

**MATERIALDIGITAL**

---

# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

Platform MaterialDigital- Semantic Interoperability Team

---



# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

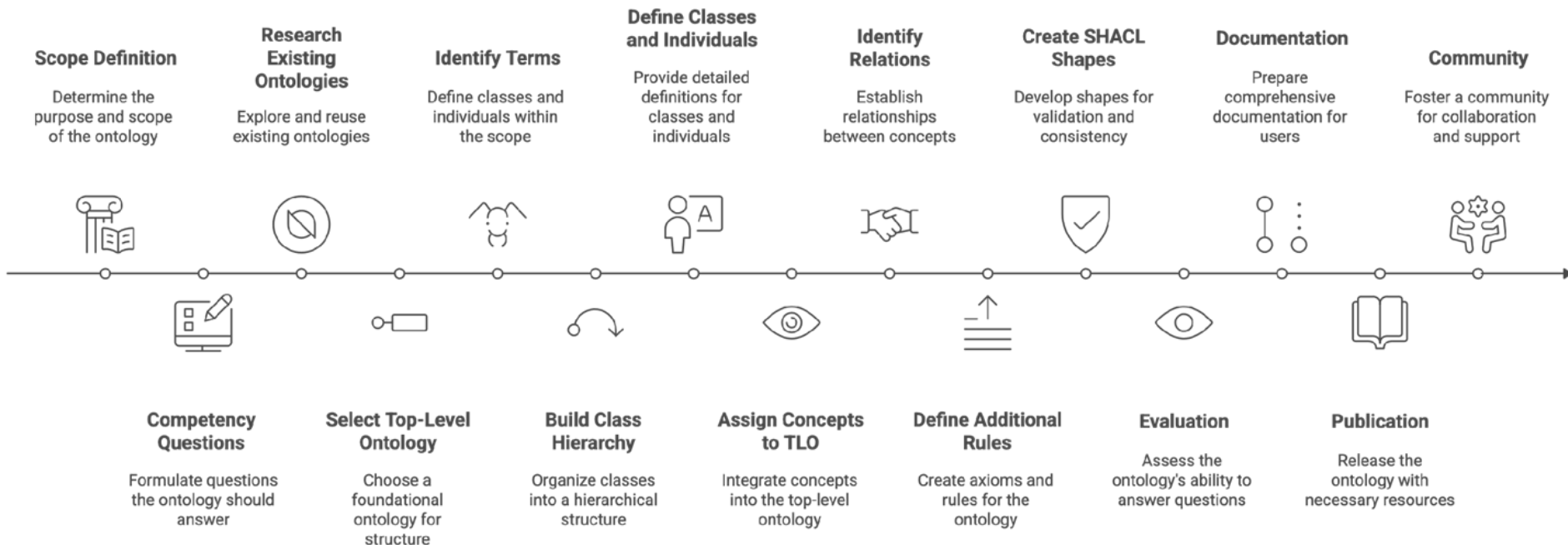
- 1- Ontology development and beginners learning materials
- 2- Ontology levels
- 3- Basic Formal Ontology (BFO) classes
- 4- Platform MaterialDigital Core Ontology (PMDco) classes  
**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**
- 5- Platform MaterialDigital Core Ontology (PMDco) object properties  
**Tutorial 2: Using appropriate object properties ([Miro board](#))**
- 6- How to develop your application ontologies using PMDco and OBO+ODK best practices?
- 7- Ontology Development Kit (ODK)  
**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**
- 8- Collaborative ontology development workflow, adding taxonomy and axioms
- 9- PMDco Ontology Design Patterns (ODPs)  
**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**
- 10- Ontology evaluation, release, documentation and maintenance

# 1- Ontology development and beginners learning materials

An ontology is an explicit, formal specification of a shared conceptualization.

conceptualization: abstract model (domain, relevant terms, relations)  
explicit: meaning of all terms is clearly and unambiguously defined  
formal: interpretable by machines  
shared: consensus

## Ontology development process:



# 1- Ontology development and beginners learning materials

🏠 PMD Core Ontology

Welcome

☰ Introduction to Ontologies

- What is an Ontology?
- Why do we Need Ontologies?
- Motivation for Developing Ontologies in Materials Science and Engineering (MSE)
- Ontology Levels and Example MSE-Related Ontologies
- Ontology Language
- Key Components of an Ontology
- Ontology Development Tools and Resources
- Ontology Learning Materials for Beginners**
- Introduction to PMDco v3.x.x
- Reused Ontologies
- Ontology Structure

## Ontology Learning Materials for Beginners

- [A Practical Ontology Development Guide](#): In cooperation with participants in the MaterialDigital initiative, a guide for the development of ontologies in general was created, which contains basic aspects and recommended procedures. This guide may be expanded and developed further and is hosted at [The Scientific Ontology Network](#).
- [Ontology Development 101](#): Stanford University's guide provides a foundational, step-by-step methodology for creating your first ontology.
- [Pizza Tutorial](#): A practical guide to building OWL ontologies using Protégé 5.5 and plugins
- [OBO Academy's Ontology Design Course](#): An interactive, free curriculum that provides lessons on ontology construction, reasoning, and design principles (based on the OBO Foundry and Semantic Web standards)
- [ISE FIZ YouTube channel: Playlist of Lectures "Knowledge Graphs - Foundations and Applications"](#)
- [Barry Smith's YouTube channel](#)
- [PMD YouTube channel](#)



[PMDco documentation page](#)

**Knowledge Graphs - Foundations and ...**  
by ISE FIZ Karlsruhe  
Playlist • 67 videos • 70,963 views

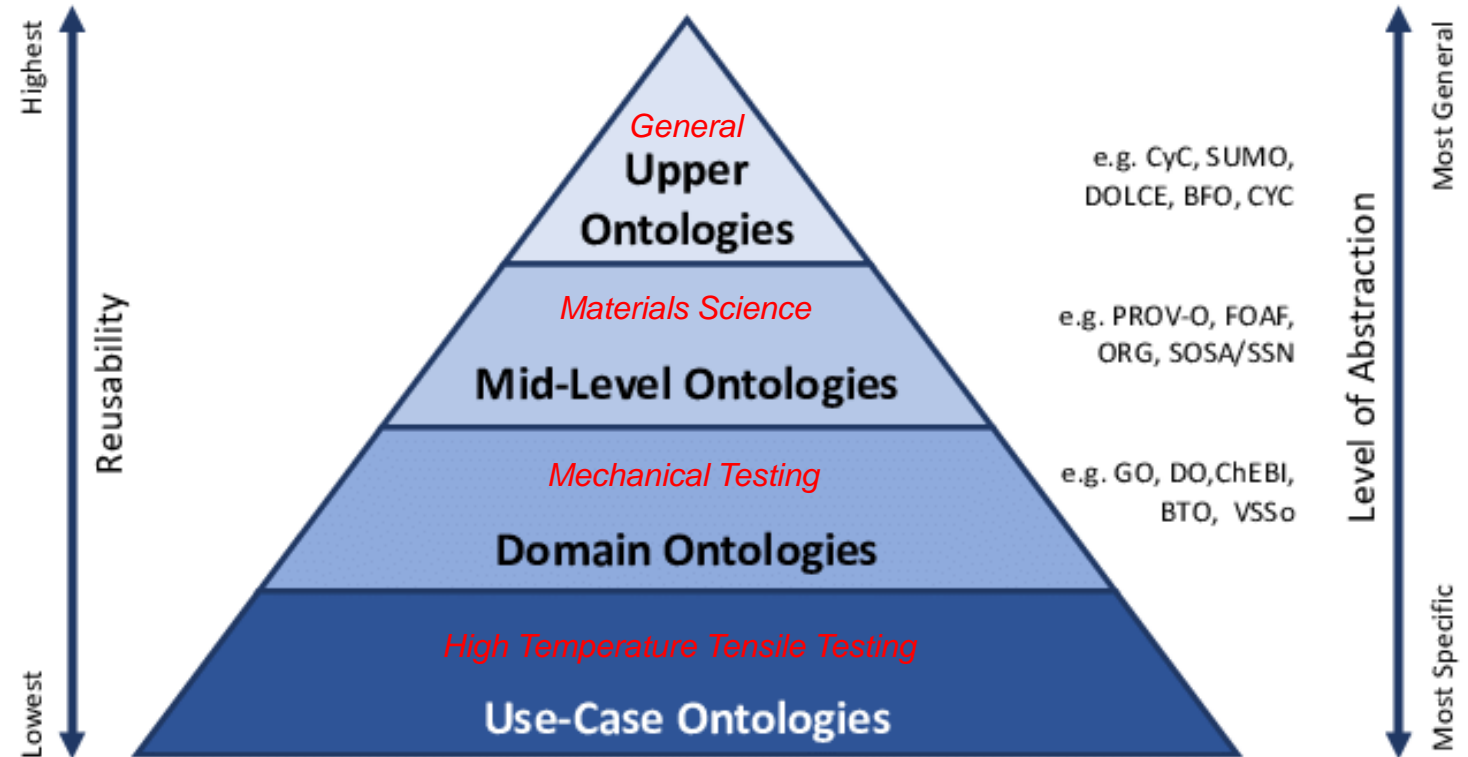
▶ Play all

- 1 Knowledge Graphs - 0.0 Lecture Overview  
ISE FIZ Karlsruhe • 11K views • 1 year ago  
16:08
- 2 Knowledge Graphs - 1.0 Knowledge Representation with Graphs  
ISE FIZ Karlsruhe • 4.4K views • 1 year ago  
3:45
- 3 Knowledge Graphs - 1.1 From Data to Knowledge  
ISE FIZ Karlsruhe • 4.5K views • 1 year ago  
10:18
- 4 Knowledge Graphs - 1.2 Knowledge and how to represent it  
ISE FIZ Karlsruhe • 3.8K views • 1 year ago  
9:19
- 5 Knowledge Graphs - 1.3 The Art of Understanding  
ISE FIZ Karlsruhe • 3.2K views • 1 year ago  
9:10
- 6 Knowledge Graphs - 1.4 Graphs and Triples  
ISE FIZ Karlsruhe • 3.7K views • 1 year ago  
10:22

SCAN ME!

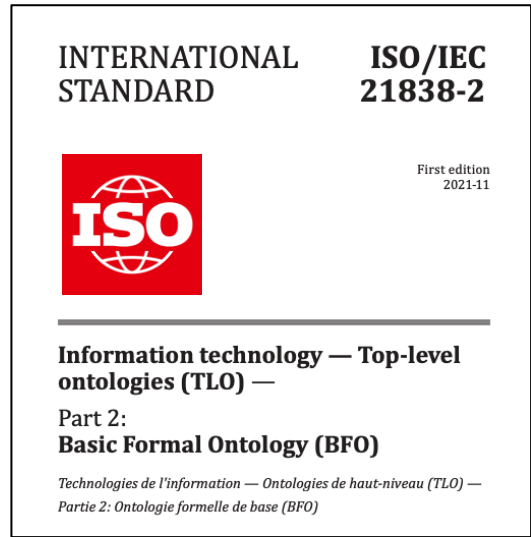
## 2- Ontology levels

- **Top-Level Ontologies:** Define very general concepts that apply across all domains
- **Mid-Level Ontologies:** Provide reusable structures shared across several domains
- **Domain-Level Ontologies:** Describe concepts specific to a particular field
- **Application-Level Ontologies:** Highly specific to individual applications or projects



# 3- Basic Formal Ontology (BFO)

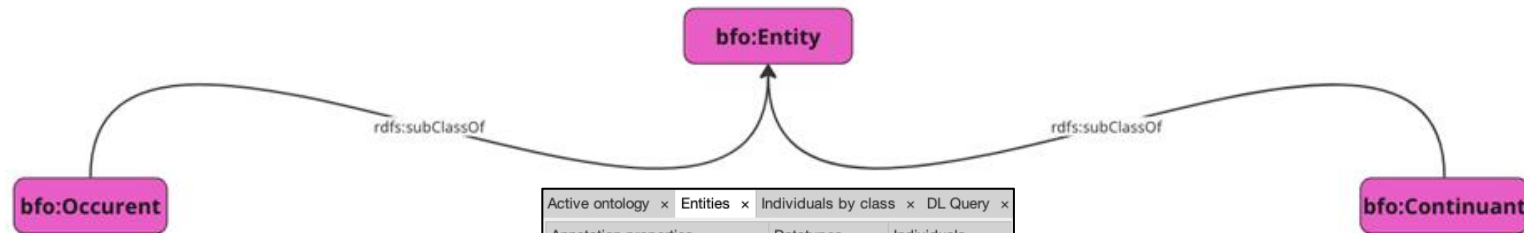
- ISO standard top-level ontology
- A small top-level ontology that is designed for use in supporting information retrieval, analysis and integration in scientific and other domains.
- Used by more than 800 ontology-driven endeavors throughout the world.



