Trustworthy AI Standard: Diving into AI Standards for Aviation

Fly AI Forum, April 22-23, 2025

Fateh Kaakai (Thales cortAIx Labs)

Joseph MACHROUH (Thales)

www.thalesgroup.com

Organization of the breakout session

> 2 presentations (40’):

- Fateh Kaakai (Thales): Update on ED-324/ARP6983 and deep dive into the ML Constituent concept
- Joseph Machrouh (Thales): Specifics of instantiating ED-324/ARP6983 in the ATM/ANS domain

> Open Discussion (20’)

Update on ED-324/ARP6983 and deep dive into the ML Constituent concept

Fly AI Forum, April 22-23, 2025

Fateh Kaakai (Thales cortAIx Labs)

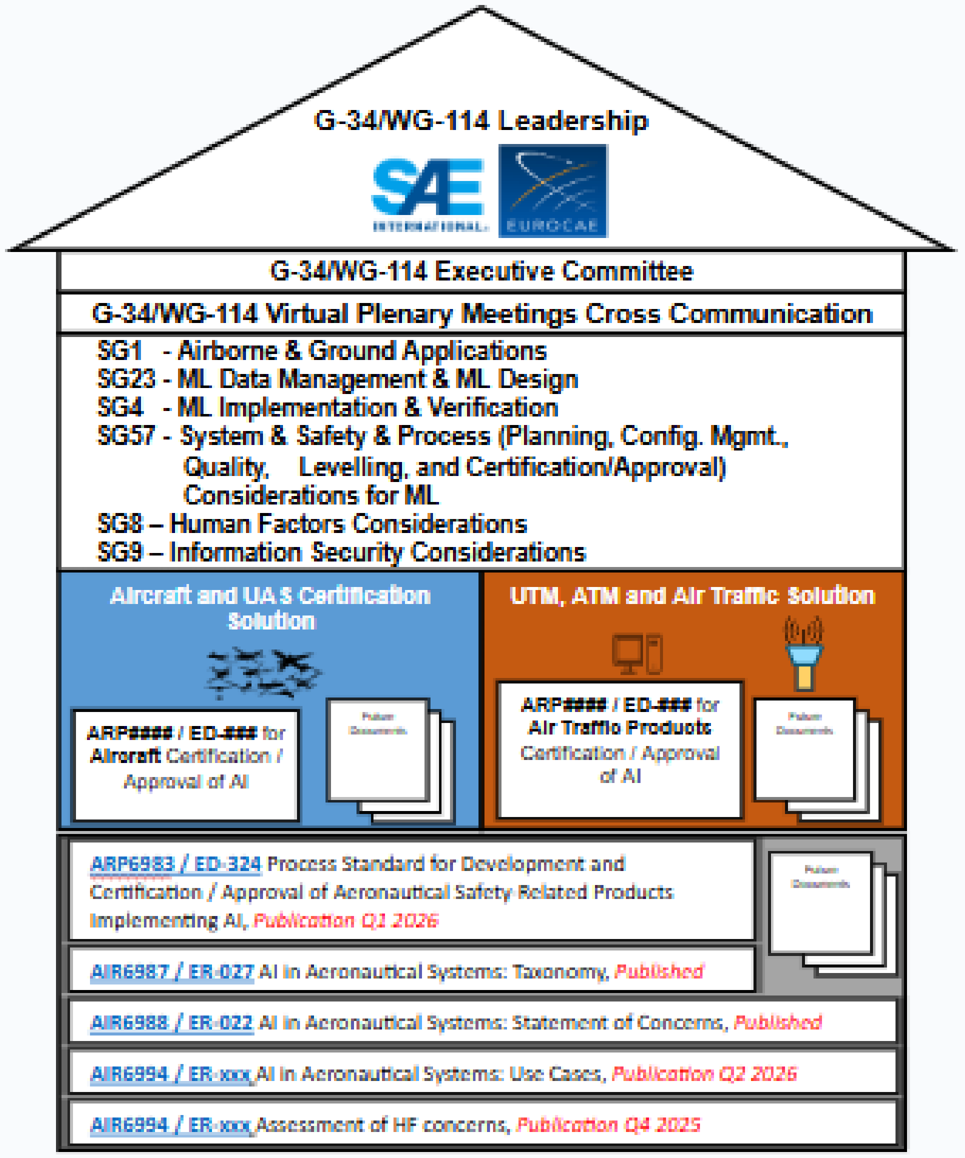
www.thalesgroup.com

About SAE G-34 / EUROCAE WG-114 – Global picture

Joint Standards Committee

- **Created in 2019**
- **2 published documents**
  - ER-022 / AIR 6988 (Statement of concerns)
    Published on 30 April 2021
  - ER-027 / AIR 5987 (Taxonomy)
    Published on 12 December 2024

- **WIPs**
  - ED-324 / ARP 6983 – **Process Standard for Development and Certification / Approval of Aeronautical Safety-Related Products Implementing AI** - Publication planned for Q1 2026
  - ER-0xx / AIR 6994 (Use Cases) - Publication planned for Q2 2026



About SAE G-34 / EUROCAE WG-114 – Leadership, SG Leads, Editors

SAE G-34 / EUROCAE WG-114 Leadership

SAE
- Mark Roboff
- Paula Olivio
- Gary Brown (S)
- David Redman (EIC)

Election

EUROCAE
- Fateh Kaakai
- Sandrine Serres
- Radek Zakrzewski (S, EIC)

SG1

SAE
- Gabriel Pedroza
- Alicija Kwasniewska

EUROCAE
- Alexis de Cacqueray

SG23

SAE
- Shreeder Adibhatla
- Elgiz Baskaya

EUROCAE
- Florence De Grancey
- Konstantin Dmitriev

SG4

SAE
- Darren Cofer
- Tammy Reeve

EUROCAE
- Louis Fabre
- Marc-Emmanuel Coupvent-des-Graviers

SG57

SAE
- Aharon David
- Ganesh Pai
- Misty Davies (S)

EUROCAE
- Giacomo Gentile
- Sandrine Serres
- Ursula Hoffmann (E)

SG8

SAE
- Ricardo Reis

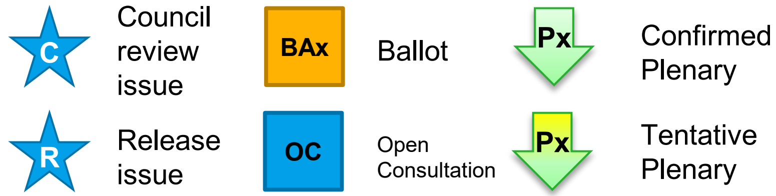
EUROCAE
- Denys Bernard
- Stathis Malakis

SG9

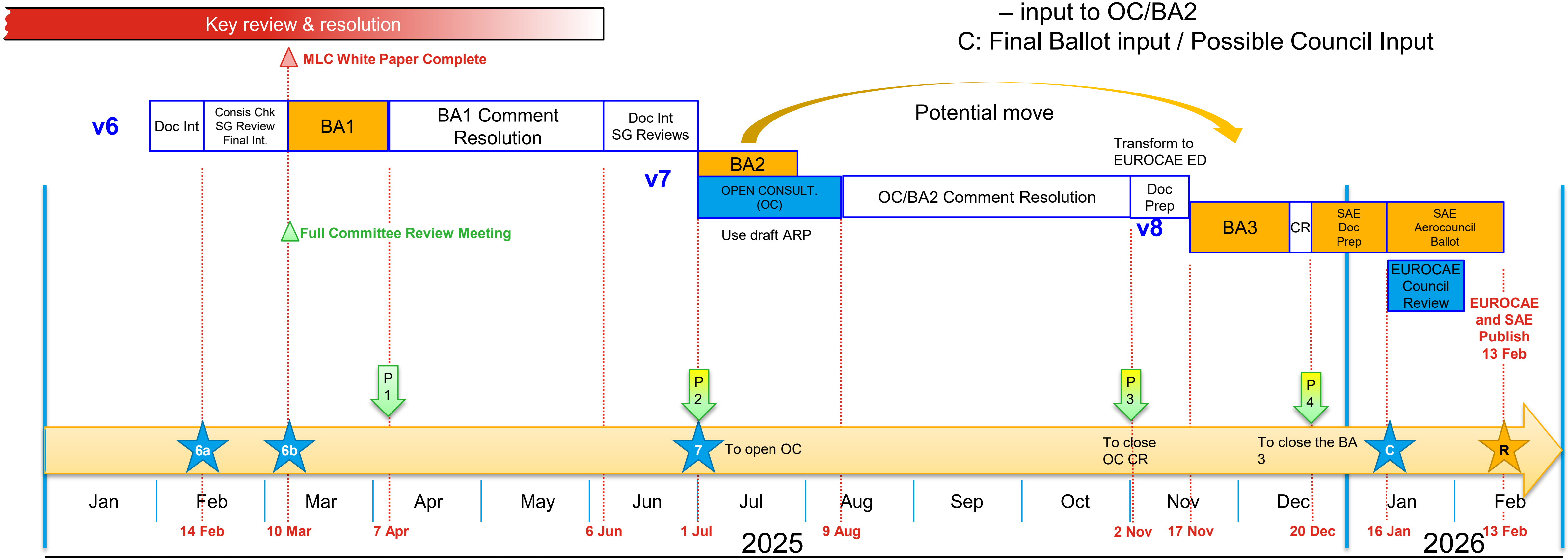
SAE
- Aharon David

EUROCAE
- Stefan Marwedel

Master Schedule (revision 7e - 11032025)

