SAFEXPLAIN

Safe and Explainable
Critical Embedded Systems based on AI

A Tale of Machine Learning Process Models

ASPICE Machine Learning Engineering (MLE) *vs* SAFEXPLAIN AI Functional Safety Management (AI-FSM)

Jaume Abella[1], Irune Agirre[2], Javier Fernandez[2], Lorea Belategi[2], Carlo Donzella[3], Giuseppe Nicosia[3], Francesca Guerrini[3]

[1] Barcelona Supercomputing Center, Spain

[2] Ikerlan Technology Research Centre, Basque Research and Technology Alliance (BRTA), Spain

[3] Exida Development s.r.l. , Italy

Funded by
the European Union

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

SAFEXPLAIN (HORIZON project) already presented at SPIN 2023

- The scene
  - **Critical Embedded Systems (CES)** increasingly rely on Artificial Intelligence (AI): automotive, space, railway, avionics, etc.
  - CES must undergo **certification/qualification**
  - AI apparently at odds with traditional functional safety **certification/qualification** processes and methods (such as those of ISO 61508, ISO 26262, SOTIF, etc.)
- SAFEXPLAIN ambition: architecting DL solutions **enabling certification/qualification**
  - Making them **explainable and traceable** and **adhere to "safety culture"** through extension of traditional process lifecycle
  - Preserving **high performance**
  - Tailoring solutions to **varying safety requirements**

OCT 2022 – SEP 2025

SAFEXPLAIN

Safe and Explainable
Critical Embedded Systems based on AI

BARCELONA SUPERCOMPUTING CENTER (BSC)
https://www.bsc.es/

IKERLAN, S. Coop (IKR)
https://www.ikerlan.es/

AIKO SRL (AIKO)
https://www.aikospace.com/

RISE RESEARCH INSTITUTES OF SWEDEN AB (RISE)
https://www.ri.se/

NAVINFO EUROPE BV (NAV)
https://www.navinfo.eu/

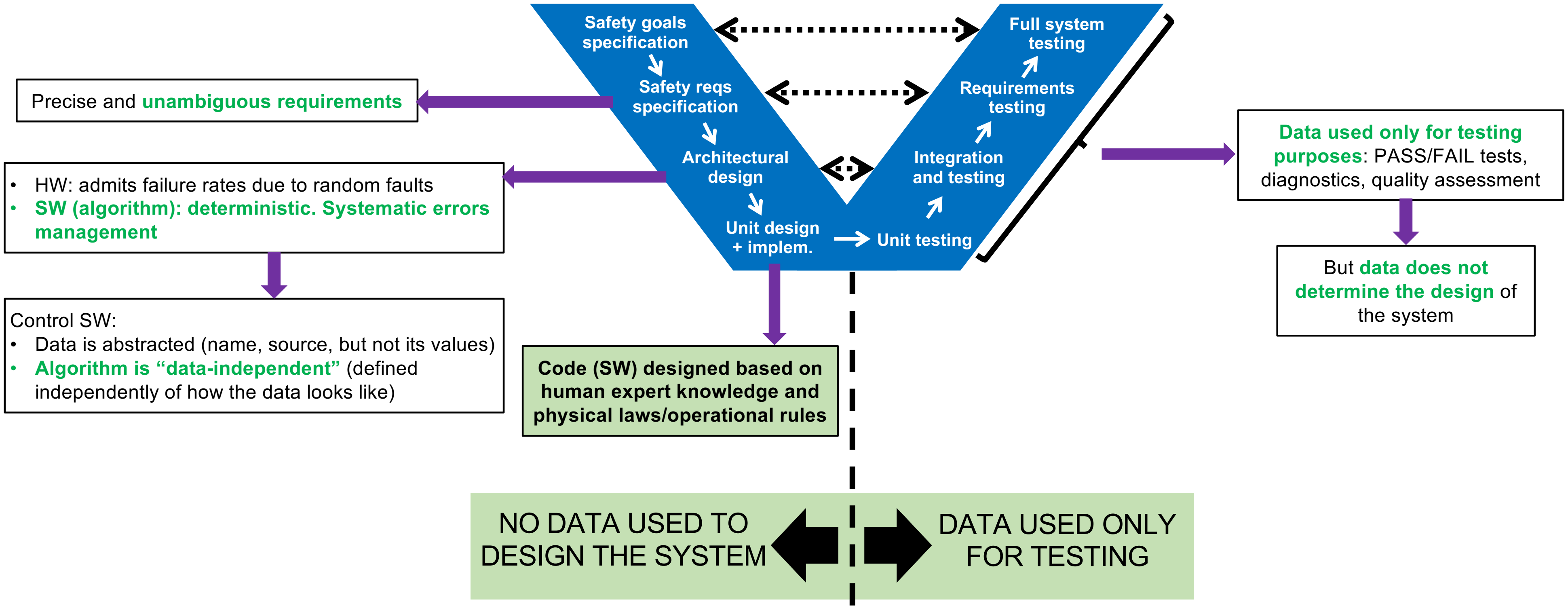
EXIDA DEVELOPMENT SRL (EXI)
https://www.exida-eu.com/

SAFEXPLAIN HAS 5 STATED GOALS…

- At ***SPIN Italia 2023***, presentation was on an early result of **Goal 2**: *Adapt (software) safety lifecycle steps and the architecture of solutions based on DL components so that certification is viable,* that is to say*:*
    - …
    - *V&V model (based on SOTIF, developed by exida, reviewed by IKERLAN)*
    - …
- At ***SPIN Italia 2024***, presentation is on another consolidated results of **Goal 2,** , that is to say*:*
    - …
    - *AI-FSM model (based on IEC 61508, developed by IKERLAN, reviewed by TUV and exida)*
    - …

Safety-related Development Processes

- ISO 26262 software V-model



Safety goals specification

Safety reqs specification

Architectural design

Unit design + implem.

Unit testing

Integration and testing

Requirements testing

Full system testing

Precise and **unambiguous requirements**

- HW: admits failure rates due to random faults
- **SW (algorithm): deterministic. Systematic errors management**

Control SW:
- Data is abstracted (name, source, but not its values)
- **Algorithm is "data-independent"** (defined independently of how the data looks like)

Code (SW) designed based on human expert knowledge and physical laws/operational rules

