ONTOLOGY GOVERNANCE

British Musical Experience

BME

BME Governance Mechanisms

Governance mechanisms are the structured instruments that support and regulate the decision-making process within ontology governance. They establish what should be achieved, what must be fulfilled, and how it can be implemented. These elements are organized in a hierarchical structure that connects abstract governance goals with concrete operational actions:

Principles. (*what should be achieved)* Represent the highest level of the hierarchy. They are the foundational ideas that establish the values and intentions of the governance. They define the desired outcomes and states to be achieved.

> **Requirements.** (*what must be fulfilled*) Derived from each Principle, the requirements are statements that express needs, obligations and conditions that must be fulfilled to comply with the principles.
>
> > **Guidelines.** (*how it can be implemented*) Set of recommendations and methodological instructions on how to implement the Requirements.

Through this hierarchy, each Principle is supported by one or more Requirements, and each Requirement is implemented through one or more Guidelines.

Within the British Music Experience Ontology project, there have been defined 9 foundational principles that guide the whole lifecycle of the ontologies developed, used and managed in the project.

Principle 1. Availability

The ontology and its related components (e.g. code, documentation, CQs, etc.) are openly available for access and consumption by the target users, including both humans and machines.

Requirement 1.1. Localization of the resources

To ensure the availability of the ontology and all its resources, it is essential to clearly define where the ontology is hosted and where it is published:

- ***Hosting*** refers to the platform or infrastructure where the ontology and related assets (e.g. documentation, metadata) are stored, maintained, and updated by its authors.
- ***Publication*** refers to the means by which the ontology is made accessible and discoverable by target users, both humans and machines.

Within the context of this project, the ontology developed is both hosted and published in the GitHub repository belonging to the Knowledge Engineering Group of the University of Liverpool.

Guideline 1.1.1. Hosting and Publishing ontologies in the British Music Experience Server

Each version of the ontology is stored and published in the GitHub repository (add link to the repository) of the Knowledge Engineering Group at the University of Liverpool. All resources developed during the ontology's life cycle will be published in the repository:

- The ontology code in JSON-LD
- The Competency Questions
- The user stories

Once the ontology has been developed, it will be stored and published by the BME server.

Requirement 1.2. Licensing conditions

It must be explicitly stated what terms and conditions apply to access and use of the ontologies and their associated resources.

Clear licensing ensures that users understand whether, how, and under what conditions they can use, modify, or redistribute the ontology. This information should be easy to find and written in a standard, machine-readable format.

The BME ontology will be published under an open license, such as the Creative Commons Attribution 4.0 International (CC-BY-4.0).

However, there will be ontology-related assets that won't be available for users outside the BME organization.

Guideline 1.2.1. Declaring the licensing conditions

In order to ensure legal clarity regarding the access and re-use of the ontology, the licensing information will be given at two complementary levels:

1) **At the documentation level:** Include a human-readable file called LICENSE in the GItHub repository. This file states the licensing terms not only for the ontology code, but also for related resources published in the GitHub repository such as documentation, diagrams, etc.
2) **At the ontology metadata level:** Declare the license in a machine-readable way using standard properties in the metadata of the ontology. The license is identified with the URI that resolves to the description of the license. To do so, well-known standard vocabularies will be reused. For example:

   - From DCMI Metadata Terms: dct:license.
   - From DCMI Metadata Terms : dct:rights
   - From schema.org : schema:license

- From Creative Commons Rights Expression Language : cc:license

Principle 2. Scope

Ontologies are expected to have an explicitly defined scope, ensuring alignment between their content and purpose. A well-scoped ontology facilitates its discovery and reuse. It should be specific enough to represent its intended domain while remaining flexible enough to support its extension.

Requirement 2.1. Scope statement

The scope of the ontology must be indicated in the documentation of the ontology and in its description.

Guideline 2.1.1. Where to declare the scope of the ontology

As stated in the requirement, the ontology's scope should be stated at the documentation of the ontology. Specifically, the knowledge domain that the ontology covers has to be indicated in the README file of the GitHub repository where the ontology is stored and published.