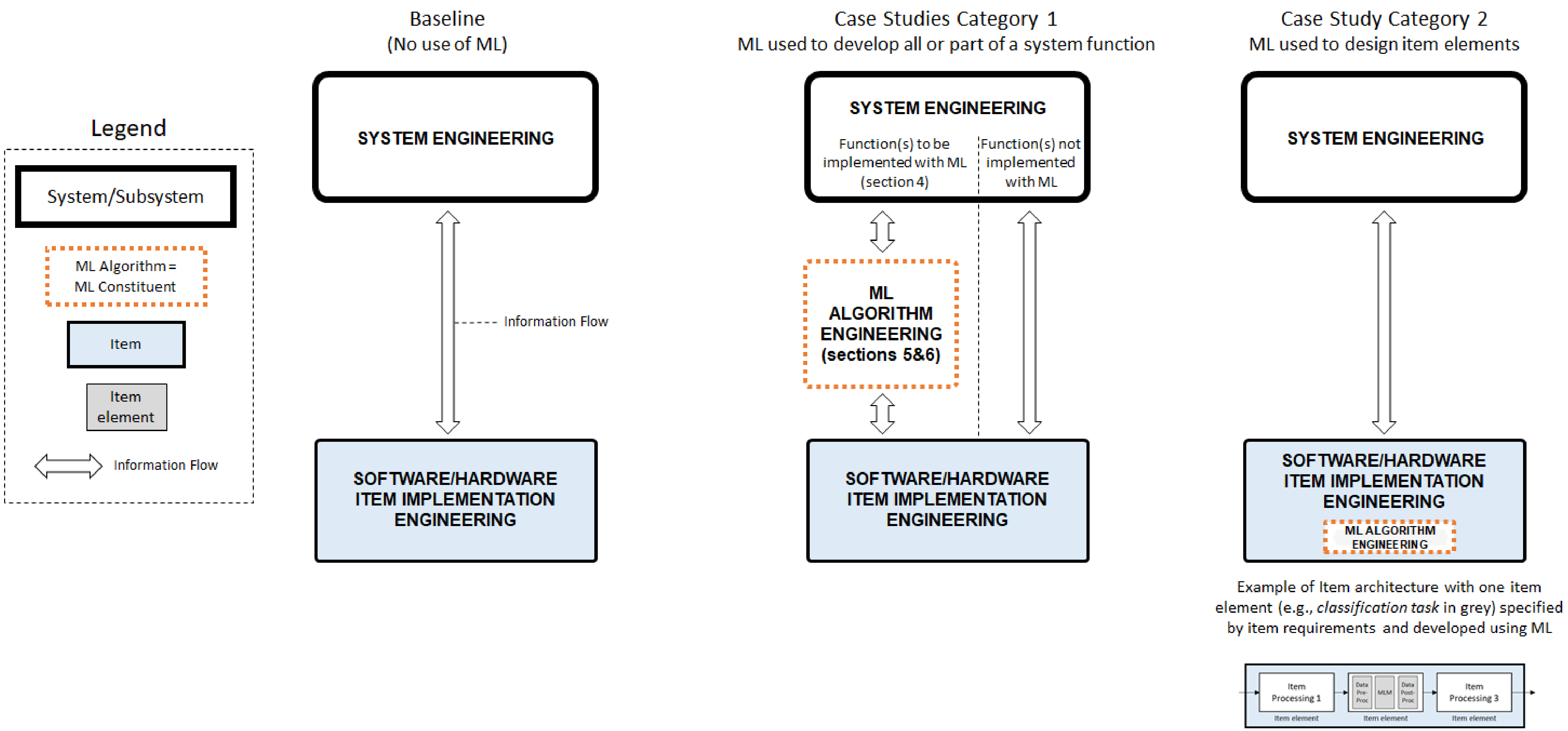


6a: SG revisions integrated into master document
6b: Consistency checks and SG review integrated into document; Grey text managed – input to BA1
7: Ballot 1 comments resolved and integrated; – input to OC/BA2
C: Final Ballot input / Possible Council Input



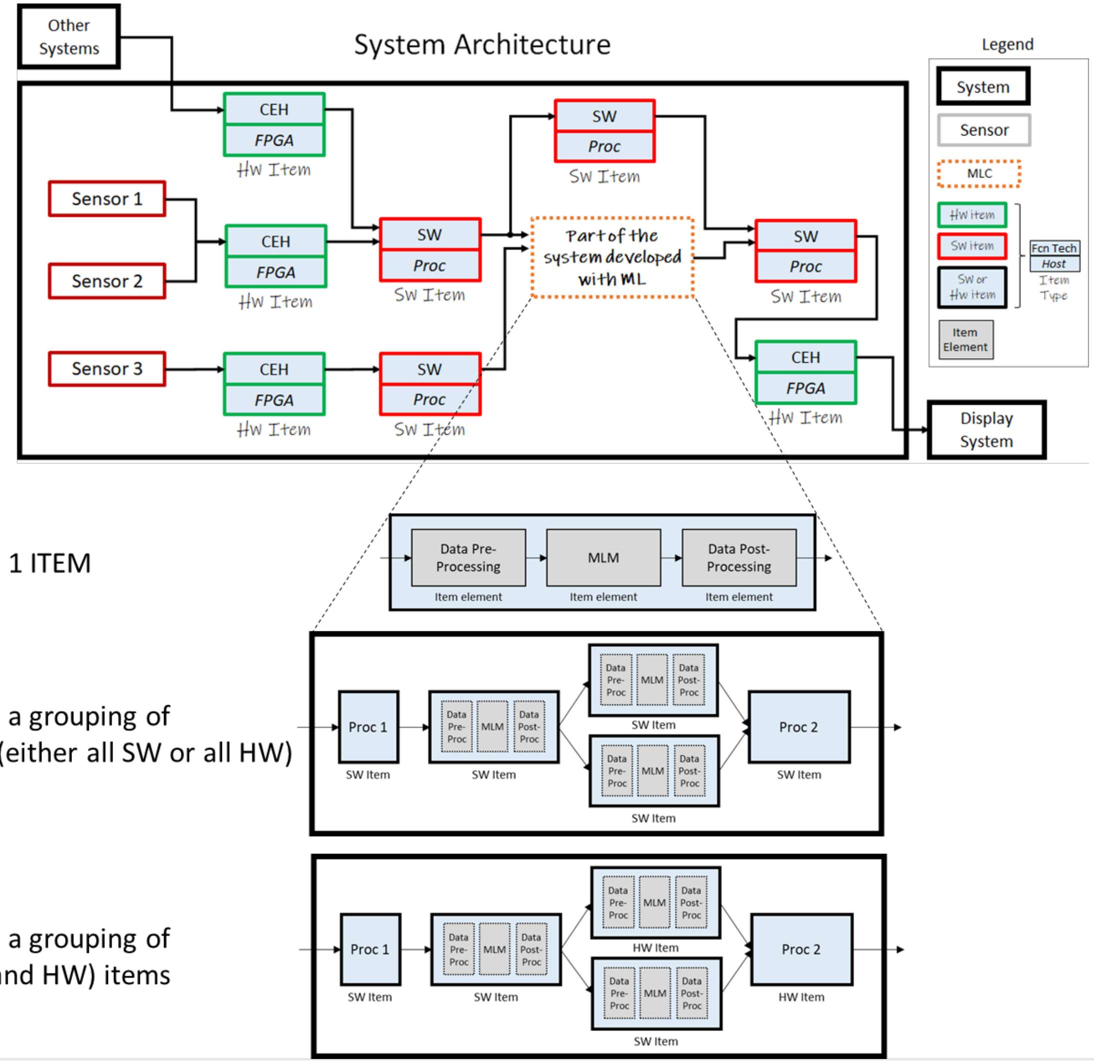
Deep dive into the ML Constituent (MLC) concept (15’)

Two Categories of Case Studies with ML



Category 1: ML used to develop all or part of a system function

Methodology Adopted for the Case Studies



Case Study 1: MLC as 1 ITEM

Case Study 2: MLC as a grouping of homogeneous items (either all SW or all HW)

Case Study 3: MLC as a grouping of heterogeneous (SW and HW) items

Proposed analysis methodology

A) System before the application of ARP6983/ED-324?