<b>Common foundation</b>	<b>Consistency</b>	<b>Facilitate reasoning</b>	<b>Interoperability</b>	<b>Reusability</b>
Enable integration across different domain ontologies	Avoid ambiguity by defining core categories (e.g., object, process, quality)	Logical structure helps automated reasoning and validation	Allow systems and datasets to “speak the same language”	Save time and effort by building on established upper-level concepts

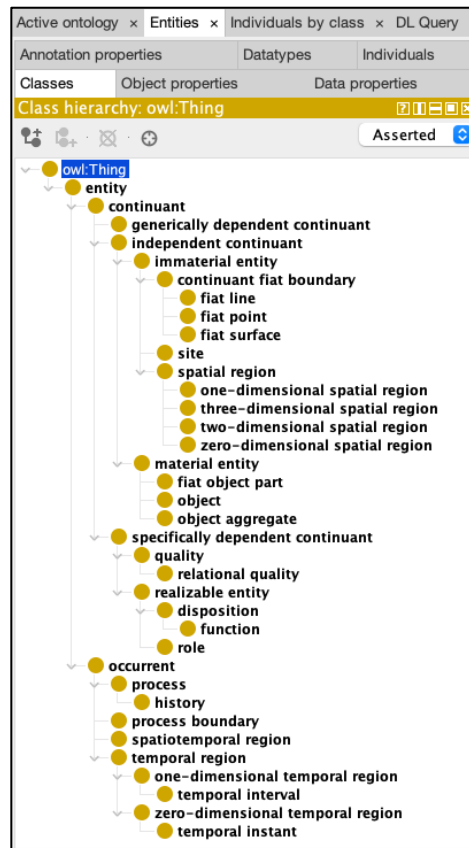
# 3- Basic Formal Ontology (BFO) classes: the very top

**BFO** sits at the very top as a domain-neutral upper ontology  
Divides all entities into two fundamental categories:



## Occurrents

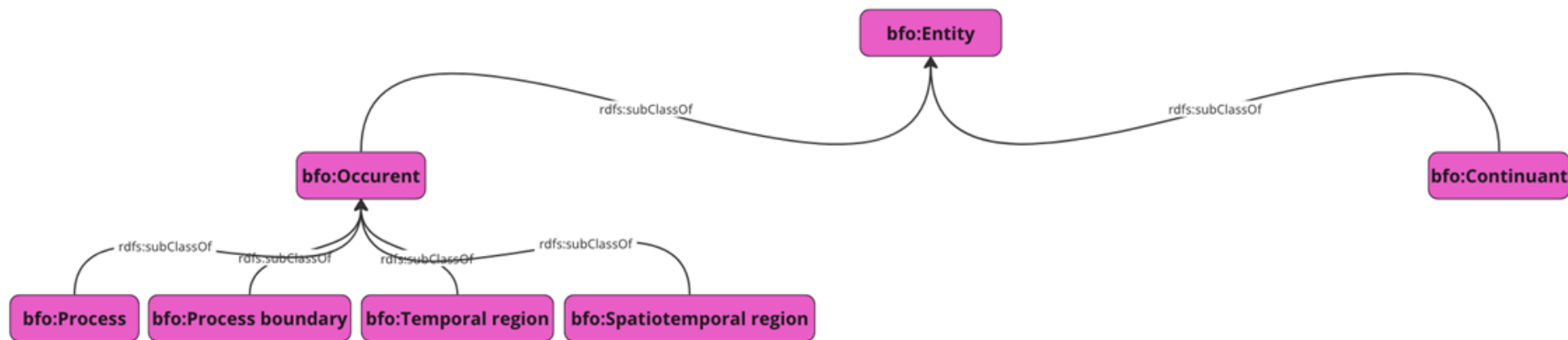
- Happen or unfold **over time**
- Have temporal parts and duration
- Examples: reading a book



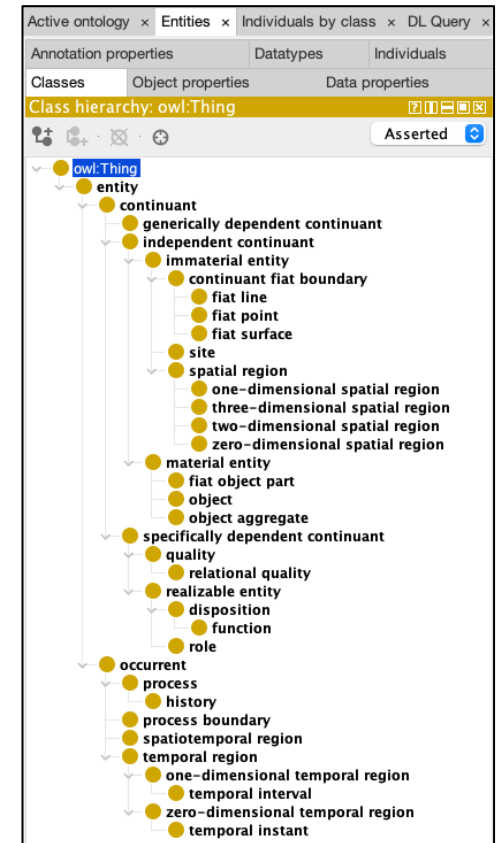
## Continuants

- Persist through time while possibly undergoing change
- Exist **wholly at any moment** in time
- Example: a book

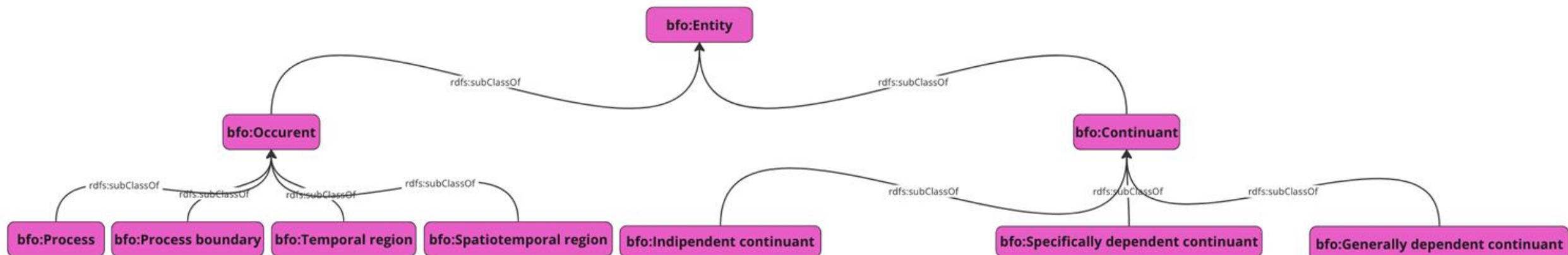
# 3- Basic Formal Ontology (BFO) classes: Occurrent



- **Process:** Ongoing event with temporal parts (e.g., a surgery, running)
- **Process Boundary:** Instant where a process starts or ends (e.g., moment of impact)
- **Temporal Region:** Span of time itself (e.g., a day, a year)
- **Spatiotemporal Region:** Combination of space and time (e.g., the area of a concert during its performance)



# 3- Basic Formal Ontology (BFO) classes: Continuant



## Independent Continuants

- Exist on their own
- Do not depend on other entities for their existence
- Examples: a rock, a human body, a tree

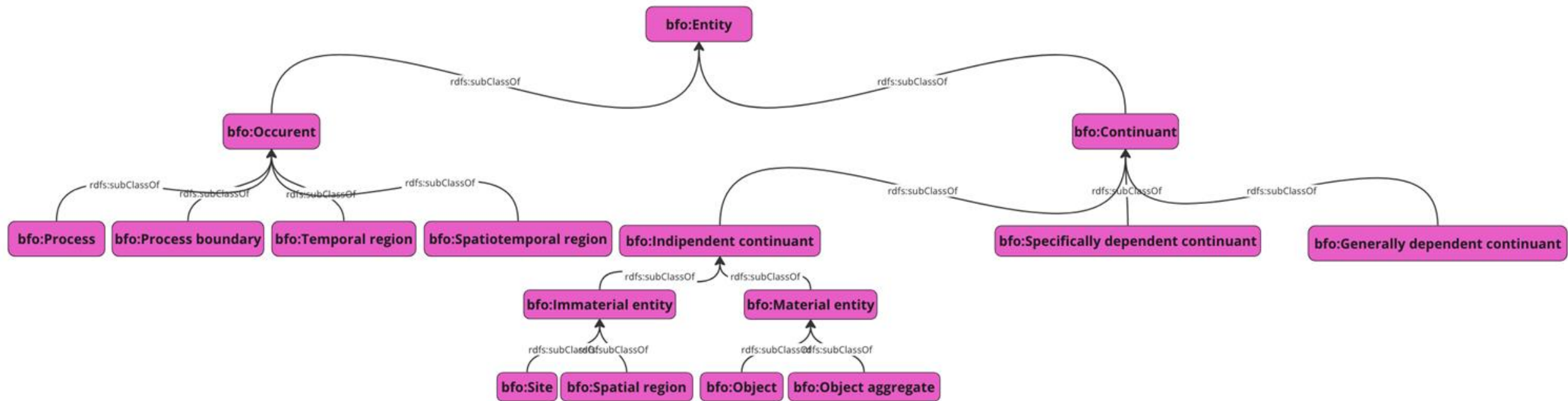
## Specifically Dependent Continuants

- Depend on **one specific** bearer
- If the bearer ceases to exist, then its quality, function, role ceases to exist
- Examples: the color of *this* apple, my weight

## Generically Dependent Continuants

- Can be **copied or instantiated** in different bearers
- Examples: a digital file, a PDF document, a gene sequence

# 3- Basic Formal Ontology (BFO) classes: Independent continuant



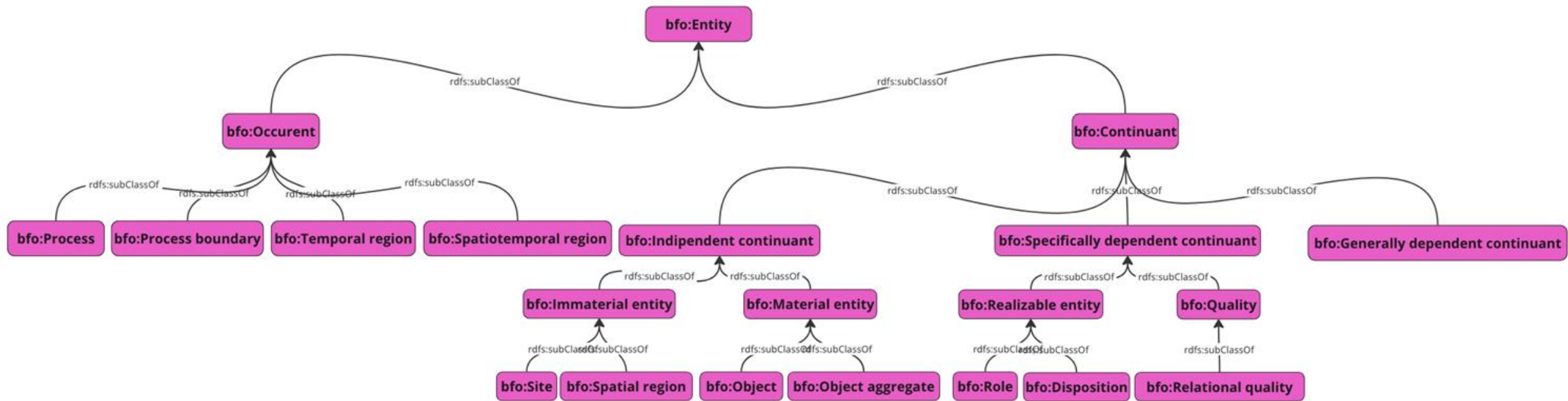
**Immaterial Entity:** Has no physical mass but still exists spatially

- **Site:** location occupied by something (e.g., parking space)
- **Spatial Region:** abstract space (e.g., 3D coordinates)
- **Fiat Boundary:** non-physical border (e.g., equator)

**Material Entity:** Has physical presence and occupies space

- **Object:** one whole, independent thing (e.g., a car)
- **Object Aggregate:** group of objects (e.g., a fleet of cars)
- **Fiat Object Part:** arbitrary part of an object (e.g., front half of a car)

### 3- Basic Formal Ontology (BFO) classes: Specifically Dependent Continuant



**Realizable Entity:** Exists as a potential; realized in certain conditions

- **Role:** A realizable entity that depends on external **social or** institutional context

*Example:* A person as a teacher, a molecule as a tracer

- **Disposition:** A realizable entity that reflects an inherent tendency

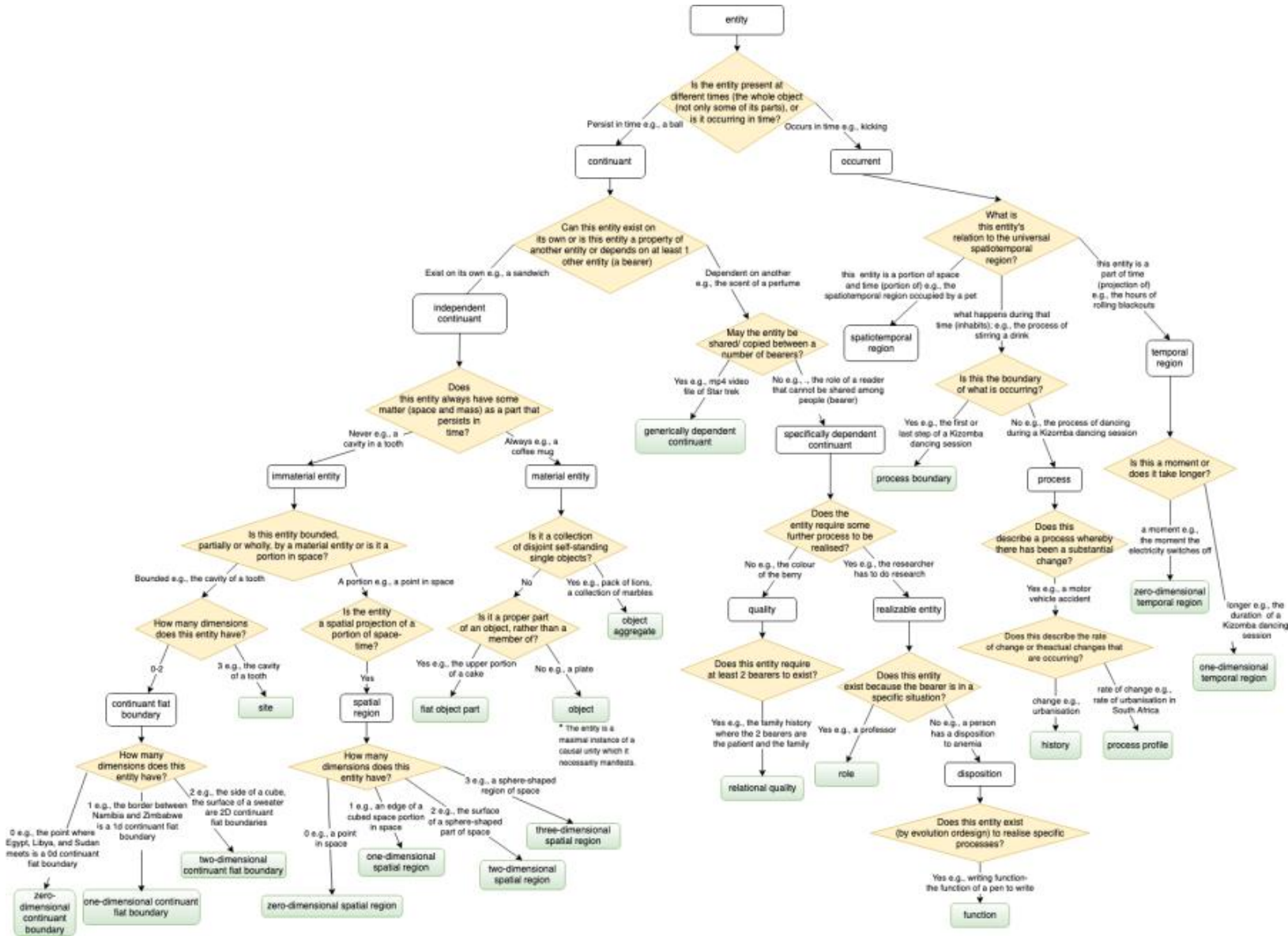
*Example:* Fragility of glass, flammability of gasoline