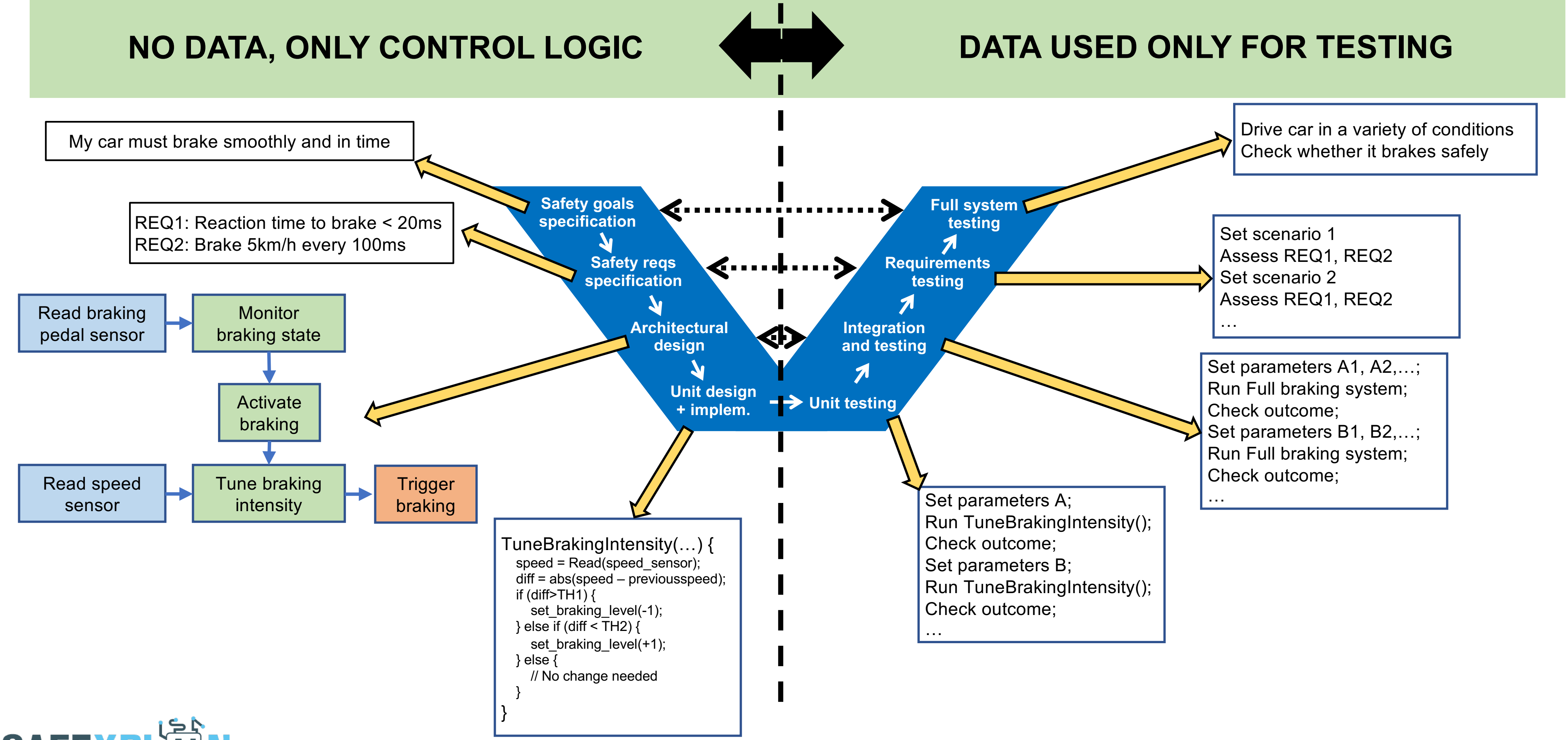
Data used only for testing purposes: PASS/FAIL tests, diagnostics, quality assessment

But **data does not determine the design** of the system

NO DATA USED TO DESIGN THE SYSTEM

DATA USED ONLY FOR TESTING

Logic behind certification/qualification



NO DATA, ONLY CONTROL LOGIC

DATA USED ONLY FOR TESTING

My car must brake smoothly and in time

REQ1: Reaction time to brake < 20ms
REQ2: Brake 5km/h every 100ms

```
TuneBrakingIntensity(…) {
   speed = Read(speed_sensor);
   diff = abs(speed – previousspeed);
   if (diff>TH1) {
      set_braking_level(-1);
   } else if (diff < TH2) {
      set_braking_level(+1);
   } else {
      // No change needed
   }
}
```

Drive car in a variety of conditions
Check whether it brakes safely

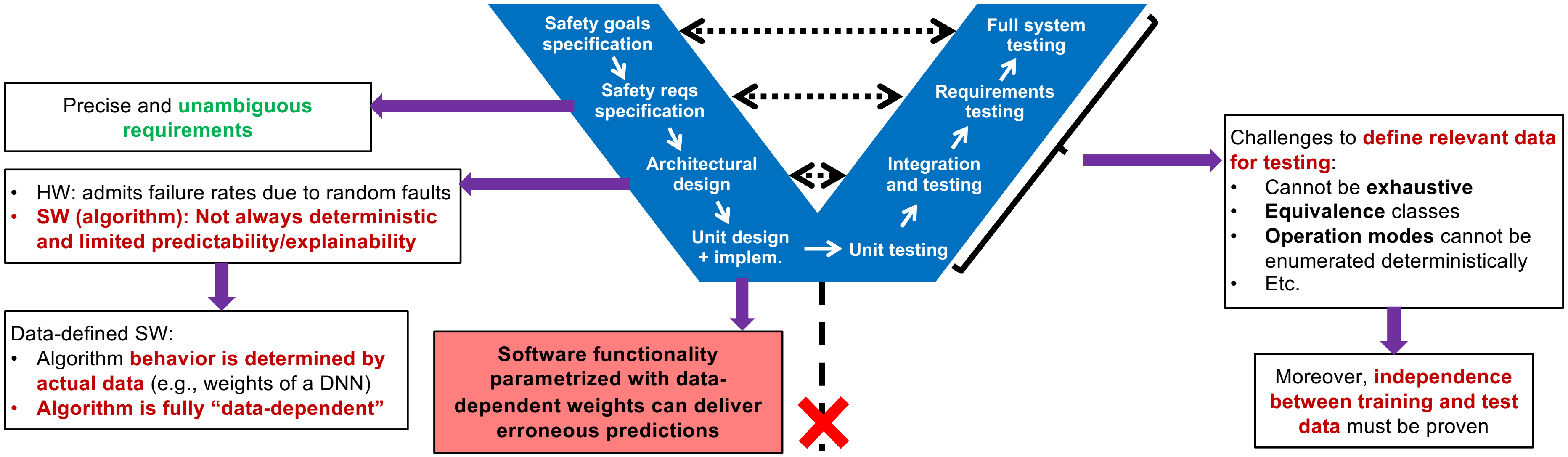
Set scenario 1
Assess REQ1, REQ2
Set scenario 2
Assess REQ1, REQ2
…

Set parameters A1, A2,…;
Run Full braking system;
Check outcome;
Set parameters B1, B2,…;
Run Full braking system;
Check outcome;
…

Set parameters A;
Run TuneBrakingIntensity();
Check outcome;
Set parameters B;
Run TuneBrakingIntensity();
Check outcome;
…

Safety-related Systems Development Process

- AI-related challenges



Precise and **unambiguous requirements**

- HW: admits failure rates due to random faults
- **SW (algorithm): Not always deterministic and limited predictability/explainability**

Data-defined SW:
- Algorithm **behavior is determined by actual data** (e.g., weights of a DNN)
- **Algorithm is fully "data-dependent"**