In addition the scope of the ontology must be indicated in a brief paragraph in the ontology description in its metadata. It is recommended to use these well-known annotations properties:

- DCMI Metadata Terms dct:abstract. Like in a scientific paper, here a brief abstract and explanation of the resource should be added.
- DCMI Metadata Terms dct:description . As the name says, it should be a short description of the resource.
- rdfs:comment . Normally used to add additional information of the resource, although it is normally used for the definition of terms.

Principle 3. Documentation

Human and machine readable documentation contributes to promote the transparency, traceability, and understandability of ontologies. This documentation enables diverse users to understand the ontology's content, context and its evolution across its lifecycle.

With regard to ontology documentation, we identified two categories of documentation:

Requirement 3.1. Embedded documentation

The ontology must include embedded documentation through metadata and annotations, expressed directly in the ontology code.

This documentation should be provided at two complementary levels:

1. Ontology-level metadata (applies to the whole ontology). The metadata allows us to gather and show more information about the context of the ontology. The minimum that should be included for documenting the ontology is:

- Ontology name
- Ontology main namespace URI
- Ontology preferred prefix: A recommended prefix established by the authors to be used by other users when using their ontology.
- Ontology creator(s): To provide credit to the people who created the ontology.
- Ontology contributor(s): To credit people who contribute in the ontology development, even if they didn't have a highly important role.
- License and Copyrights rights: Indicate the conditions of (re)use of the ontology.
- Version IRI of the ontology: The IRI that identifies the current version of the ontology.
- Scope and description of the ontology

2. Term-level annotations (applies to individual classes and properties). Users should be able to understand all the elements of the ontology in general but also individually. For that, each element must have:

- A label: A descriptive and (if possible) unique name assigned to each term.
- A textual definition: Natural language definition that facilitates the understanding of the notion. These are recommended to be unique within the ontology, and they should avoid ambiguity. It is also recommended that the individual name of each element must not be used in the definition itself.

Guideline 3.1.1. Ontology-level metadata documentation

There are well-known annotations properties used by the community to describe information about the ontology:

- Ontology name: From RDF Schema Vocabulary rdfs:label or from SKOS skos:prefLabel
- Ontology main namespace URI: From the VANN vocabulary vann:preferredNamespaceUri
- Ontology preferred prefix: From the VANN vocabulary vann:preferredNamespacePrefix
- Ontology creator: From DCMI dct:creator
- Ontology contributor: From DCMI dct:contributor
- License: From DCMI dct:license
- Version: From OWL owl:versionIRI
- Scope and description: From DCMI dct:abstract and dct:description

These annotations should be embedded in the ontology code file.

Guideline 3.1.2. Term-level metadata documentation

Each individual element of an ontology should be understood by the user through its textual definition, which is also embedded in the ontology code. To do so, here are some recommendations:

1. **Use well-known annotation properties**. There are three main properties that are highly used to defined terms:

- From RDF Schema the annotation rdfs:label or from SKOS skos:prefLabel to indicate the human-readable version of the name of the term. Sometimes the label of the term can be different from its identifier. There are some cases in which the authors prefer to create their own property to label their terms. In this case, it is recommended to connect your labeling property to **rdfs:label** explicitly with the rdfs:subPropertyOf property.
- From SKOS (Simple Knowledge Organization System) the property skos:definition to add the formal definition of the term.

From RDF Schema the annotation property rdfs:comment to add the formal definition of the term or other type of additional information.

- In fact, both rdfs:comment and skos:definition can be used at the same time. In this case, normally skos:definition is used to add the formal definition of the term, and rdfs:comment is used to add additional information, e.g. an example where the term can be used.

2. Reuse well-known sources.

If applicable, it is recommended to use already existing definitions from standard references for the definition such as terminologies, dictionaries, glossaries, ISO standards, etc. When applying this process, the original source should be indicated using the annotation from RDF Schema rdfs:isDefinedBy.

Note: For the annotation isDefinedBy it is recommended to add the link to the specific version of the ontology or vocabulary source.

3. Clear, natural and simple text.

Last but not least, it is important that the definition is well-written in natural language that is understood by domain experts and ontology users. The term's definition should be defined as simply and accurately as possible, trying to avoid circularity (using the term itself in the definition).

Requirement 3.2. External documentation

In addition to the embedded metadata, external human-readable documentation must be created and provided to describe the ontology's context, usage and maintenance:

- A document describing all the terms of the ontology. This is a file that collects all the terms documentation embedded in the ontology and that also contains additional information like the scope of the ontology and examples of usage.

- Use cases/user stories: Examples describing how and by whom the ontology is intended to be used, explaining which data is going to be represented.

- Requirements: The document detailing both functional (e.g., a list of Competency Questions) and the non-functional requirements (e.g., the language of the ontology, licensing conditions).

- Diagrams: Visual representation of the ontology. A common notation should be used to represent the different elements of the ontology.

- Examples of usage: Illustrative examples that show how ontology classes and properties are instantiated with real or synthetic individuals. These examples help demonstrate the ontology's structure and its application to model actual data.

- SPARQL queries: example queries to show how the ontology can be queried and used.

- Changelog or similar document to keep track of the changes.

Guideline 3.2.1. Publishing external ontology documentation

Documenting the ontology is hard-working and continuous activity throughout all the ontology's lifecycle. As the ontology, the documentation is also an evolving resource that reflects the changes, decisions, and developments in the ontology over time. Here are some best practices to generate human-readable documentation:

1. Documentation is an iterative process

Documentation must accompany the ontology through all its stages. Any change to the ontology (e.g., creation of new requirements, adding or deprecating elements, etc.) should be reflected in the corresponding documentation. This is important not only for the future users, but also for the authors themselves.

2. Choose an appropriate format for documenting and publishing the ontology:

It is important to document each element of the ontology in a human-readable way since it is not always easy to go through the Turtle or RDF/XML file and understand searching and trying to understand the terms. The most recommended practice is to publish the documentation of the ontology in an interactive HTML page. Although this can be as a difficult way, in fact there are several tools created by the Semantic Web Community that allows to create this pages from the metadata and annotation properties of the ontology:

- **WIDOCO**: Generates comprehensive HTML documentation from ontology metadata and annotations.

- **LODE**: Lightweight and quick HTML renderer for OWL ontologies.

- **ProtégéOWLDoc**: plugin of Protégé that allows to export to HTML

- **Ontoology**: Web-based platform that supports validation, documentation, and publication via GitHub.

However there are also other options for publishing the ontology documentation. For example in static documents, such as PDFs or Word files. It is a more conservative and easy to manage option, but it is also a really time-consuming task.

3. Indicate how to cite the ontology

It is common to find that if an ontology has a related scientific publication, users reference the article to cite the ontology. However, articles are static resources, unlike ontologies, which are subject to different changes and versions. That is why it is important to tell users how to cite the ontology, and specifically how to cite each version of the ontology used. That is why it must be specified in the ontology's external documentation (e.g., in the README file) as well as in its metadata (e.g., using the dct:bibliographicCitation or schema:citation property).

Principle 4. Metadata

The ontology is made findable and traceable through its metadata, which helps users identify and interpret its context.

Requirement 4.1. Metadata at ontology level

The ontology must include a minimal set of metadata annotations at the ontology level to ensure its discoverability, identification, and contextualization across systems. This metadata allows users and machines to retrieve and interpret the ontology in a standardized way. The minimum metadata recommended for the BME ontology is:

1) Ontology identification and description:

- Ontology name
- Ontology main namespace URI
- Ontology preferred prefix: A recommended prefix established by the authors to be used by other users when using their ontology.

- Scope and description of the ontology (see Principle 2 - Scope)

2) Attribution:

- Ontology author(s): Provide credit to the people who created the ontology.
- License and Copyrights rights: Indicate the conditions of (re)use of the ontology.
- Citation: Indicate a way to cite the ontology.

3) Versioning:

- Version IRI of the ontology: IRI that identifies the current version of the ontology.

Guideline 4.1.1. Recommended metadata for describing the ontology

For the BME there will be used standard well-known vocabularies such as Dublin Core.

There are a set of well-known annotation properties from standard vocabularies that are used by the Semantic Web Community as metadata for ontologies:

1) Ontology identification and description:

- Ontology name: From RDF Schema Vocabulary rdfs:label or from DCMI Metadata Terms dct:title
- Ontology main namespace URI: From the VANN vocabulary vann:preferredNamespaceUri
- Ontology preferred prefix: From the VANN vocabulary vann:preferredNamespacePrefix
- Scope and description: From the DCMI Metadata Terms dct:abstract and dct:description

2) Attribution:

- Ontology creator: From the DCMI Metadata Terms dct:creator
- License and Copyrights: From the DCMI Metadata Terms dct:license
- Citation: From DCMI Metadata Terms dct:bibliographicCitation

3) Versioning:

- Version IRI of the ontology: From the OWL 2 Schema Vocabulary owl:versionIRI
- If applicable, indicate the previous version of the ontology (the previous version IRI): From OWL 2 Schema Vocabulary owl:priorVersion

Requirement 4.2. Metadata at term level

Each term in the ontology must include a set of metadata annotations to ensure it can be identified, understood, and reused:

- **A label:** A descriptive and human-readable name assigned to each term.

- **A textual definition**: Natural language definition that facilitates the understanding of the notion. These are recommended to be unique within the ontology, and they should avoid ambiguity. It is also recommended that the individual name of each element must not be used in the definition itself.
- **Reuse source**: If the term is reused from another ontology or vocabulary, provide the URI of the original resource.

Guideline 4.2.1. Recommended metadata for describing the terms

For the BME there will be used standard well-known vocabularies. There are well-known annotations for the terms individually:

- **Label:** From RDF Schema the annotation rdfs:label, used with a language tag such as @en.
- **Definition source:** From RDF Schema rdfs:isDefinedBy
- **Resource source:** From RDF Schema rdfs:isDefinedBy is also used to indicate the IRI of the ontology from which the term was reused.

Principle 5. Identifiers

The BME ontology follows a designing policy for the creation and management of identifiers. This policy addresses identifiers at three levels: Ontology and Term IRI, Version identifiers, and Naming Convention.

Requirement 5.1. Ontology and Terms IRI

The ontology must be assigned a unique and persistent IRI to ensure its long-term accessibility. The base IRI serves as the base for constructing the IRIs of all terms defined within the ontology.

The BME ontology follows the next pattern:

https://w3id.org/BME/NameOntology

Guideline 5.1.1. Designing and managing the ontology IRI

The base IRI is the unique and persistent identifier that serves as the main access point for both human and automatic agents. It represents the ontology as a whole and must remain stable over time to facilitate its long-term location and reuse.

Persistent domain

To ensure the ontology remains accessible in the long term, it is recommended to register it under a persistent IRI service rather than a personal, project-specific, or institutional domain that might expire or change. For the BME ontology, the IRI will be assigned a persistent domain using the w3id.org initiative. A W3C community-driven initiative that allows users to register persistent identifiers that redirect to a controlled destination. You can manage redirections via a GitHub repository.

This service allows to separate the ontology's identifier from its physical location, so if the ontology files are ever moved or restructured, the IRI remains unchanged by simply updating the redirection rules.

In this case, the IRI of the ontology will be direct from the w3id.org to the specific GitHub repository where the ontology is hosted.

Content negotiation

The ontology IRI should resolve via HTTP and support content negotiation, which allows users and systems to retrieve either:

- A human-readable HTML documentation page

- A machine-readable serialization (e.g., RDF/XML, Turtle)

Design of the name and prefix of the ontology

In order to facilitate the findability of the ontologies, it is highly recommended to use short names or acronyms for the ontologies. In addition, the prefix should be in line with this name. In order to avoid copying already existing ontologies, it is important to check if there are ontologies with the same prefix in the public register of common namespaces and prefixes prefix.cc. It allows users to look up or register short, ontology namespace prefixes and their corresponding namespace IRIs. Once the ontology IRI and its preferred prefix are defined, they must be registered in prefix.cc.

Guideline 5.1.2. Designing the Terms IRIs

As the ontology, each class, property, and individual in the ontology will have a unique and persistent IRI. Within the BME, the following strategy will be used to design the term IRI:

Slash-based IRIs (e.g., https://w3id.org/biodiv-onto/Species)

Starting from the base IRI of the ontology, the IRIs of the terms are created by appending the chosen identifier for each term to the base IRI, separated by a slash. Here each term is treated as a resolvable resource. Then pattern to be followed is:

https://w3id.org/BME/NameOntology/Term

Requirement 5.2. Ontology versions

Ontology development is an iterative process in which multiple versions of the ontology are created and released. Each official version of the ontology must be assigned a specific and persistent IRI to ensure that users can access and cite any past version. The base ontology IRI must resolve to the latest available version of the ontology.

To do so, the ontology authors must define a versioning policy, including the pattern used to construct version-specific IRIs.

G5.2.1. Defining IRIs for the ontology's versions

Version-specific IRIs are essential to ensure that users can access, cite, and reuse a specific release of the ontology. A well-defined versioning policy supports traceability, reproducibility, and transparency across the ontology's lifecycle. The most common practice in ontology IRI versioning is to follow the Semantic Versioning model.:

X.Y.Z (e.g., https://w3id.org/example/1.2.0) where:

- X (major): for changes that break compatibility or significantly alter the structure (e.g., adding a whole new module to the ontology).

- Y (minor): for backward-compatible changes (e.g., adding new terms, metadata).

- Z (patch): for small corrections or non-structural updates (e.g., fixing typos, updating labels or definitions).

These types of changes and how to handle them must be specified in your versioning policy

Keep in mind that the version IRI should not replace the base IRI of the ontology. The base ontology IRI (e.g., https://w3id.org/example) should always resolve to the latest version. The version IRI (e.g., https://w3id.org/example/1.0.1) points to a specific release and remains immutable. This separation ensures backward compatibility and avoids breaking references in reused data.

However, embedding version numbers in term IRIs (e.g., https://w3id.org/example/1.0.1#Organization) is discouraged because it forces consumers to update instance data when the ontology evolves. Instead, use versioning only at the ontology IRI level (with owl:versionIRI) while keeping the namespace consistent.

Finally, it is important to declare version metadata in the ontology file:

- owl:versionIRI: the IRI of the current version of the ontology.

- owl:versionInfo: a human-readable string describing the version (e.g., "Version 1.2.0 - Minor update").

- owl:priorVersion: link to the previous version helps to see how the ontology has changed.

Requirement 5.3. Naming convention

To support consistency and maintainability, the ontology must adopt a naming convention policy that defines how term identifiers are constructed.

Within the BME ontology a naming convention policy has been defined for all identifiers in the ontology. This policy ensures consistency, and clarity across classes, properties, and individuals.

Guideline 5.3.1. Naming convention policy

To support a coherent and maintainable ontology design, a naming convention policy was defined. The policy includes the following guidelines:

1. **Use of natural language terms**
    - All identifiers are constructed using natural language words that are descriptive of the concept or relation they represent.
2. **Syntax rules by type of term**
    - Classes: Use CamelCase (e.g., Instrument).
    - Object and data properties: Use mixedCase starting with a lowercase letter (e.g., playsInstrument).
    - Individuals (if applicable): Use CamelCase or another consistent pattern agreed within the project.

Principle 6. Reuse

To ensure interoperability and avoid duplication of efforts, for the construction of the BME ontology, there will be reused existing, and well-established ontologies and vocabularies.

Requirement 6.1. Reuse of existing ontologies

Existing ontologies and controlled vocabularies must be reused when they are relevant to the domain and aligned with the ontology's requirements.

To be eligible for reuse, external ontologies must demonstrate clear relevance to the domain, in this case, alignment with LinkedArt, and should preferably belong to recognized controlled vocabularies such as AAT, PBCore, MIMO, or Europeana Fashion.

Guideline 6.1.1. How to select ontologies for reuse

The selection of ontologies for reuse follows a structured process that ensures coverage of the domain requirements. The recommended steps include:

1. **Start from a preliminary model**
    - Begin by drafting a preliminary conceptual model using a general schema such as schema.org to identify the main entities, relations, and missing elements.
2. **Identify suitable upper-level ontologies**

    - For high-level, cross-domain concepts, reuse well-established ontologies such as LinkedArt and the Europeana Data Model (EDM).
3. **Evaluate relevance and alignment**
    - Assess whether candidate ontologies and vocabularies sufficiently cover the domain needs and are conceptually compatible with the BME model.
4. **Consider established controlled vocabularies**
    - Prioritize vocabularies recognized in the cultural heritage domain, such as AAT, PBCore, MIMO, and Europeana Fashion.

Guideline 6.1.2. How to reuse ontologies in practice

Once identified a suitable ontology, there are two main approaches to reuse ontologies in the BME ontology:

Soft reuse

Individual terms or collection of terms are reused but not the whole ontology:

- The terms are referenced by reusing their IRIs. This allows more control and stability, as the ontology is not automatically affected by changes in the source ontology.
- It is important to document the source of the reused terms using the annotation property rdfs:isDefinedBy to indicate the version IRI of the original ontology.

Ontology mapping

Map local concepts to external ontologies using equivalence or hierarchical relations (e.g., owl:equivalentClass, skos:exactMatch, rdfs:subClassOf), ensuring semantic interoperability.

Principle 7. Format

The BME ontology is available in the ontology implementation language RDF(S) and in the serialization JSON-LD, to ensure its interpretability and interoperability by tools, systems, and users.

Requirement 7.1. Ontology implementation language

The ontology must be developed using RDF/RDFS as the foundational representation language.

Additionally, the ontology must be made available in at least one widely adopted serialization format, specifically JSON-LD, to ensure interoperability and ease of integration with web-based systems.

Guideline 7.1.1. How to implement the ontologies in the right format

The ontology is implemented and maintained using Protégé, configuring the project to:
- model the ontology in RDF/RDFS, and
- export the ontology in JSON-LD format.

Principle 8. Adoption

To ensure the impact and relevance of the ontology, its adoption by the intended users is actively promoted and monitored. In this context, dissemination refers to the set of actions aimed at communicating, explaining, and making the ontology accessible to different audiences.

Requirement 8.1. Dissemination

A dissemination strategy must be defined to ensure that the ontology reaches its intended user groups. This strategy must address the specific needs of two audiences:

- Non-expert users within the BME organization, and
- The scientific community of the knowledge domain.

Guideline 9.1.1. Dissemination activities and materials

The dissemination of the ontology should be planned and executed according to the needs and characteristics of its two main user groups: the scientific community and the non-expert users within the BME organization.

The objective of these activities is to ensure that each audience can understand the ontology, its purpose, and how to use it in their respective contexts.

1. Scientific community

Dissemination to the scientific community focuses on contributing to ongoing research discussions, ensuring visibility within relevant academic environments. To achieve this:

- The ontology is described and analyzed in scientific articles, presenting its motivation, scope, modeling choices, and potential applications.
- The ontology is also presented in scientific conferences, where it can be explained to peers, discussed, and validated by experts in related fields.

These activities help position the ontology within academic discourse and promote its reuse or extension by other research groups.

2. Non-expert users

For internal users who are not experts in ontology engineering, dissemination prioritizes clarity, and practical relevance. To ensure that these users can understand the ontology and apply it effectively in their daily work:

- Workshops should be organized to provide hands-on sessions where the ontology's structure, key concepts, and use cases are explained interactively.

These materials and activities ensure that the ontology is not only technically correct, but also usable and meaningful for BME staff who interact with it in a non-technical capacity.

Requirement 9.2. Adoption by community

There are a set of indicators that demonstrate the ontology is being used by its intended community. These indicators not only reflect the level of adoption, but can also serve for assessing the ontology's quality.

Guideline 9.2.1. Internal indicators of the ontology's use

To monitor whether the ontology is being adopted by its target community, a set of internal usage indicators are tracked and documented. These indicators provide the authors with concrete evidence of how the ontology is being used, how it evolves, and how the community interacts with it.

The following indicators are recommended:

1. GitHub activity

Monitor contributions, feedback, and issue reports submitted by users through the GitHub repository.