**Quality:** Inherent and always present in the bearer, Measurable or observable features

Examples: color of an apple, mass of a rock

- **Relational Quality:** a quality that depends on another entity (e.g., distance between two objects)

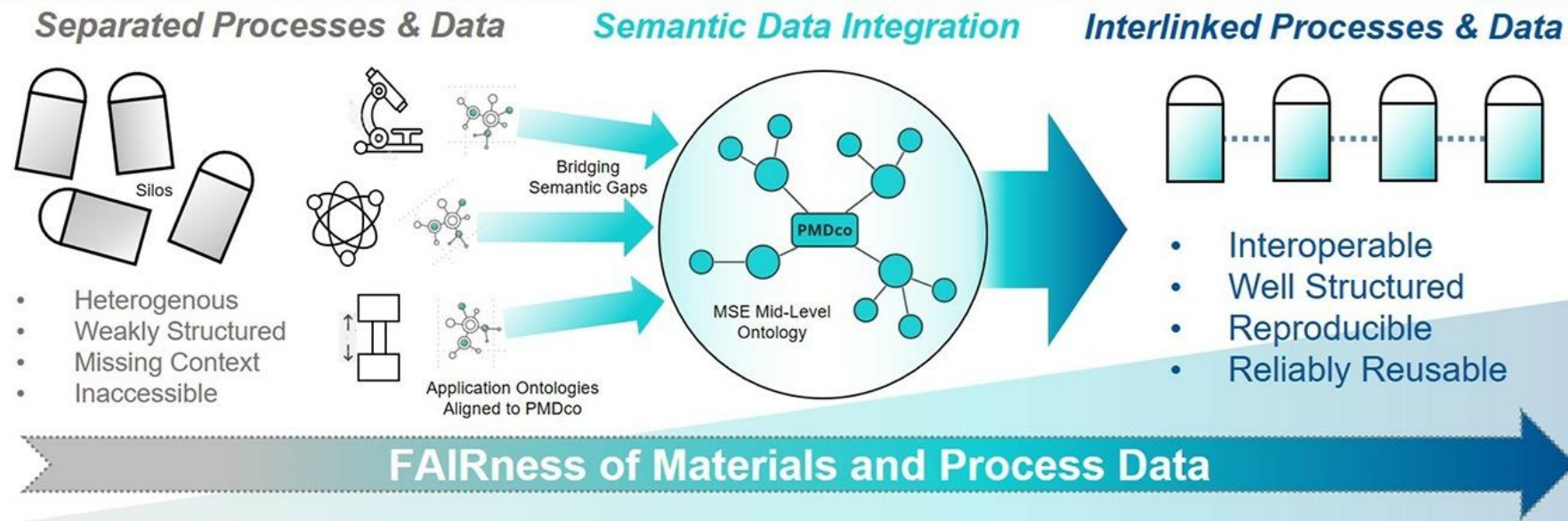
# BFO Classifier



# 4- Platform MaterialDigital Core Ontology (PMDco) classes

A mid-level ontology to support the digital transformation of the Materials Science and Engineering (MSE) domain.

## Achieving Semantic Interoperability for Materials Science and Engineering



### Objectives

- ✓ Semantic Interoperability
- ✓ FAIR Data Principles
- ✓ Workflow Modeling

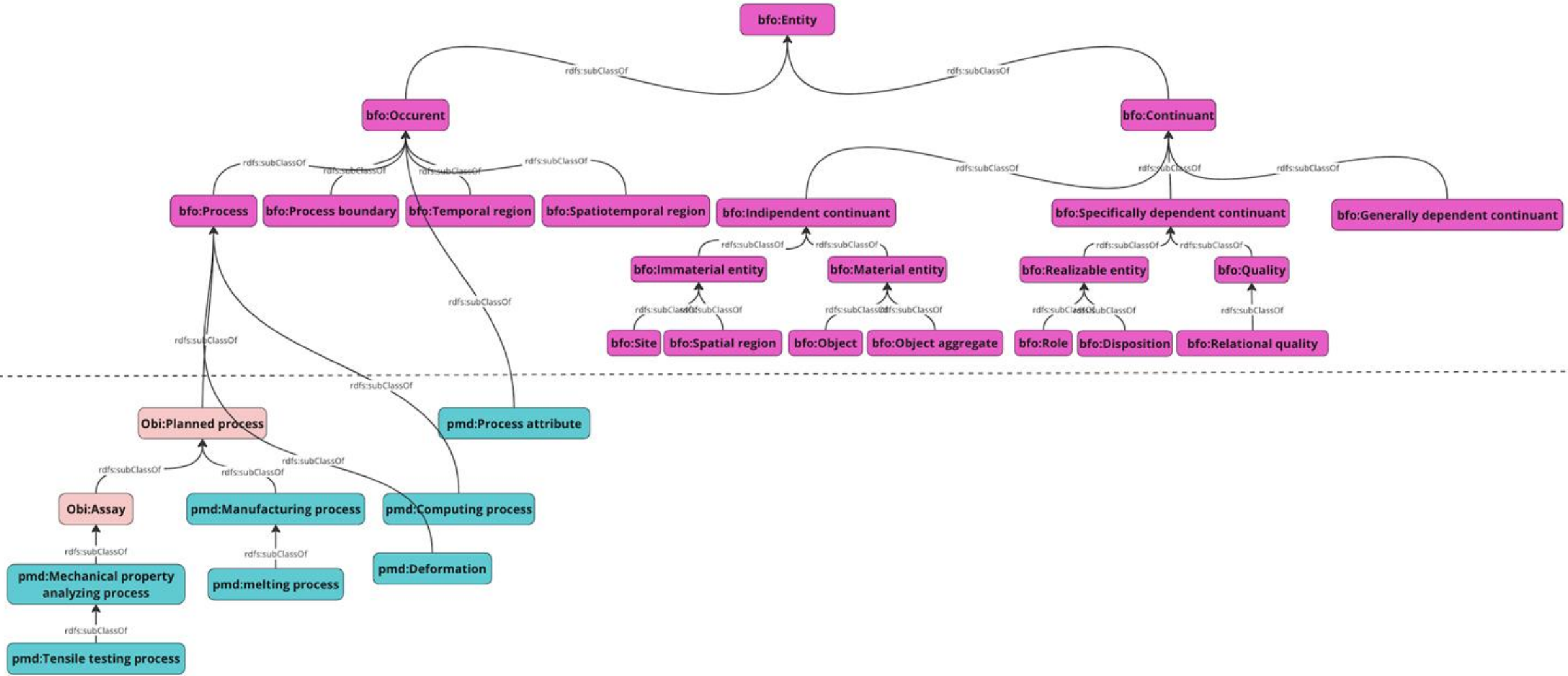
### Applications

- ✓ Material Specifications
- ✓ Process Chains
- ✓ Data Integration and Analysis
- ✓ Device and Function Modeling

