An OWL Ontology for Hackathons, Skills, Projects, and Teams

Eisha Tir Raazia
Semantic Web Technologies

Autumn 2025

Abstract

Hackathons bring together participants with complementary skills to work on time bounded projects, but information about these events is usually stored in ad hoc formats that lack explicit semantics. This project presents an OWL ontology for hackathons, teams, projects, skills, topics, and awards, implemented in Protégé and populated with a small example dataset. The ontology reuses FOAF and SKOS, includes OWL axioms and SWRL rules for reasoning about team composition, and is queried with SPARQL. The work illustrates how Semantic Web technologies can provide a structured, interoperable view on hackathon ecosystems.

Contents

1 Introduction

Hackathons are widely used in education, innovation, and civic tech to bring together participants with diverse skills to work on intensive projects over a short time period. Data about hackathons is often maintained in spreadsheets or event platforms without a shared schema, which makes it difficult to query across events, reason about teams, or analyze skill gaps.

The goal of this project is to design and implement an ontology that captures the core concepts of a hackathon ecosystem, while remaining simple and reasoner friendly. The ontology is intended to:

- represent hackathons, teams, projects, participants, skills, topics, and awards;
- reuse well established vocabularies such as FOAF and SKOS;
- support basic reasoning about team composition using OWL and SWRL;
- enable SPARQL queries that answer competency questions about hackathons.

The ontology is implemented in OWL 2 DL using Protégé, stored in RDF Turtle format, and populated with fictional but realistic individuals for three hackathons, three projects, three teams, six participants, several skills and topics, and three awards. SWRL rules classify teams based on their skills, and SPARQL queries are used to interrogate the ontology.

2 Background and reused vocabularies

2.1 RDF and OWL

The ontology is encoded in RDF using the Turtle syntax. OWL 2 constructs are used to define classes, object properties, data properties, and logical axioms such as subclass relationships, disjointness, and property restrictions. A DL reasoner (HermiT in Protégé) is used to check consistency and to compute inferred class memberships. The design stays within OWL 2 DL to preserve decidability and tool support.

2.2 FOAF

FOAF (Friend of a Friend) is a vocabulary for describing people and organizations and their basic attributes. In this project FOAF is used to avoid redefining generic notions of persons and organizations:

- `foaf:Person` is the parent class for hackathon participants, mentors, and judges.
- `foaf:Organization` is the parent class for organizations that host hackathons.
- `foaf:name` and `foaf:homepage` are reused as standard data properties.

2.3 SKOS

SKOS (Simple Knowledge Organization System) is used to represent controlled vocabularies, in this case skills and topics:

- `skos:Concept` is the parent class for `hack:Skill` and `hack:Topic`.
- `skos:prefLabel` stores human readable labels such as "Data Science".
- `skos:ConceptScheme` groups skills into a skill scheme and topics into a topic scheme.

Skills such as data science, web development, and cloud infrastructure are modeled as individuals of `hack:Skill` with `skos:prefLabel` and membership in a `hack:SkillScheme`. Topics such as "AI for Good" or SDG 11 are modeled as individuals of `hack:Topic`.

2.4 SWRL

The Semantic Web Rule Language (SWRL) combines OWL with Horn style rules of the form body → head. SWRL rules are evaluated over the OWL knowledge base and can derive additional class memberships or property assertions. In this project SWRL is used to classify teams as data science oriented or well balanced based on the skills of their members. The rules are monotonic and purely classificatory, which keeps reasoning straightforward.

2.5 SPARQL

SPARQL is the query language used to retrieve and combine information from the ontology. Queries are executed in the Protégé SPARQL tab and operate over the in memory model. The project provides several example queries that answer competency questions defined in Section 3.

3 Requirements and competency questions

Before designing the ontology, a set of competency questions was defined to guide the modeling and to serve as targets for the SPARQL queries:

CQ1 For a given hackathon, which teams participated and which projects did they work on?

CQ2 For a given team, who are the members and what skills do they have?

CQ3 For a given project, which skills are required and what topic or SDG does it address?

CQ4 Which projects received which awards?

CQ5 Which teams can be considered data science oriented?

CQ6 Which teams are well balanced, combining data science and web development skills?

CQ7 Across all hackathons, which participants have worked together on multiple projects?

The ontology and dataset are designed so that CQ1 to CQ6 can be answered directly with SPARQL, while CQ7 can be answered with a more complex query that detects repeated co membership in teams.

4 Ontology design

4.1 Namespaces

The main namespaces used in the ontology are:

- `hack: http://example.org/hackathon#`
- `foaf: http://xmlns.com/foaf/0.1/`
- `skos: http://www.w3.org/2004/02/skos/core#`
- `dct: http://purl.org/dc/terms/`

The ontology IRI is `http://example.org/hackathon`.

4.2 Core classes

Table 1 summarises the main domain classes and their roles in the ontology.

Class	**Description**
`Hackathon`	Event where teams work on projects during a limited time window.
`Team`	Group of participants collaborating on a project.
`Project`	Outcome developed during a hackathon, such as an application or dashboard.
`Participant`	Human participant in hackathons; subclass of `foaf:Person`.
`Mentor`	Participant providing guidance; subclass of `hack:Participant`.
`Judge`	Participant evaluating projects; subclass of `hack:Participant`.
`Skill`	Capability such as data science or UX design; subclass of `skos:Concept`.
`Topic`	Thematic area or SDG addressed by a project or hackathon; subclass of `skos:Concept`.
`Organization`	Hosting institution or partner; subclass of `foaf:Organization`.
`Award`	Prize or recognition given to a project.
`DataScienceTeam`	Category of team inferred by SWRL when at least one member has data science skills.
`WellBalancedTeam`	Category of team inferred by SWRL when it has both data science and web development skills (possibly in different members).

Table 1: Main domain classes in the hackathon ontology

To improve error checking, disjointness axioms are added between the main categories (for example `Hackathon` is disjoint with `Team`, `Project`, `Participant`, `Skill`, `Organization`, `Award`, `Topic`). This prevents an individual from being both a hackathon and a project.

4.3 Class diagram

Figure 1 shows the class hierarchy as exported from Protégé. The diagram highlights the main classes, the reuse of FOAF and SKOS and the specialization of teams into `DataScienceTeam` and `WellBalancedTeam`.

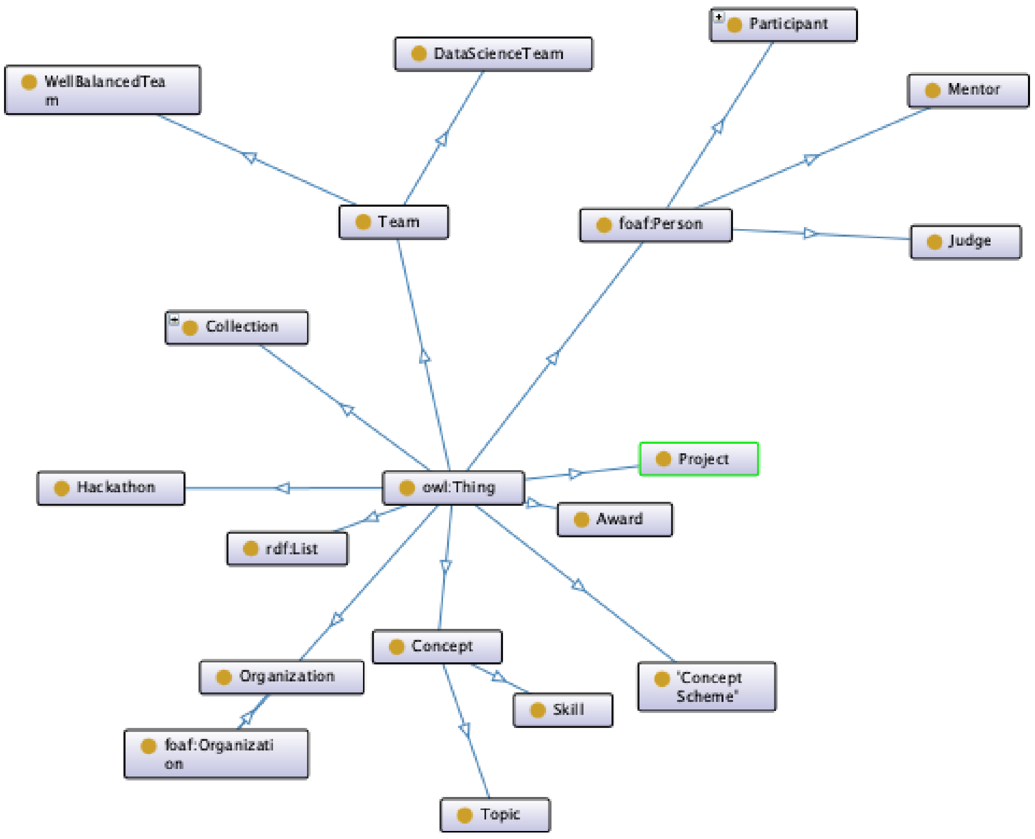

Figure 1: Class diagram of the hackathon ontology

4.4 Reuse of FOAF and SKOS

Reuse of external vocabularies is achieved through subclass relationships:

- `hack:Participant rdfs:subClassOf foaf:Person`
- `hack:Mentor rdfs:subClassOf hack:Participant`
- `hack:Judge rdfs:subClassOf hack:Participant`
- `hack:Organization rdfs:subClassOf foaf:Organization`
- `hack:Skill rdfs:subClassOf skos:Concept`
- `hack:Topic rdfs:subClassOf skos:Concept`

Two concept schemes are defined, `hack:SkillScheme` and `hack:TopicScheme`, and skills and topics are associated with them using `skos:inScheme`. Labels such as "Data Science" and "AI for Good" are stored in `skos:prefLabel`.

4.5 Object and data properties

The main object properties are:

- `hack:hasTeam` (domain `Hackathon`, range `Team`) and its inverse `hack:isTeamOf`.

- `hack:hasParticipant` (domain `Team`, range `Participant`) and its inverse `hack:memberOfTeam`.
- `hack:worksOn` (Team to Project).
- `hack:organizedBy` (Hackathon to Organization).
- `hack:hasSkill` (Participant to Skill).
- `hack:requiresSkill` (Project to Skill).
- `hack:hasTopic` with range `Topic`. This property is used by both projects and hackathons; therefore it deliberately has no domain.
- `hack:hasAward` (Project to Award).

Data properties:

- `hack:startDate`, `hack:endDate`, `hack:deadline` (Hackathon to `xsd:dateTime`).
- `hack:githubUrl` (Project to `xsd:anyURI`).

4.6 OWL restrictions

To move beyond a simple taxonomy, several OWL restrictions are defined:

- `Hackathon` is a subclass of `hasTeam some Team` and `organizedBy some Organization`. Every hackathon has at least one team and at least one organizer.
- `Team` is a subclass of an at least cardinality restriction: `hasParticipant min 2 Participant`. Every team must have at least two participants.
- `Project` is a subclass of `requiresSkill some Skill` and `hasTopic some Topic`. Every project requires at least one skill and has at least one topic.

These axioms are checked by the reasoner and help ensure that the example instances are properly populated.

4.7 Axioms in the ontology

From the OWL perspective, the ontology contains several kinds of axioms:

Class axioms These introduce the classes and relate them. The declarations `hack:Hackathon a owl:Class`, `hack:Team a owl:Class`, and so on, and the subclass statements such as `hack:Participant rdfs:subClassOf foaf:Person` and `hack:Skill rdfs:subClassOf skos:Concept` are class axioms. Disjointness statements such as `hack:Hackathon owl:disjointWith hack:Project` are also class axioms, and they enforce that the main categories cannot overlap.

Property axioms Object and data properties are introduced with their domains, ranges, and inverses. For example, `hack:hasTeam a owl:ObjectProperty` with `rdfs:domain hack:Hackathon` and `rdfs:range hack:Team` and the inverse relation `hack:isTeamOf` are property axioms. Similar axioms exist for `hasParticipant`, `worksOn`, `organizedBy`, `hasSkill`, `requiresSkill`, `hasTopic`, and `hasAward`, as well as the data properties `startDate`, `endDate`, `deadline`, and `githubUrl`.

Restriction axioms Subclass restrictions such as `Hackathon rdfs:subClassOf (hasTeam some Team)` and `Team rdfs:subClassOf (hasParticipant min 2 Participant)` are axioms that add semantic constraints. They state that every hackathon must have at least one team and one organizing organization, every team has at least two participants, and every project has at least one required skill and one topic.

Assertion axioms The individuals in the dataset are described with class assertions and property assertions. For example, `hack:HackathonSDG2025 a hack:Hackathon` and `hack:HackathonSDG2025 hack:hasTeam hack:TeamDataHeroes` are axioms that type an individual and connect it to other individuals. Similar assertions exist for teams, projects, participants, skills, topics, organizations, and awards.

Rule axioms Finally, the SWRL rules are rule axioms on top of the OWL axioms. The rules that infer `DataScienceTeam` and `WellBalancedTeam` use the existing properties `hasParticipant` and `hasSkill` to derive additional class assertions for teams when the skill patterns match.

Together, these different kinds of axioms turn the ontology into a proper OWL 2 DL model rather than just a flat taxonomy.

5 Population and example dataset

The ontology is populated with a small but meaningful dataset.

Organizations Three organizations are modeled as instances of `hack:Organization`: OrgUNIGE (University of Geneva), OrgCERN (CERN), and OrgEPFL (EPFL), each with a FOAF name and homepage.

Hackathons Three hackathons are defined:

- `HackathonSDG2025`, titled "SDG Innovation Hackathon 2025", organized by OrgUNIGE, with topic AI for Good.
- `HackathonHealth2025`, "HealthTech for SDG3 Hackathon 2025", organized by OrgCERN, focused on SDG 3.
- `HackathonClimate2025`, "Climate Resilience Hackathon 2025", organized by OrgEPFL, focused on SDG 11.

Each hackathon has start date, end date, and deadline.

Participants Six participants (`hack:ParticipantEisha`, `hack:ParticipantHassan`, `hack:ParticipantAlex`, `hack:ParticipantSofia`, `hack:ParticipantLi`, `hack:ParticipantAmina`) are modeled with FOAF names and multiple skills through `hack:hasSkill`.

Skills and topics Seven skills and several topics are modeled as individuals of `hack:Skill` and `hack:Topic`, with SKOS labels and membership in concept schemes.

Teams and projects Three teams and three projects are created:

- `TeamDataHeroes` works on `ProjectGreenCity`, a sustainability dashboard.
- `TeamInfinity` works on `ProjectHealthConnect`, a health application.
- `TeamCityGuard` works on `ProjectFloodWatch`, a flood monitoring dashboard.

Each team has at least two participants, and each project has required skills and a topic.

Awards Three awards are modeled, each associated with one project using `hack:hasAward`.

6 SWRL rules and reasoning

Two SWRL rules are defined in Protégé to classify teams based on the skills of their members.

6.1 Data science oriented teams

The first rule states that any team that has at least one participant with the Data Science skill is a `DataScienceTeam`:

```
hack:Team(?t) ^
hack:hasParticipant(?t, ?p) ^
hack:hasSkill(?p, hack:SkillDataScience)
->
hack:DataScienceTeam(?t)
```

After executing this rule with the reasoner, teams such as TeamDataHeroes and TeamCityGuard are inferred as instances of `hack:DataScienceTeam`.

6.2 Well balanced teams

The second rule identifies well balanced teams that combine data science and web development skills, possibly in different members:

```
hack:Team(?t) ^
hack:hasParticipant(?t, ?p1) ^
hack:hasParticipant(?t, ?p2) ^
hack:hasSkill(?p1, hack:SkillDataScience) ^
hack:hasSkill(?p2, hack:SkillWebDev)
->
hack:WellBalancedTeam(?t)
```

Once the rule is applied, teams that contain at least one data scientist and at least one web developer become instances of `hack:WellBalancedTeam`. This captures a simple notion of balance often used when assigning or evaluating teams.

