Ontology Construction Approach

The development of the ontology for integrating Digital Product Passports (DPPs) into Enterprise Asset Management (EAM) systems started by identifying the key data and concepts needed. This was done through literature review, interviews with industry professionals, and a survey aimed at understanding current practices. The findings highlighted asset information is often scattered across multiple systems, stored in various formats, and difficult to access or integrate efficiently. There is also a lack of common standards for how product data should be structured and shared. The insights from the review, interviews, and survey helped define the important elements and relationships that the ontology should include. This work forms the basis for building a structured, flexible model that can improve data integration, support better maintenance planning, and help organizations move toward more sustainable and circular asset management.

Table 1: Interview and Survey Findings

Interview Findings	**Survey Findings**
Asset data is fragmented across spreadsheets, PDFs, and internal systems like Maximo.	Most organizations use manual methods (e.g., spreadsheets) for asset tracking.
Lack of standardized structure and validation limits interoperability.	Top challenges: lack of standardized data, difficulty tracking maintenance history.
QR codes are already used in some products for reordering and identification.	DPPs are seen as beneficial for traceability, compliance, and sustainability.
There are concerns about data confidentiality and intellectual property when sharing detailed product information.	Main barriers: cost of upgrades, lack of standards, resistance to new technology.
Semantic technologies not yet used but recognized for their potential.	Regulatory compliance and standardization are seen as key adoption drivers.
Confidentiality concerns limit data sharing, especially from manufacturers.	Most valued DPP data: maintenance history, energy consumption, environmental impact.

MOMo methodology Workflow

The MOMo workflow methodology presented by Shimizu *et al.* (2023) is applied to develop the modular ontology in a structured and reusable manner, aligned with stakeholder needs and expectations from the survey and interviews. For the development of DPPs for EAM systems, the MOMo workflow plays a vital role in guiding how to address reuse or adapting ontologies for a new use case or purpose. It also defines processes for creating modular ontologies, which is a more suitable approach, in this context for developing DPPs since it contains a large amount of data (Kebede *et al.* 2024). The modular ontology was developed using the software Protégé and Python to exemplify data in a DPP.

Table 2: The MOMo methodology workflow adapted according to ***(Shimizu et al. 2023)***

Step	Responsible	Output
1. Describe use cases & data sources	Entire team	Use case descriptions
2. Gather competency questions	Entire team	List of CQs
3. Identify key notions	Entire team	List of key notions
4. Identify existing ODPs	Ontology engineers	Selected ODP(s) for each key notion.
5. Create module diagrams	Entire team	Diagrammatic representation of the solution module.
6. Document modules & axioms	Ontology engineers & domain experts	Module documentation with embedded schema diagrams, axiomatization, etc. (e.g., in LaTeX, Word, HTML format).
7. Create ontology diagram	Ontology engineers	Diagrammatic representation of the whole composed ontology.
8. Add spanning axioms	Ontology engineers	Documentation of the entire ontology with embedded schema diagrams, axiomatization, etc. (e.g., in LaTeX, Word, HTML format).
9. Review naming & axioms	Ontology engineers	Updated module and ontology documentation.
10. Create OWL file & axioms	Ontology engineers	An OWL file for publication and use.

1. Describe use cases & data sources

Industries such as manufacturing, infrastructure, and construction that rely on physical assets, an effective lifecycle management is crucial for ensuring reliability, cost efficiency and sustainability. Many organizations still depend on spreadsheets and manual documentation, resulting in poor traceability and limited data interoperability. The survey responses indicated the challenges related to non-standardized asset tracking methods leading to inconsistent data across the EAM system. The interview results highlighted the current reliance on static documents and the requirement for machine-readable data. DPPs offers an opportunity to address these issues by providing machine-readable, structured data on a product's origin, composition, usage, maintenance history, and environmental impact. This use case focuses on integrating DPP data into an EAM system using semantic technologies to reduce fragmented data and to improve maintenance activities and sustainability tracking. For example, each lighting fixture is equipped with a QR code linked to its DPP, which contains semantically structured data such as technical specifications, carbon footprint, lifespan, and maintenance instructions. When scanned in the field, the data is accessed through a mobile interface, allowing maintenance staff to retrieve maintenance instructions, log issues, and trigger feedback loops. The ontology is designed to support the integration of heterogeneous product data, providing a modular, standard compliant, and reusable framework for managing asset information within EAM systems.

2. Gather competency questions

1. What Digital Product Passport (DPP) is associated with a product?
2. What repair instructions are available for a given product?
3. What are the characteristics and composition of a product as per its DPP?
4. Who is the responsible actor for a product's DPP?
5. What is the model and serial number of a product?
6. What materials and components make up a product?
7. What certifications does a product have?
8. Who manufactured a product?
9. What are the physical and chemical properties of a material?
10. Is a component made of hazardous material?
11. What is the expected lifetime of a component?
12. Who manufactured a material or component?
13. Who is the manufacturer of a product or component?
14. What role does an agent play in manufacturing or distribution?
15. What is the provenance of a product? Which certifications do a product hold?
16. Who is the certifying authority for a product?
17. What documents are associated with a certification?
18. What lifecycle stage is the product currently in?
19. What environmental impacts are associated with a product?
20. Which agent performed the design or manufacturing of a product?
21. What maintenance tasks are associated with a product?
22. What is the cost or duration of a maintenance activity?
23. Who performed a given maintenance task?
24. What work orders have been created for the product's service and maintenance?
25. What inspections, repairs, replacements, or calibrations have been recorded for the product?
26. When was the product last serviced?
27. Who performed the maintenance or inspection?
28. What costs are associated with specific maintenance services?
29. What maintenance reports are available for the product?
30. Are there recurrent maintenance schedules for the product?
31. Who is the sender or receiver in a transportation event?
32. What is the identifier of a supply chain event?
33. Where and when did a supply chain event occur?

3. Identify key notions

Table 3: Identified Keynotions

Product	Access Control	Failure Event
Digital Product Passport (DPP)	Repair Instruction	Data Format
Component	Installation	QR Code
Usage	Energy Consumption	RFID Tag
Lifecycle	Environmental Impact	System Identifier
Maintenance Activity	Material Composition	log issues
Maintenance History	Service Log	Process
Data Carrier	Location	Data
User Role	Disassembly Information	Operational Status
Instructions	Technical Specification	Type
Supplier	Carbon Footprint	Certification
Manufacturer	Ownership History	Task
Regulatory Standard	End-of-Life Process	Asset information
Carbon footprint	Maintenance staff	organizations

4. Identify existing ODPs

Key Notion	**Ontology Design Pattern**
Digital Product Passport (DPP) and DPP information Reused from (Jansen *et al.* 2024)	- **Dpp-info**: Describes the DPP information types - **Dpp-ODP**: This is a minimal core Ontology Design Pattern defining what a DPP is, i.e. information about a product. - **DPP-core**: This is a minimal core ontology for DPPs, specializing the DPP ODP. - **DPP-prov**: Describes provenance information for DPP information.
Product Reused from (Kebede *et al.* 2024)	- **Name Stub and Identifiers**: The “Name Stub” and ‘Identifier’ patterns are used to name and uniquely identify products and components. - **AgentRole** including Role-Dependent Names: The ‘AgentRole’ is used for defining various roles that entities play in the supply chain as well as to model who is responsible for various aspects such as design, manufacturing, etc. - **Product**: This pattern enables the modeling of product components and compositions. Material - **Quantities and Units**: This pattern is selected to represent product dimensions, weight, density, and other similar technical specifications. - **Explicit Typing**: This pattern provides a way of explicitly specifying the types of entities, in this case, different types of products.

	- **Provenance**: This pattern can provide a more comprehensive understanding of the history and derivation of each product, including answering the following competency questions: - **Property Reification**: This pattern is used to model instructions such as assembly, disassembly, and installation instructions.
Material and Component	- **Name Stub and Identifiers**: The “Name Stub” and ‘Identifier’ patterns are used to name and uniquely identify material and components. **- AgentRole**: This pattern defines the actors and their specific functions related to a component or material. **- Provenance**: This pattern tracks the origin and history of a material or component. It creates a detailed record of its lifecycle by linking it to a series of events. **- Quantity and Units**: This pattern provides a structured way to describe a physical Property. Instead of just using a number, it captures a Quantity by combining a numeric value with its corresponding Unit of measure and the QuantityKind (e.g., mass, density), ensuring the data is clear and computationally useful. - **Material property**: Is used to describe the characteristics and attributes of a Material. It categorizes properties into PhysicalProperty, ChemicalProperty, BiochemicalProperty, SustainabilityProperty, and TemporalProperty.
Manufacturer	**- Provenance**: This pattern is used to track the origin and history of an entity. It details the specific activities such as Producing, Importing, or Supplying, that an entity has been involved in. **- AgentRole**: Is used to separate the agent from its role. Easy to see if an agent has multiple roles. - **Name Stub and Identifiers**: The “Name Stub” and ‘Identifier’ patterns are used to name and uniquely identify the manufacturer.
Regulatory Compliance and Certifications	- **Identifier**: Captures the type of certification. - **AgentRole**: Is used to separate the agent from its role. Easy to see if an agent has multiple roles. - **Provenance**: To see how a certification was issued and by whom and what activity was made (e.g., certification creation).
Lifecycle & Environmental Impact	- **Event**: Represents lifecycle stages/events of the product (e.g., Manufacturing, End-of-Life). - **AgentRole**: This pattern is used to describee the involved agents and their role within the lifecycle stages. - **Provenance**: Captures how, who, when and what when it comes to environmental data and lifecycle events. - **Quantity**: Captures numeric measures and units from a lifecycle event and environmental impact. - **SpatiotemporalExtent**: Assiociates each lifecycle event with a time and place.
Maintenance	- **Status**: Captures the state of a maintenance activity (e.g., complete or incomplete) - **Recurrent Event**: Handles event that will be repeated (e.g., inspections, service) - **Document**: Links workorders to maintenance reports, manual or logs. It supports that documentation is digitally attached.

	- **Reporting Event**: To see what agent reported an event. - **Provenance**: The provenance ODP is used to see who performed the maintenance, type of activity. - **AgentRole**: Is used to separate the agent from its role. Easy to see if an agent has multiple roles.
Supply chain Information	- **SpatiotemporalExtent**: This pattern is used to anchor each Event to a specific context in place and time. It directly links an event to its geographical Place and chronological Time, answering the "where?" and "when?" questions. - **Event**: This pattern represents the key activities or stages that occur within the supply chain. The diagram shows specific examples like Manufacturing, Storage, Distribution, and Transportation. It acts as the central hub connecting who did what, where, and when - **ParticipantRole**: This pattern describes the function or role that a participant plays in a supply chain Event. - **Identifier**: This pattern provides a standardized way to reference supply chain information. It allows for an object to have an **Identifier** that includes both the ID value itself (as text) and the type of ID it is (**IDType**), such as a product code, batch number, or shipment ID.

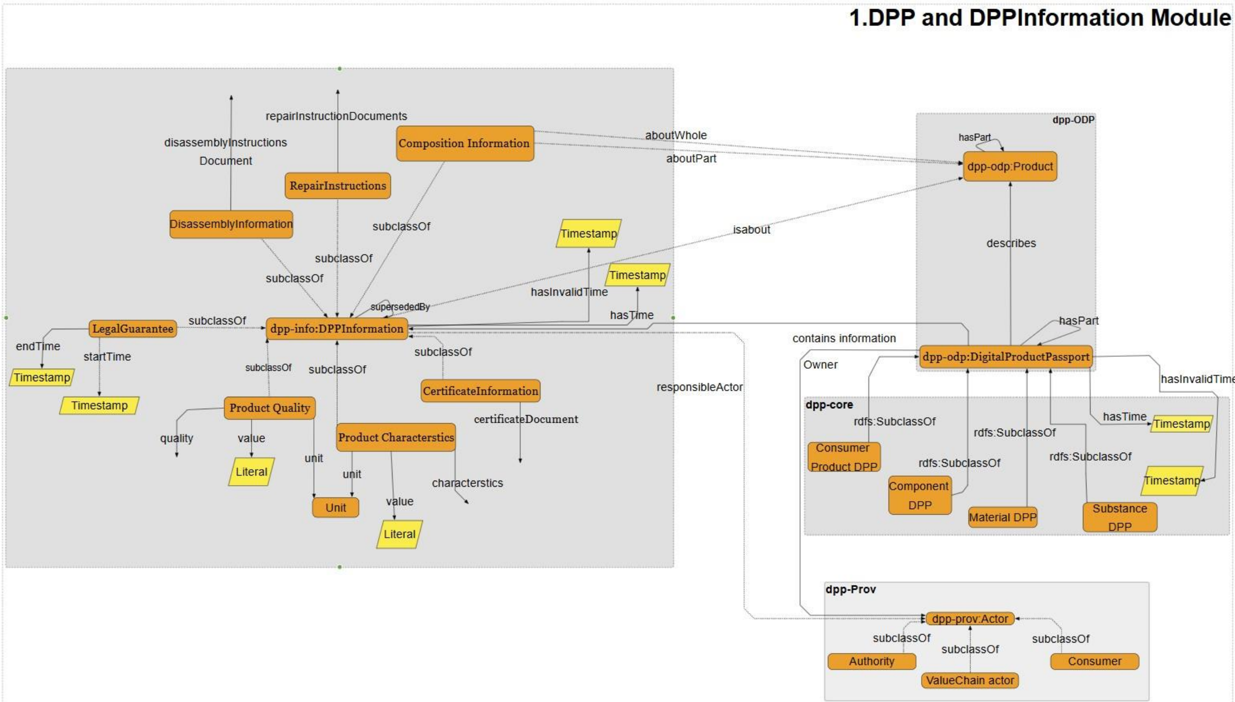
To promote interoperability and avoid redundancy, the proposed ontology strategically reuses existing ontological constructs from well-established vocabularies. Table 4 summarizes the reused elements. Core domain concepts such as *Quantity*, *QuantityKind*, *QuantityValue*, *Unit*, *Product*, *Material*, *Agent*, *Place*, and *Manufacturer* are integrated primarily from QUDT, Schema.org, and PROV-O, ensuring alignment with widely adopted Linked Data standards.Object properties such as *used*, *wasGeneratedBy*, *atTime*, and *attributedTo* are reused from **PROV-O** to describe processes, roles, and how data is connected over time. The *numericValue* property from **QUDT** is used to represent numbers related to quantities in a consistent way.By reusing these standard terms, the ontology becomes easier to integrate with other systems and supports better data sharing across different use cases.