# 4- Platform MaterialDigital Core Ontology (PMDco) classes

TOP-Level: BFO

MID-Level: PMD



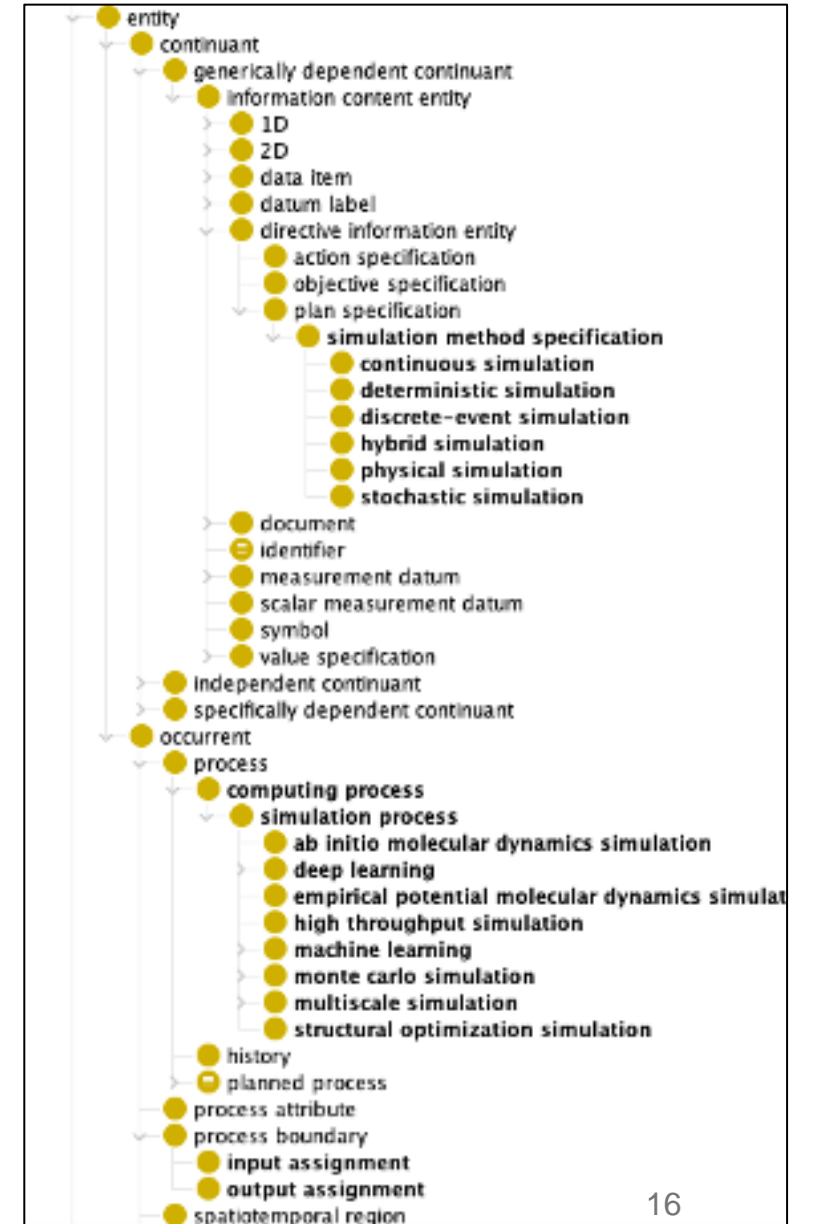
## Manufacturing module



## Material characterization module

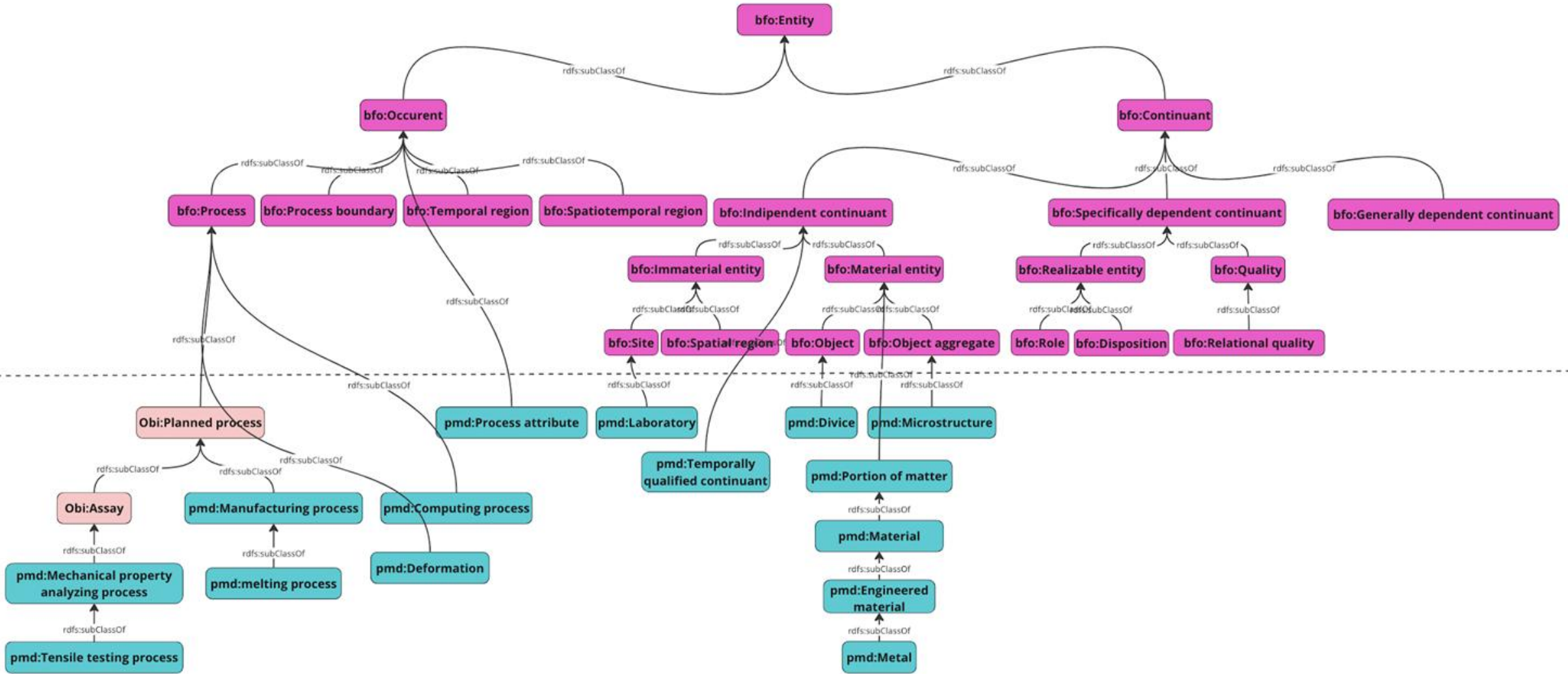


## Data transformation module

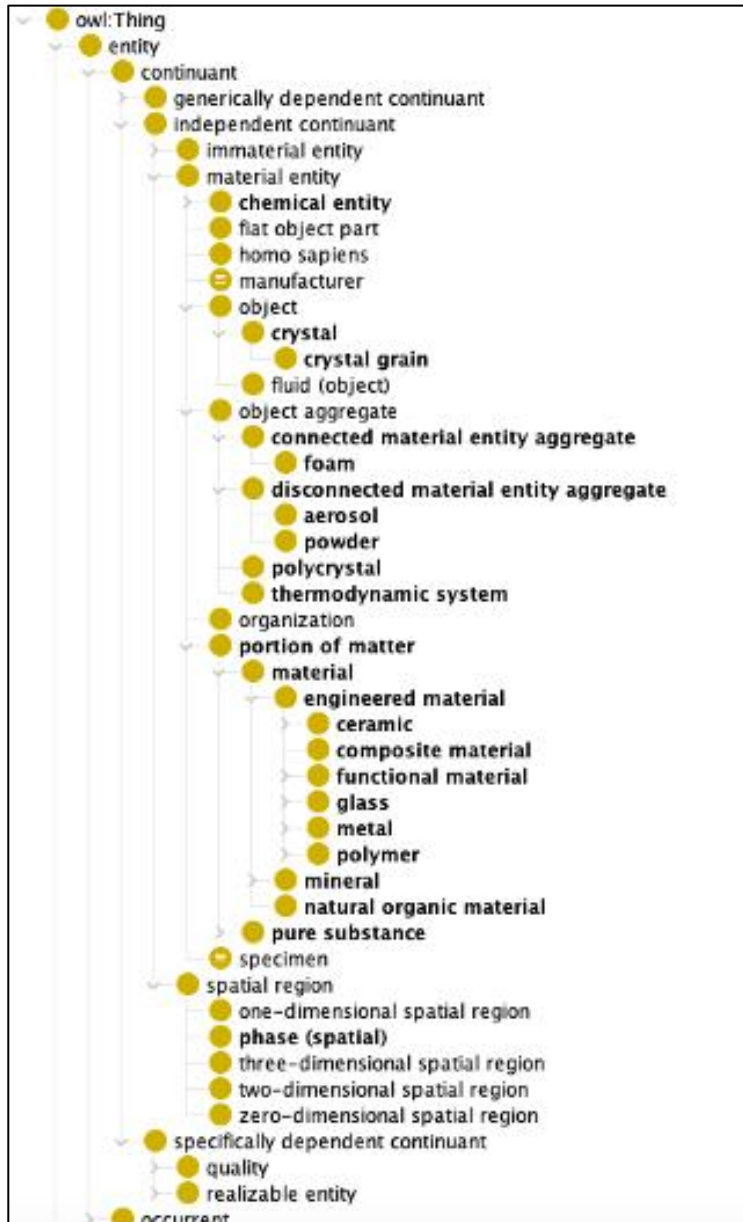


# 4- Platform MaterialDigital Core Ontology (PMDco) classes

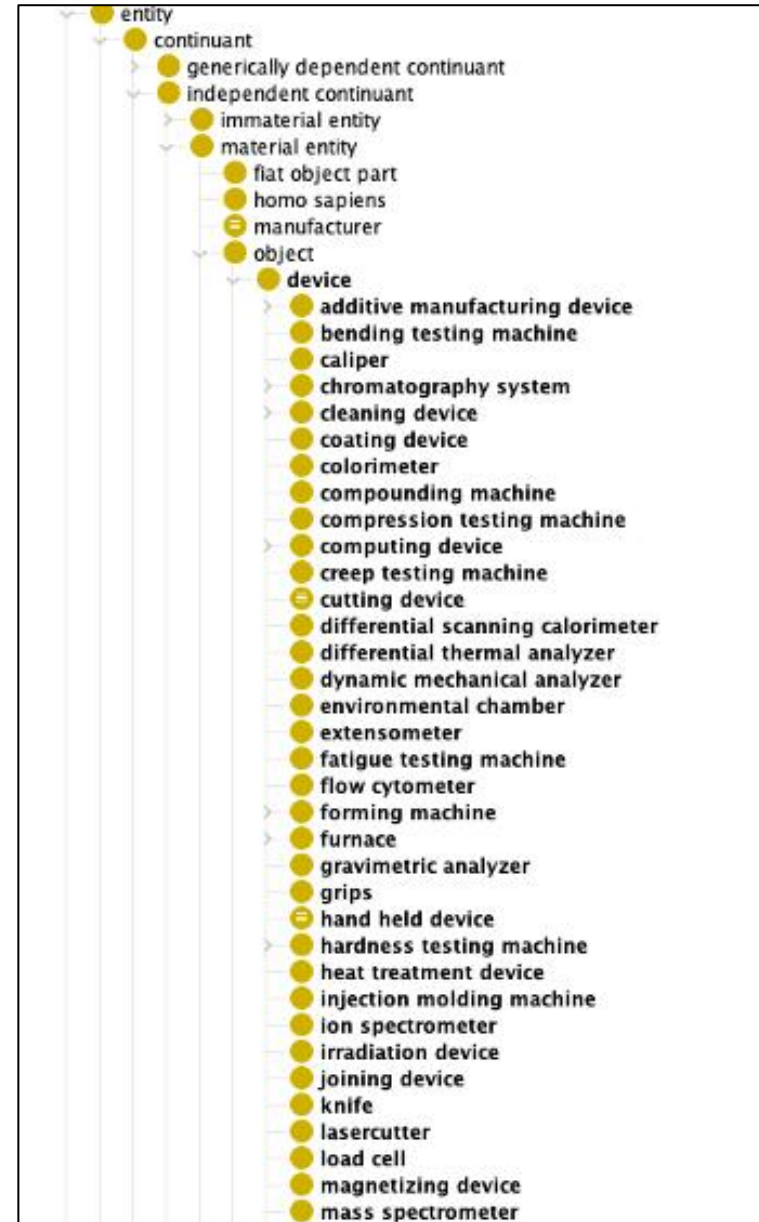
TOP-Level: BFO  
MID-Level: PMD



## Materials module

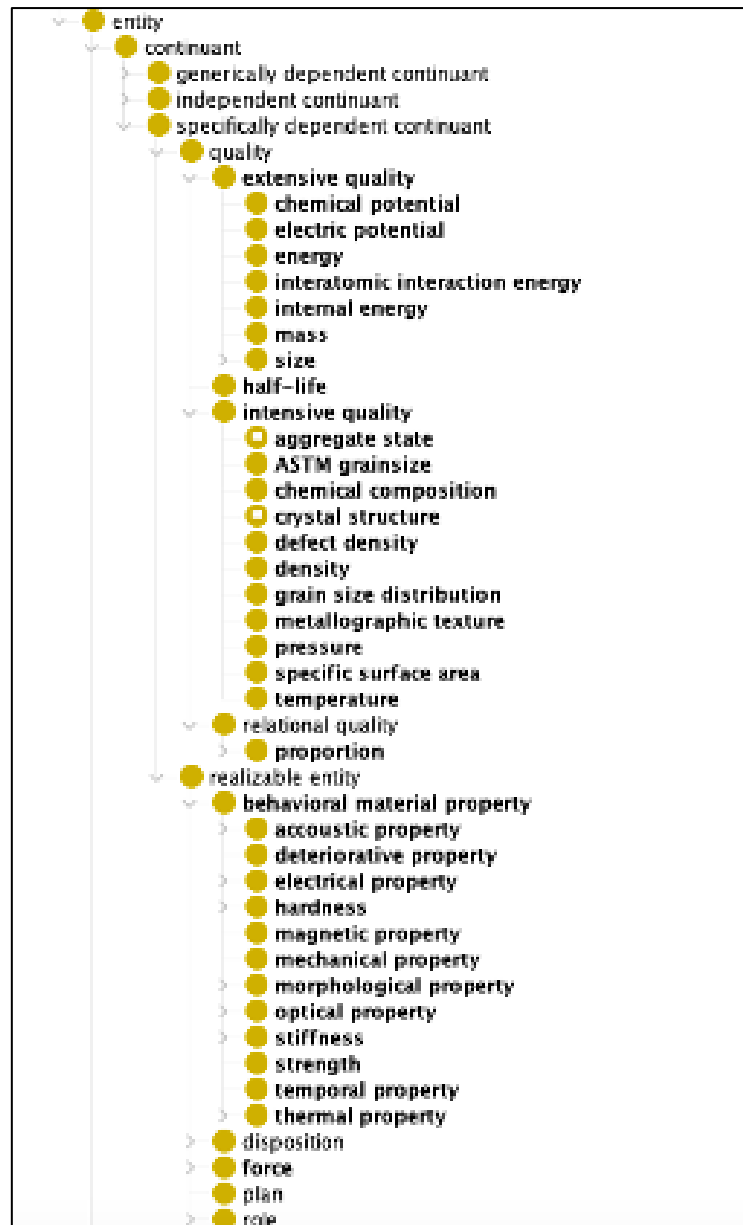


## Devices module





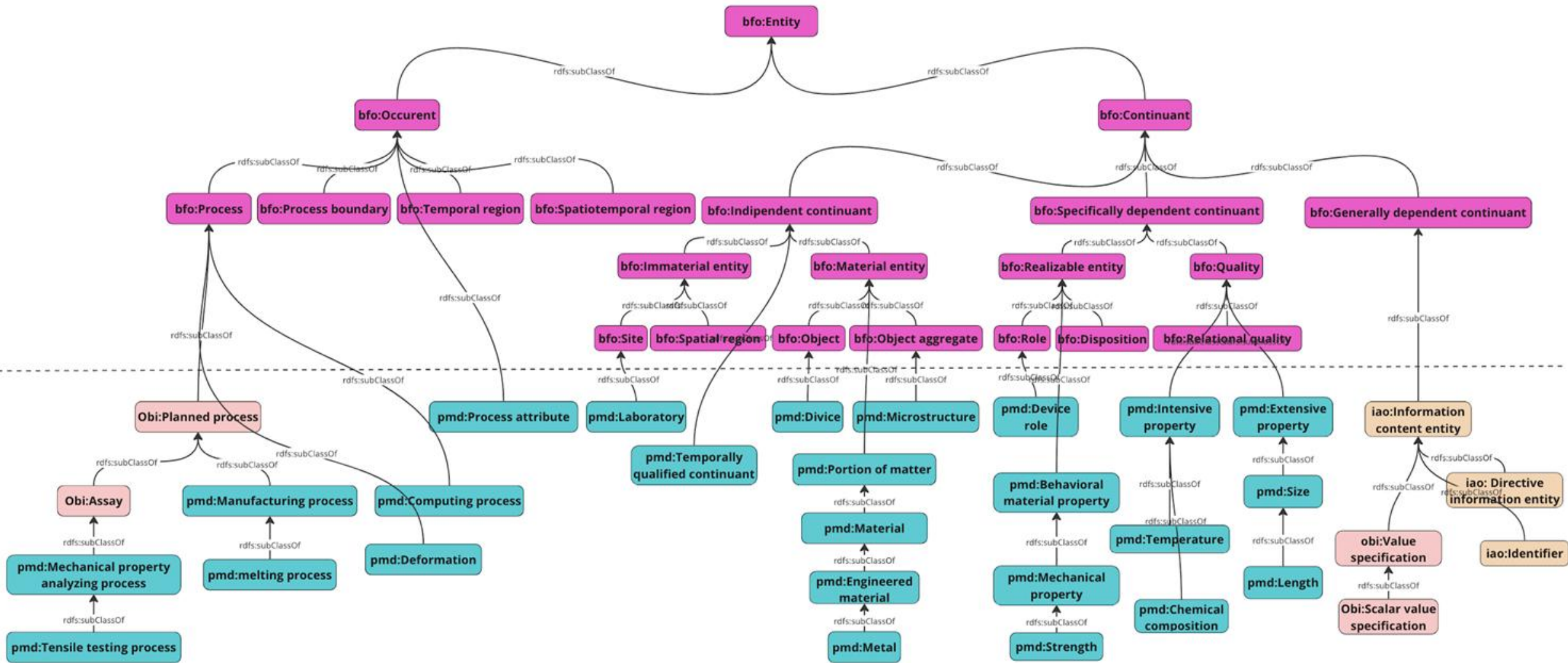
# Qualities module



# 4- Platform MaterialDigital Core Ontology (PMDco) classes

TOP-Level: BFO

MID-Level: PMD





# Workshop example: "High temperature tensile testing ontology" by reusing PMDco

## Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))

*\* This tutorial designed based on PMDco v3.0.0.rc1*



[Miro Board](#)

### Platform MaterialDigital Core Ontology (PMDco) Workshop

This repository provides PMDco workshop materials, featuring the HTTTO (High Temperature Tensile Test Ontology) as a hands-on use case. HTTTO is an application-level ontology built on top of the Platform MaterialDigital Core Ontology (PMDco) and is designed to support workshops on [How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices](#)

#### Workshop materials

- Workshop slides [here](#)
- Tutorials 1 and 2: [Miro Board](#)
- Tutorial 3: [GitHub ODK template](#)
- Tutorial 4: [GitHub PMDco Workshop](#) (this repository) and [http://ttt.ttl](#)