2. Scientific and technical citations

If applicable, track how often the ontology (and the possible publication describing it) is cited in:

- scientific papers,
- technical reports,
- other ontologies or data models.

3. Integration in projects, tools, or platforms

Document cases in which the ontology is integrated into:

- research or industrial projects,
- software tools,
- platforms or services.
- References in project websites, deliverables, or documentation where the ontology is explicitly mentioned are strong indicators of adoption.

Principle 9. Maintenance → Change to the beginning of the Principles

To ensure the long-term relevance, accuracy, and usability of the ontologies, its resources are regularly reviewed and updated to incorporate new requirements, corrections, and improvements. This is done following the maintenance policy defined, which covers these four elements: Ontology Evolution, Users Communication, Versioning, and Contribution Guidelines.

Requirement 9.1. Ontology evolution

Ontologies are not static resources, they evolve over time to incorporate new requirements, correct errors, and adapt to changing knowledge domains. To manage this process effectively, it must be defined with a clear evolution strategy. This includes establishing guidelines and procedures for identifying and managing changes, documenting decisions, and involving stakeholders.

It is important that this strategy establishes a classification of the types of changes that are handled within the ontology. For each change, it must specify which artefacts are affected, what are the roles responsible for the execution and validation, and the monitoring and documentation mechanisms to ensure traceability.

Guideline 9.1.1. Ontology Evolution Strategy

Ontologies represent knowledge domains that evolve over time. New requirements may emerge, existing definitions may be refined, and modelling errors may need correction. To manage this process in a controlled, consistent and transparent way, ontology teams should define a clear Ontology Evolution Strategy.

This guideline provides a generic template that any organisation can adopt to manage ontology evolution. It explains how to classify changes, how to handle them, and which aspects should be documented and communicated.

1. Classification of Changes

Following the logic of Semantic Versioning (X.Y.Z), ontology changes can be grouped into three categories:

Major changes (X → X+1.0.0):
Large modifications that alter the ontology's conceptual structure or scope. These include adding new modules, introducing new core concepts or relations, or deprecating key classes or properties.

Minor changes (X.Y → X.Y+1.0):
Extensions or refinements that do not affect the main conceptual structure, such as adding new classes or properties within existing sections of the model, or improving metadata annotations.

Patch changes (X.Y.Z → X.Y.Z+1):
Small corrections that do not alter meaning, such as fixing typos, syntax issues, prefixes, or updating metadata and examples.

2. Elements to Define for Each Change Type

For each category (Major, Minor, Patch), the evolution strategy should describe:

- How the change is identified (e.g., issue tracking, stakeholder input, domain updates).

- How the change is implemented and validated, including modelling, encoding and review activities.

- Which roles are involved, such as ontology engineers, domain experts or project leaders.

- Which artefacts are affected, including ontology files, diagrams, documentation, metadata, mappings or release notes.

- How the change is documented, using a changelog, requirements documentation or design notes.

- How the change is communicated, ensuring that users and stakeholders are informed as appropriate.

3. Procedures by Change Type

3.1 Major Changes

Major changes begin with identifying a significant modelling need, often linked to new requirements or domain evolution. Domain experts, stakeholders and ontology engineers collaborate to analyse the impact and record it in requirement artefacts. The implementation phase includes conceptualisation, modelling and validation of the new material. Once complete, the updated ontology is released, documentation is updated, and the change is communicated broadly to users.

A new major version number is assigned (X → X+1.0.0).

3.2 Minor Changes

Minor changes start with identifying smaller modelling needs, such as adding new classes or refining existing ones. Ontology engineers, often with domain experts, implement and validate these updates. After updating documentation and publishing the revised release, the change is recorded in the changelog and communicated through regular channels.

A new minor version number is assigned (X.Y → X.Y+1.0).

3.3 Patch Changes

Patch changes arise from detecting non-semantic issues such as typos, syntax errors or metadata corrections. Ontology engineers implement these fixes directly and update the changelog accordingly. Documentation and communication depend on the severity and visibility of the correction.

A new patch version number is assigned (X.Y.Z → X.Y.Z+1).