- Initial set of system requirements
- Initial system architecture including the MLC
- Initial allocation of system requirements to the MLC item(s)
- Initial assignement of IDAL(s) to MLC item(s)

B) What is the outcome of ARP6983/ED-324 application?

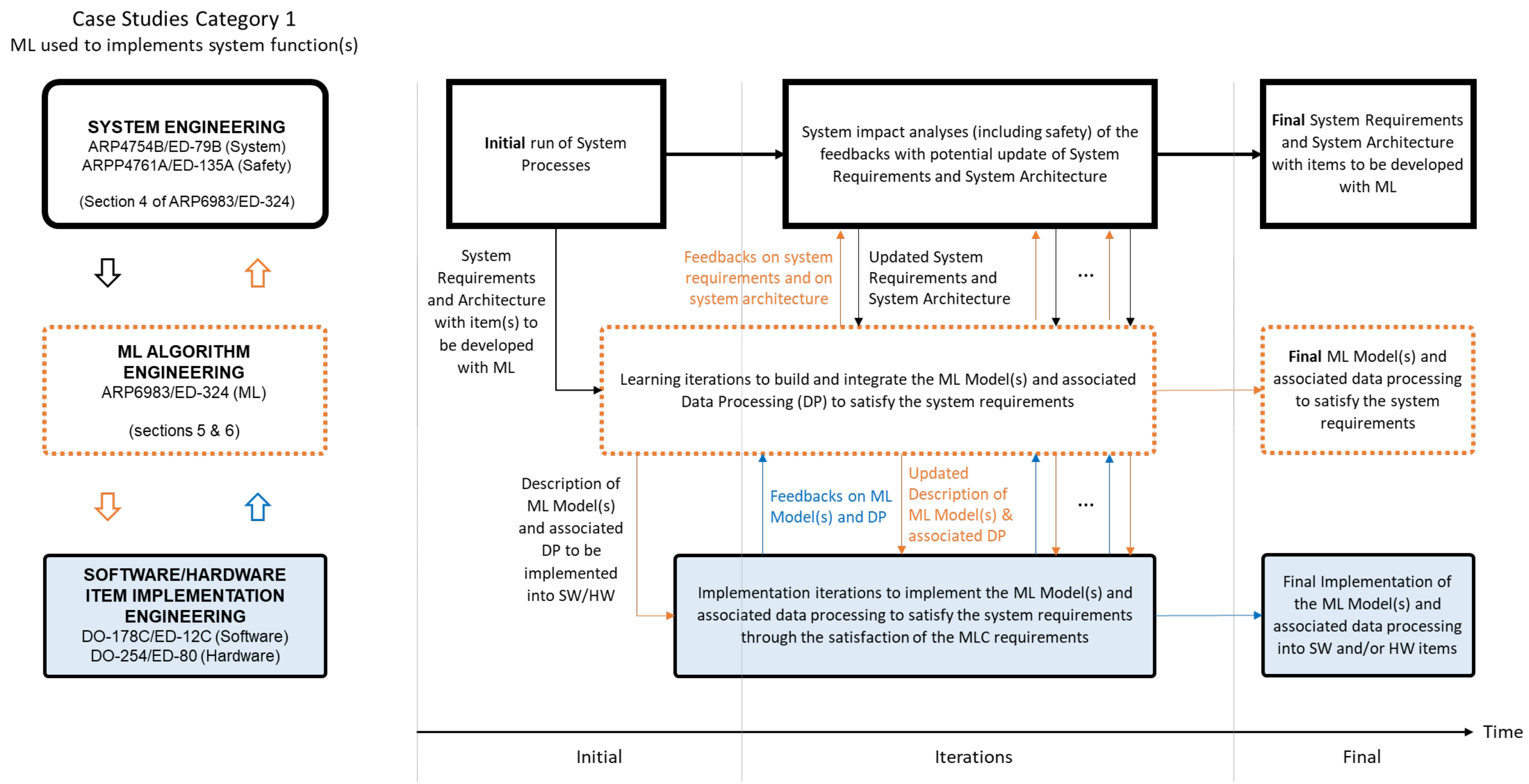
- MLC logical architecture
- MLC Implementation architecture
- Derived requirements / Problem or change documentation
- Integration and verification of the implemented MLC

C) System after the application of the outcome of ARP6983/ED-324?

- Final set of system requirements
- Final system architecture including the MLC
- Final allocation of system requirements to the MLC item(s)
- Final assignement of IDAL(s) to MLC item(s)

The complexity of the ML model(s) and the associated data processing cannot be fully known until the end of the learning process.

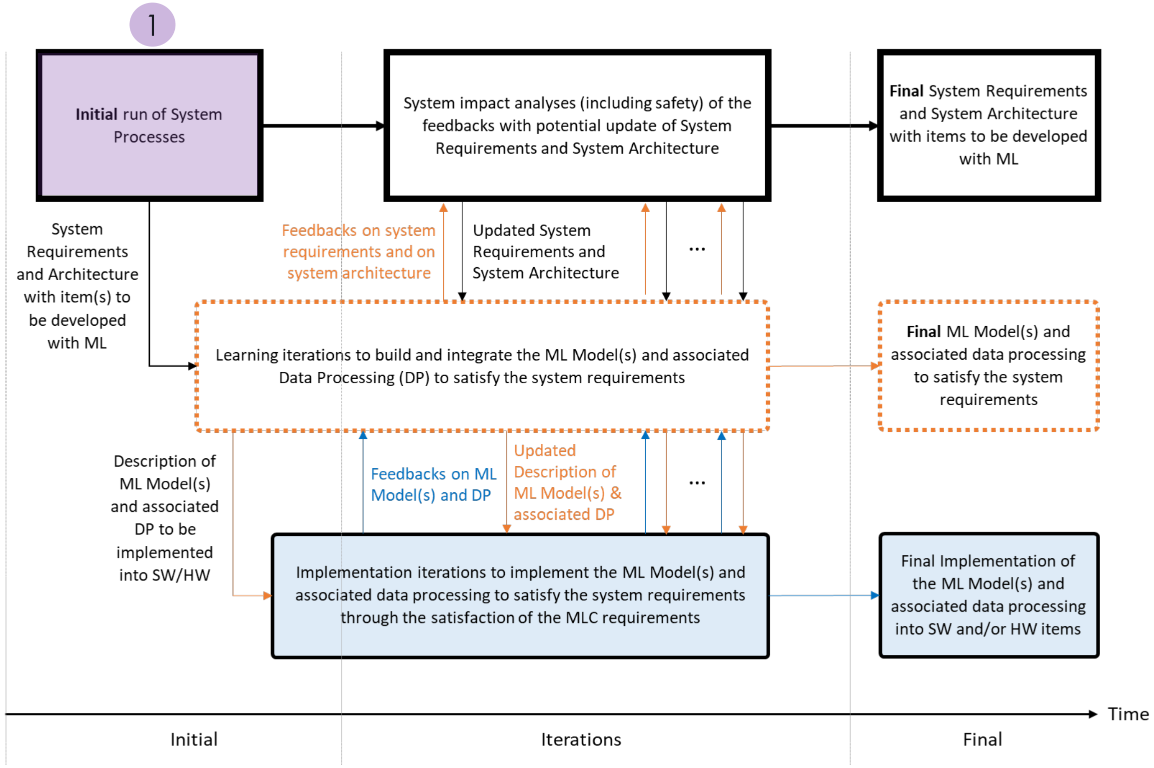
MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (1/6)



MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (2/6)