#### PMDco documentation

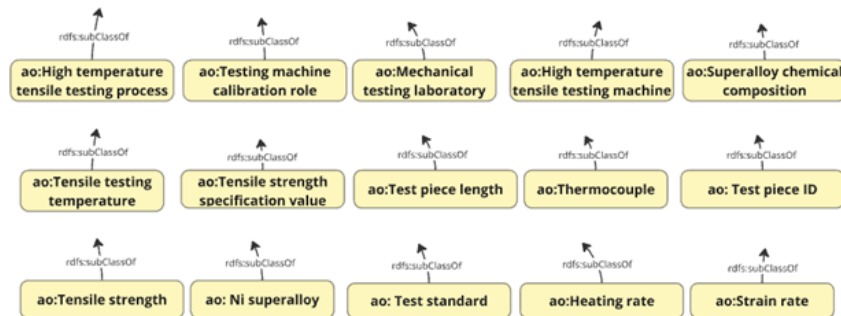
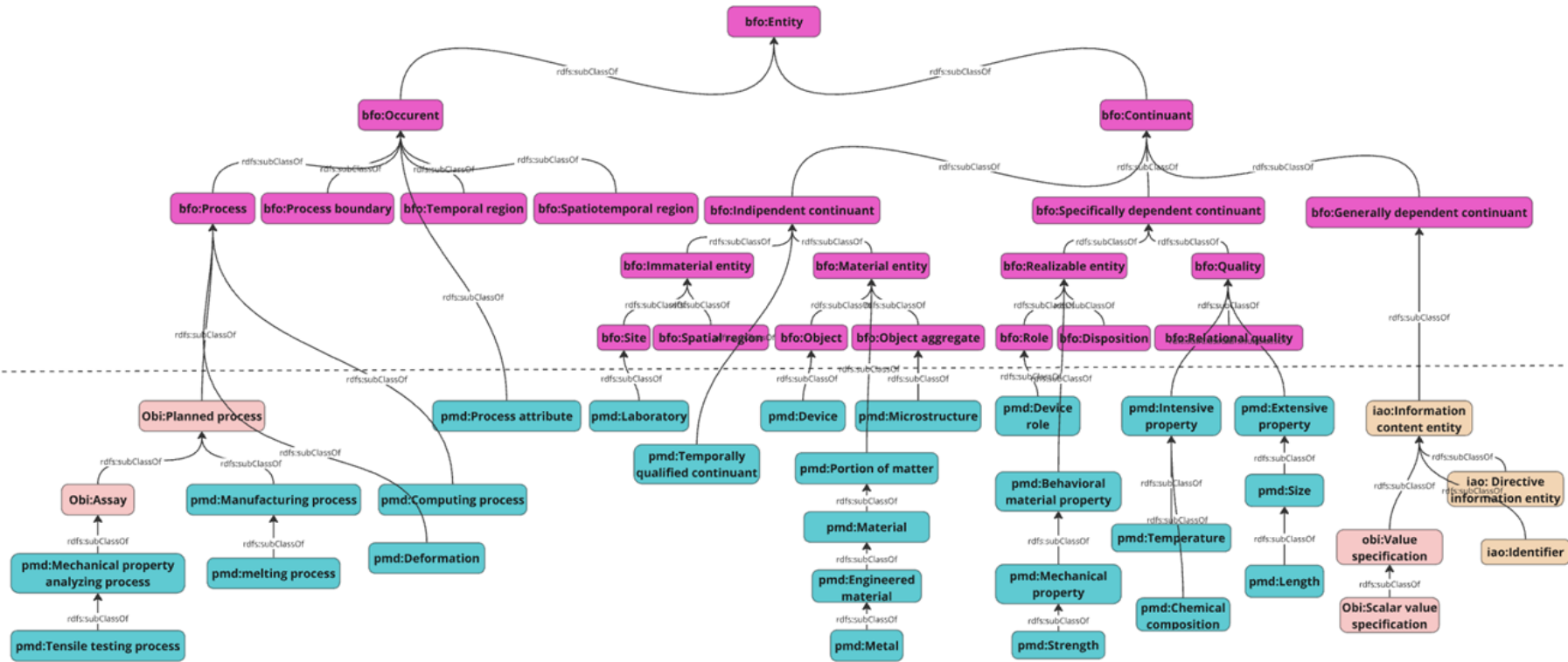
- [PMDco GitHub repository](#)
- [PMDco documentation page](#) for detailed information on the ontology structure, beginner learning materials, and a complete user guide.
- [PMDco list of classes and properties in Widoco](#)
- [PMDco ontology design patterns](#)
- [PMDco-related publications](#)

# Tutorial 1: Structure given classes according to PMDCo hierarchy

TOP-Level: BFO

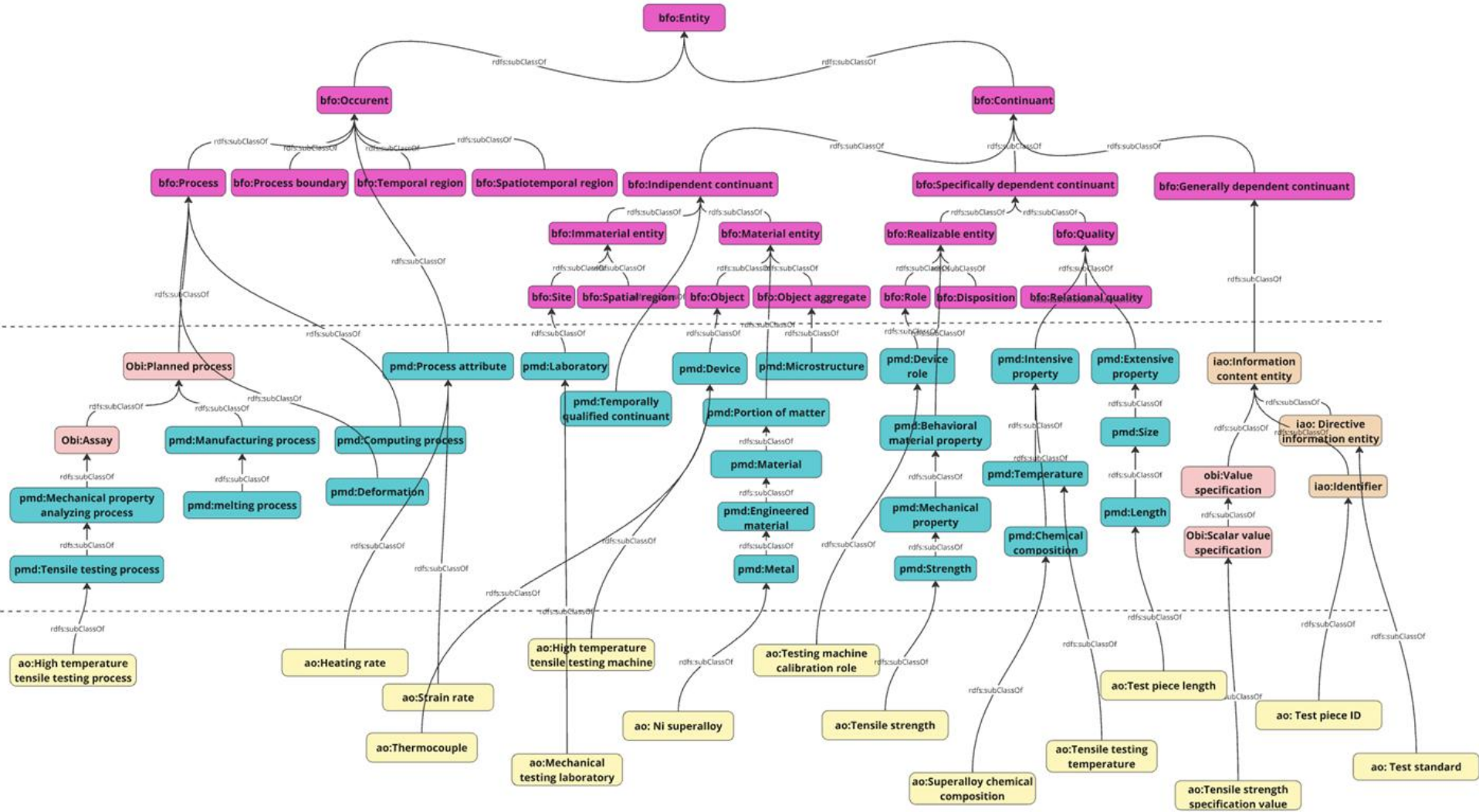
MID-Level: PMD

Application-Level: AO



# Tutorial 1: Answer

Application-Level: AO MID-Level: PMD TOP-Level: BFO



# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

1- Ontology development and beginners learning materials

2- Ontology levels

3- Basic Formal Ontology (BFO) classes

4- Platform MaterialDigital Core Ontology (PMDco) classes

**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**

**5- Platform MaterialDigital Core Ontology (PMDco) object properties**

**Tutorial 2: Using appropriate object properties ([Miro board](#))**

6- How to develop your application ontologies using PMDco and OBO+ODK best practices?

7- Ontology Development Kit (ODK)

**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**

8- Collaborative ontology development workflow, adding taxonomy and axioms

9- PMDco Ontology Design Patterns (ODPs)

**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**

10- Ontology evaluation, release, documentation and maintenance

# 5- Platform MaterialDigital Core Ontology (PMDco) object properties

Object property hierarchy: owl:topObjectProperty [?] [I] [E] [M] [X]

Asserted [v]

- owl:topObjectProperty
  - achieves\_planned\_objective
  - assay measures characteristic
  - characteristic measured by assay
  - characteristic of
    - disposition of
    - function of
    - quality of
    - role of
  - concretizes
  - denoted by
  - derives from
  - derives into
  - environs
  - executes
  - exists at
  - first instant of
  - has characteristic
    - has disposition
    - has function
    - has quality
    - has role
  - has first instant
  - has history
  - has last instant
  - has part
    - consists of
    - ends with
    - has measurement unit label
    - has member
    - has occurrent part
      - has proper occurrent part
      - has temporal part
        - has proper temporal part
    - has value specification
    - starts with
  - has participant
    - has input
    - has output
    - has specified input

Object property hierarchy: owl:topObjectProperty [?] [I] [E] [M] [X]

Asserted [v]

- has specified output
- has realization
  - stimulated by
- has relational quality
- has state
- history of
- IAO\_0000413
- in response to
- intensive bearer of
- interacts with
  - causally influences
- is about
  - denotes
  - is quality measurement of
  - is quality specification of
  - specifies value of
- is concretized as
- is quality measured as
- is state of
- is subject of
  - complies with
  - specified by value
- last instant of
- located in
- location of
- occupies spatiotemporal region
- occupies temporal region
- occurs in
- part of
  - ends
  - occurrent part of
    - proper occurrent part of
    - temporal part of
      - proper temporal part of
  - starts
- participates in
  - input of
  - is specified input of
  - is specified output of
  - output of

Annotations: PMD\_0000009 [?] [I] [E] [M] [X]

Annotations +

- label** [language: de] [in pmldco-shared] @ x o  
has process attribute
- skos:definition** [language: en] [in pmldco-shared] @ x o  
A relation between a process and a process attribute that depends on it.
- rdfs:seeAlso** [language: en] [in pmldco-shared] @ x o  
has process attribute from OEO [https://openenergyplatform.org/ontology/oeo/OEO\\_00000500](https://openenergyplatform.org/ontology/oeo/OEO_00000500)
- skos:example** [language: en] [in pmldco-shared] @ x o  
Tensile testing process has process attribute tensile rate

Characteristic [?] [I] [E] [M] [X] Description: PMD\_0000009 [?] [I] [E] [M] [X]

- Functional
- Inverse functional
- Transitive
- Symmetric
- Asymmetric
- Reflexive
- Irreflexive

Equivalent To +

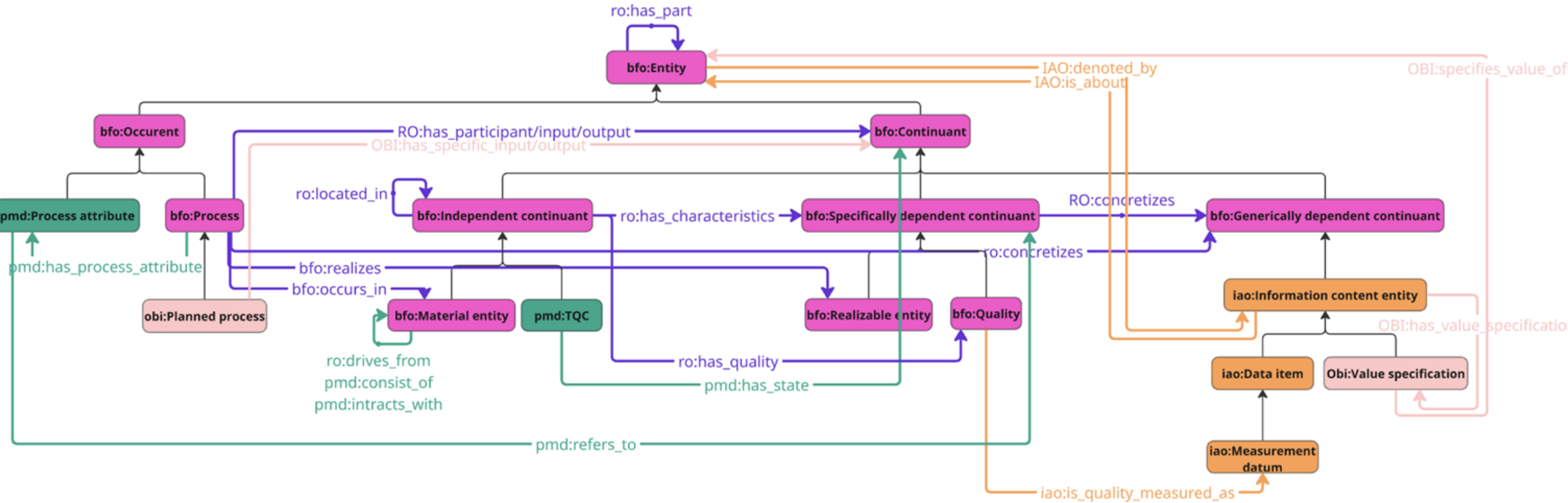
SubProperty Of +

Inverse Of +

Domains (intersection) +  
● process @ x o

Ranges (intersection) +  
● 'process attribute' @ x o

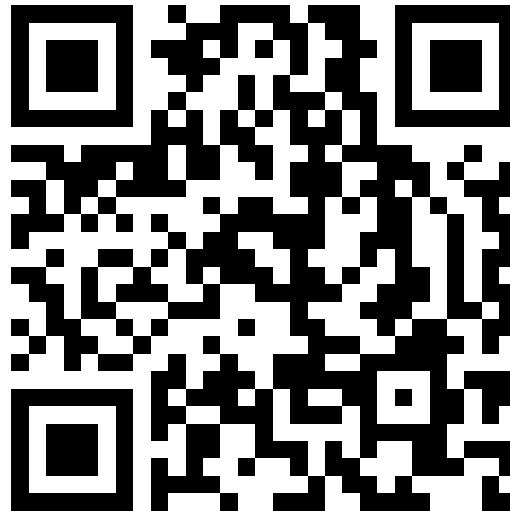
# 5- Platform MaterialDigital Core Ontology (PMDco) object properties



# Workshop example: "High temperature tensile testing ontology" by reusing PMDco

## Tutorial 2: Using appropriate object properties ([Miro board](#))

*\* This tutorial designed based on PMDco v3.0.0.rc1*



[Miro Board](#)

### Platform MaterialDigital Core Ontology (PMDco) Workshop

This repository provides PMDco workshop materials, featuring the HTTTO (High Temperature Tensile Test Ontology) as a hands-on use case. HTTTO is an application-level ontology built on top of the Platform MaterialDigital Core Ontology (PMDco) and is designed to support workshops on [How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices](#)

#### Workshop materials

- Workshop slides [here](#)
- Tutorials 1 and [2: Miro Board](#)
- Tutorial 3: [GitHub ODK template](#)
- Tutorial 4: [GitHub PMDco Workshop](#) (this repository) and [http://ttt.ttl](#)



