvements confirm that rigorous ontology engineering — grounded in clear domain understanding, consistent modelling, and logical validation — is crucial for the reliability, scalability, and semantic precision of AI-powered job matching systems.

5. References

- Antoniou, G. and van Harmelen, F., 2004. A Semantic Web Primer. Cambridge, MA: MIT Press.

- Horridge, M., 2011. A Practical Guide to Building OWL Ontologies Using Protégé 4 and CO-ODE Tools. University of Manchester.
- Horridge, M., Knublauch, H., Rector, A.L., Stevens, R. and Wroe, C., 2004. A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools.
- Gruber, T.R., 1993. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2), pp.199–220.
- Noy, N.F. and McGuinness, D.L., 2001. Ontology Development 101: A Guide to Creating Your First Ontology. [online] Stanford Knowledge Systems Laboratory.
- Motik, B., Patel-Schneider, P.F. and Parsia, B., 2012. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. [online] Available at: https://www.w3.org/TR/owl2-syntax/ [Accessed 12 July 2025].
- Staab, S. and Studer, R., 2009. Handbook on Ontologies. 2nd ed. Berlin: Springer.
- W3C, 2012. OWL 2 Web Ontology Language Document Overview. [online] Available at: https://www.w3.org/TR/owl2-overview/ [Accessed 12 July 2025].
- W3C, 2013. SPARQL 1.1 Query Language. [online] Available at: https://www.w3.org/TR/sparql11-query/ [Accessed 12 July 2025].

Appendix

The Protégé file with the ontology described above is attached:

Jobseeker_Protege.rdf