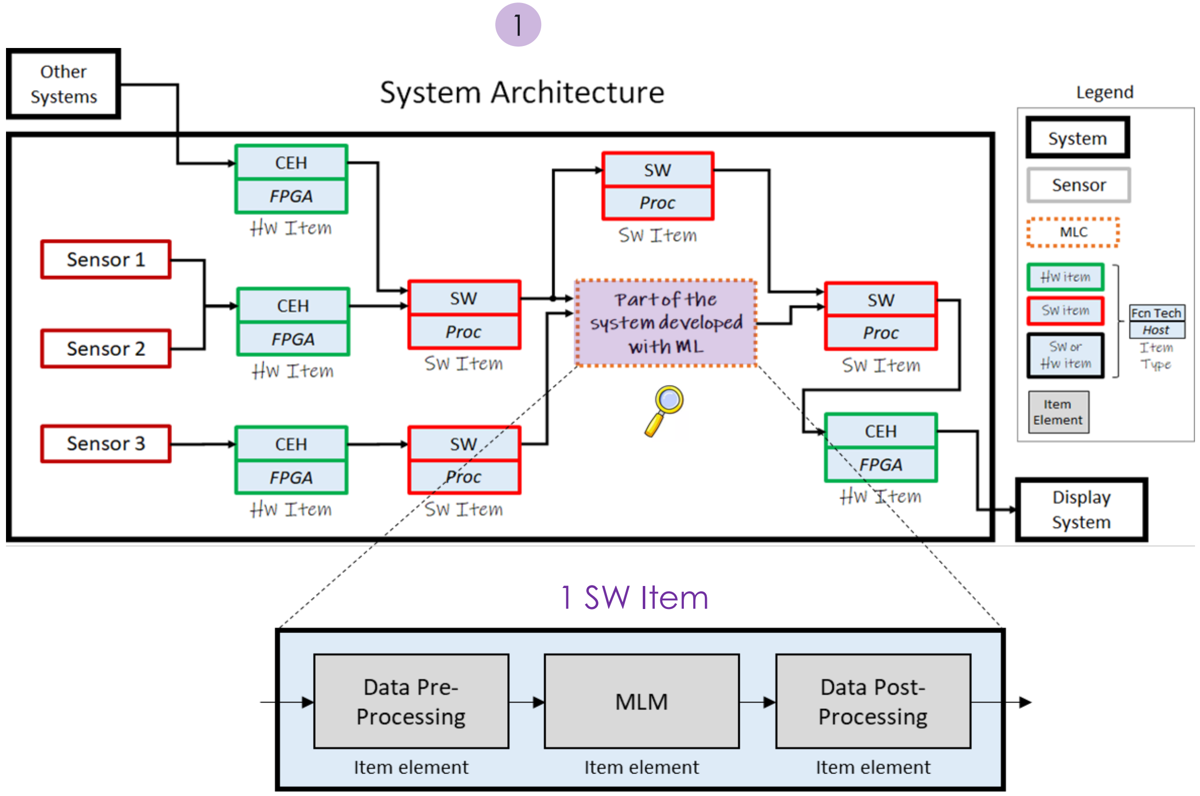
Overview of the MLC Development Lifecycle

Current Step



Detailed view of the Current step

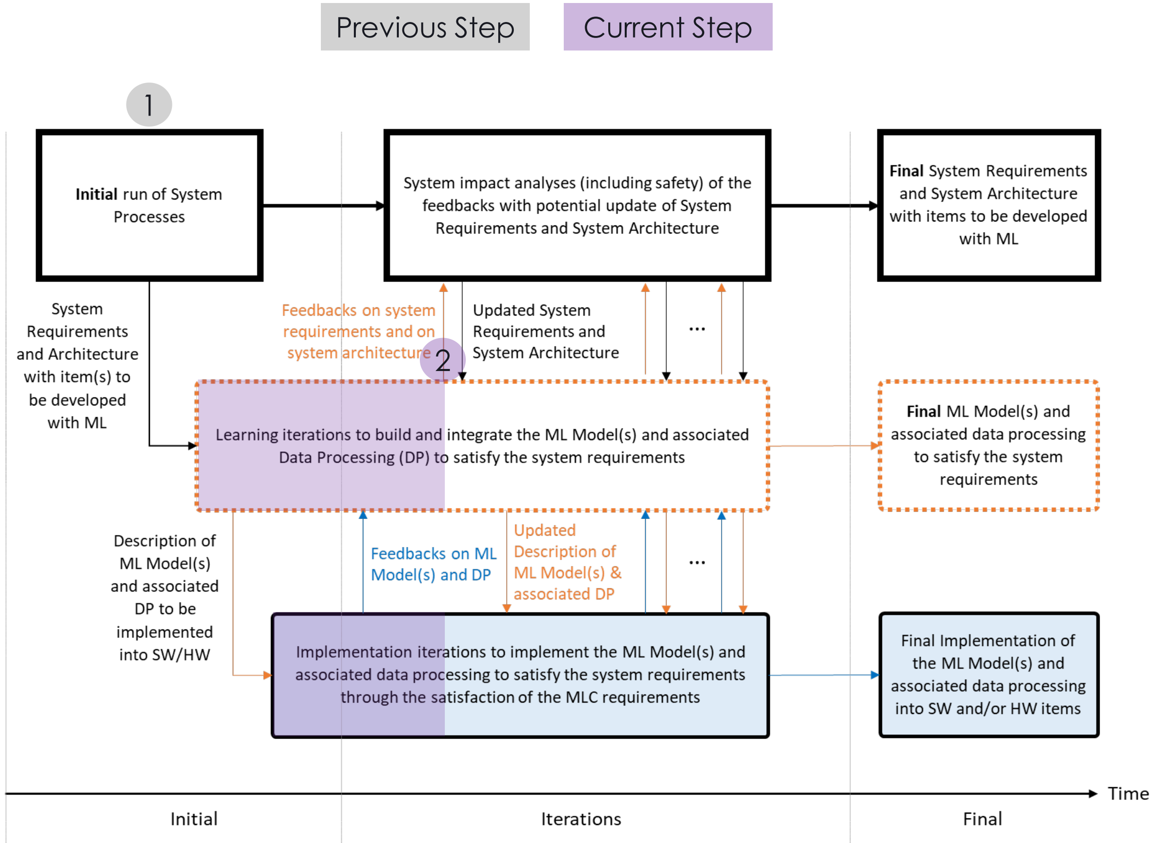
Current Step



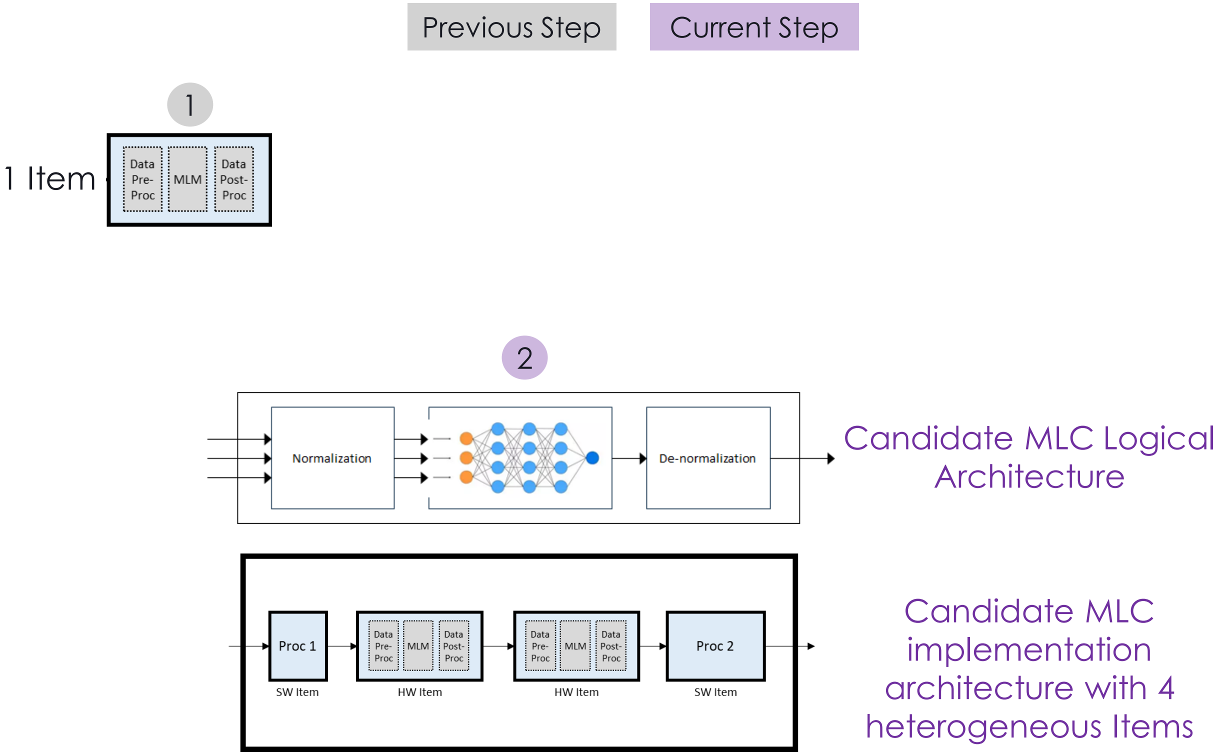
THALES Building a future we can all trust

MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (3/6)

Overview of the MLC Development Lifecycle

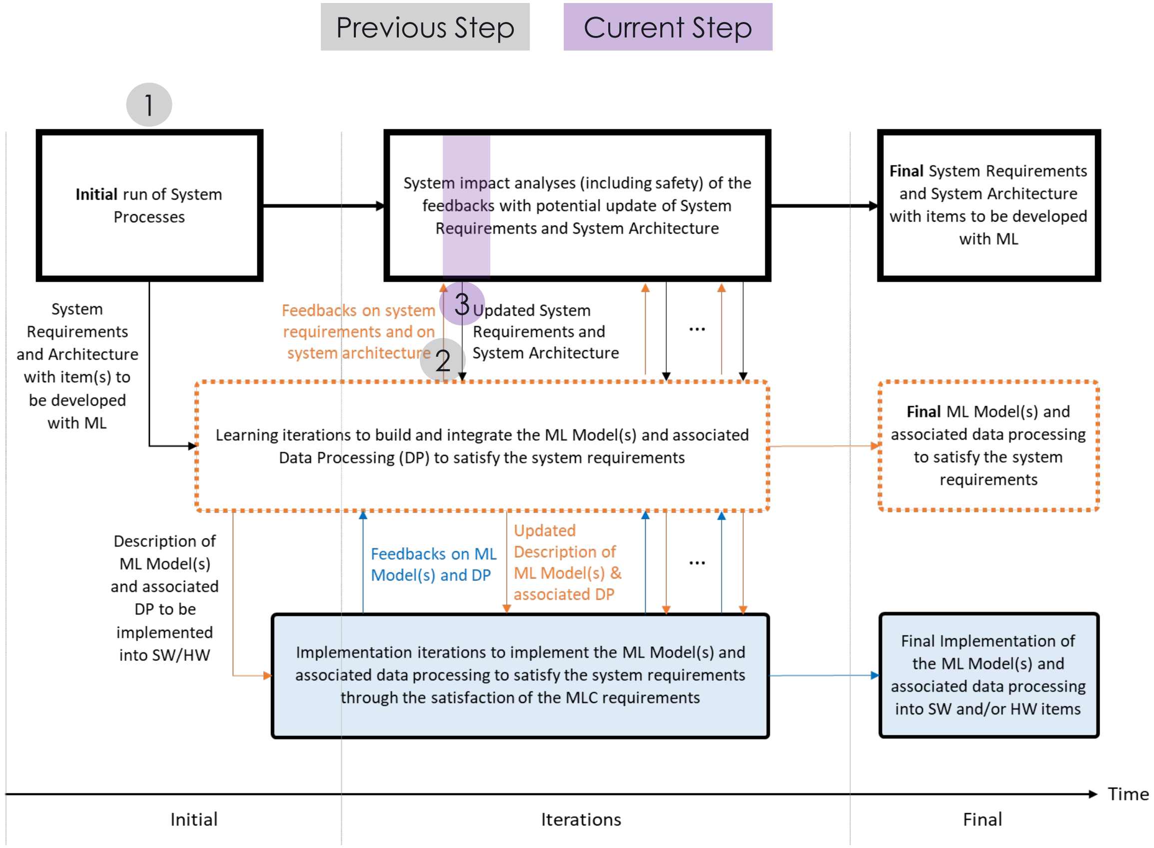


Detailed view of the Current step

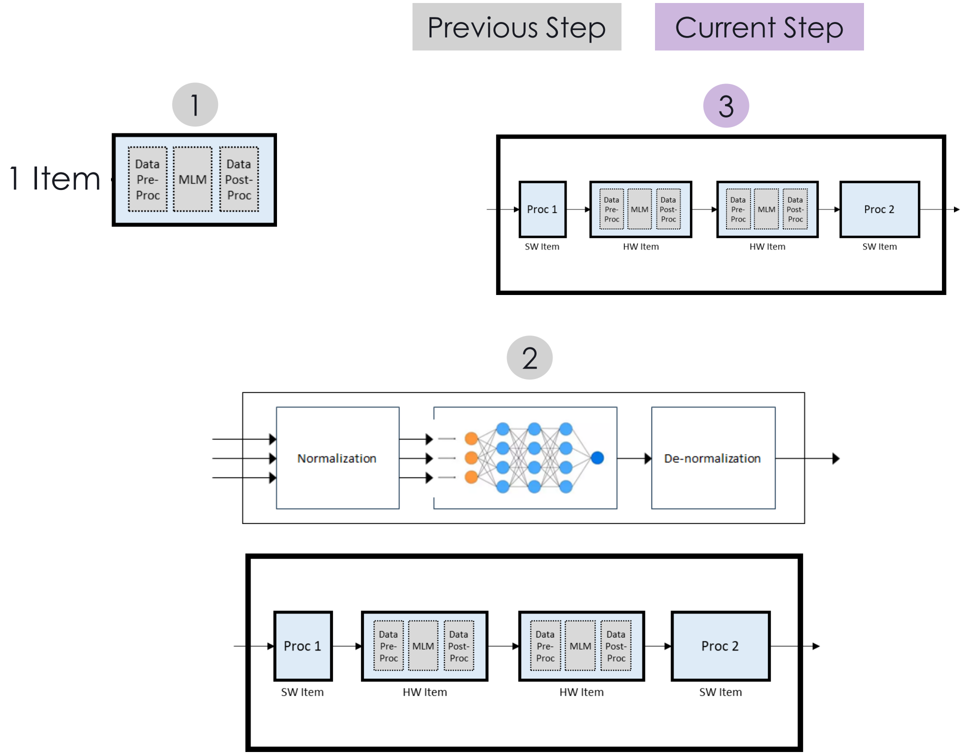


MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (4/6)

Overview of the MLC Development Lifecycle



Detailed view of the Current step

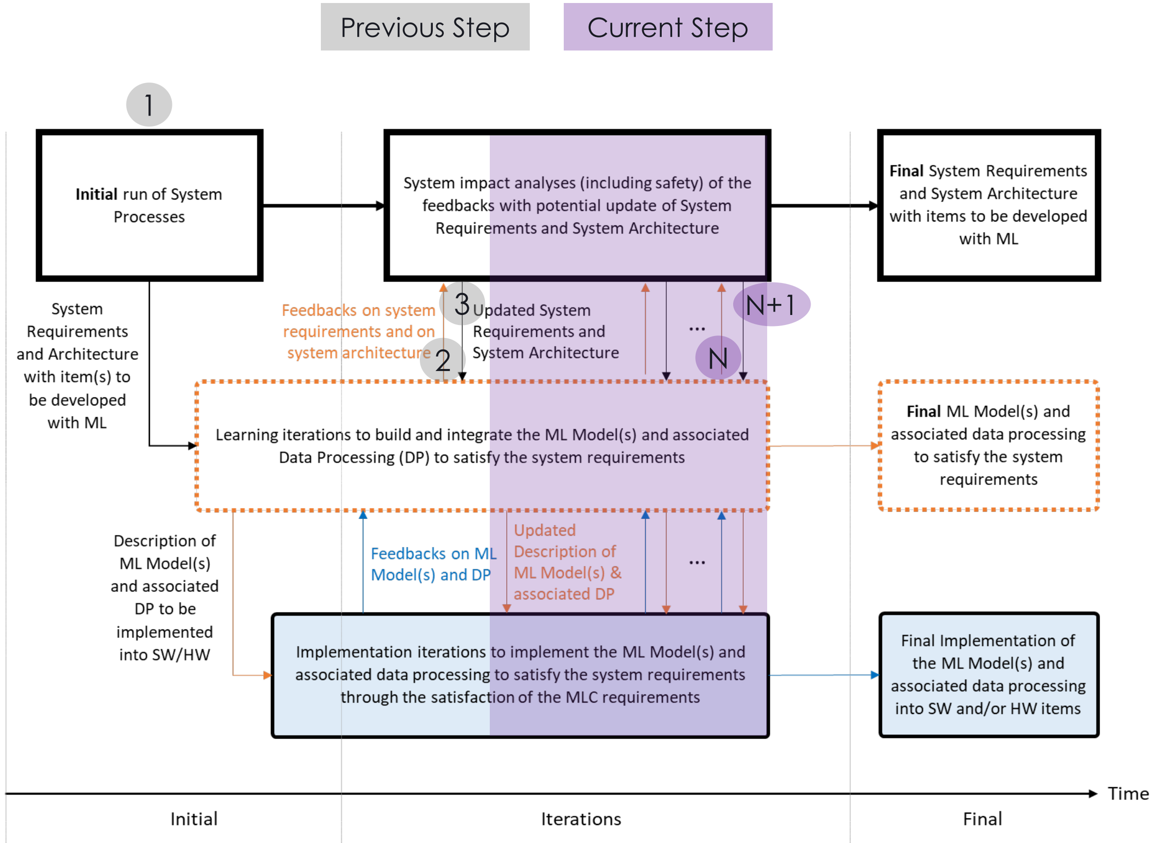


Candidate MLC implementation architecture integrated into the system architecture after system impact analyses (including safety)

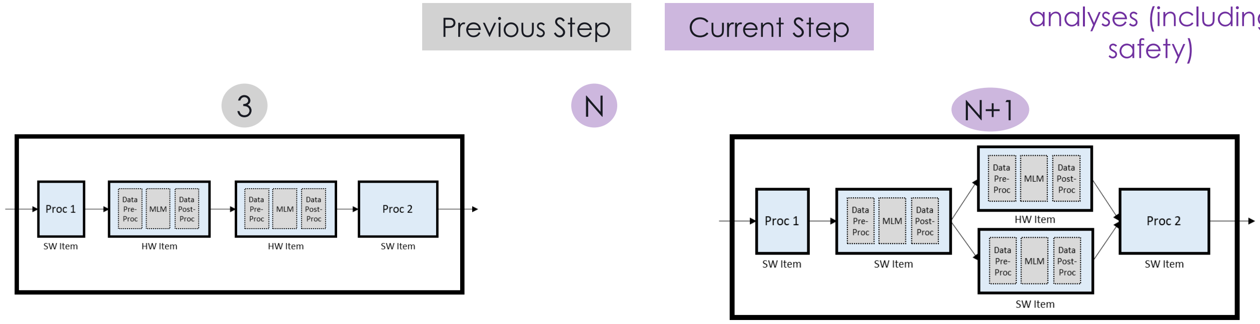
MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (5/6)

Last candidate MLC implementation architecture integrated into the system architecture after system impact analyses (including safety)

Overview of the MLC Development Lifecycle



Detailed view of the Current step



Extract from ARP4754B/ED-79B – section 4.5

4.5 Development of System Architecture and Allocation of System Requirements to Items

In practice, system architecture development and the allocation of system requirements to items are tightly coupled and iterative processes. Requirements arising from allocation may be system-related or item-related. With each cycle, the identification and understanding of derived requirements increases and the rationale for the allocation of system-level requirements to items becomes clearer. The process is complete when all requirements can be accommodated within the final architecture. The development of the system architecture and the allocation of requirements to items ensures that the items perform all intended system functions.

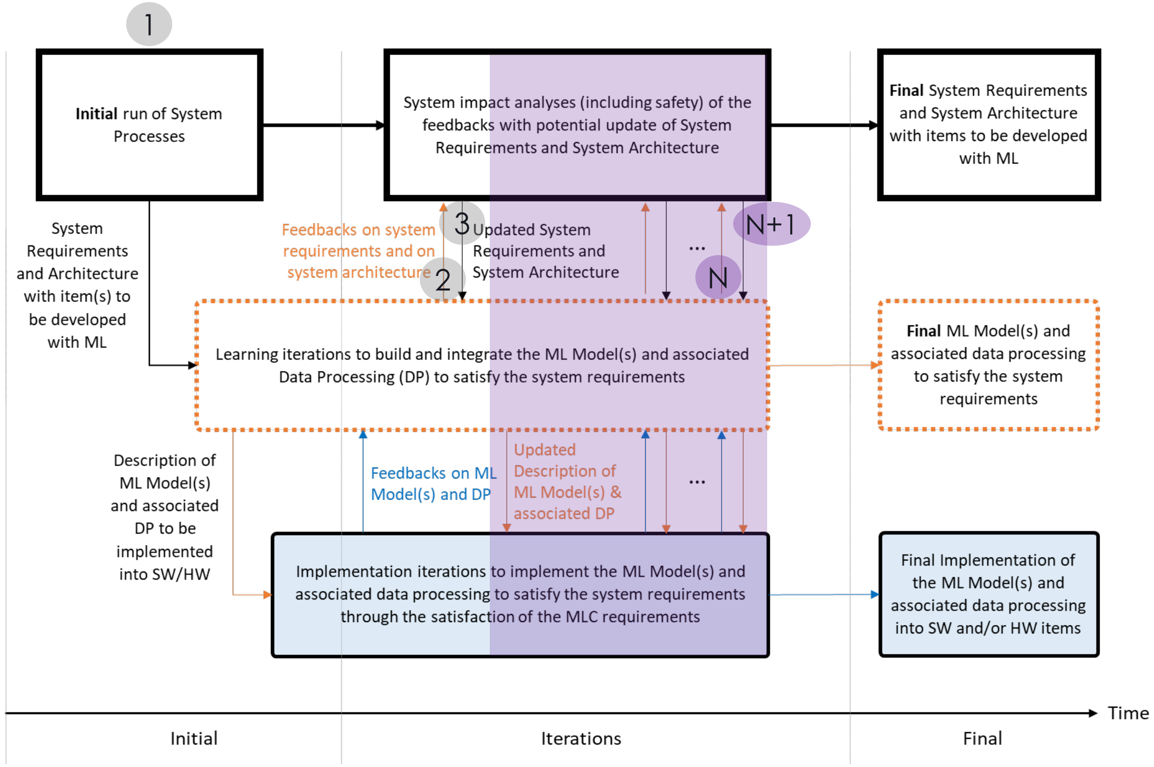
The system architecture establishes the structure and boundaries within which specific item designs are implemented to meet the established requirements. More than one candidate system architecture may be considered for implementation. These candidate system architectures may be evaluated using such factors as technology readiness, implementation schedules, producibility, contractual obligations, economics, prior experience, and industry precedence. The candidate architectures are then iteratively evaluated using functional and performance analyses. In particular, the PASA and PSSA processes establish feasibility with respect to top-level safety objectives assigned to each system. PASA/PSSA activities are summarized in Section 5.1.2 and Section 5.1.4, respectively.

Requirements stemming from technology, architecture, system, equipment, and item interfaces, system constraints (physical, environmental, etc.), integration, and implementation choices become more clearly visible as work on the system architecture progresses.

MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (5/6)

Overview of the MLC Development Lifecycle

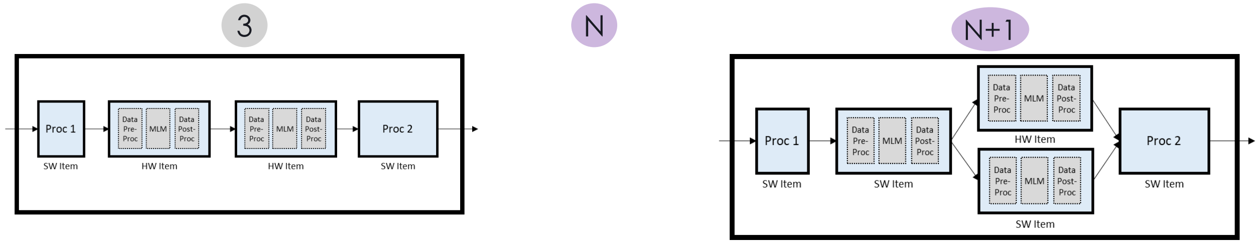
Previous Step | Current Step

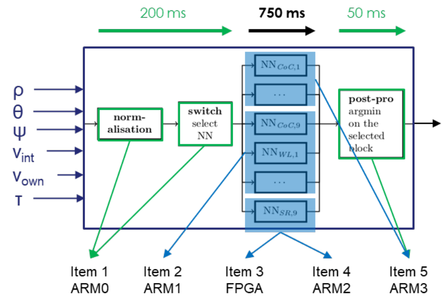


Detailed view of the Current step

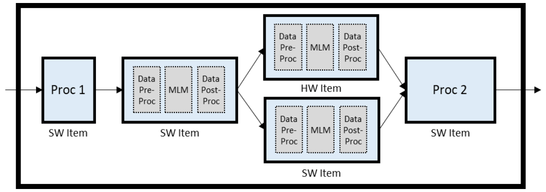
Previous Step | Current Step

Last candidate MLC implementation architecture integrated into the system architecture after system impact analyses (including safety)