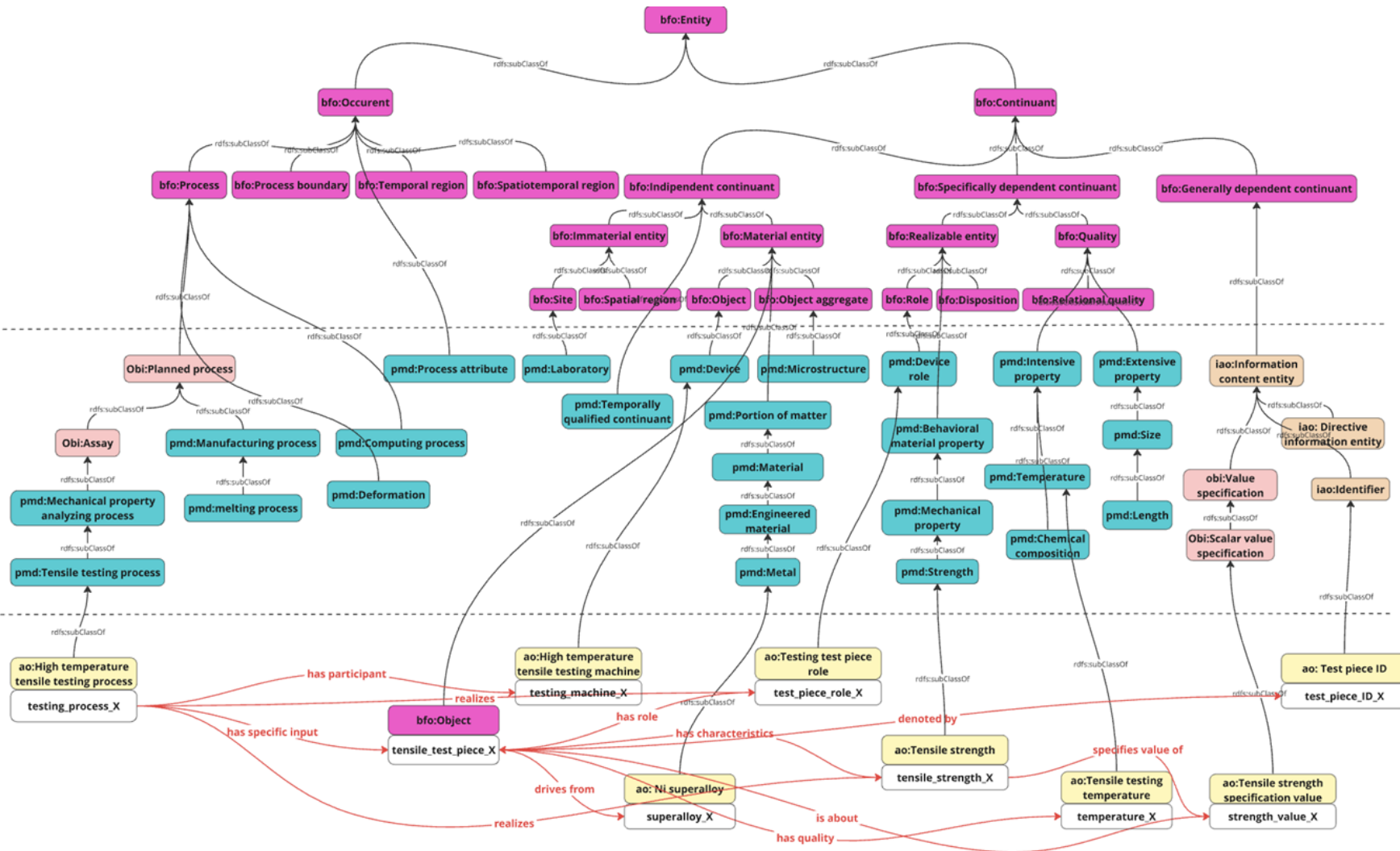
#### PMDco documentation

- [PMDco GitHub repository](#)
- [PMDco documentation page](#) for detailed information on the ontology structure, beginner learning materials, and a complete user guide.
- [PMDco list of classes and properties in Widoco](#)
- [PMDco ontology design patterns](#)
- [PMDco-related publications](#)



# Tutorial 2: Answer

Application-Level: AO MID-Level: PMD TOP-Level: BFO



# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

1- Ontology development and beginners learning materials

2- Ontology levels

3- Basic Formal Ontology (BFO) classes

4- Platform MaterialDigital Core Ontology (PMDco) classes

**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**

5- Platform MaterialDigital Core Ontology (PMDco) object properties

**Tutorial 2: Using appropriate object properties ([Miro board](#))**

**6- How to develop your application ontologies using PMDco and OBO+ODK best practices?**

7- Ontology Development Kit (ODK)

**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**

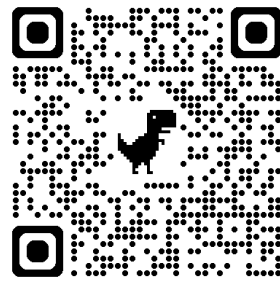
8- Collaborative ontology development workflow, adding taxonomy and axioms

9- PMDco Ontology Design Patterns (ODPs)

**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**

10- Ontology evaluation, release, documentation and maintenance

# 6- How to develop your application ontologies using pmdco and OBO+ODK best practices?



## ☰ PMDco User Guide

### ☰ How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow
- 8- Ontology release, updating, and versioning workflow
- 9- Ontology documentation and publishing

### ⊕ List of Ontologies Reusing PMDco V3.x.x:

Versions

Who We Are & How to Join

Publications

## PMDco User Guide

### How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

Although several guidelines and tutorials for ontology development are available (see *Ontology Learning Materials for Beginners*), we provide here a dedicated list of recommendations for creating application ontologies based on PMDco, making use of PMD guidance, ontology design patterns, OBO principles, and ODK features.

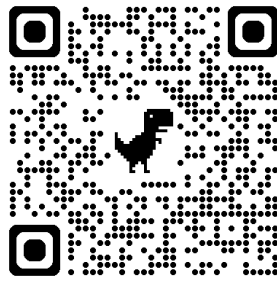
#### 1- Define the scope and domain of the ontology

- Clearly define the domain (e.g., materials testing, patient data, food production)
- Collaborating with the domain experts, specify the use cases or competency questions the ontology should address
- Determine the target audience (e.g., researchers, software agents, data engineers)
- Formulate a narrow and specific ontology scope according to [OBO principles](#).

#### 2- Decide on the level of detail and sources for terminology collection

- Determine whether the ontology will cover a broad area at a general level or focus on a specific topic in more detail,
- Identify reliable sources for terminology and concepts, such as books, scientific publications, domain experts, and relevant standards,
- Through such resources, identify the essential concepts within your domain,
- Provide accurate labels and clear definitions for each class (e.g., documented in a spreadsheet).

# 7- Ontology Development Kit (ODK)



## ☐ PMDco User Guide

### ☐ How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow
- 8- Ontology release, updating, and versioning workflow

## 3- Create an ODK-based ontology repository

- If you already know ODK, create a GitHub repository for ontology with a reasonable x.yaml file configuration and importing the latest PMDco version,
- If you don't know ODK or don't want to install it, we made a template for you. Simply create your repository following the instructions mentioned in "[application-ontology-template](#)". This template applies the same framework used for PMDco and mirrors PMDco with all its modules.



odk

ONTOLOGY DEVELOPMENT KIT

- Open-source toolkit and ontology life cycle management system for a standardized and transparent ontology development process.
- **Automation** of tasks like: import and release management, consistency checks, reasoning, and quality assurance.
- Supports us to comply to the **best-practices** of ontology development and avoid errors in imports and inconsistencies.
- **Modularized** approach through clearly separable components and modules.
- Uses technologies like ROBOT, Docker, GitHub Actions.
- Community-based tool with regular updates

# Workshop example: "High temperature tensile testing ontology" by reusing PMDco

## Tutorial 3: Creating ODK repository for PMDco application ontologies (GitHub)

### How to participate:

**Case A) I plan developing professional AO:**

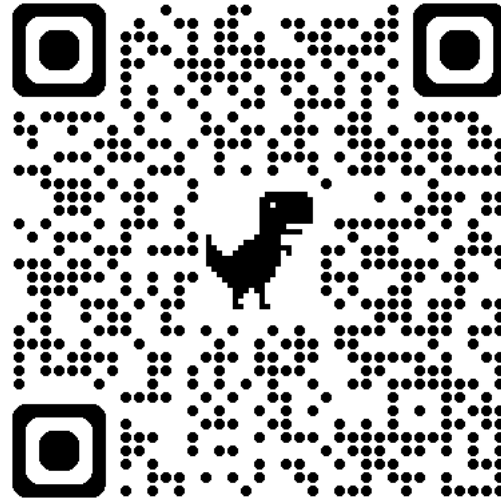
**We recommend learning ODK and use it for creating your repo. Now you can just watch how an ODK is looking like.**

**Case B) I want to test ODK, I may use it:**

**Participate in this tutorial and make your first ODK repo.**

**Case C) I don't plan developing AO and use ODK:**

**You can just watch how an ODK is looking like.**



[GitHub ODK template](#)

### Platform MaterialDigital Core Ontology (PMDco) Workshop

This repository provides PMDco workshop materials, featuring the HTTTO (High Temperature Tensile Test Ontology) as a hands-on use case. HTTTO is an application-level ontology built on top of the Platform MaterialDigital Core Ontology (PMDco) and is designed to support workshops on *How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices*

#### Workshop materials

- Workshop slides [here](#)
- Tutorials 1 and 2: [Miro Board](#)
- Tutorial 3: [GitHub ODK template](#)
- Tutorial 4: [GitHub PMDco Workshop](#) (this repository) and [http://ttt](#)

#### PMDco documentation

- [PMDco GitHub repository](#)
- [PMDco documentation page](#) for detailed information on the ontology structure, beginner learning materials, and a complete user guide.
- [PMDco list of classes and properties in Widoco](#)
- [PMDco ontology design patterns](#)
- [PMDco-related publications](#)



# Tutorial 3: Creating ODK repository for PMDCo application ontologies (GitHub)

1

2

3

4

**application-ontology-template** Public template Watch 0 Fork 5 Star 0 Use this template

main Go to file Code About

ThHanke removed ao from uri schema cf03811 · 5 days ago 30 Commits

template repository for starting a PMDCo application ontology

**Create a new repository**  
Repositories contain a project's files and version history. Have a project elsewhere? [Import existing code from a file, a VCS, or a website.](#)  
Required fields are marked with an asterisk (\*).

**Start with a template**  
Templates pre-configure your repository with files. [materialdigital/application-ontology](#)

**Include all branches**  
If enabled, all branches from the template repository will be included.

**1 General**

**Owner \*** HosseinBeygiNasrabadi

**Repository name \*** PMDCo workshop  
Your new repository will be created as PMDCo-workshop. The repository name can only contain ASCII letters, numbers, and underscores.

Great repository names are short and memorable. How about [symmetrical-broccoli?](#)

**Description**  
0 / 350 characters

**2 Configuration**

**Choose visibility \***  
Choose who can see and commit to this repository

Public

Create repository

**PMDco-workshop** Public  
generated from [materialdigital/application-ontology-template](#)

main Go to file Code About

HosseinBeygiNasrabadi Initial commit bd33f8e · 12 minutes ago 1 Commit

**Files:**

- .github/workflows Initial commit 12 minutes ago
- .gitignore Initial commit 12 minutes ago
- LICENSE Initial commit 12 minutes ago
- README.md Initial commit 12 minutes ago
- seed-template.yaml Initial commit 12 minutes ago

**README** Apache-2.0 license

**application-ontology-template**

template repository for starting a PMDCo application ontology. it comes preconfigured with github workflows using [ontology development kit](#).

**About**

No description, website, or topics provided.

Readme

Apache-2.0 license

Activity

0 stars

0 watching

0 forks

**Releases**

No releases published  
[Create a new release](#)

**Packages**

No packages published  
[Publish your first package](#)

Code Issues Pull requests **Actions** Projects Wiki Security Insights Settings

### Actions

New workflow

All workflows

- build
- create widoco documentation
- seed odk**

Management

- Caches
- Attestations
- Runners
- Usage metrics
- Performance metrics

#### seed odk

Filter workflow runs

Help us improve GitHub Actions

0 workflow runs

This workflow has a workflow\_dispatch event trigger.

Use workflow from

Branch: main

Ontology ID (replaces id in seed.yaml) \*

htto

URI base suffix (replaces uribase\_suffix in seed.yaml) \*

htto

ODK Docker tag to use

latest

Configuration file name

seed-template.yaml

Run workflow

**5**

This workflow has

### ODK Seed: htto #1

Open github-actions wants to merge 1 commit into main from fresh-seed-20251122-200221

Conversation 0 Commits 1 Checks 0 Files changed 41

github-actions bot commented 2 minutes ago

#### ODK Seed Generated Files

- Ontology ID: htto
- URI Base Suffix: htto
- GitHub Organization: HosseinBeygiNasrabadi
- Repository: PMDco-workshop
- Config File: seed-template.yaml
- ODK Tag: latest

This PR contains the generated ontology files from ODK seed.

ODK seed generated files for htto using seed-template.yaml

No conflicts with base branch

Merge pull request

**7**

3 workflow runs

seed odk

seed odk #1: Manually run by HosseinBeygiNasrabadi

main

2 minutes ago

1m 31s

create widoco documentation

create widoco documentation #1: completed by HosseinBeygiNasrabadi

15 minutes ago

7s

**6**

### PMDco-workshop

generated from materialdigital/application-ontology-template

main

Go to file

Code

About

No description, website, or topics provided.

- Readme
- Apache-2.0 license
- Contributing
- Activity
- 0 stars
- 0 watching
- 0 forks

#### Releases

No releases published

Create a new release

#### Files

.github/workflows	Initial commit	19 minutes ago
src	ODK seed generated files for htto usi...	3 minutes ago
.gitignore	ODK seed generated files for htto usi...	3 minutes ago
CONTRIBUTING.md	ODK seed generated files for htto usi...	3 minutes ago
LICENSE	Initial commit	19 minutes ago
README.md	ODK seed generated files for htto usi...	3 minutes ago
htto.ttl	ODK seed generated files for htto usi...	3 minutes ago
issue_template.md	ODK seed generated files for htto usi...	3 minutes ago
seed-template.yaml	ODK seed generated files for htto usi...	3 minutes ago

**8**

# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

1- Ontology development and beginners learning materials

2- Ontology levels

3- Basic Formal Ontology (BFO) classes

4- Platform MaterialDigital Core Ontology (PMDco) classes

**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**

5- Platform MaterialDigital Core Ontology (PMDco) object properties

**Tutorial 2: Using appropriate object properties ([Miro board](#))**

6- How to develop your application ontologies using PMDco and OBO+ODK best practices?

