Safe and Explainable
Critical Embedded Systems based on AI

PhIMT0001 Inference Requirements Specifications

Version 2.0

Documentation Information

Contract Number	101069595
Project Website	www.safexplain.eu
Contratual Deadline	31.04.2024
Dissemination Level	SEN
Nature	R
Author	Javier Fernández
Modified by	Lorea Belategi
Reviewed by	Irune Agirre
Approved by	Irune Agirre
Keywords	AI, Functional Safety, Inference Management, Inference 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	13/11/2023	Modifications and improvements based on internal review
V0.1	08/10/2023	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 facilitate/guide the specification and collection of the inference requirements.

3 Scope

This document applies to the inference requirements specification of the Inference Management phase performed through the Artificial Intelligence - Functional Safety Management (AI-FSM). These requirements encompass safety, operational, functional and non-functional 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 required for each inference requirement are defined in the following table.

Table 1. 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 **INFRS** (*INFerence Requirement Specification*). Depends 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***: This 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 inference requirements case, the Inference 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 Inference Requirements Specifications

This document shall define the requirements associated with the inference of the DL model. The requirements for deploying the verified learning model in the final inference platform may require conversions and optimizations to meet them. Therefore, deployment requirements are specified first, and then the conversion and optimization needs are specified.

5.1 Requirements Associated with Model Deployment

Some requirements associated with the runtime operation from the Learning requirements and DL requirements shall refined. Among them:

1. Computing power associated with the inference for ensuring compatibility of the verified learning model from Learning Management phase with the deployment environment.
2. Execution time restrictions. Establish restrictions on execution time, outlining the maximum duration allowed for the model to make predictions efficiently.
3. Memory limitations. Address memory limitations of the underlying platform, necessitating the minimization of the model's footprint to optimize resource utilization or the application of model optimizations.
4. Allocation of resources. Define the allocation of resources to the DL constituent, such as specifying tensor core utilization in GPUs or CPUs in multicores.
5. Independence of execution of the model. Define spatial and temporal isolation requirements for the DL model, if any.

5.2 Requirements Associated with Model Conversion

After finishing the Learning Management phase, the verified learning model shall be converted to a format compatible with the inference framework and the resources (both hardware and software) of the underlying platform in which the model will be implemented. Any differences between the verified learning model and the inference model shall be identified and documented. The following may be specified:

1. Platform where the model is executed: including the hardware and the software stack.
2. Choice of the computer arithmetic. Specify the arithmetic representation of numbers and operations (for instance, integer, floating-point, or fixed-point arithmetic) while seeking a trade-off between accuracy and computational efficiency for the specific target platform. For example, employing fixed-point arithmetic instead of floating-point arithmetic can be more resource-efficient, especially on traditional embedded platforms.
3. Software dependencies: including libraries, operating systems, and frameworks, which might vary between the training and deployment platforms.
4. Toolset differences: Identification and documentation of differences in tools, such as compilers, used during the learning and inference phases.

In cases in which the training and inference are performed on the same platform, this conversion is not necessary.

5.3 Requirements Associated with Model Optimization

On certain occasions, model optimization is sometimes essential to meet inference requirements such as execution time, datatype compatibility with the platform, and memory constraints. During this phase, it is crucial to identify and list potential optimizations allowed in the inference stage, including, but not limited to, the following:

- Model quantization: For instance, the reduction of model variable precision from 32-bit floating point to 8-bit integers. This not only diminishes memory usage but also expedites inference, thereby reducing execution time. Among the specifications are the following:
    - Quantization scheme: For instance, symmetric quantization.
    - Framework: The same framework used for training will be utilized for quantification, provided that the chosen framework offers tools and methods for effective model quantization.
    - Technique for quantization: Specify the techniques to be applied, such as weight quantization or integer quantization.
    - Techniques to Recover Accuracy: Specify the strategy to be followed in case of reduction of the accuracy, for instance, Quantization-Aware Training.
- Model pruning. For example, the application of pruning techniques to identify unimportant weights or neurons from the model and, consequently, reduce the memory footprint and the computational requirements. Among the specifications are the following:
    - Pruning criteria: The pruning criteria determine the significance of a group of parameters, deciding whether these parameters should undergo pruning. As an example of requirements to be defined, we can list the threshold (local, global) or the magnitude (weight, gradient…).
    - Pruning patterns: Specifies the arrangement or structure of elements intentionally preserved and not set to zero. For instance, it shall be defined that the pruning pattern shall be applied at the element-wise, vector-wise, or block-wise level.
- Optimized frameworks: Utilizing frameworks optimized for inference operations, such as TensorFlow Lite, TensorFlow Lite Micro, or ONNX runtime, to streamline processes.
- Sparsity Techniques: Incorporating techniques that introduce sparsity in the model, such as sparse convolutions or sparse matrix representations.

6 Inference Requirements Overview

To obtain a comprehensive overview of all the requirements established in this phase, it is essential to collect them in Table 2:

Table 2. Inference 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, data and learning requirements specifications with respect to the inference 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
- OoD – Out-of-distribution
- DL – Deep Learning
- TPU – Tensor Processing Units

8 Bibliography

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