4. Additional Recommendations

The evolution strategy should align with the versioning policy, define the roles involved in each step, specify the tools used for tracking and documenting changes (e.g. WIDOCO), and ensure that all released versions remain accessible through persistent identifiers.

Maintaining a clear changelog and consistent communication strategy supports transparency and helps users understand how and why the ontology evolves.

Requirement 9.2. Versioning policy

A versioning protocol is essential to ensure the traceability and long-term management of the ontology. It allows both users and maintainers to track the evolution of the ontology across time. For that, a clear versioning strategy has been defined which includes:

1. The use of **unique and persistent identifiers (IRIs)** to each released version. As mentioned before, one recommended approach is the use of the Semantic Versioning (X.Y.Z) model to design the path IRI to each version. Each version of an ontology represents a specific type of change (see Requirement 10.1. Ontology evolution):

    - **Major change** involves this path of IRI:
      www.w3id.org/BME/NameOntology/X.Y.Z to
      www.w3id.org/BME/NameOntology/X+1.0.0

    - **Minor change** involves this path of IRI:
      www.w3id.org/BME/NameOntology/X.Y.Z to
      www.w3id.org/BME/NameOntology/X.Y+.1.0

    - **Patch change** involves this path of IRI:
      www.w3id.org/BME/NameOntology/X.Y.Z to
      www.w3id.org/BME/NameOntology/X.Y.Z+1

2. The **storage and accessibility of versioned files**, ensuring that each version remains available even after new versions are released.

Guideline 9.2.1 Guidelines for Versioning

The following are the procedures and recommendations to follow for ontology versioning in BME:

1. Assign persistent version identifiers:

As mentioned before, each official version of the ontology should have a unique and stable IRI. The pattern to be followed is based on the Semantic Versioning model X.Y.Z. The design of the paths of the version IRIs is the following one:

www.w3id.org/BME/NameOntology/X.Y.Z

2. Maintain a change record document:

It is important to establish a system for saving and tracking changes and versions obtained for each ontology. That is why each ontology must have a document assigned to it for saving changes, a changelog file, published and maintained in the ontology's GitHub repository. The following must be defined within the document:

- The date of each release
- The version of the ontology affected
- A summary of the changes
- Contributors involved in the changes

3. Preserve previous versions

Make previous ontology versions accessible. Avoid deleting or overwriting old versions, as they may be still used by external systems.

4. Synchronize documentation and metadata:

The documentation and the ontology metadata must reflect the current version and the changes from the previous version. Remember to update:

- The **metadata** of the ontology using the annotations:
  - Include the version of the ontology with owl:versionIRI
  - Include some specific information about this version with owl:versionInfo
  - Indicate the IRI of the previous version to facilitate comparison with owl:priorVersion
  - Include a list of the changes or reference to a resource that describes the changes between previous and new versions with vann:changes

- The external documentation such as diagrams, requirements, and user stories.

Requirement 9.3. Users communication

There must be defined communication channels to facilitate the interaction between the ontology owners, its users and the community. These channels enable users to report issues, request changes, or provide feedback, and allow ontology owners to notify users about updates, new releases, or important decisions.

Within the BME ontology, the main channels to communicate with users are:

- The GitHub Issue trackers
- Dedicated pages on the project’s website

Guideline 9.3.1 Users communication mechanisms

Communication channels are essential to keep users informed throughout the lifecycle of the ontology. Recommended communication mechanisms include:

It's important to clearly indicate how users can access these channels (e.g., through links in the ontology documentation or metadata) and to ensure their maintenance over time.

Requirement 9.4. Contribution policy

External users may contribute through the project's GitHub platform by submitting changes, reporting issues, or providing suggestions.

However, the final decision regarding the acceptance, modification, or rejection of any contribution MUST remain under the responsibility of the ontology's internal team.

Guideline 9.4.1 Contribution guidelines

For the BME ontology, contributions are handled through the GitHub issue tracker. External contributors may:

- report issues or errors,
- suggest improvements or modifications,
- request new terms or changes to existing ones,
- propose enhancements to documentation or metadata.

Contributors should describe the proposed change, its motivation, and any relevant context. The internal ontology team is responsible for reviewing all submissions.