7- Ontology Development Kit (ODK)

**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**

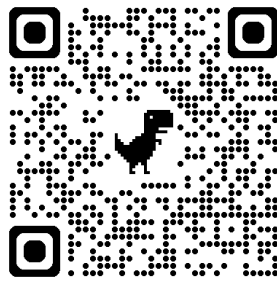
**8- Collaborative ontology development workflow, adding taxonomy and axioms**

9- PMDco Ontology Design Patterns (ODPs)

**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**

10- Ontology evaluation, release, documentation and maintenance

# 8- Collaborative ontology development workflow, adding taxonomy and axioms



## PMDCo User Guide

### How to Develop Your Application Ontologies Using PMDCo and OBO+ODK Best Practices

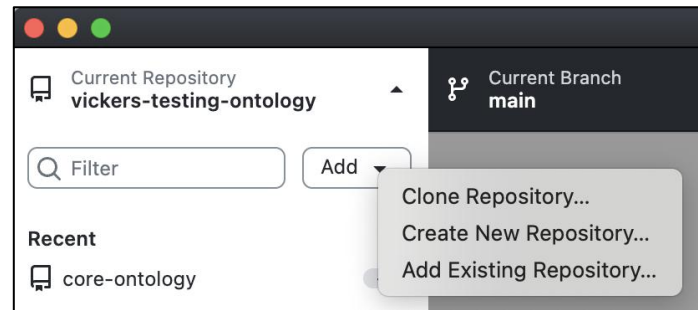
- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow
- 8- Ontology release, updating, and versioning workflow
- 9- Ontology documentation and publishing

### List of Ontologies Reusing PMDCo V3.x.x:

- Versions
- Who We Are & How to Join
- Publications
- Acknowledgements

## 4- Collaborative ontology development workflow

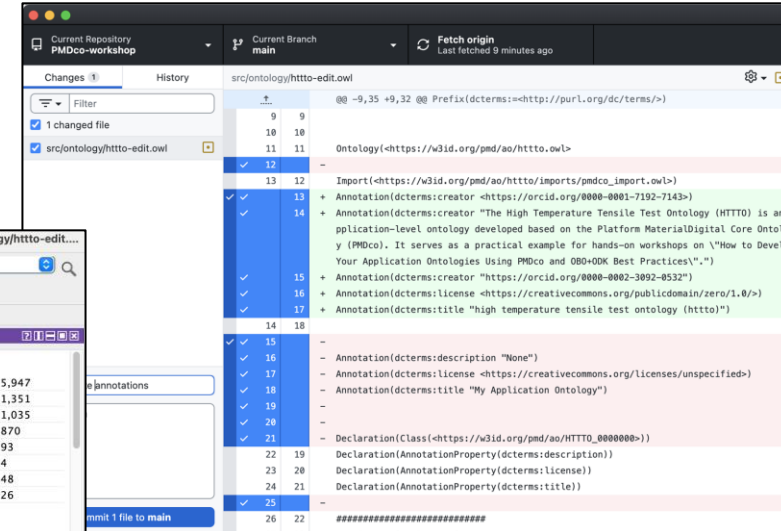
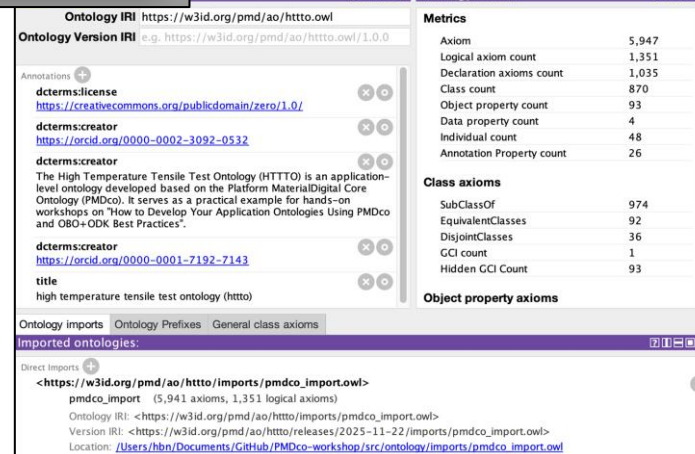
- Any person who wants to contribute to the collaborative development of ontology should clone the ODK repository to the local machine,
- In Protégé, open the editable file: `src/ontology/*-edit.owl` (Any editor can manage an assigned ID range),
- Edit the ontology, like add classes/properties, update definitions, and organize axioms,
- Save and validate locally,
- Commit your edits and push them to your branch,
- Create a Pull Request, submit a PR with a short summary and justification of your changes,
- Team members review; after approval, the PR is merged.



### 1- GitHub desktop: clone the repository

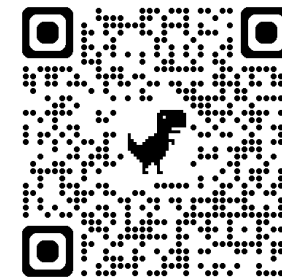


### 2- Protege: src/ontology/httpo-edit.owl Edit Ontology



### 3- GitHub desktop: commit and push

# 8- Collaborative ontology development workflow, adding taxonomy and axioms



## ☐ PMDco User Guide

### ☐ How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow
- 8- Ontology release, updating, and versioning workflow
- 9- Ontology documentation and publishing

### ⊕ List of Ontologies Reusing PMDco V3.x.x:

Versions

Who We Are & How to Join

Publications

## 5- Add your desired taxonomy

- Edit ontology in protégé,
- Create new classes, properties, and individuals according to your terminology lists,
- Structure them hierarchically in alignment with reused top- and mid-level ontologies, where subclasses inherit properties from their superclasses,
- Evaluate other OBO Foundry ontologies (e.g., using [Ontobee](#)) to check for existing classes and to guide your classification. In the case you want to import entities from other ontologies, please follow the instructions [here](#),
- Following [OBO principles](#), build the ontology primarily by extending the class hierarchy and minimizing the introduction of new object properties.
- Add rich annotations for new entities (such as definitions, references, examples, and editorial notes).
- Include comprehensive ontology annotation metadata (e.g., title, label, description, creators, contributors, version info, previous versions, license, and citation).

## 6- Define a richer semantic model with multiple relationship types, rules, and constraints

- Define logical axioms and property constraints (e.g., domain and range),
- Use graphical tools (e.g., Ontopanel or Miro) to visually discuss the modeling issues,
- Use our [Ontology Design Patterns](#) as a guideline for the semantic representation of your concepts,
- See other PMDco-based application ontologies (listed below this page) for similar modeling concepts.
- If you still need support for specific modelling issues, please feel free to mention it in our GitHub [issues](#) or [discussion forum](#).

# 9- PMDco Ontology Design Patterns (ODPs)

<https://github.com/materialdigital/core-ontology/tree/main/patterns>



## Patterns

### Pattern 1 - Temporal Region

Pattern 2 - Process Chain

Pattern 3 - Process Inputs and Outputs

Pattern 4 - Realizable Entities (Role)

Pattern 5 - Realizable Entities (Qualities)

Pattern 6 - Scalar Measurement

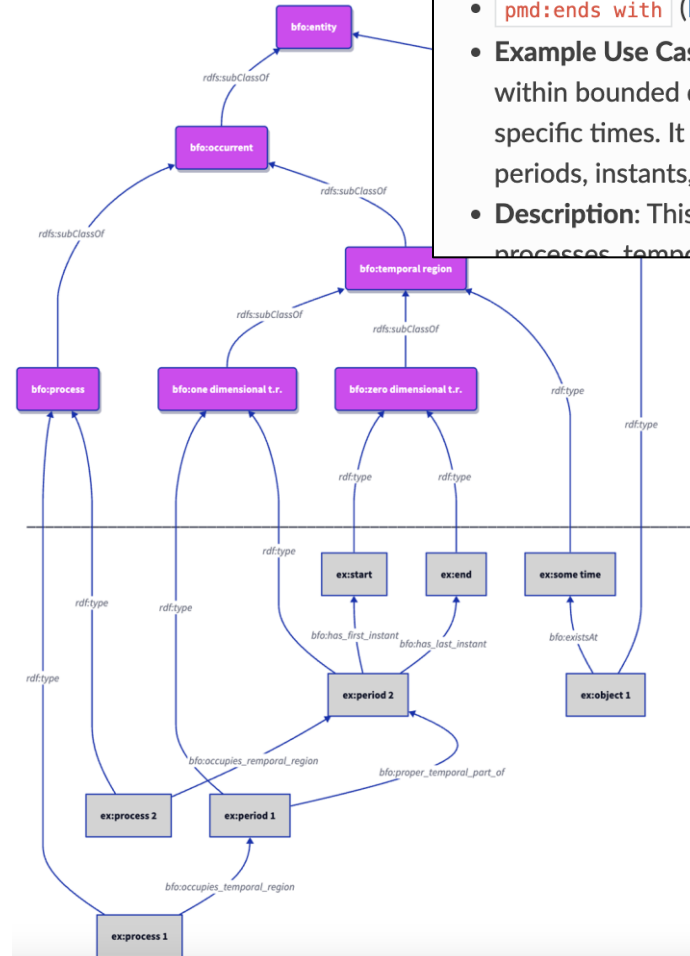
Pattern 7 - Scalar Value Specification

Pattern 8 - Categorical Value Specification

Pattern 9 - Material and Device Specification

## Pattern 1 - Temporal Region

- **Purpose:** Specifying the boundaries of a process on the time axis.
- **Core Properties:**
  - `bfo:occupies temporal region` (BFO\_0000199)
  - `bfo:proper temporal part of` (BFO\_0000136)
  - `bfo:has first instant` (BFO\_0000222)
  - `bfo:has last instant` (BFO\_0000224)
  - `pmd:ends with` (PMD\_0060003)
- **Example Use Case:** the pattern defines a temporal ontology where processes unfold within bounded durations, the specific times. It provides a duration, periods, instants, and enduring processes, temporal regions.
- **Description:** This graph describes the relationships between various temporal entities.



```
@prefix : <https://w3id.org/pmd/co/test#> .
@prefix ex: <https://www.example.org/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <https://w3id.org/pmd/co/test#> .
```

```
@prefix process: <http://purl.obolibrary.org/obo/BFO_0000015> . #Class
@prefix continuant: <http://purl.obolibrary.org/obo/BFO_0000002> . #Class
@prefix temporal_region: <http://purl.obolibrary.org/obo/BFO_0000008> . #Class
@prefix one-dimensional_temporal_region: <http://purl.obolibrary.org/obo/BFO_0000038> . #Class
@prefix zero-dimensional_temporal_region: <http://purl.obolibrary.org/obo/BFO_0000148> . #Class
```

```
@prefix occupies_temporal_region: <http://purl.obolibrary.org/obo/BFO_0000199> . #ObjectProperty
@prefix proper_temporal_part_of: <http://purl.obolibrary.org/obo/BFO_0000136> . #ObjectProperty
@prefix has_first_instant: <http://purl.obolibrary.org/obo/BFO_0000222> . #ObjectProperty
@prefix has_last_instant: <http://purl.obolibrary.org/obo/BFO_0000224> . #ObjectProperty
@prefix exists_at: <http://purl.obolibrary.org/obo/BFO_0000108> . #ObjectProperty
```

```
<https://w3id.org/pmd/co/test/shape/temporal_region> rdf:type owl:Ontology .
```

```
ex:process_1 a process: .
ex:process_2 a process: .
ex:period_1 a one-dimensional_temporal_region: .
ex:period_2 a one-dimensional_temporal_region: .
ex:some_time a temporal_region: .
ex:object_1 a continuant: .
ex:start a zero-dimensional_temporal_region: .
ex:end a zero-dimensional_temporal_region: .
```

```
ex:process_1 occupies_temporal_region: ex:period_1 .
ex:process_2 occupies_temporal_region: ex:period_2 .
ex:period_2 has_first_instant: ex:start .
ex:period_2 has_last_instant: ex:end .
ex:period_1 proper_temporal_part_of: ex:period_2 .
ex:object_1 exists_at: ex:some_time .
```

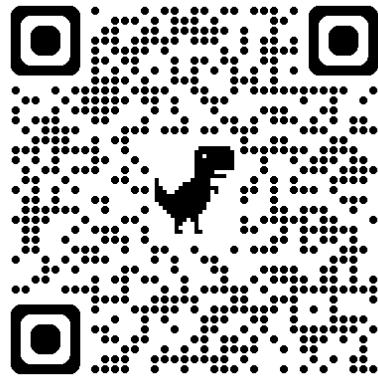
# Workshop example: "High temperature tensile testing ontology" by reusing PMDco

## Tutorial 4: Ontology editing; adding classes, annotations and axioms (Protege)

### How to participate:

#### Option A (commit and push possibility):

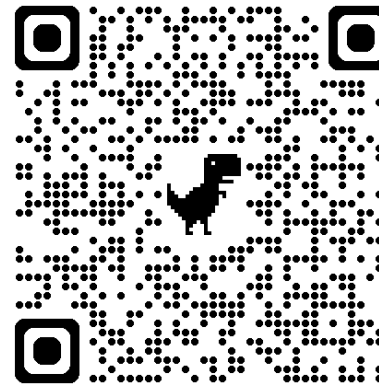
- 1- GitHub desktop: clone the "PMDco Workshop" repository (QR code link below)
- 2- Protégé: open file > src/ontology/httto-edit.owl



[GitHub PMDco Workshop](#)

#### Option B (faster, just for participating in tutorial):

- 1- Protégé: open from URL > QR code link below)