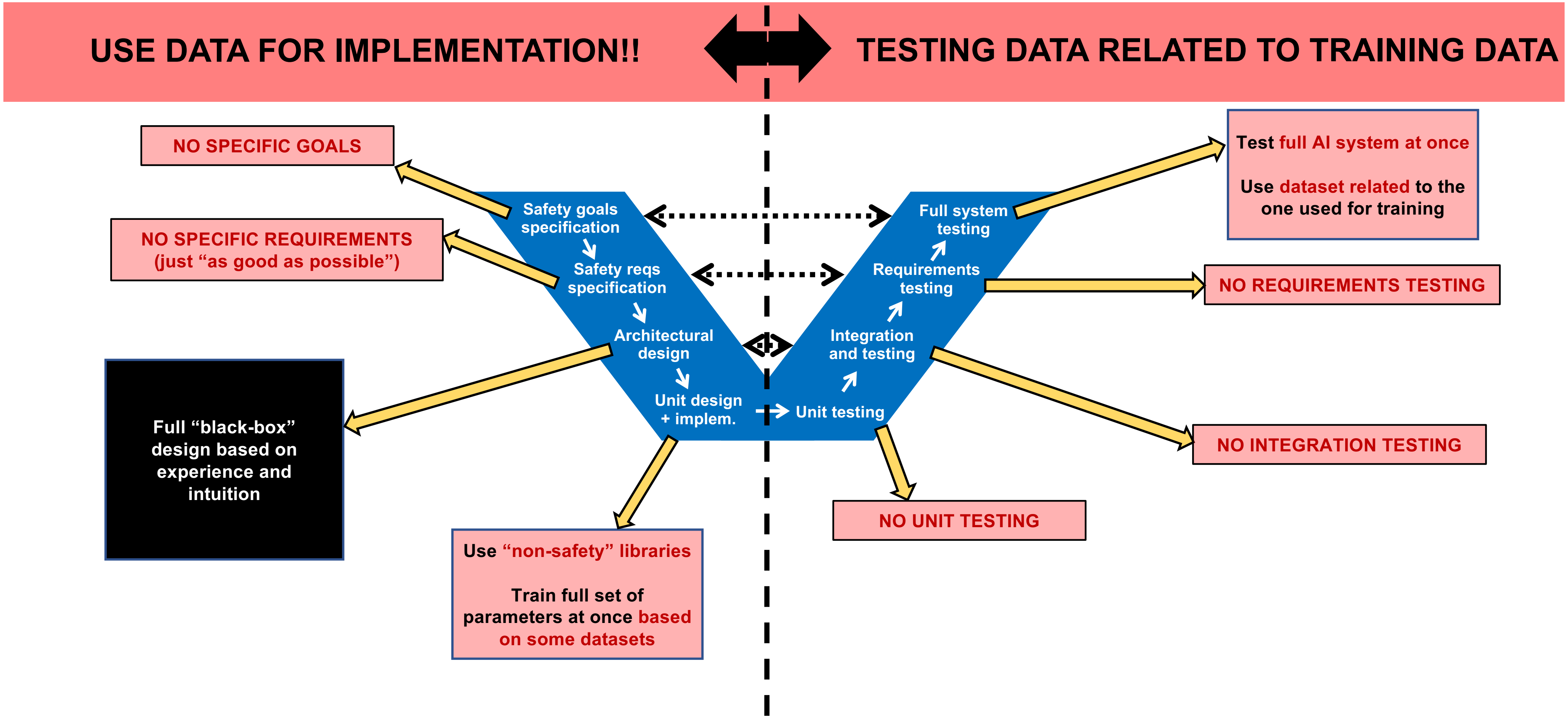
Software functionality parametrized with data-dependent weights can deliver erroneous predictions

Challenges to **define relevant data for testing**:
- Cannot be **exhaustive**
- **Equivalence** classes
- **Operation modes** cannot be enumerated deterministically
- Etc.

Moreover, **independence between training and test data** must be proven

DATA DETERMINES SYSTEM DESIGN

Logic behind AI software design



USE DATA FOR IMPLEMENTATION!!

TESTING DATA RELATED TO TRAINING DATA

NO SPECIFIC GOALS

NO SPECIFIC REQUIREMENTS (just "as good as possible")

Full "black-box" design based on experience and intuition

Use "non-safety" libraries

Train full set of parameters at once based on some datasets

Test full AI system at once

Use dataset related to the one used for training

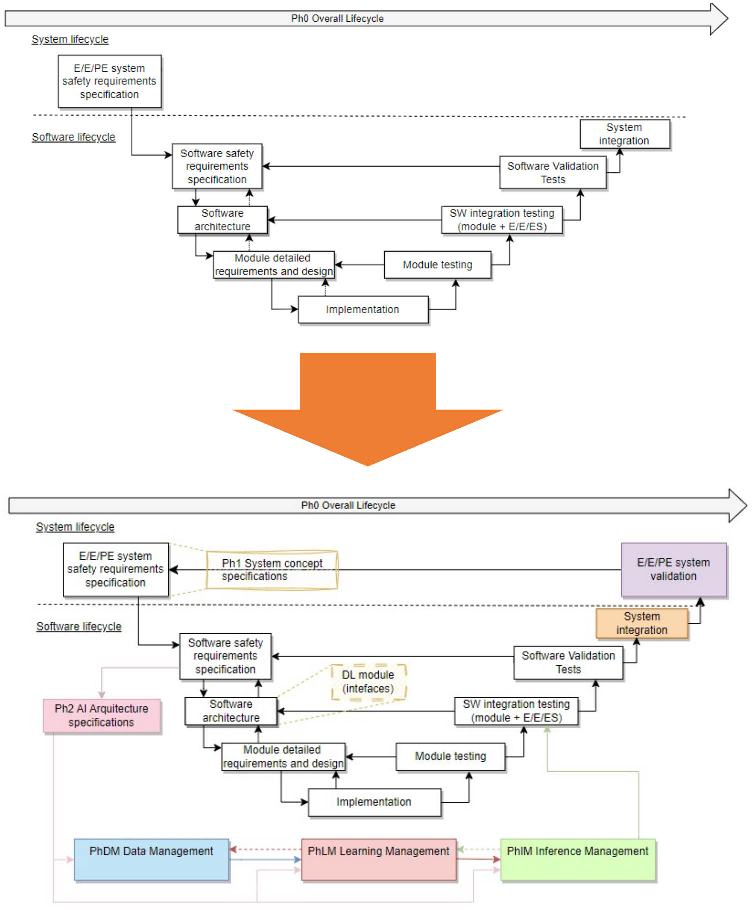
NO REQUIREMENTS TESTING

NO INTEGRATION TESTING

NO UNIT TESTING

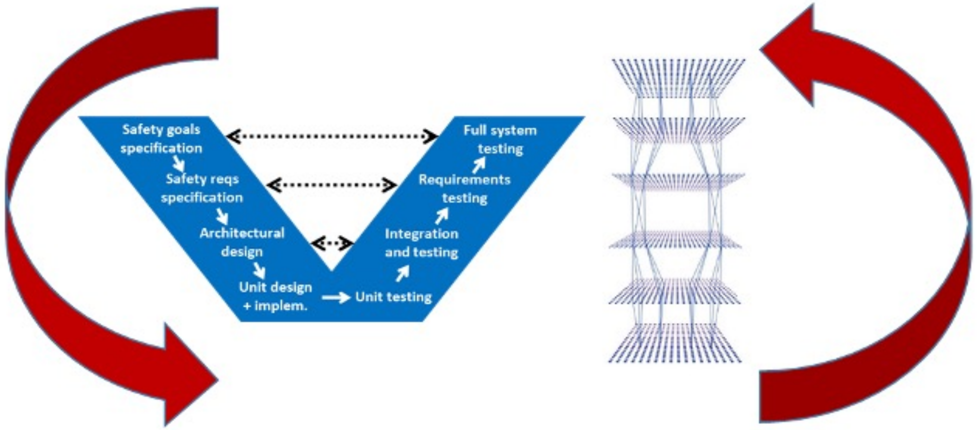
Ambition/objective for SAFEXPLAIN Goal 2