Table 4: concept and properties that are used in DPP4EAM Ontology

Ontology Part	Element Names	Ontology
Concepts	Quantity, QuantityKind, QuantityValue, Unit, Role, Entity, Activity, Product, Manufacturer, Agent, Material, Name, Identifier, Place	QUDT, PROV-O, Schema.org
Object Properties	used, wasGeneratedBy, attributedTo, derivedFrom, hasPart, supersededBy, atTime	PROV-O, Schema.org
Datatype Properties	numericValue	QUDT

5-6. Create module diagrams - Document modules & axioms

1.DPP and DPP info Module



Axiom:

Product ⊑ ∃hasDPP.DigitalProductPassport

→ **Explanation:** A product must be linked to its Digital Product Passport (DPP).

DigitalProductPassport ⊑ ∃containsInformation.DPPInformation → **Explanation:** A DPP must contain relevant product information.

DigitalProductPassport ⊑ ∃hasPart.ComponentDPP

→ **Explanation:** A DPP must describe at least one component.

ComponentDPP ⊑ ∃hasTime.Time

→ **Explanation:** Every component's lifecycle in the DPP must include a timestamp.

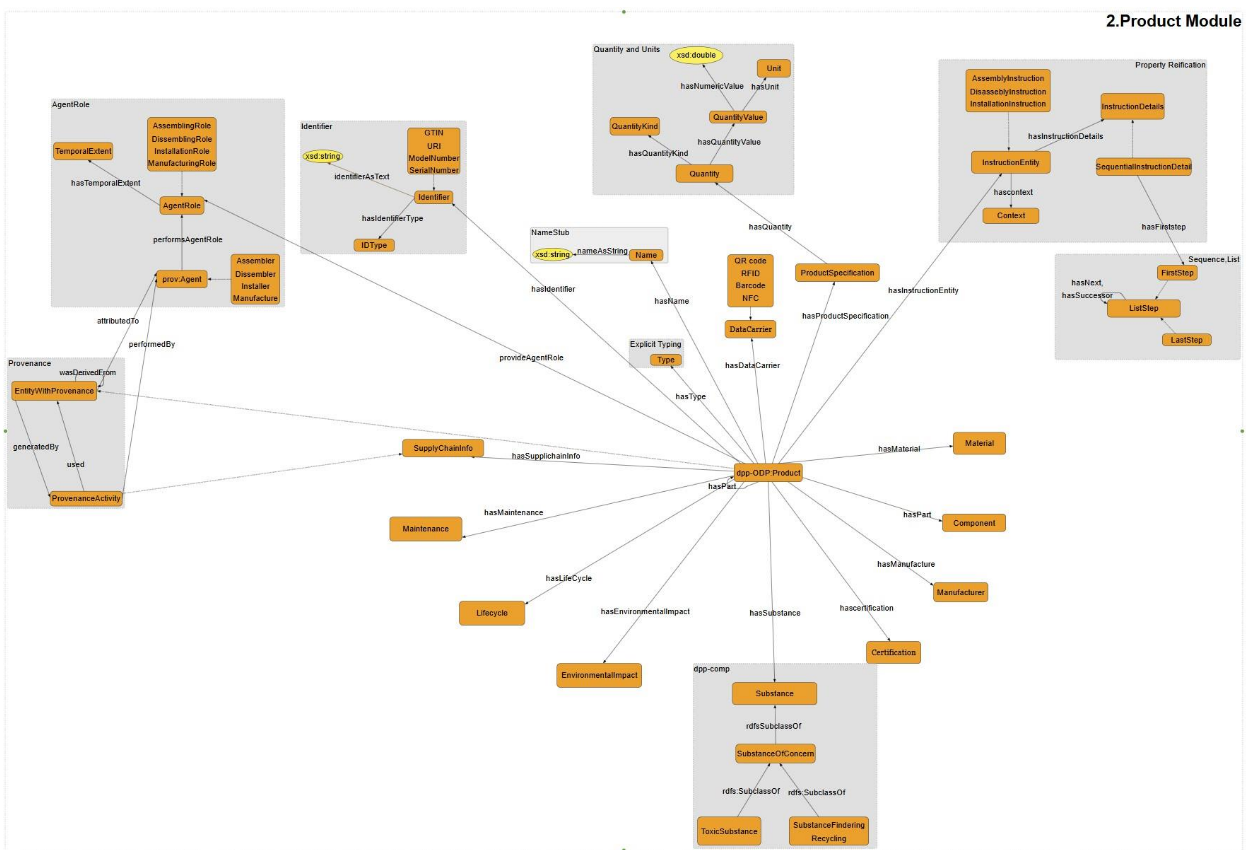
CertificateInformation ⊑ ∃certificateDocument.Document

→ **Explanation:** Certification information must reference an official document.

SPARQL Query:

```
SELECT ?dpp ?product WHERE {
  ?dpp dpp:describes ?product .
}
SELECT ?dpp ?actor WHERE {
  ?dpp dpp:responsibleActor ?actor .
}
SELECT ?product ?certificate WHERE {
  ?product dpp:certificateDocument ?certificate .
}
```

2.Product Module



Axiom:

Product ⊑ ∃hasIdentifier.Identifier

→ **Explanation:** Every product must be associated with a unique identifier.

Product ⊑ ∃hasName.Name

→ **Explanation:** Each product must have a name to be identified by humans.

Product ⊑ ∃hasPart.Component

→ **Explanation:** A product is composed of one or more components.

Product ⊑ ∃hasSpecification.Specification

→ **Explanation:** Products must have associated specifications.

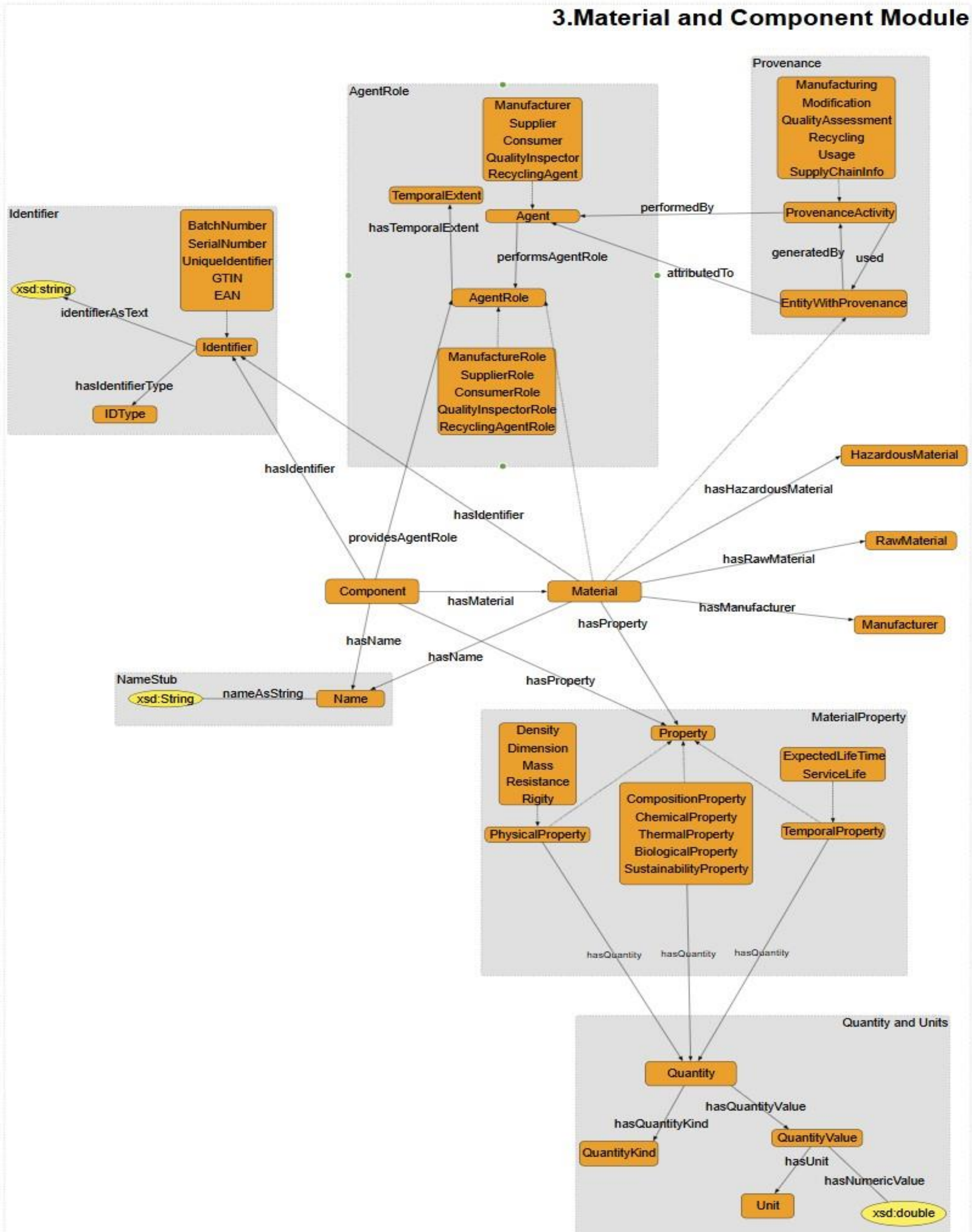
Product ⊑ ∃hasType.Type

→ **Explanation:** Each product must be classified under a type.

SPARQL Query:

```
SELECT ?product ?name
WHERE {
  ?product prod:hasName ?name .
}
SELECT ?identifier ?type WHERE {
  ?identifier prod:hasIdentifierType ?type .
  OPTIONAL { ?identifier a schema:identifier . }
}
SELECT ?product ?spec WHERE {
  ?product prod:hasSpecification ?spec .
}
SELECT ?product ?instruction WHERE {
  ?product prod:hasInstructionDetail ?instruction .
}
```

3.Material and Componet Module



Axiom:

Component ⊑ ∃hasMaterial.Material

→ **Explanation:** A component must be associated with some material.

Material ⊑ ∃hasProperty.PhysicalProperty

→ **Explanation:** A material must have at least one physical property.

Product ⊑ ∃hasComponent.Component

→ **Explanation:** Every product must be composed of at least one component.

Component ⊑ ∃hasIdentifier.Identifier

→ **Explanation:** Each component must have a unique identifier.