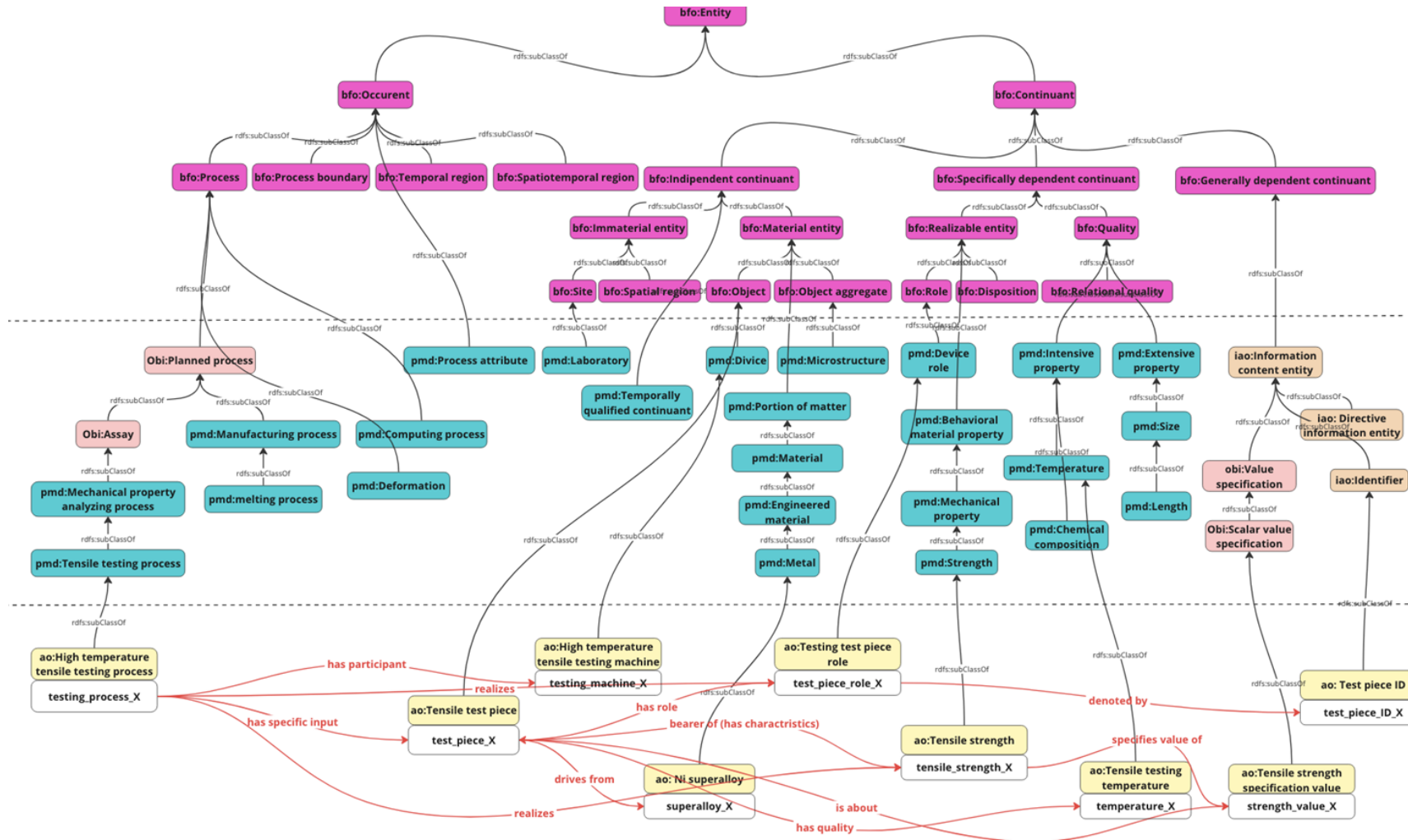
[htto.ttl](http://htto.ttl)

#### Workshop materials

- Workshop slides [here](#)
- Tutorials 1 and 2: [Miro Board](#)
- Tutorial 3: [GitHub ODK template](#)
- Tutorial 4: [GitHub PMDco Workshop](#) (this repository) and [htto.ttl](http://htto.ttl)

# Tutorial 4: Ontology editing; adding classes, annotations and axioms (Protege)

- 1- Add ao classes (yellow ones) in their designed hierarchy
- 2- Create annotations (at least definitions) for your created classes
- 3- Create axioms (as shown in the graph) for your created classes



## Tutorial 4: Example adding classes, annotations, and axioms

The screenshot shows the Protégé ontology editor interface. The browser address bar displays the URL: `https://w3id.org/pmd/ao/httto.owl`. The breadcrumb trail indicates the current location: `> entity > occurrent > process > planned process > assay > mechanical property analyzing process > tensile testing process > high temperature tensile testing process`. The top navigation bar includes tabs for 'Active ontology', 'Entities', 'Individuals by class', and 'DL Query'. Below this, there are tabs for 'Annotation properties', 'Datatypes', and 'Individuals'. The main interface is divided into three main sections:

- Class hierarchy:** A tree view on the left showing the ontology structure. The class `high temperature tensile testing process` is selected and highlighted in blue. Other visible classes include `dynamic mechanical analysis process`, `fatigue testing process`, `fracture toughness testing process`, `hardness testing process`, `impact testing process`, `nanoindentation process`, `rheological property analyzing process`, `scratch testing process`, `shear testing process`, `spalling testing process`, `tensile testing process`, `thermomechanical analysis process`, `torsion testing process`, `wear testing process`, `optical property analyzing process`, `structural property analyzing process`, `thermal property analyzing process`, `identifier creating process`, `manufacturing process`, `publishing process`, `process chain`, `project`, `responding process`, `reversible process`, and `stimulating process`.
- Annotations:** A panel on the right showing the annotations for the selected class. It includes:
  - label [language: en]:** `high temperature tensile testing process`
  - skos:definition:** `High temperature tensile testing is a tensile testing testing process used to evaluate a material's strength, ductility, and deformation behavior under elevated temperatures.`
- Description:** A panel on the right showing the description of the class, which is a disjunctive definition:
  - Equivalent To:**
    - `'tensile testing process' and ( inverse ('participates in') some 'high temperature tensile test machine')`
    - `'tensile testing process' and ( inverse ('input of') some 'tensile test piece')`
    - `'tensile testing process' and ( realizes some 'tensile strength')`
    - `'tensile testing process' and ( realizes some 'tensile test piece role')`
  - SubClass Of:**
    - `'tensile testing process'`

# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

1- Ontology development and beginners learning materials

2- Ontology levels

3- Basic Formal Ontology (BFO) classes

4- Platform MaterialDigital Core Ontology (PMDco) classes

**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**

5- Platform MaterialDigital Core Ontology (PMDco) object properties

**Tutorial 2: Using appropriate object properties ([Miro board](#))**

6- How to develop your application ontologies using PMDco and OBO+ODK best practices?

7- Ontology Development Kit (ODK)

**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**

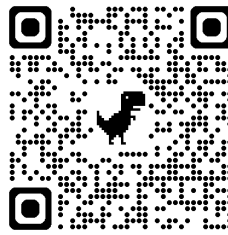
8- Collaborative ontology development workflow, adding taxonomy and axioms

9- PMDco Ontology Design Patterns (ODPs)

**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**

10- Ontology evaluation, release, documentation and maintenance

# 10- Ontology evaluation, release, documentation and maintenance



## ☐ PMDco User Guide

### ☐ How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow**
- 8- Ontology release, updating, and versioning workflow
- 9- Ontology documentation and publishing

### ☐ List of Ontologies Reusing PMDco V3.x.x:

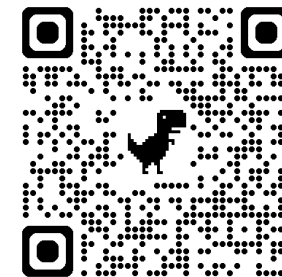
## 7- Ontology evaluation and testing workflow

- Plan regular meetings with both ontology and MSE domain experts in your project to assess the ontology's structure, resolve modeling issues, and ensure its logical correctness,
- Use ODK evaluation and testing workflows through GitHub Actions Continuous Integration to validate the ontology's structure, syntax, annotation completeness, and logical consistency,
- Use reasoning tools (e.g., reasoners Hermit and Pellet) to ensure logical and technical consistency and identify potential redundancies,
- Analyzes the ontology for common modeling pitfalls, structural issues, and best-practice violations using tools like OOPS! (Ontology Pitfall Scanner!),
- We will be pleased to invite you to present your ontology in one of our "Ontology Playground meetings" (See Section Who We Are & How to Join)

## 8- Ontology release, updating, and versioning workflow

- Use ODK automated workflow for the release process, which follows these steps: running the release with ODK, reviewing the output, merging changes into the main branch, and creating a GitHub release.
- Track actively the GitHub issue to gather community feedback, recognize bugs, and identify necessary additions and refinements.
- Implement a curation process to assess and prioritize updates based on community needs.
- Employ the robust versioning system hosted on GitHub to ensure clear tracking of changes over time, management of previous versions, and clear tracking of the ontology's evolution over time.

# 10- Ontology evaluation, release, documentation and maintenance



## ☐ PMDco User Guide

### ☐ How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

- 1- Define the scope and domain of the ontology
- 2- Decide on the level of detail and sources for terminology collection
- 3- Create an ODK-based ontology repository
- 4- Collaborative ontology development workflow
- 5- Add your desired taxonomy
- 6- Define a richer semantic model with multiple relationship types, rules, and constraints
- 7- Ontology evaluation and testing workflow
- 8- Ontology release, updating, and versioning workflow
- 9- **Ontology documentation and publishing**

### ☐ List of Ontologies Reusing PMDco V3.x.x:

Versions

Who We Are & How to Join

Publications

Acknowledgements

## 9- Ontology documentation and publishing

- Create detailed documentation, including best practices, examples, and guidelines in your ontology GitHub repository.
- Use [Widoco template](#), a documentation generator for ontologies that automatically produces human-readable HTML documentation including metadata, hierarchies of classes, properties, and annotations,
- If interested, create [MkDocs](#) documentation (like this repository).
- If you are using “[application-ontology-template](#)”, your ontology is automatically documented in [obofoundry.org](#),
- To enhance your ontology accessibility, please also publish your ontology in [MatPortal](#) and [PMD DataPortal](#).

## List of Ontologies Reusing PMDco V3.x.x:

- [Logistic Application Ontology \(LOG\)](#)
- PMDco application ontology for logistics and supply chain adopted from iof-supplychain-module
- [Vickers Testing Ontology \(VTO\)](#)
- An Ontology for representing the Vickers testing process, testing equipment requirements, test pieces characteristics, and related testing parameters and their measurement procedure according to DIN EN ISO 6507-1 standard.
- [Tensile testing Ontology \(TTO\)](#)
- An ontology for representing tensile testing of metals at room temperature in accordance with the associated testing standard ISO 6892-1:2019-11.
- [Heat Treatment Ontology \(HTO\)](#)
- An application ontology of PMDco to model heat treatment processes with a focus on metals.

# How to Develop Your Application Ontologies Using PMDco and OBO+ODK Best Practices

## Workshop content (10 topics, 4 tutorials):

1- Ontology development and beginners learning materials

2- Ontology levels

3- Basic Formal Ontology (BFO) classes

4- Platform MaterialDigital Core Ontology (PMDco) classes

**Tutorial 1: Structure given classes according to PMDco hierarchy ([Miro board](#))**

5- Platform MaterialDigital Core Ontology (PMDco) object properties

**Tutorial 2: Using appropriate object properties ([Miro board](#))**

6- How to develop your application ontologies using PMDco and OBO+ODK best practices?

7- Ontology Development Kit (ODK)

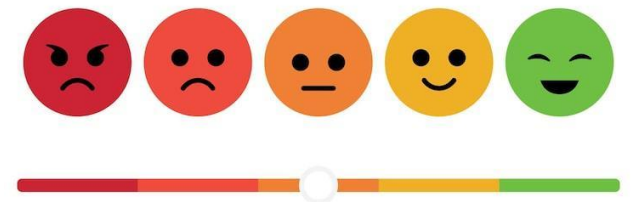
**Tutorial 3: Creating ODK repository for PMDco application ontologies ([GitHub](#))**

8- Collaborative ontology development workflow, adding taxonomy and axioms

9- PMDco Ontology Design Patterns (ODPs)

**Tutorial 4: Ontology editing; adding classes, annotations and axioms ([Protege](#))**

10- Ontology evaluation, release, documentation and maintenance



# Invitation to PMD Ontology Playground Meetings

## Bi-weekly public sessions

Fridays, 1-2 pm

## Audience

Ontology practitioners & MSE domain experts

## Key activities

- Knowledge transfer
- Experience exchange
- Modeling challenges – User insights
- Collaborative PMDco enhancement

## Participate in our PMD Playground Meetings

Please register via our [mailing list](#).



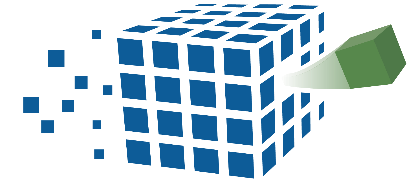
Check out our YouTube channel

YouTube



The screenshot shows a YouTube channel page for 'Show & Tell' on the 'Plattform MaterialDigital' channel. The channel has 10 videos and 74 views, last updated on 16.04.2024. The video list includes:

Video Title	Views	Time
KupferDigital_Miriam Eisenbart, Thomas Hanke & Gordian Dziwis Show & Tell	39 Aufrufe	vor 4 Monaten
SmaDi_Mena Leemhuis & Özgür Özcep Show & Tell	15 Aufrufe	vor 5 Monaten
LeBeDigital_Melissa Telong & Stephan Pirskawetz	72 Aufrufe	vor 5 Monaten
DigiBatMat_Vincent Nebel & Marcel Mutz Show & Tell 2023.10.13	61 Aufrufe	vor 7 Monaten
DiProMag_Basil Ell & Moritz Blum Show & Tell 2023.12.08	38 Aufrufe	vor 8 Monaten
DIGITRUBBER_Lars Vogt & Akhilesh Vyas Show & Tell 2023.11.24	25 Aufrufe	vor 8 Monaten
SensoTwin_Ursula Pähler Show & Tell 2023.11.10	43 Aufrufe	vor 9 Monaten
GlasDigital_Ya-Fan Chen Show & Tell 2023.10.27	43 Aufrufe	vor 9 Monaten
iBain_Akhil Thomas Show & Tell 2023.09.29	38 Aufrufe	vor 10 Monaten



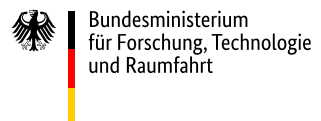
**MATERIALDIGITAL**

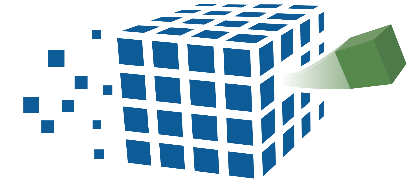
# Thank You!

PMDco workshop materials have been prepared by Platform  
MaterialDigital- Semantic Interoperability Team.

For more information, please contact us via  
[hossein.beygi\\_nasrabi@fiz-karlsruhe.de](mailto:hossein.beygi_nasrabi@fiz-karlsruhe.de).

The Material Digitalization Platform - a joint project by:





**MATERIALDIGITAL**

# Contact us!

[info@material-digital.de](mailto:info@material-digital.de)



[www.material-digital.de](http://www.material-digital.de)

The Material Digitalization Platform - a joint project by:

