
# PhLMT0001 Learning Requirements Specifications

**Version 2.0**

## Documentation Information

| | |
|---|---|
| **Contract Number** | 101069595 |
| **Project Website** | www.safexplain.eu |
| **Contractual Deadline** | 31.03.2024 |
| **Dissemination Level** | SEN |
| **Nature** | R |
| **Author** | Javier Fernández |
| **Modified by** | Lorea Belategi |
| **Reviewed by** | Irune Agirre |
| **Approved by** | Irune Agirre |
| **Keywords** | AI, Functional Safety, Learning Management, Learning Requirements |

This project has received funding from the European Union's Horizon Europe programme under grant agreement number 101069595.

# Table of Contents

# 1 Review / Modification History

| Version | Date | Description Change |
|---|---|---|
| V2.0 | 15/02/2024 | Changes Applied as a result of TÜV Review 2024-01-19 |
| V1.0 | 01/12/2023 | First version after complete internal review |
| V0.2 | 28/11/2023 | Modifications and improvements based on internal review |
| V0.1 | 11/10/2023 | First draft |

*Note: The paragraphs/name of the project/Rev./Ref./history table in **blue** must be replaced with the information for the specific project. The paragraphs written in **red** are instructions that can be used as a guide, so they must be deleted.*

# 2 Objective

The objective of this document is to collect the learning requirements specification.

# 3 Scope

This template applies to the learning requirements specifications of the Learning Management phase performed through the Artificial Intelligence - Functional Safety Management (AI-FSM). In fact, the documents generated at this step shall encompass safety, operational, functional and non-functional requirements specifications requirements.

# 4 Requirements Features and Conventions

This section provides a set of characteristics and attributes that are required for each requirement.

## 4.1 Description of the Requirements

All the requirements of a safety-related project must be specified following the characteristics below:

- Shall be described in such a way that they are*:*
  - **Clear.** The requirement must be easy to understand and not misleading. The requirements are written in a way that allows them to be understood by all stakeholders in the project.
  - **Concise.** The requirement must not be too long. It must be easy to read and understand, and it must not contain definitions, descriptions of its use, or reasons for its need.
  - **Unambiguous.** The requirements can be interpreted only one way.
  - **Verifiable.** The implementation of the requirement can be determined through one of four possible methods: inspection, analysis, demonstration, or test.
  - **Traceable.** A good requirement is traceable: it must be easily traced through to specifications, design and testing. Besides, it must have a unique identity or number.
  - **Complete.** All conditions under which the requirement applies should be stated.
  - **Feasible.** The requirement can be implemented within the constraints of the project.
    - Shall be written to aid comprehension by those who are likely to use the information at any phase of the E/E/PE safety-related system.
    - Shall be expressed in natural or formal language and/or logic, sequence or cause and effect diagrams that define the necessary safety functions.

## 4.2 Requirements Specifications Table

The attributes needed for each requirement to facilitate its identification and description are specified in Table 1 as follows:

*Table 1. Table of attributes for each requirement*

| <Identifier> | <Title> |
|---|---|
| Description | |
| Source | |
| Phase of the lifecycle | |
| Reference | |
| Type | |
| Validation criteria | |
| Date | |
| Version | |

A brief description of each field of the previous table has been done below:

- *<Identifier>.* Each Safety Requirement must be identified with a code like **REQ-YYY-OOO-ZZZ** where **REQ** refers to *requirement* and **YYY** is a code that identifies the requirement category. In this case the code selected is **LRNRS** (*LeaRNing Requirement Specification*). Depending on the project more subdivisions can be necessary to identify the requirements (The identifier "**OOO**" can be used to identify these subdivisions). Besides, the identifier "**ZZZ**" represents an incremental number used to identify the requirements in each sub-division. The last two identifiers are optional and they can be replaced for another approach.
- *<Title>.* Title of the requirement. The requirements can be divided into some more specific groups that can be directly related to the application, such as components and devices of the system, the regulations, tools, programming languages, and others. In order to identify these specific sub-divisions it can be useful to define first the name of the sub-division and then the title of the requirement.
- *<Description>.* Brief description of the requirement. The description must be short (max. 2-3 lines), aligning with the characteristics outlined in Subsection 5.1, and it must only include one requirement. Besides, it must be defined by the following keywords:
  - ***MUST.*** This word, or the adjective "required", means that the definition is mandatory.
  - ***MUST NOT.*** This phrase means that the definition is an absolute prohibition of the specification.
  - ***SHOULD.*** This word, or the adjective "recommended", means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications must be understood and carefully considered before choosing a different course. A requirement expressed as "should" could become a mandatory requirement in a future version of the document.
- *<Source>.* Source of information relevant to the requirement, i.e. department, contact person, etc.
- *<Reference>.* References relevant to the requirement, i.e. documents, files, figures, emails, etc.
- *<Type>.* Mandatory/Desirable/Optional.
- <Validation Criteria> The requirements will be validated using one, or several of the following methods (at different process phases), depending on the attributes assigned to them:
  - ***Inspection***: It takes place when a direct examination of the design documentation is enough to show that the requirement has been implemented. The requirements must be traced to the proposed design for the system. This design should be defined in the documents.
  - ***Analysis***: This is a non-functional validation, which may include simulation, quantitative analysis, statistical analysis and comparison between the results obtained analytically and numerically. The requirements will be tested by a dedicated analysis.

- ***Test***: Implies a functional validation, which can verify the suitability of the implemented requirement by direct measurement. The requirements will be verified using a test procedure. A test is defined by a set of input parameters, an environment, a procedure and a set of outcomes or outputs.

- *<Phase>:* Specification/Architecture/Detail. Phase in which the requirement is defined as specified in the V-model. For the learning requirements case, the Learning Management phase.
- *<Date>:* Date of creation of the requirement version.
- *<Version>:* Requirement version. All versions are included in the report in a consecutive order.

# 5 Learning Requirements Specifications

This template proposes decomposing the learning requirements specifications into two categories explained in the following two subsections: i) qualitative learning requirements, explained in Subsection 5.1 and ii) quantitative learning requirements, explained in Subsection 5.2. In addition, the model election criteria shall be defined to be used during the model verification step in those cases in which several candidate models provide the required performance. However, the user can employ that structure that better fits its necessities.

## 5.1 Qualitative Learning Requirements Specifications

Among the qualitative learning requirements specifications, we can list the following ones:

1. Define the methodology for searching the hyperparameters of the model (configuration settings not learned from the data). This methodology encompasses a systematic approach to efficiently explore the hyperparameter space, along with the constraints or rules guiding this exploration, ensuring that the search is conducted within practical and feasible boundaries. Key aspects include:
    1.1. Learning algorithm
    1.2. Cost/loss function specification
    1.3. Model bias and variance methodology

    The *PhLMG0002_Learning_Management_guideline.docx* guideline includes in the model design step a set of aspects to be considered.
2. Define a strategy to select the model among the candidates. Clearly define a plan or set of criteria for choosing the best-performing model from a pool of candidate models.
3. An acceptance criterion of the model should be defined for the training, validation and verification of the model. Establish quantitative properties such as performance, robustness, the model's capability to extrapolate, and explainability. The acceptance criteria should specify acceptable or boundary levels and the assessment metrics.
4. Define the training environment. Clearly specify the deep learning requirements or the requirements allocated to deep learning components to identify the necessary hardware (GPUs, Tensor Processing Units (TPUs), etc.) and software (TensorFlow, PyTorch, etc.) required for training the model.
5. Define the methodology to initialize the parameters of the model. Common methodologies include pre-defined parameters, random initialization, or techniques such as Glorot initialization (reference).

## 5.2 Quantitative Learning Requirements Specifications

Some of the previously qualitative requirements are linked to measuring DL safety-related properties. In the following

1. Define conditions for the hyperparameter of the models, such as boundaries. The *PhLMT0002_Model_Election_Log_template.docx* provides a list of hyperparameters that can be helpful

in identifying them, but the hyperparameters are not restricted to the ones mentioned in the document.

2. Define model bias and variance boundaries to avoid underfitting and overfitting.
3. Define performance and robustness requirements, along with the metrics for evaluating their compliance. These specifications are evaluated based on the quantifiable properties of the DL model. The following is a list of model properties that can be utilized to assess the DL requirements:
   3.1. Examples of DL performance requirements: accuracy (i.e., for each frame: the pedestrian should be detected with an accuracy of at least 0.95, the bounding box shall not differ in height, weight and centre position more than 8 pixels…), precision, recall and F1 score (aiming to find a balance between precision and recall) …
   3.2. Regarding the DL robustness requirements. The model shall perform within the performance boundaries thresholds within unseen data in the training dataset but foreseeable data within the Operational Design Domain (ODD). For example, in an object detection application: Evaluate the performance metrics with new ranges of light or fluency of people, asses new people poses …

## 5.3 Model Election Criteria Definition

The definition of model election criteria involves establishing specific parameters or characteristics that are considered when choosing one model over another. These criteria serve as guidelines or benchmarks to evaluate and compare different models, ultimately aiding in the selection of the most suitable or effective one for a given purpose. Common examples of model criteria include accuracy, explainability, and robustness, among others. The process of defining model criteria is essential for making informed decisions and ensuring that the chosen model aligns with the desired outcomes and requirements of a particular task or problem. Examples in object detection applications:

- Prioritizing classes accuracy. Example: If all candidate models demonstrate similar overall performance across different classes, a prioritization strategy can be applied. In this scenario, models excelling in accuracy specifically for pedestrian detection could be favored.
- Robustness regarding specific environmental conditions. Example: If all candidate models demonstrate similar overall performance across different environmental conditions, a prioritization strategy can be applied by selecting the model with the highest performance for the most common environmental conditions in which the system will operate.
- Emphasis on explainability. When accuracy results among candidate models are not significantly distinguishable, the focus may shift to explainability. This is particularly relevant if explainability can be quantified and applied consistently throughout the model's lifecycle. In such cases, prioritizing models with high levels of explainability ensures a clearer understanding of the decision-making process.

# 6 Learning Requirements Overview

To have a complete view of all the requirements defined in this phase, the following table must be completed with the learning requirements defined in the previous sections:

*Table 2. Learning requirements overview*

| <Identifier> | Description |
|---|---|
| | |
| | |
| | |
| | |

***Reminder**:*

*- Update the state of REF_Ph0D0003_AI Document_List.docx when a document is generated or modified, including the last version generated.*

*- The tools and frameworks employed must be listed in REF_Ph0D0011_AI_Tools_Selection.docx.*

*- The traceability between the DL requirements specifications and the learning requirements specifications must be updated in REF_Ph0D0013_AI_Traceability_Matrix.docx.*

# 7 Acronyms and Abbreviations

Below is a list of acronyms and abbreviations employed in this document:

- AI - FSM – Artificial Intelligence - Functional Safety Management
- ODD – Operational Design Domain
- DL – Deep Learning
- TPU – Tensor Processing Units

# 8 Bibliography

Add here the reference to used bibliography / references (if any).