7 SPARQL queries

7.1 Teams, members, and skills

The following query retrieves each team, its members, and their skills. It answers competency question CQ2.

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>

SELECT ?teamName ?personName ?skillLabel
WHERE {
  ?team a hack:Team ;
        dct:title ?teamName ;
        hack:hasParticipant ?p .
  ?p foaf:name ?personName ;
     hack:hasSkill ?skill .
  ?skill skos:prefLabel ?skillLabel .
}
ORDER BY ?teamName ?personName ?skillLabel
```

7.2 Required skills per project

The next query lists projects with their required skills, addressing CQ3.

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>

SELECT ?projectTitle ?skillLabel
WHERE {
  ?project a hack:Project ;
           dct:title ?projectTitle ;
           hack:requiresSkill ?skill .
  ?skill skos:prefLabel ?skillLabel .
}
ORDER BY ?projectTitle ?skillLabel
```

7.3 Teams and projects per hackathon

The following query links hackathons to their teams and projects (CQ1).

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?hackathonTitle ?teamTitle ?projectTitle
WHERE {
  ?hack a hack:Hackathon ;
        dct:title ?hackathonTitle ;
        hack:hasTeam ?team .
  ?team dct:title ?teamTitle ;
        hack:worksOn ?project .
  ?project dct:title ?projectTitle .
}
ORDER BY ?hackathonTitle ?teamTitle
```

7.4 Data science and well balanced teams

After running the SWRL rules, the following queries answer CQ5 and CQ6.

Data science oriented teams:

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?teamName
WHERE {
  ?team a hack:DataScienceTeam ;
        dct:title ?teamName .
}
```

Well balanced teams:

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?teamName
WHERE {
  ?team a hack:WellBalancedTeam ;
        dct:title ?teamName .
}
```

7.5 Projects and awards

To answer CQ4, the following query lists each project and the award it received, if any:

```
PREFIX hack: <http://example.org/hackathon#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?projectTitle ?awardTitle
WHERE {
  ?project a hack:Project ;
           dct:title ?projectTitle ;
           hack:hasAward ?award .
  ?award dct:title ?awardTitle .
}
ORDER BY ?awardTitle ?projectTitle
```

8 Discussion and limitations

The ontology provides a compact but expressive model of a hackathon ecosystem. The reuse of FOAF and SKOS avoids reinventing basic concepts. Disjointness and property restrictions ensure that the model is not just a simple taxonomy but a proper ontology that supports reasoning. SWRL rules add lightweight classification based on intuitive patterns.

There are several limitations and possible extensions:

- Mentors and judges are modeled as subclasses of Participant but are not yet populated with instances or linked to evaluation activities.
- Temporal reasoning about repeated participation across multiple years is not addressed.
- Only SWRL is used; SHACL shapes could be added for data validation, for example to enforce that projects have exactly one GitHub URL.
- The dataset is fictional and small. Connecting the ontology to real hackathon data would be a natural next step.

9 Conclusion

This project designs and implements an OWL ontology for hackathons, teams, participants, skills, topics, and awards. The ontology reuses FOAF and SKOS, adds OWL restrictions for key constraints, and uses SWRL rules to classify teams as data science oriented or well balanced. A set of SPARQL queries answers predefined competency questions about team composition, project requirements, awards, and hackathon structure. The work demonstrates how Semantic Web technologies can be used in practice to give structure and reasoning capability to information about collaborative innovation events.

References

[1] Protégé Project. *Class Expression Syntax in Protégé.*
Available at `https://protegeproject.github.io/protege/class-expression-syntax/`.

[2] W3C OWL Working Group. *OWL 2 Web Ontology Language Manchester Syntax.*
W3C Recommendation. Available at `https://www.w3.org/TR/owl2-manchester-syntax/`.