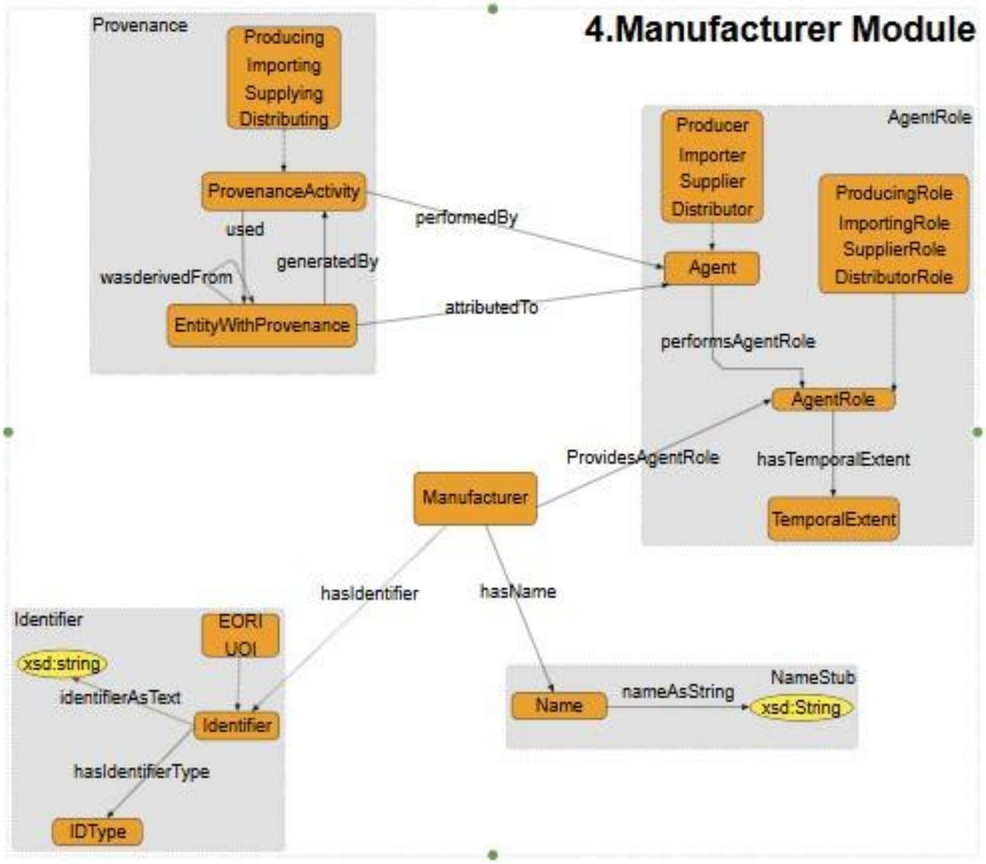
Material ⊑ ∃hasQuantity.Quantity

→ **Explanation:** Materials must be linked to a measurable quantity.

SPARQL Query:

```
SELECT ?product ?material WHERE {
  ?product mc:hasMaterial ?material .
}
SELECT ?product ?component
WHERE {
  ?product mc:hasComponent ?component .
}
SELECT ?entity ?identifier ?name WHERE {
  ?entity mc:identifierAsText ?identifier .
  OPTIONAL { ?entity mc:hasName ?name . }
}
```

4.Manufacturer Module



Axiom:

Product ⊑ ∃hasManufacturer.Manufacturer

→ **Explanation:** Every product must be linked to a manufacturer.

Manufacturer ⊑ ∃hasIdentifier.Identifier

→ **Explanation:** Each manufacturer must be uniquely identifiable.

Manufacturer ⊑ ∃providesAgentRole.ManufacturingRole

→ **Explanation:** Manufacturers provide the role of manufacturing agents.

ManufacturingRole ⊑ ∃performedBy.Manufacturer

→ **Explanation:** Each manufacturing role must be linked to the manufacturer that performs it.

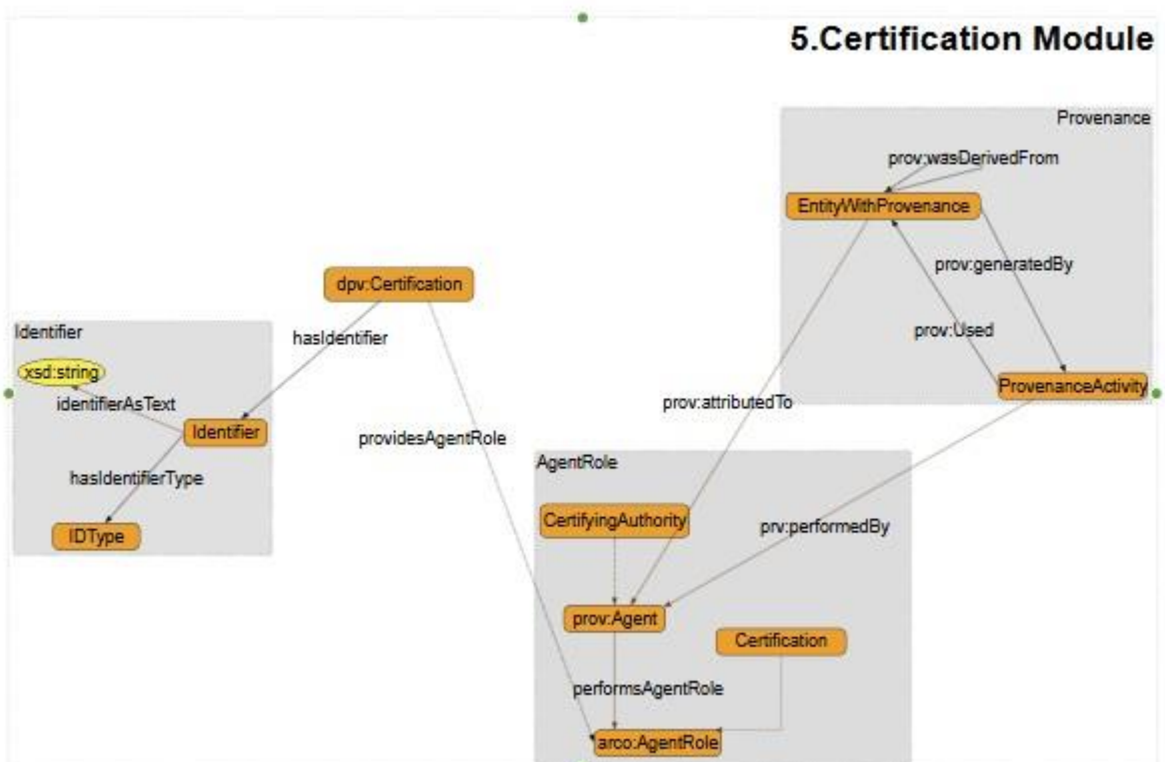
Manufacturer ⊑ ∃hasName.Name

→ **Explanation:** A manufacturer must have a name as an identifying attribute.

SPARQL Query:

```
SELECT ?product ?manufacturer
WHERE {
  ?product manuf:hasManufacturer ?manufacturer .
}
SELECT ?identifier ?idType WHERE {
  ?identifier manuf:hasIdentifierType ?idType .
}
SELECT ?entity ?idText WHERE {
  ?entity manuf:identifierAsText ?idText .
}
```

5.Certification Module



Axiom:

Product ⊑ ∃hasCertification.Certification

→ **Explanation:** Every product must be linked to its certification details.