- Re-think safety lifecycle
  - **Keep principles** but with AI implementation in mind
  - Enable the **use of some AI models** first, and generate requirements, goals, unit testing, etc. from there (**bottom-up approach instead of top-down**)
  - Specific steps to **manage data, learning** and **inference**


Ph0 Overall Lifecycle
System lifecycle
E/E/PE system safety requirements specification
Software lifecycle
System integration
Software safety requirements specification
Software Validation Tests
Software architecture
SW integration testing (module + E/E/ES)
Module detailed requirements and design
Module testing
Implementation
Ph0 Overall Lifecycle
System lifecycle
E/E/PE system safety requirements specification
Ph1 System concept specifications
E/E/PE system validation
Software lifecycle
System integration
Software safety requirements specification
Software Validation Tests
DL module (intefaces)
Ph2 AI Arquitecture specifications
Software architecture
SW integration testing (module + E/E/ES)
Module detailed requirements and design
Module testing
Implementation
PhDM Data Management
PhLM Learning Management
PhIM Inference Management

Stated Goal 2/5 of SAFEXPLAIN is explicitly about:

- Adapt software safety lifecycle steps and the architecture of solutions based on DL components so that certification is viable
  - E.g., add additional lifecycle steps to contemplate model training, and adapt requirement specification, data management and testing approaches



Tailor safety life cycle to enable DNN certification

Tailor DNNs to match properties needed by functional safety standards

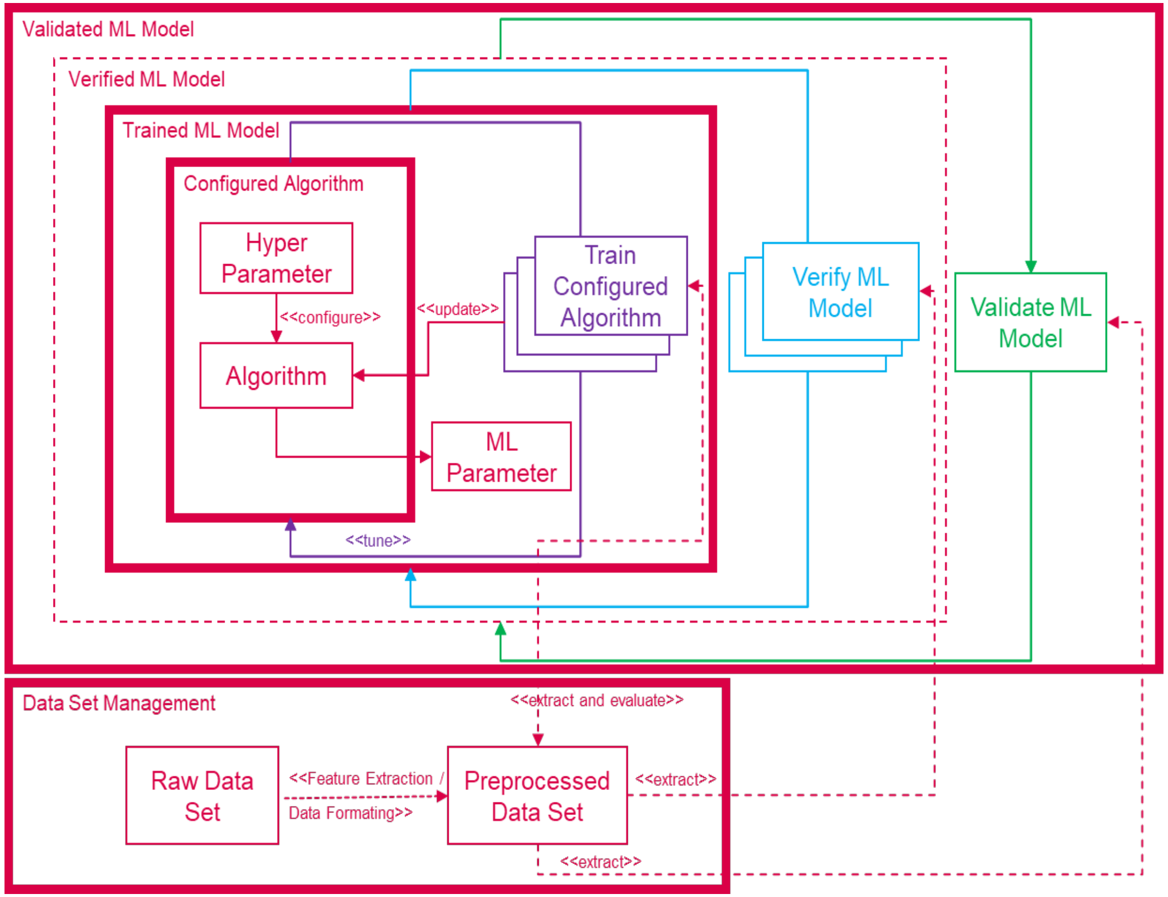
SAFEXPLAIN approach has been inspired by guidelines, papers and models from aerospace and automotive domain

Comparable life-cycle/process models

- **SAFEXPLAIN** approach to the SW development life-cycle (when ML/DL components are integrated in the overall SW architecture) has been initially inspired by guidelines (*EASA Concept Paper: guidance for Level 1 & 2 machine learning applications*) and papers (*Proposing the Use of Hazard Analysis for Machine Learning Data Sets*) from influential aerospace (**EASA** - European Union Aviation Safety Agency and **DEVCOM** - U.S. Army Combat Capabilities Development Command Aviation & Missile Center) stakeholders.
- Thanks to an agreement with the **AK13 of VDA**, an exchange of pre-publication drafts took place in early 2023, allowing **SAFEXPLAIN** to become acquainted with the **MLE Model** now integrated in the recently published draft of the **ASPICE 4.0**.
- The rest of this presentation will focus on the striking similarities of all these approaches in terms of **ML/DL processes** identification and description, despite some significant differences mostly in terms of terminology.

Origin of the (A)SPICE ML Model

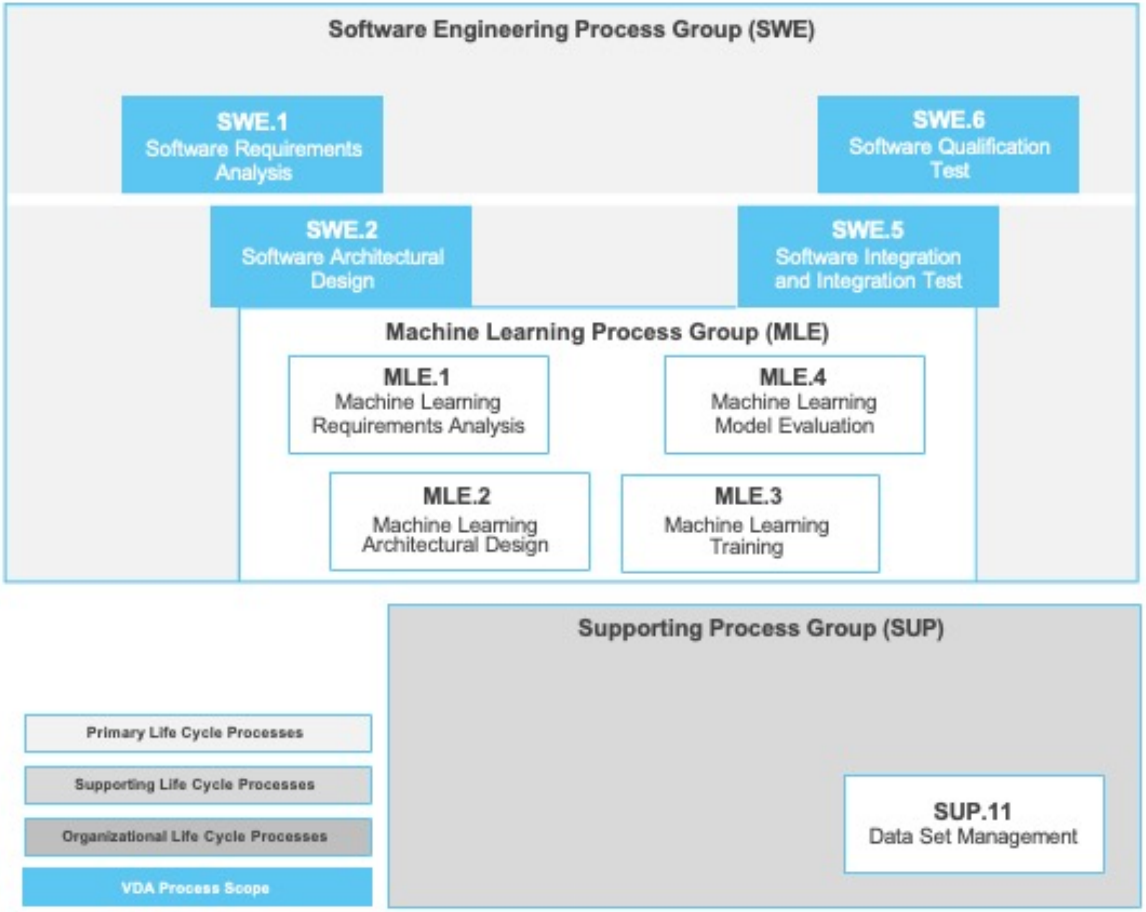
- Originally developed according to the “Plug-in” concept as the Hardware model by a dedicated Working Group
- It started later than other plugins (MECH, HWE) but as ML is affecting many aspects of automotive development it was given a special priority for integration in the full ASPICE 4.0
- Here an early public presentation of the key ML activities…



Origin of the (A)SPICE ML Model

- …and the original idea of "positioning" the **4 new MLE processes** as a distinct "mini-V" taking place of the "tip of the V" in the traditional SWE V-model

- A distinct process belonging to a different process group was created to be in charge of ML **Data Set Management**


Software Engineering Process Group (SWE)
SWE.1 Software Requirements Analysis
SWE.6 Software Qualification Test
SWE.2 Software Architectural Design
SWE.5 Software Integration and Integration Test
Machine Learning Process Group (MLE)
MLE.1 Machine Learning Requirements Analysis
MLE.4 Machine Learning Model Evaluation
MLE.2 Machine Learning Architectural Design
MLE.3 Machine Learning Training
Supporting Process Group (SUP)
Primary Life Cycle Processes
Supporting Life Cycle Processes
Organizational Life Cycle Processes
VDA Process Scope
SUP.11 Data Set Management

Current status of (A)SPICE MLE - integrated in ASPICE PAM 4.0 (I)

- The previous schemes have been further elaborated and finally included into **ASPICE 4.0, Annex C.3 “Integration of Machine Learning Engineering Processes”,** where, expectedly, special relevance is given to the concept of **ML architecture**…

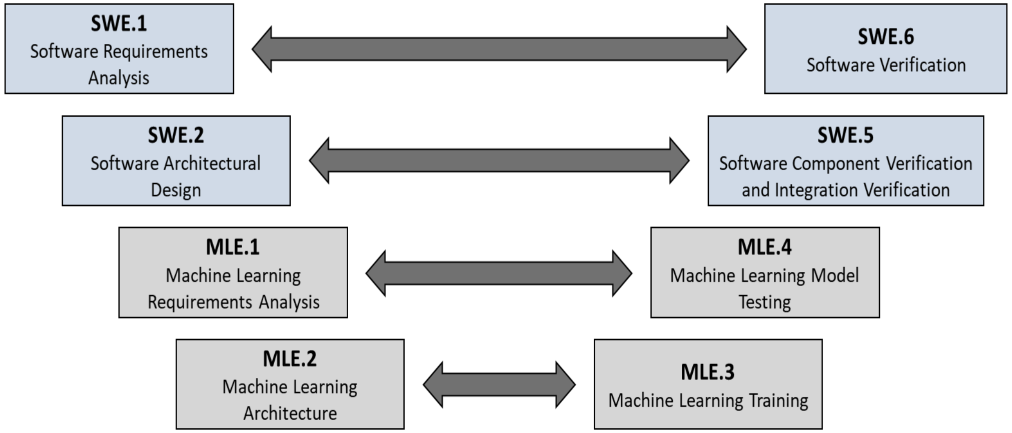

SWE.1
Software Requirements Analysis
SWE.6
Software Verification
SWE.2
Software Architectural Design
SWE.5
Software Component Verification and Integration Verification
MLE.1
Machine Learning Requirements Analysis
MLE.4
Machine Learning Model Testing
MLE.2
Machine Learning Architecture
MLE.3
Machine Learning Training

Figure C.3 — Integration of MLE Processes

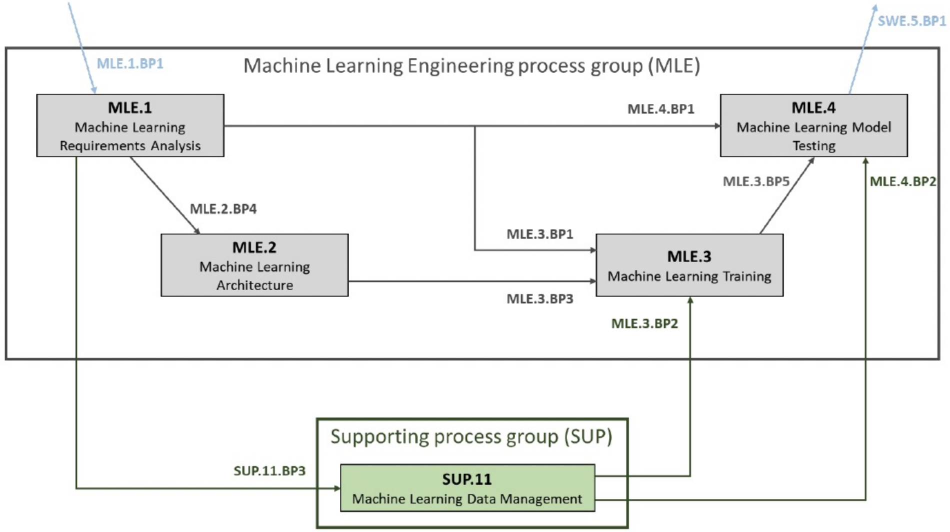

SWE.5.BP1
MLE.1.BP1
Machine Learning Engineering process group (MLE)
MLE.1
Machine Learning Requirements Analysis
MLE.4.BP1
MLE.4
Machine Learning Model Testing
MLE.3.BP5
MLE.4.BP2
MLE.2.BP4
MLE.3.BP1
MLE.2
Machine Learning Architecture
MLE.3
Machine Learning Training
MLE.3.BP3
MLE.3.BP2
Supporting process group (SUP)
SUP.11.BP3
SUP.11
Machine Learning Data Management