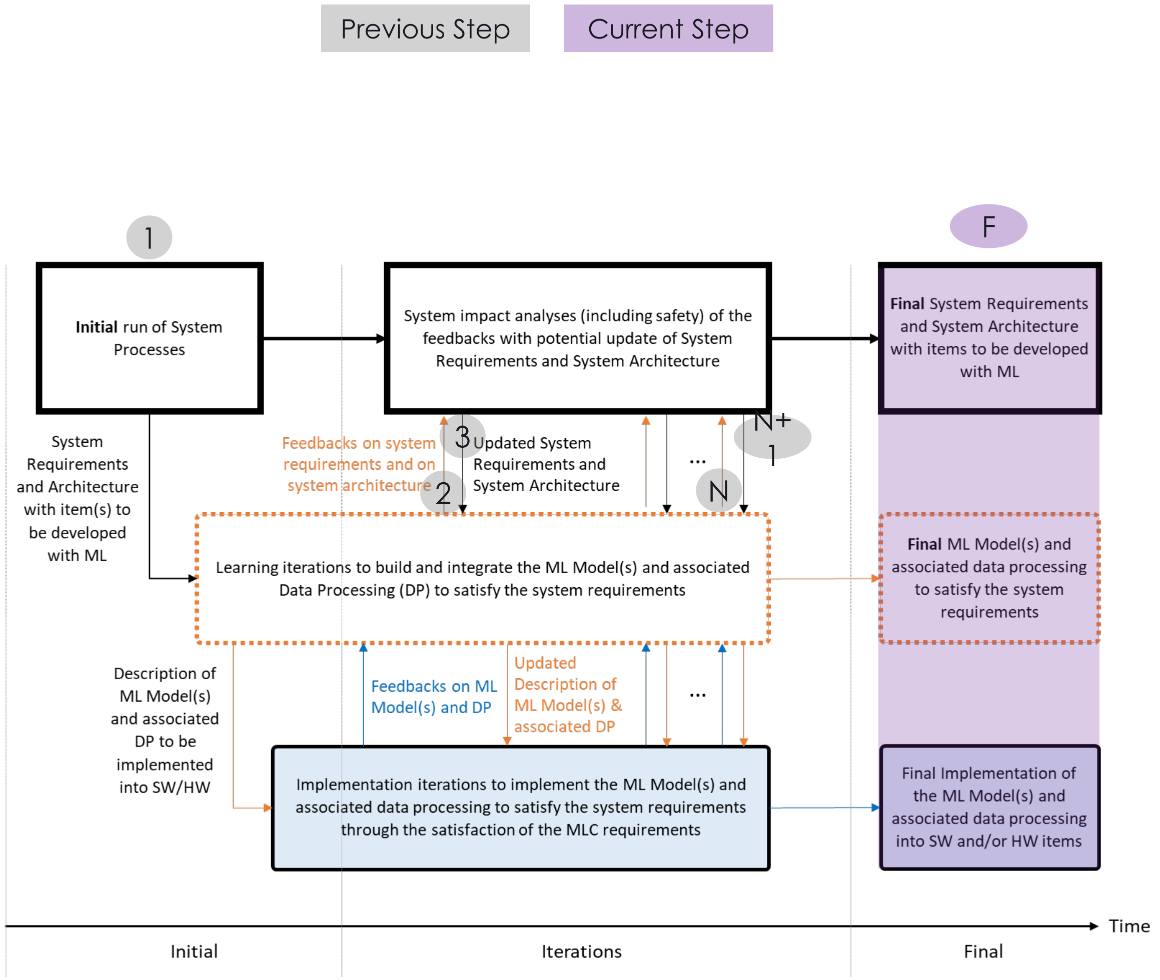
Last candidate MLC Logical Architecture



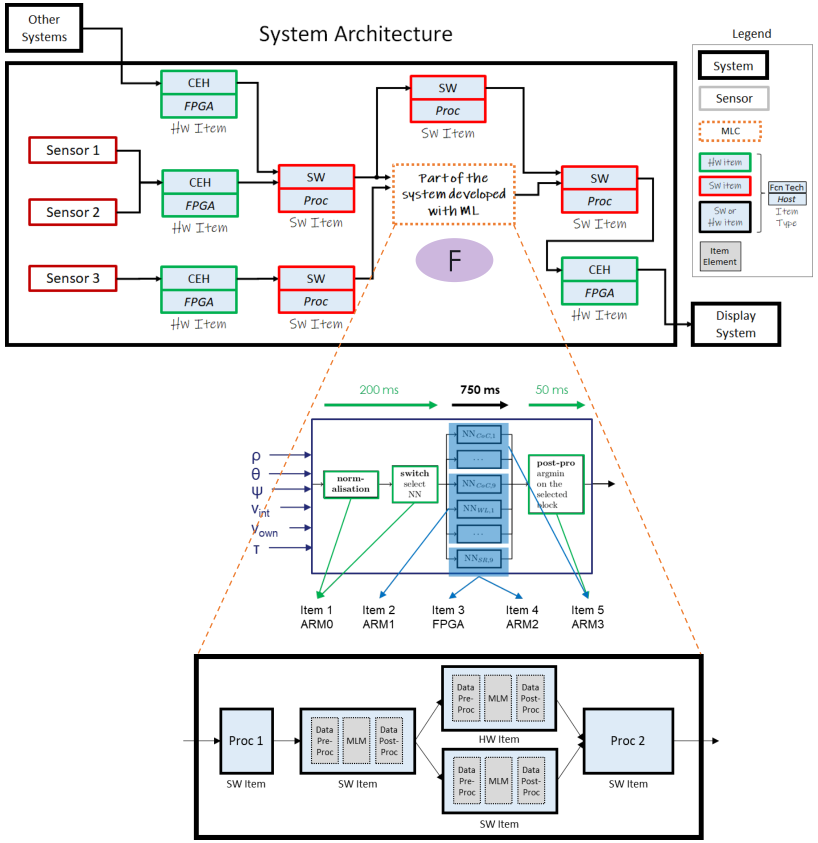
Last candidate MLC implementation architecture with 5 heterogeneous Items

MLC Development Lifecycle Interfaces: from system to item(s) via the MLC (6/6)

