PhLMG0002 Learning Management Guideline

Version 2.0

Documentation Information

Contract Number	101069595
Project Website	www.safexplain.eu
Contratual Deadline	31.03.2024
Dissemination Level	SEN
Nature	R
Author	Javier Fernández, Ana Adell, Jon Imaz,
Modified by	Lorea Belategi
Reviewed by	Irune Agirre
Approved by	Irune Agirre
Keywords	DL, Functional Safety, Learning Management

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	04/12/2023	First version after complete internal review
V0.5	04/12/2023	Modifications and improvements based on internal review
V0.4	11/09/2023	Modifications and improvements based on internal review
V0.3	30/08/2023	Modifications and improvements based on internal review
V0.2	30/05/2023	Modifications and improvements
V0.1	14/04/2023	First draft

Note: Since Artificial Intelligence – Functional Safety Management (AI-FSM) utilizes templates from both the traditional Functional Safety Management (FSM) and its own templates, this annex distinguishes the AI-FSM templates by color-coding them in orange and the traditional FSM templates in green. Additionally, the files' names created from the templates are written in italics and underlined. It is worth mentioning that all the templates' names are preceded by "REF_" which should be changed to reflect the specific safety project reference.

2 Objective

The purpose of this document is to provide guidance for the Learning Management process. Learning Management is carried out in parallel with Data Management, initiating both processes simultaneously. It can be broken down into five primary steps, as illustrated in Figure 1. In that figure, the three numbered blue rhombuses represent inputs from the Data Management phase, which correspond to the training dataset (rhombus labelled with the number 1.1), the validation dataset (rhombus labelled with the number 1.2) and the verification datasets (rhombus with the number 2). Additionally, there is an additional red rhombus, which serves as a condition to check the results of the model evaluation. In case the model evaluation does not meet the criteria, a new iteration of the model design, model training and model evaluation steps must be performed until the model is successfully validated.

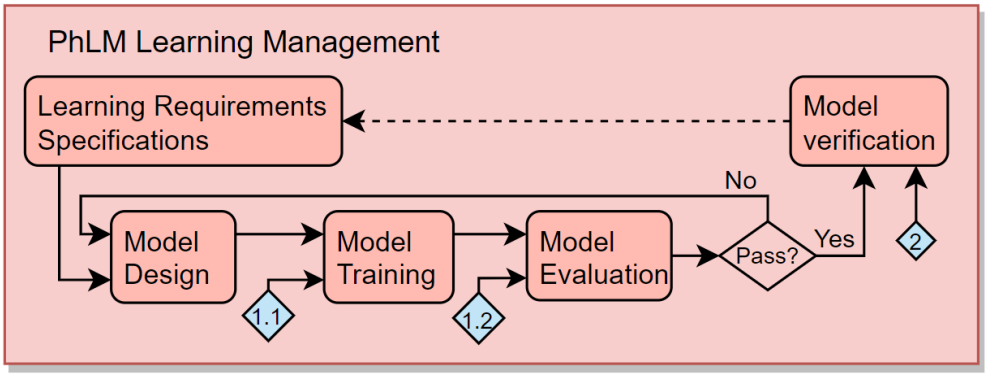

Figure 1. Learning Management phase

3 Scope

This guideline applies to all activities and documentation required to perform the Learning Management phase, where a DL model is trained and verified based on the datasets coming from the Data Management phase.

4 Introduction

In order to provide an overall insight into this process before delving into the individual explanation of each step, here is a list of all the documents that need to be generated at this phase:

1. *REF_PhLMD0001_Learning_Requirements_Specifications.docx*. This step refines the Deep Learning (DL) requirements specifications previously defined in Phase 2, focusing on the needs of the Learning process.
2. *REF_PhLMD0002_Learning_Requirements_Specifications_IR.xlsx*. Internal review document to be checked after completing *REF_PhLMD0001_Learning_Requirements_Specifications.docx.*
3. Learning requirement tests encompass a set of metrics to assess whether the learning requirement specifications have been fulfilled, the test definitions, and their corresponding outcomes. In the Learning Management phase, the following two test sets should be defined and performed:

a. *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx*
b. *REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx*

And their associated internal reviews:

1. *REF_PhLMD0006_Learning_Requirements_Evaluation_Tests_IR.xlsx*. Internal review document to be checked after completing *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx.*
2. *REF_PhLMD0008_Learning_Requirements_Verification_Tests_IR.xlsx*. Internal review document to be checked after completing *REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx.*

4. *REF_PhLMD0003_Model_Election_Log.docx*. Collecting the DL models designed and the criteria for the election of the most suitable DL model.
5. *REF_PhLMD0004Model_Election_Log_IR.xlsx*. Internal review document to be checked after completing *REF_PhLMD0003_Model_Election_Log.docx.*

Additionally, the following artifacts must be generated and stored:

1. Trained model(s). Models that have undergone training on labeled datasets (training dataset) to learn patterns and relationships for making predictions on new data.
2. Evaluated model(s). Models that have been evaluated using separate datasets (validation dataset) to assess if the model achieves a predefined performance and, in some cases, stops the training phase.
3. Verified Learning Model(s). Models that have been evaluated using separate datasets (verification dataset) to assess their generalization capabilities and identify potential issues.

Table 1 presents the inputs and outputs associated with each step of Learning Management, which will be elaborated in the subsequent sections: Section 5 guides the development of the learning requirements specifications. Sections 6 and 7 guide model design and training, respectively. Section 8 evaluates the trained model and Section 9 conducts the model verification to ascertain whether the datasets meet the data requirements stated in Section 5. Finally, Sections 0 and 11 collect the acronyms and bibliography relevant to this document, respectively.

Table 1. Inputs and outputs of each step of the Learning Management phase

Phase	Step	Inputs	Outputs	Corresponding templates
PhLM Learning Management	Learning Requirements Specifications	*REF_Ph2D0001_DL_Requirements_Specifications* *REF_Ph2D0005_DL_Component_Description*	*REF_PhLMD0001_Learning_Requirements_Specifications* *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests* *REF_PhLMD0007_Learning_Requirements_Verification_Tests*	*PhLMT0001_Learning_Requirements_Specifications_template* *Ph0T0009_Test_definition_and_results_template*
		REF_PhLMD0001_Learning_Requirements_Specifications *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests* *REF_PhLMD0007_Learning_Requirements_Verification_Tests*	*REF_PhLMD0002_Learning_Requirements_Specifications_IR* *REF_PhLMD0006_Learning_Requirements_Evaluation_Tests_IR* *REF_PhLMD0008_Learning_Requirements_Verification_Tests_IR*	*PhLMT0001_Learning_Requirements_Specifications_template_IR* *Ph0T0009_Test_definition_and_results_template_IR*
	Model Design	*REF_PhLMD0001_Learning_Requirements_Specifications*	*REF_PhLMD0003_Model_Election_Log*	*PhLMT0002_Model_Election_Log_template*
		REF_PhLMD0003_Model_Election_Log	*REF_PhLMD0004_Model_Election_Log_IR*	*PhLMT0002_Model_Election_Log_template_IR*
	Model Training	*REF_PhLMD0003_Model_Election_Log* Training dataset	Trained Model(s)	*There is not a template, it should be considered as an implementation.*
	Model Evaluation	*REF_PhLMD0005_Learning_Requirements_Evaluation_Tests* Trained Model(s) Validation dataset [1]	*REF_PhLMD0005_Learning_Requirements_Evaluation_Tests* Evaluated Model(s)	*Document previously generated*
	Learning Model Verification	*REF_PhLMD0007_Learning_Requirements_Verification_Tests* Evaluated Model(s) Verification dataset	*REF_PhLMD0007_Learning_Requirements_Verification_Test* Verified Learning Model(s)	*Document previously generated*

[1] Although this document maintains the name "validation" according to AI nomenclature, it would not correspond to validation in the context of safety

5 Learning Requirements Specifications

Regarding the learning requirements specifications, this template directly addresses the user to the *PhLMD0001_Learning_Requirements_Specifications.docx* generated from *PhLMT0001_Learning_Requirements_Specifications_template.docx* template which provides a set of instructions and recommendations to carry out the definition of requirements. These requirements shall be collected in the *PhLMD0001_Learning_Requirements_Specifications.docx* document.

Additionally, *PhLMD0001_Learning_Requirements_Specifications.docx* shall include the post-training model selection criteria for those cases in which the several models meet the learning requirements specifications from *PhLMD0001_Learning_Requirements_Specifications.docx*.

According to the tests to be defined in this step, learning requirements evaluation tests shall be stored in the *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx* and *REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx*, respectively. The main distinctions are as follows:

1. The learning requirements evaluation tests shall be defined in *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx.* These evaluation tests determine when the training step is complete.
   It is important to note that since these tests are not verification or validation tasks according to functional safety standards, they do not necessarily need to be performed by a different person or team from those responsible for the design and/or training of the model.
   These tests go beyond evaluating the performance of the model; they also focus on the training process to prevent overfitting, which occurs when the model learns noise from the dataset, and to mitigate data bias. Techniques such as early stopping, regularization, dropout, holdout validation, or cross-validation are employed for this purpose.
   The outcomes of the evaluation process may require iterative adjustments in the Data Management process. For instance, increasing the size of the validation dataset (part of the development dataset) might be considered.
2. The learning requirements verification tests to be defined in the *REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx* document evaluate whether the model can generalize effectively to new, unseen data and provide accurate predictions within a representative dataset of the Operational Design Domain (ODD).

6 Model Design

This step focuses on the specification of a set of candidate DL models that best suit the application. In this step, the *REF_PhLMD0003_Model_Election_Log.docx* document generated from the *PhLMT0002_Model_Election_log.docx* template must be fulfilled. This involves documenting all the elections conducted within this process. If additional information is required, the template can be tailored to suit those needs. In cases where multiple models are designed, each distinct model should be meticulously documented within the same document.

The choice of the most appropriate candidate model for the problem before training is often based on a previous analysis of the state of the art and the DL designer. This process begins with analyzing well-known models that have succeeded in similar task domains. Among the aspects to consider in the selection and development of the model, we list the following ones:

- **Model architecture:** Choose a well-suited model architecture for your data type. A data scientist could use his expertise to choose a model type considering the use case, model complexity, resource availability, and deployment scenario, among others. For example, Convolutional Neural Networks (CNNs) are often used for image data due to their ability to automatically extract meaningful features from images; Recurrent Neural Networks (RNNs) are often used for tasks like action recognition where temporal information matters because they use internal memory state that allows them to capture

the temporal dependencies; Autoencoders, an unsupervised learning technique for neural networks, are often used for anomaly detection because they can learn a compact representation of normal data and are sensitive to deviations from this learned representation or Generative Adversarial Networks (GANs) which are often used for generating realistic synthetic data or enhancing image quality.

- **Initial model weights configuration.** It focuses on how the initial weights of a model are set before training begins. This is an important aspect as it can significantly impact the model´s ability to learn and converge to an optimal solution. There are different strategies to follow for the initial weights configuration. For example:
  - **Random initialization**: weights are initialized with small random values. The main disadvantage is that randomly initialized weights might be far from the optimal values, resulting in slower convergence during training. Consequently, the model might require more epochs to reach a satisfactory level of accuracy, thereby increasing training time and computational cost.
  - **Pretrained models**: Consider using a pre-trained model that has been trained on a similar dataset for a similar task. Pretrained models can save time and resources by providing a good starting point for training. However, their use can lead to unexpected model behavior if the training data employed to develop these pre-trained models does not meet the data requirements specifications of the specific application. For example, it must be ensured that training data is representative of the application-specific Operational Design Domain (ODD), or the data preparation process has been performed according to the specifications. Additionally, it poses additional challenges in the data management phase, which in these cases shall collect data that complements the training dataset employed for the pre-trained model to ensure requirements such as balancing of the classes presented in the dataset. Then, the team leading the data management phase must have access to and be familiar with this dataset.
- **Hyperparameter tuning**: It involves exploring different combinations of hyperparameters to find the combination that produces the best performance on a validation dataset. Among the hyperparameters to be defined, we can list the following:
  - **Type of activation function**: For example, regression, binary classification, multiclass classification…
  - **Number of nodes** (**neurons**) and **hidden layers**. This configuration directly affects the network's capacity to learn intricate patterns and its ability to generalize to new, unseen data. A larger number of neurons or layers may enable the model to capture intricate features but could also lead to overfitting, while a smaller network might struggle to learn complex relationships.
  - **Type of optimization algorithm:** Batch gradient descent, Stochastic gradient descent, Mini-batch gradient descent, Gradient descent with momentum, Adam, Adagrad, Adadelta, Adamax…
  - **Learning rate-α**: The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Examples:
    - $\alpha = \frac{\alpha_0}{1+(decay\_rate * epoch\_number)}$
    - $\alpha = \frac{k}{(epoch\_number/2) * \alpha_0}$
  - **Type of loss function:** Mean Squared Error (MSE), Mean Absolute Error (MAE), Huber Loss, Binary Cross-entropy, Multi-class Cross-entropy/categorical Cross-entropy…
  - **Batch size** refers to the number of training instances in the batch or the number of instances used per gradient update (each update equivalent to an iteration).
  - **Epochs:** number of times the model evaluates the entire training dataset
  - **Training steps (iterations) per epoch**. This is normally defined following the next equation: (size of the entire dataset /batch size) +1, but it can be different based on the designer's expertise.
  - **Optimizer**: The optimizer steers the model toward the optimal set of weights and biases by minimizing the chosen loss function. A spectrum of optimizers, including Stochastic Gradient Descent (SGD) [1], ADAM [2], RMSProp [3], and others, are available, each with its unique

approach to adjusting the model's parameters. The choice of optimizer can significantly impact the model's convergence speed and its ability to escape local minima.
The process of obtaining the best performance of the model can be done manually by adjusting hyperparameters and evaluating the model's performance on the validation dataset or by using automated methods such as grid search, random search, or Bayesian optimization [4]. Regarding hyperparameter tuning, the following aspects must be taken into account:

- Technique to use: Grid Search, Random Search, etc.
- Parameters to tune: decision tree depth, activation function in neural networks, etc.