Figure C.4 — Interdependencies within MLE and SUP.11

Current status of (A)SPICE MLE - integrated in ASPICE PAM 4.0 (II)

- …with even a specific **example of ML architecture**
- *“ML architecture typically consists of an **ML model and other ML architectural elements**, which are other (classical) software components […] and provided **to train, test, and deploy the ML model.**”*

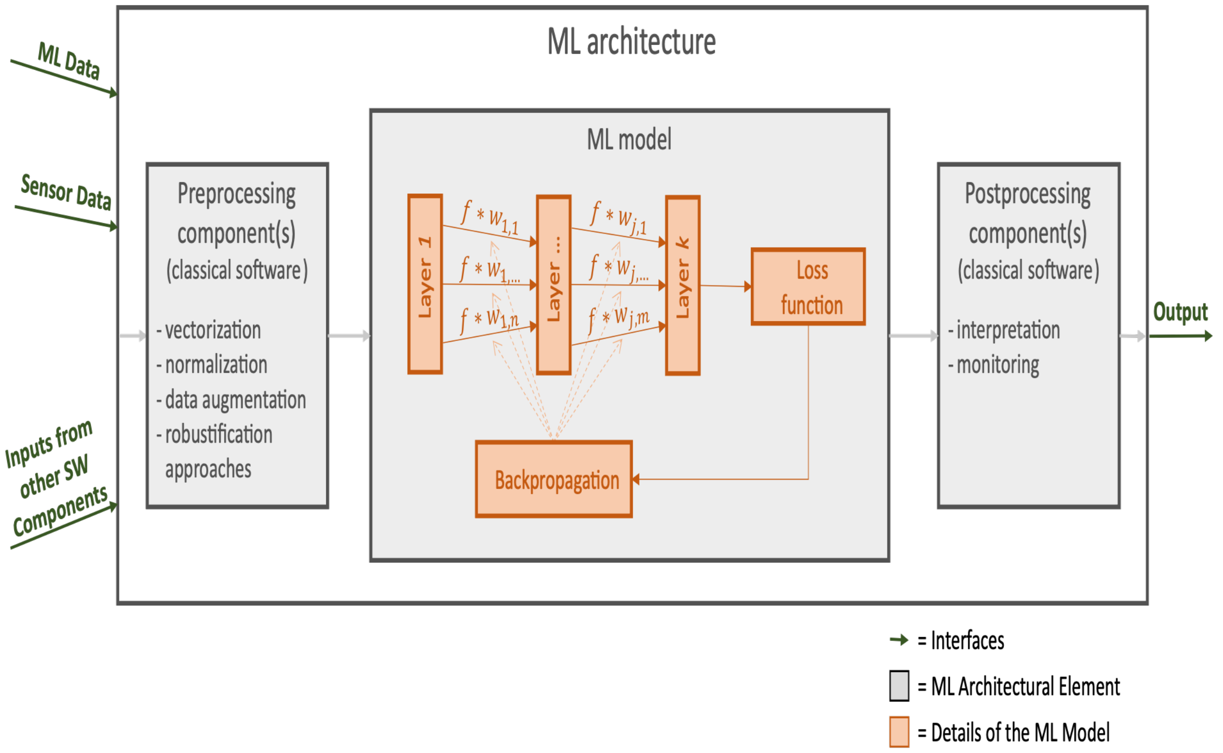


Figure C.5 — Example of an ML Architecture

ASPICE MLE Processes and ‘characterizing’ Information Items

MLE.1 Machine Learning Requirements Analysis

- *(no specific II, but specific* ML requirements *are expected)*

MLE.2 Machine Learning Architecture

- 04-51 ML architecture (includes *01-54 Hyperparameters*)
- 01-54 Hyperparameters

MLE.3 Machine Learning Training

- 08-65 ML training and validation approach (a.k.a. *strategy*)
- 03-51 ML data set
- 01-53 Trained ML model