Certification ⊑ ∃issuedBy.Authority

→ **Explanation:** Each certification must be issued by an authority.

Certification ⊑ ∃certificateDocument.Document

→ **Explanation:** Certifications should be accompanied by documentation.

Certification ⊑ ∃hasIdentifier.Identifier

→ **Explanation:** Each certification must have a unique identifier.

Certification ⊑ ∃hasTime.Time

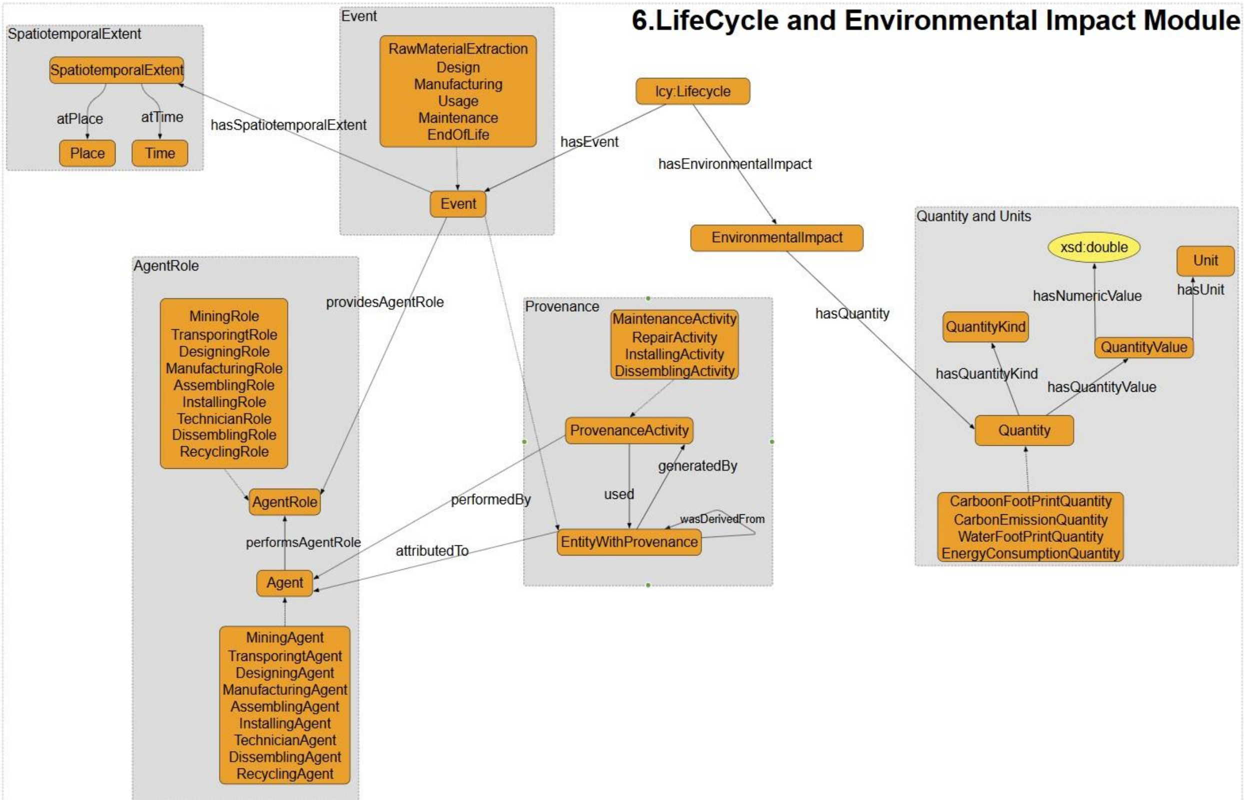
→ **Explanation:** Certifications should include issuance or expiration time.

SPARQL Queries:

```
SELECT ?identifier ?idType WHERE {
  ?identifier a cert:IDType .
}

SELECT ?product ?cert WHERE {
  ?product cert:hasCertification ?cert .
}
```

6.LifeCycle and Environmental Impact Module



Axiom:

Product ⊑ ∃hasLifeCycle.LifeCycle

→ **Explanation:** Each product must be linked to a life cycle.

LifeCycle ⊑ ∃hasEnvironmentalImpact.EnvironmentalImpact

→ **Explanation:** A lifecycle must be associated with environmental impact information.

EnvironmentalImpact ⊑ ∃hasQuantityValue.QuantityValue

→ **Explanation:** Environmental impacts are measured and expressed as quantity values.