This step is often subject to modifications since Learning Management is an iterative process. Consequently, successive iterations result in incremental versions of this documentation. To maintain a record of these evolving versions, it is recommended to archive them within the designated "*Learning Management*" folder. This practice ensures that the iterative progress is well-documented and can be tracked effectively.

7 Model Training

The third step entails generating the candidate DL model(s) as specified in the previous step. This step depends on completing the first Data Analysis step from the Data Management process guaranteeing the fulfilment of the data requirements specifications. This is because the model training process draws from the development dataset as input. During this step, the candidate model(s) are trained employing the training dataset derived from the development dataset.

8 Model Evaluation

The performance of the resultant models is subsequently assessed using the evaluation dataset included in the development dataset. This evaluation process involves applying the entire set of tests defined in *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx* for all the candidate models. All the results obtained must be documented in the same *REF_PhLMD0005_Learning_Requirements_Evaluation_Tests.docx* document. The model evaluation step focuses on the DL performance requirements from the *REF_PhLMD0001_Learning_Requirements_Specifications.docx* document.

Throughout the evaluation step, a scenario may arise in which none of the previous candidates achieve the expected performance. In such cases, an iterative repetition of the model design, training and evaluation (steps defined in Sections 6, 7, and 8, respectively) becomes necessary. These iterations continue until the stipulated performance requirements are successfully met. If they are not met, a new iteration of the Data Management process should be carried out. If multiple candidate models demonstrate the anticipated performance levels, all of them will be evaluated in the model verification step (subsequent one).

9 Model Verification

The last step of this stage is model verification. This step checks that the model meets the requirements defined in the *REF_PhLMD0001_Learning_Requirements_Specifications.docx* document when the verification is performed with data that differs from the development dataset. For that, the model verification employs as input the verification dataset. This verification process involves applying the tests previously defined in the *REF_PhLMD0005_Learning_Requirements_Verification_Tests.docx* document. All the results obtained must be documented within the same document.

Throughout the verification phase, three potential scenarios can arise:

1. If none of the previous candidate models achieves the expected requirements specified in *REF_PhLMD0001_Learning_Requirements_Specifications.docx*, an iterative approach becomes necessary. This iterative process encompasses re-designing, re-training, re-evaluating the model, and

re-verification of the model (that is, repeating the steps defined in Sections 6, 7, 8, and 9, respectively). These iterations continue until the stipulated requirements are successfully met. If they are not achieved, a new iteration of the Data Management process should be carried out.

2. If a single candidate model outperforms its peers and meets the pre-defined requirements, it will be directly selected as the model of choice.
3. If multiple candidate models demonstrate the anticipated performance levels, the post-training model selection criteria defined in *REF_PhLMD0001_Learning_Requirements_Specifications.docx* will be employed. The criteria guiding the selection of the best DL model should be documented in the *REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx* document, adhering to the post-training model selection criteria outlined in *REF_PhLMD0001_Learning_Requirements_Specifications.docx*.

***Reminder**:*

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

- The status of the tests (Not done/Pass/Fail) must be updated in the REF_Ph0D0009_AI_Log_of_Tests.docx.

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

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

- The tests results must be documented in REF_PhLMD0007_Learning_Requirements_Verification_Tests.docx

10 Acronyms and Abbreviations

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

- AI – Artificial Intelligence
- AI-FSM – Artificial Intelligence – Functional Safety Management
- CNN – Convolutional Neural Networks
- DL – Deep Learning
- FSM – Functional Safety Management
- GAN – Generative Adversarial Networks
- HPC – High Performance Computing
- KPI – Key Performance Indicator
- MAE – Mean Absolute Error
- MSE – Mean Squared Error
- ODD – Operational Design Domain
- RNN – Recurrent Neural Networks
- SGD – Stochastic Gradient Descent

11 Bibliography

[1] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv,* 2016.

[2] D. P. K. a. J. Ba, "ADAM: A Method for Stochastic Optimization," *arXiv,* 2017.

[3] V. Bushave, "Understanding RMSprop — faster neural network learning," 2018.

[4] P. I. Frazier, "A Tutorial on Bayesian Optimization," *arXiv,* 2018.