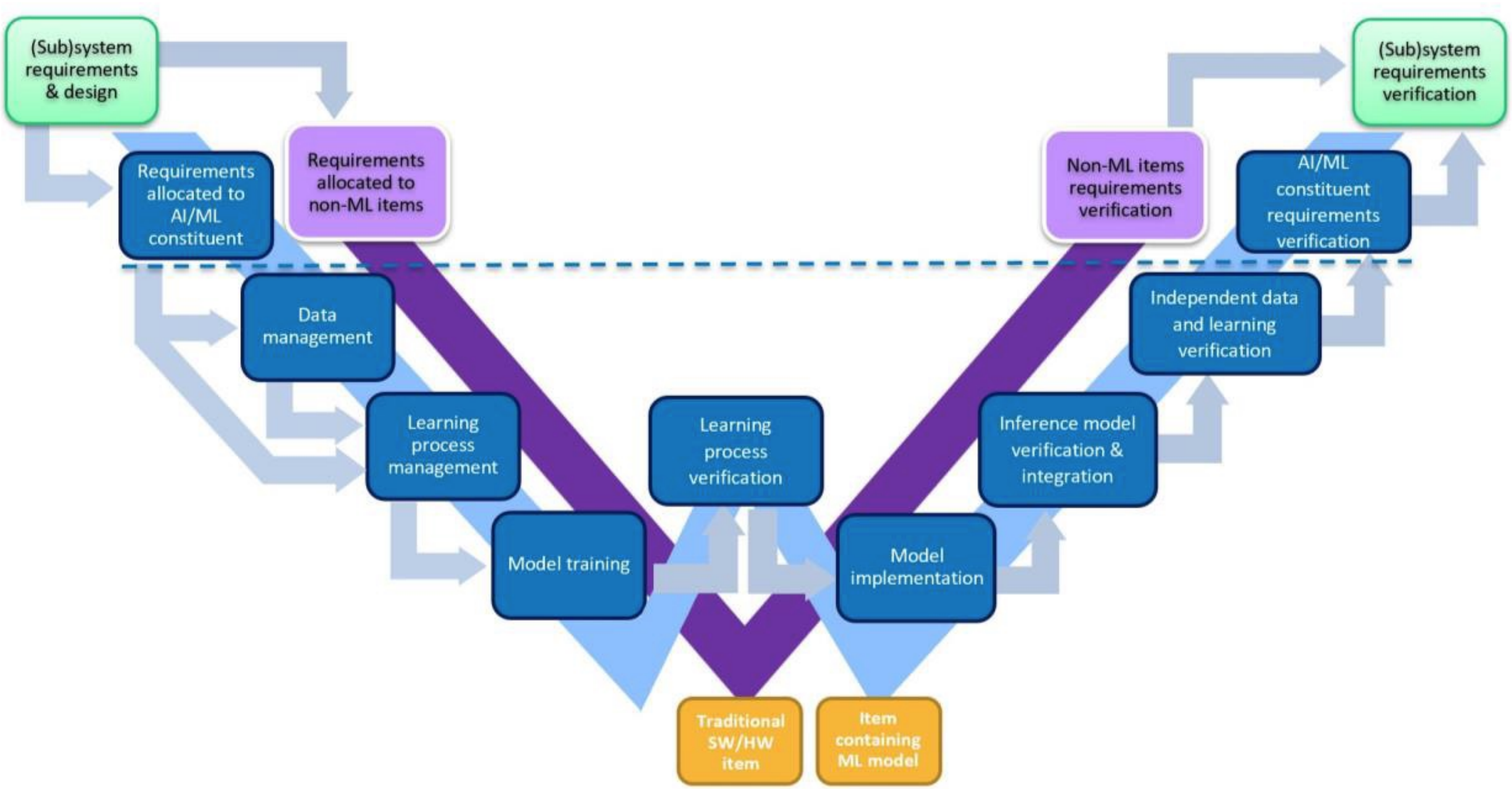
MLE.4 Machine Learning Model Testing

- 08-64 ML test approach (a.k.a. *strategy*)
- 03-51 ML data set
- 11-50 Deployed ML model
- 13-50 ML test results

SUP.11 Machine Learning Data Management

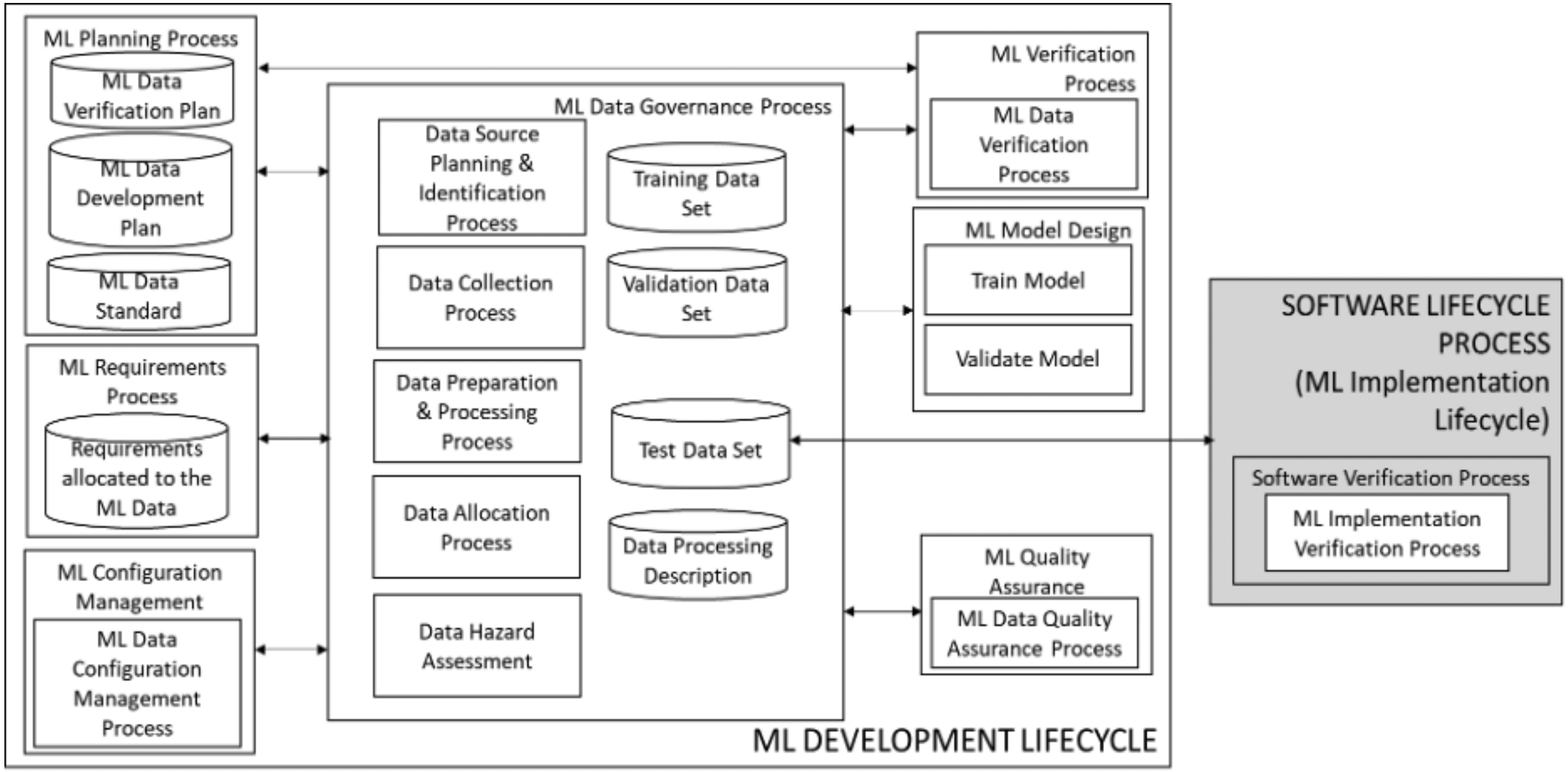
- 19-50 ML data quality approach (a.k.a. *strategy*)
- 16-52 ML data management system (part of CM system)
- 03-53 ML data (all ML-related data, includes *03-51 ML data set*)

Interlude I - EASA Concept Paper: guidance for ML (Feb 2023)


(Sub)system requirements & design
Requirements allocated to AI/ML constituent
Requirements allocated to non-ML items
Data management
Learning process management
Model training
Learning process verification
Model implementation
Inference model verification & integration
Independent data and learning verification
Non-ML items requirements verification
AI/ML constituent requirements verification
(Sub)system requirements verification
Traditional SW/HW item
Item containing ML model