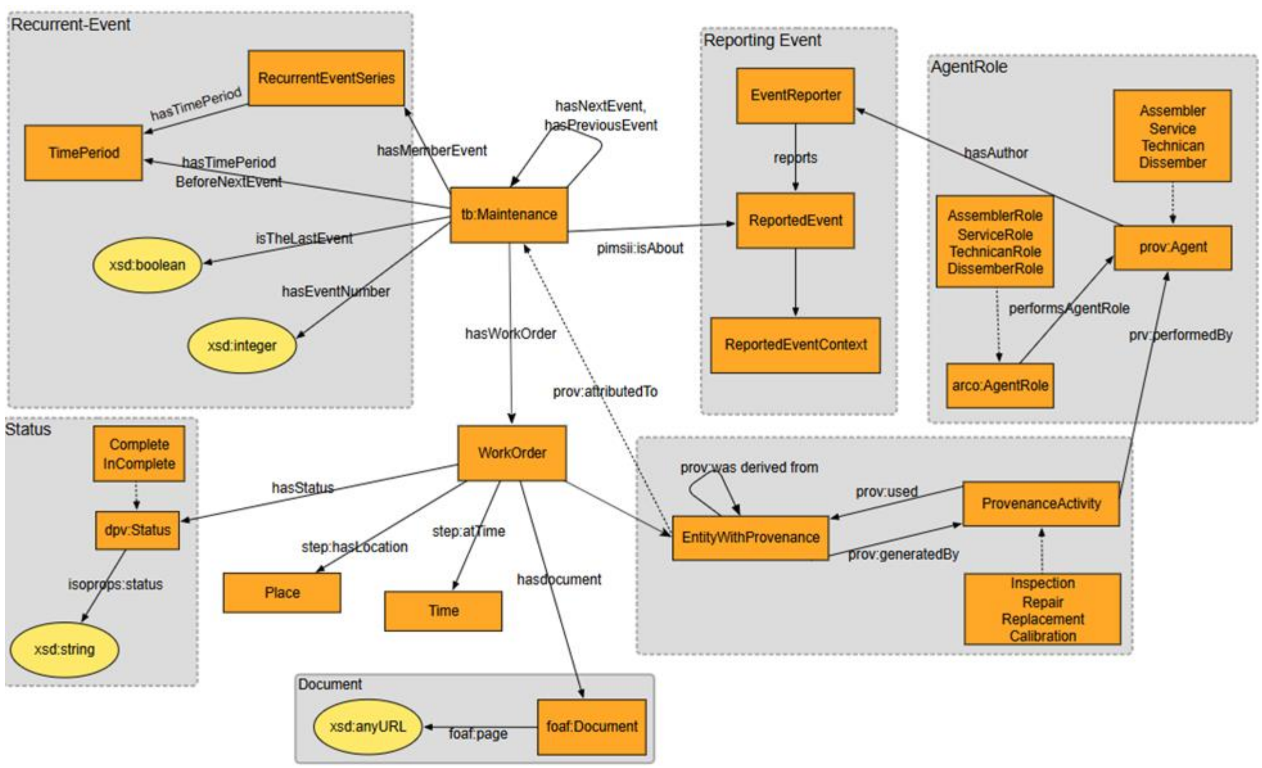
QuantityValue ⊑ ∃hasUnit.Unit

→ **Explanation:** A quantity value must be expressed using a specific unit.

SPARQL Query:

```
SELECT ?lifecycle ?event WHERE {
  ?lifecycle lcei:hasEvent ?event .
}
SELECT ?entity ?impact WHERE {
  ?entity lcei:hasEnvironmentalImpact ?impact .
}
SELECT ?event ?place WHERE {
  ?event lcei:atPlace ?place .
}
```

7.Maintenace Module



Axiom:

1. Product ⊑ ∃ hasMaintenance.Maintenance
   → Every product must be associated with a maintenance record.
2. Maintenance ⊑ ∃ hasPreviousEvent.Maintenance
   → A maintenance event may refer to a previous maintenance event.
3. Maintenance ⊑ ∃ hasNextEvent.Maintenance
   → A maintenance event may link to a next event in the series.
4. Maintenance ⊑ ∃ hasWorkOrder.WorkOrder
   → A maintenance event must have a corresponding work order.
5. Maintenance ⊑ ∃ attributedTo.Agent
   → Maintenance must be attributed to an agent (e.g., technician).
6. Maintenance ⊑ ∃ wasDerivedFrom.EntityWithProvenance
   → Maintenance data can be derived from previous data or records.
7. WorkOrder ⊑ ∃ hasLocation.Place
   → A work order must specify where the maintenance happens.
8. WorkOrder ⊑ ∃ atTime.Time
   → A work order must include when it happens.
9. WorkOrder ⊑ ∃ hasDocument.Document
   → A work order may link to related documentation.
10. Document ⊑ ∃ page.xsd:anyURI
    → The document must include a web link or source.
11. Maintenance ⊑ ∃ hasStatus.Status
    → Maintenance must include a status (e.g., complete or incomplete).
12. RecurrentEventSeries ⊑ ∃ hasMemberEvent.Maintenance
    → A recurrent event series includes individual maintenance events.
13. RecurrentEventSeries ⊑ ∃ hasTimePeriod.TimePeriod
    → A recurrent event series has a regular interval.
14. RecurrentEventSeries ⊑ ∃ hasEventNumber.xsd:integer
    → Each event series has a countable number of occurrences.
15. RecurrentEventSeries ⊑ ∃ isTheLastEvent.xsd:boolean
    → It can be indicated whether this is the final event.
16. Maintenance ⊑ ∃ hasTimePeriodBeforeNextEvent.TimePeriod
    → A maintenance event may define time until the next one.
17. ProvenanceActivity ⊑ subclassOf prov:Activity
    → Any maintenance-related action is a type of provenance activity.
18. Inspection, Repair, Replacement, Calibration ⊑ ProvenanceActivity
    → All these are specific kinds of maintenance activities.
19. ProvenanceActivity ⊑ ∃ performedBy.Agent
    → All activities must be performed by someone (agent).

20. ProvenanceActivity ⊑ ∃ used.EntityWithProvenance
   → They must use existing provenance entities.
21. EntityWithProvenance ⊑ ∃ generatedBy.ProvenanceActivity
   → Provenance entities must be generated by an activity.
22. Agent ⊑ ∃ performsAgentRole.AgentRole
   → An agent (e.g., person, company) must perform a defined role.
23. Assembler, Service, Technician, Dissembler ⊑ AgentRole
   → These are specific types of agent roles.
24. AgentRole ⊑ subclassOf Agent
   → All agent roles are agents by definition.
25. EventReporter ⊑ ∃ reports.ReportedEvent
   → A reporter (e.g., user/system) must report an event.
26. ReportedEvent ⊑ ∃ hasReportedEventContext.ReportedEventContext
   → Each reported event must include some contextual info.
27. ReportedEvent ⊑ ∃ isAbout.Maintenance
   → The report must be about a maintenance record.
28. EventReporter ⊑ ∃ hasAuthor
   → The reporter must be attributed to an author.
29. Complete, InComplete ⊑ Status
   → These are subtypes of general status.
30. Status ⊑ ∃ Status.xsd:string
   → A status is described using a string value.

SPARQL Query:

```
PREFIX mnt:
<http://www.semanticweb.org/emilfernando/ontologies/
Maintenance#>
```

What maintenance reports are available for the product?

```
SELECT ?product ?maintenance ?report

WHERE {

  ?product mnt:hasMaintenance ?maintenance .

  ?report a mnt:ReportedEvent ;

          mnt:isAbout ?maintenance .

}
```

Who performed the maintenance or inspection?

```
SELECT DISTINCT ?product ?maintenance ?agent
```

```
        WHERE {
          ?product mnt:hasMaintenance ?maintenance .
          ?maintenance mnt:attributedTo ?agent .
        }
```