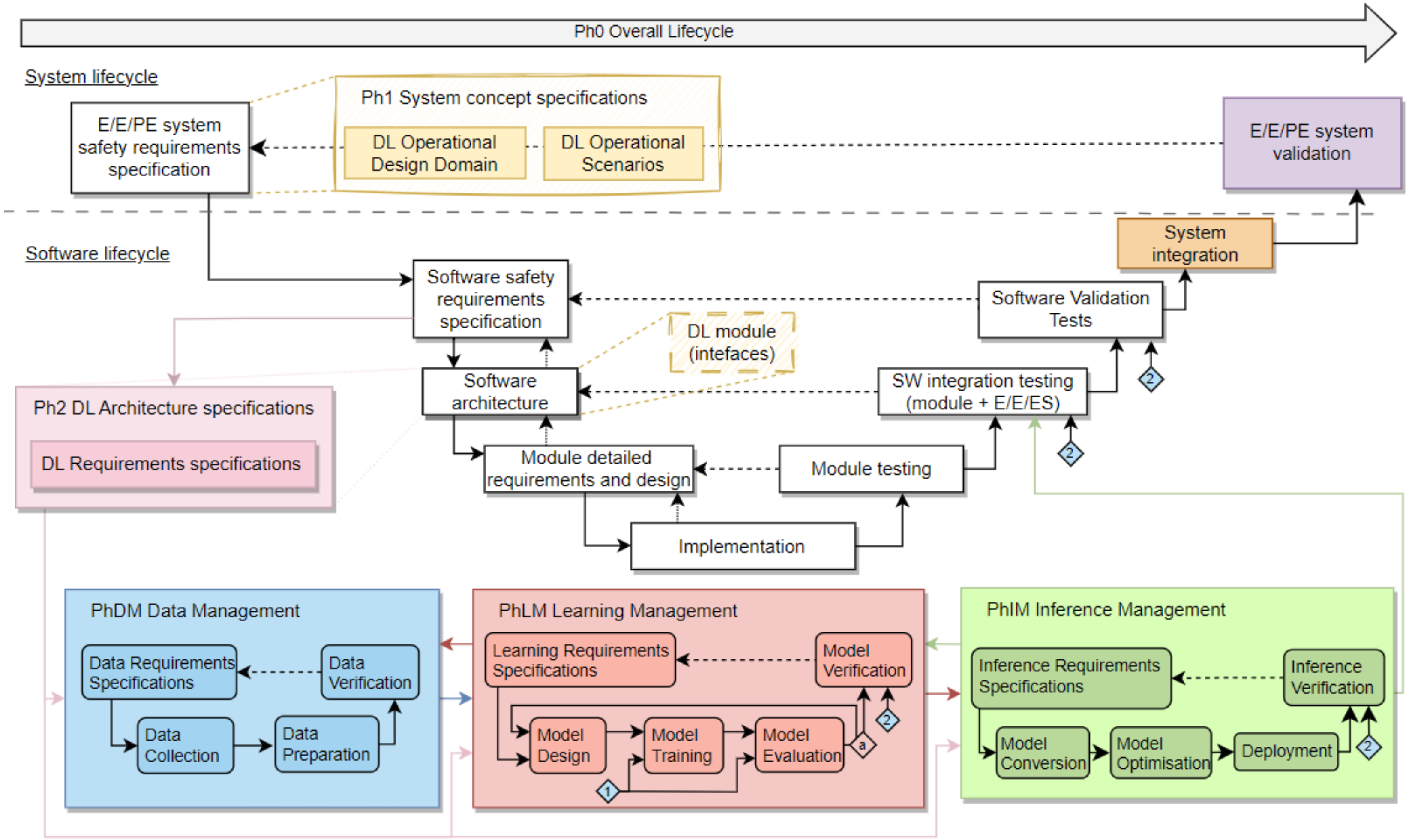
Global view of learning assurance **W-shaped** process, non-AI/ML constituent **V-cycle** process

Interlude II – DEVCOM Paper: ML Data Sets Lifecycle (Summer 2023)


ML Planning Process
ML Data Verification Plan
ML Data Development Plan
ML Data Standard
ML Requirements Process
Requirements allocated to the ML Data
ML Configuration Management
ML Data Configuration Management Process
ML Data Governance Process
Data Source Planning & Identification Process
Data Collection Process
Data Preparation & Processing Process
Data Allocation Process
Data Hazard Assessment
Training Data Set
Validation Data Set
Test Data Set
Data Processing Description
ML Verification Process
ML Data Verification Process
ML Model Design
Train Model
Validate Model
ML Quality Assurance
ML Data Quality Assurance Process
ML DEVELOPMENT LIFECYCLE
SOFTWARE LIFECYCLE PROCESS (ML Implementation Lifecycle)
Software Verification Process
ML Implementation Verification Process

ML Data Governance Process within the overall ML Development Lifecycle

Back to SAFEXPLAIN: AI-FSM as extension of FuSa lifecycle


Ph0 Overall Lifecycle
System lifecycle
E/E/PE system safety requirements specification
Ph1 System concept specifications
DL Operational Design Domain
DL Operational Scenarios
E/E/PE system validation
Software lifecycle
System integration
Software safety requirements specification
Software Validation Tests
DL module (intefaces)
Ph2 DL Architecture specifications
DL Requirements specifications
Software architecture
SW integration testing (module + E/E/ES)
2
2
2
Module detailed requirements and design
Module testing
Implementation
PhDM Data Management
Data Requirements Specifications
Data Verification
Data Collection
Data Preparation
PhLM Learning Management
Learning Requirements Specifications
Model Verification
Model Design
Model Training
Model Evaluation
a
2
1
PhIM Inference Management
Inference Requirements Specifications
Inference Verification
Model Conversion
Model Optimisation
Deployment
2

SAFEXPLAIN

Initial comparison ASPICE / SAFEXPLAIN ML models (I)

SUP.11 *vs* PhDM Data Management

Mapping is quite straightforward between *Practices and IIs* on one side and *Activities and outcomes* on the other

MLE.1 *vs* DL Requirements specifications

Mapping makes clear that all DL requirements are a subset/derived from SW requirements and that Ph2 DL Architecture specifications are there to satisfy those requirements

MLE.2 *vs* Ph2 DL Architecture specifications

Mapping makes clear that all Ph2 DL Architecture specifications are actually design (part of the overall SW architecture), and that needed complementary traditional architectural design descriptions (elements, interfaces…) are expected to be defined

Initial comparison ASPICE / SAFEXPLAIN ML models (II)

MLE.3 *vs* PhLM Learning Management

The “learning requirements specifications” appears to be mappable with the “training and verification/validation approach” and “ML data set”; Trained Model is a common basic outcome

MLE.4 *vs* PhIM Inference Management

The “inference requirements specifications” appears to be mappable with the “ML test approach” and “ML data set”; Deployed Model (i.e., Tested, Re-verified) is a common basic outcome
It is unclear the reason for the major difference in the naming (i.e. “Model Testing” vs “Inference”); please note that in early ASPICE MLE draft MLE.4 is called “ML Model Evaluation”

Some Preliminary Conclusions

- It appears there are no significant gaps in the **SAFEXPLAIN AI-FSM model** to become compliant to the **ASPICE MLE model**; **SAFEXPLAIN consortium** on one side and **VDA-QMS and Intacs** on the other side have expressed strong interest in collaborating towards further alignment
- A big advantage in adopting both approaches is that **SAFEXPLAIN AI-FSM model** (like EASA's guidelines and other draft standards dedicated to "Safe AI") are already incorporating FuSa aspects while the **ASPICE MLE Model** is "pure QM", thereby allowing a process "discipline decomposition", that has proved quite effective with ASPICE and ISO 26262 in the last decade
- By distinguishing "from the start" **Process Quality** aspects from **Functional Safety** aspects of ML/DL applications, a paradigm can be established to be further extended to **Cybersecurity**, too, addressing the most critical pillars of **Trustworthy AI**, according to both of the most important pieces of AI regulation already in place, the **EU AI Act** and the **US President Executive Order** on the **Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence**

SAFEXPLAIN

Safe and Explainable
Critical Embedded Systems based on AI

Follow us on social media:

www.safexplain.eu

Funded by
the European Union

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