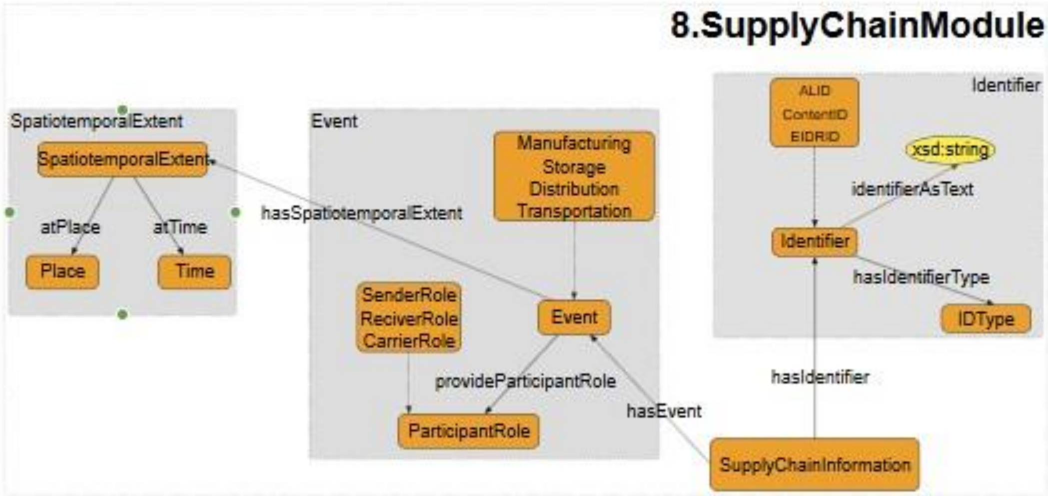
What have been recorded for the product?

```
SELECT DISTINCT ?product ?activity ?type
        WHERE {
          ?product mnt:hasMaintenance ?activity .
          ?activity a ?type .
          FILTER(?type IN (
            mnt:Inspection,
            mnt:Repair,
            mnt:Replacement,
            mnt:Calibration
          ))
        }
```

What work orders have been created for the product's maintenance?

```
    SELECT DISTINCT ?product ?maintenance
?workOrder
        WHERE {
          ?product mnt:hasMaintenance ?maintenance .
          ?maintenance mnt:hasWorkOrder ?workOrder .
        }
```

8.SupplyChain Module



Axiom:

Product ⊑ ∃hasSupplyChainInfo.SupplyChainInformation

→ **Explanation:** Each product must be associated with supply chain information.

Supplier ⊑ ∃hasName.Name

→ **Explanation:** A supplier should have an identifiable name.

SupplyChainInformation ⊑ ∃hasIdentifier.Identifier

→ **Explanation:** Supply chain data should be uniquely identified.

Supplier ⊑ ∃providesAgentRole.SupplierRole

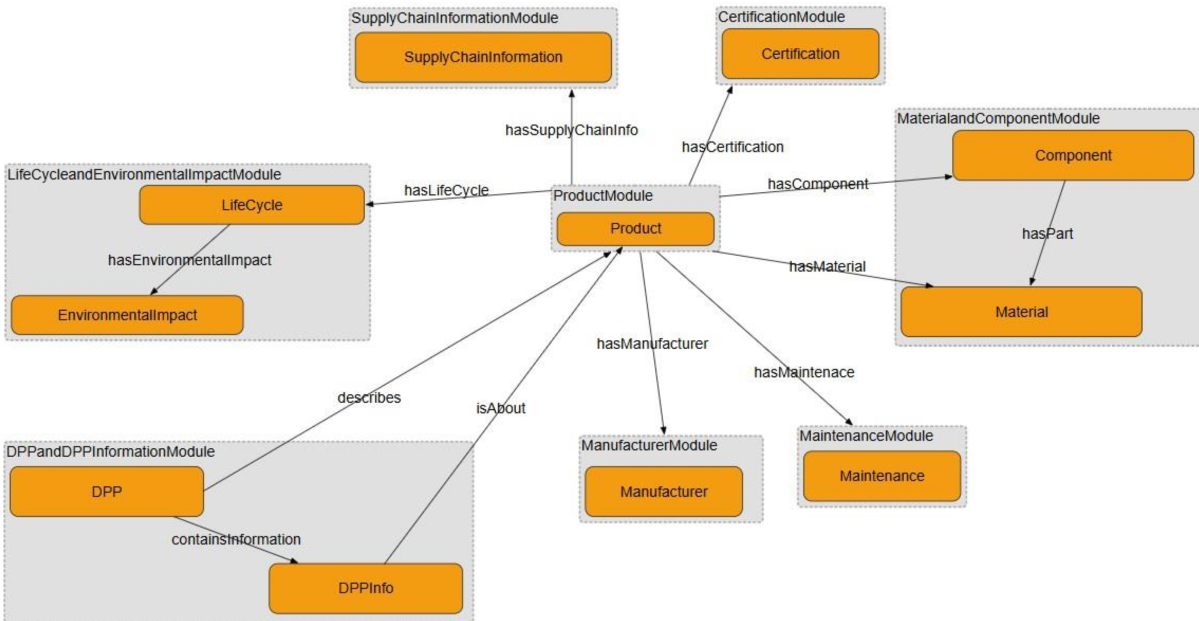
→ **Explanation:** Suppliers are defined by the roles they provide within the supply chain context.

SPARQL Query

```
SELECT ?product ?event WHERE {
  ?product sci:hasEvent ?event .
}

SELECT ?event ?time WHERE {
  ?event a sci:Event .
  ?extent prov:atTime ?time .
}
```

7-9. Create ontology diagram, Add spanning axioms - Review naming & axioms



DPPandDPPInfo ⊑ ∃describes.Product

→ **Explanation:** The DPP describes the product.

Product ⊑ ∃hasManufacturer.Manufacturer

Explanation: Every product must have a defined manufacturer entity.

Product ⊑ ∃hasLifeCycleandEnvironmentalImpact.LifeCycleandEnvironmentalImpact

Explanation: A product has lifecycle and impact data.

Product ⊑ ∃hasMaterialandComponent.MaterialandComponent

Explanation: A product is made of materials or components

Product ⊑ ∃hasCertification.Certification

Explanation: A product has at least one certification.

Product ⊑ ∃hasSupplyChainInfo.SupplyChainInformation

Explanation: A product includes supply chain information.

Product ⊑ ∃hasMaintenance.Maintenance **Explanation:** A

product may need maintenance.

SPARQL Query:

```
SELECT ?product ?component ?material ?manufacturer ?certification
WHERE {
  OPTIONAL { ?product core:hasComponent ?component . }
  OPTIONAL { ?product core:hasMaterial ?material . }
  OPTIONAL { ?product core:hasManufacturer ?manufacturer . }
  OPTIONAL { ?product core:hasCertification ?certification . }
}
```

10. Create OWL file & axioms

modules\DPPmodule.owl

modules\Productmodule .owl

modules\Material and Component module.owl

modules\ManufacturerModule.owl

modules\CertificationModule.owl

modules\MaintenanceModule.owl

modules\LifeCycle and EnvironmentalImpact Module .owl

modules\SupplychainModule.owl