Overview of the MLC Development Lifecycle



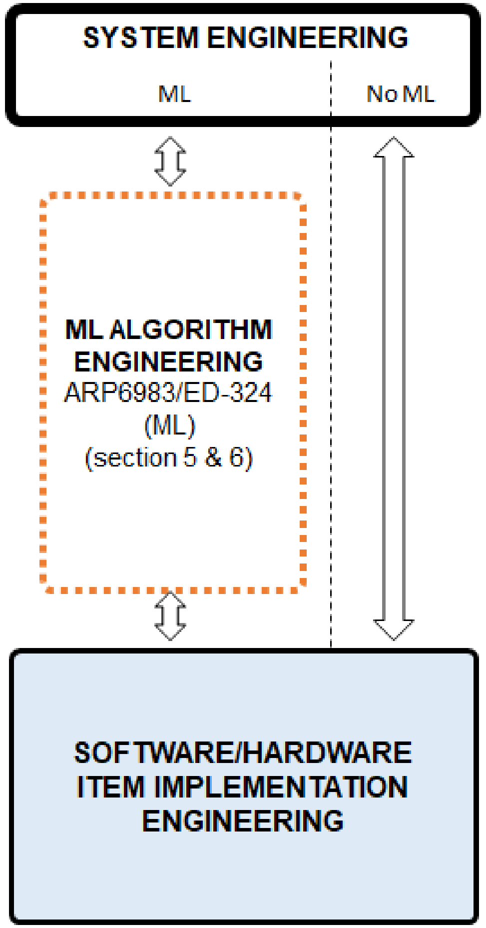
Detailed view of the Final step



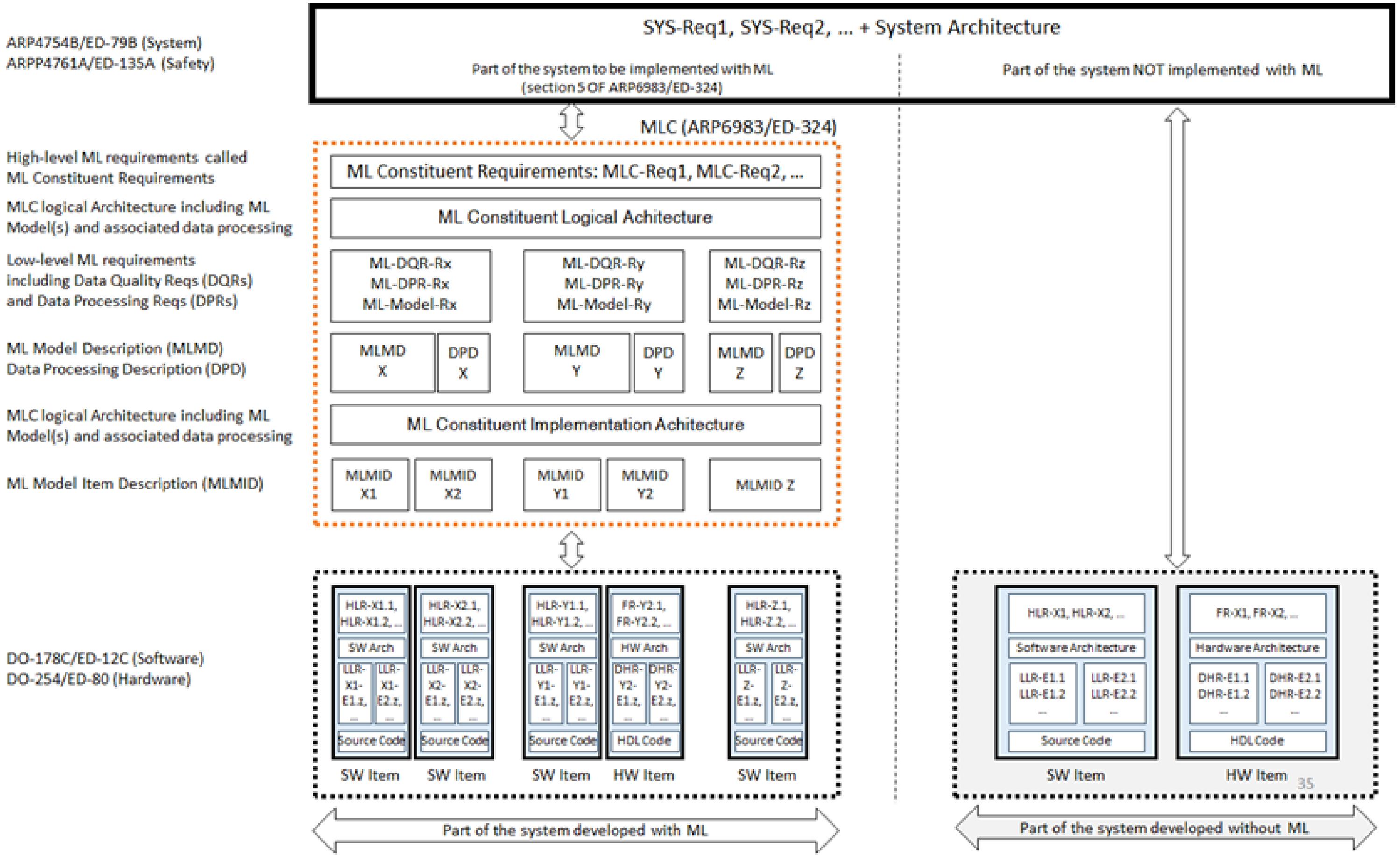
Building a future we can all trust

MLC Development Lifecycle Interfaces: focus on the final phase

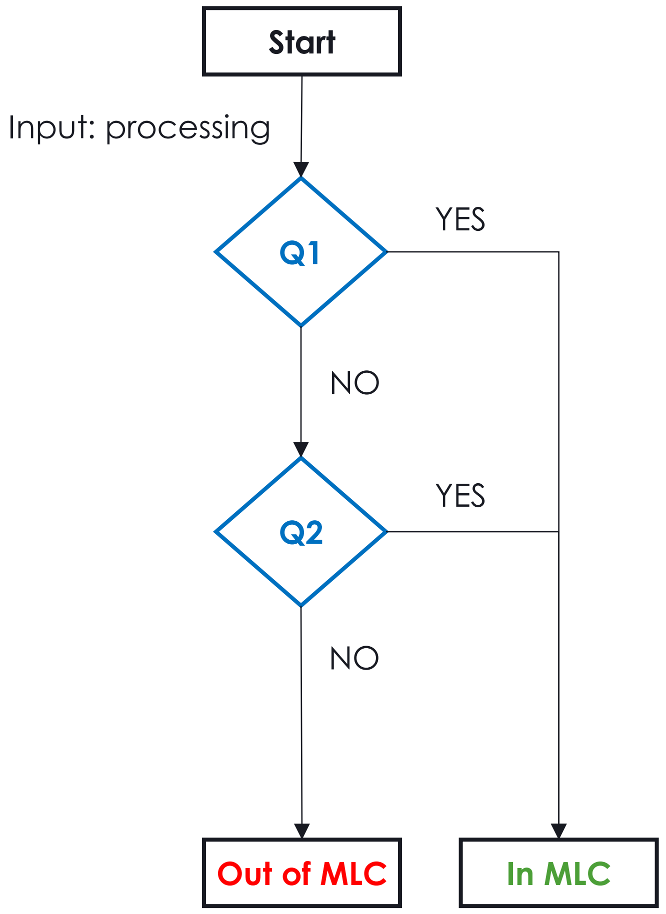
Case Studies Category 1
ML used to develop all or part of a system function



Snapshot of the Final Phase



Procedure to identify processing belonging to the MLC



> **1st question (Q1): Do you need insights from the learning process to specify this processing?**

- 1.a) **Answer is YES** => the specification of the processing strongly depend on the way the model is trained* and therefore you need the guidance of ARP6983 to be able to specify this processing (DO-178C is not enough because you cannot write HLRs without having insights on the learning process) **=> in this case your processing belongs to the MLC**
- 1.b) **Answer is NO** => **2nd question (Q2): Do you need this processing to perform any activity of the ML development assurance (learning assurance), for example to do tests?**
  - 1.b.1) **Answer is YES**: The example of performance tests (that is developed in the MLC position paper) is good because for some applications, to measure performance metrics, you may need to observe the outputs of the ML model on several cycles and compute an aggregated performance indicator (mean value and variance or other more sophisticated indicators). The processing to compute the aggregated performance indicator is independent from the learning process (no need to know how the ML model has been trained), but it is necessary to perform verification activities on the ML model (run performance tests) => **in this case your processing belongs to the MLC**
  - 1.b.2) **Answer is NO**: **your processing is outside the MLC**

Synthesis of the outcome of Category 1 Case Studies

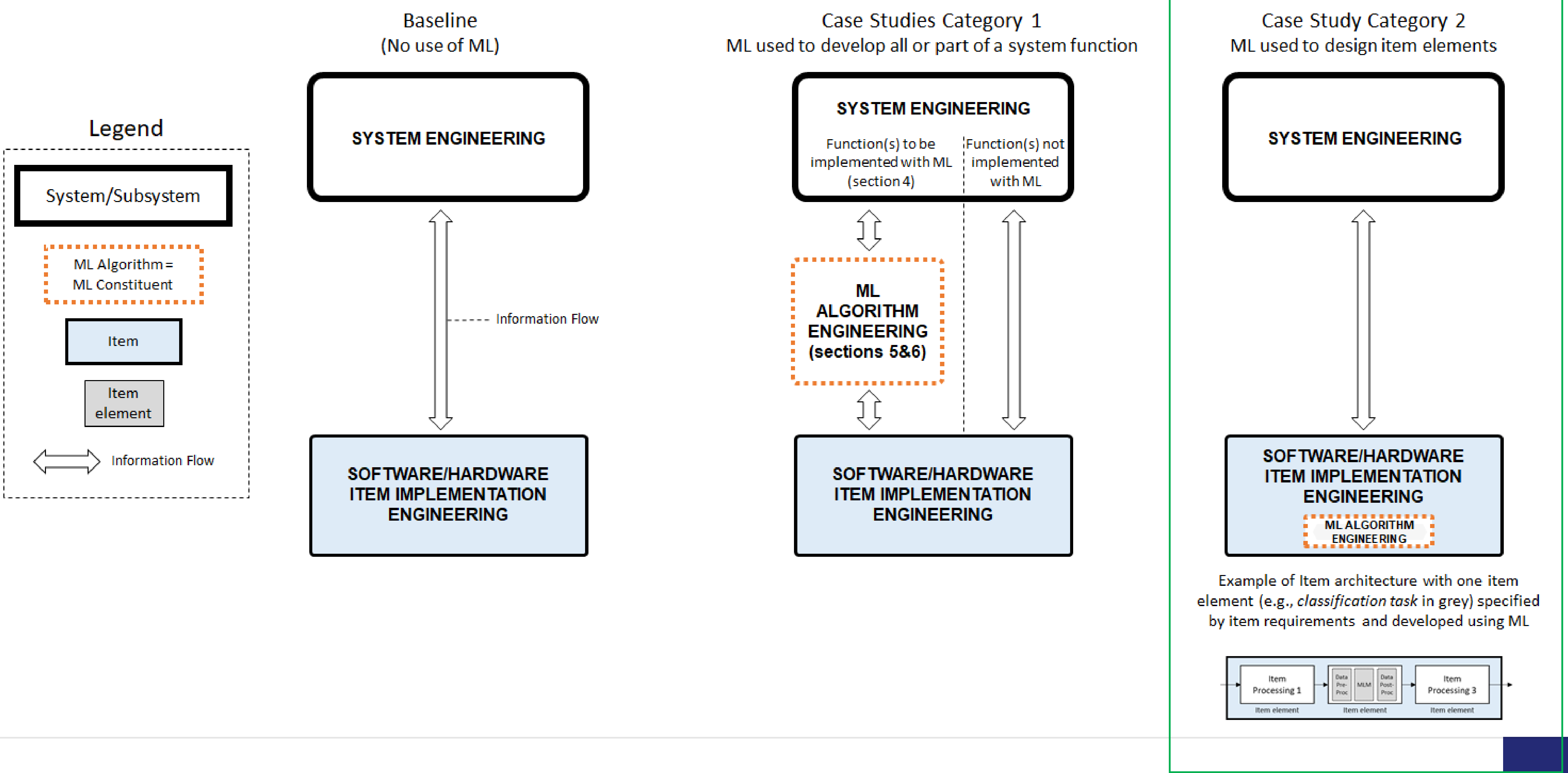
The main outcome of these case studies shows that:

- The concept of ML Constituent in ARP6983/ED-324 is introduced to designate the "item(s) in the system architecture that will subsequently be implemented using ML".
- The MLC concept does not affect the way system processes (system requirements, system architecture, safety assessment, etc.) are performed as per ARP4754B/ED-79B and ARP4761A/ED-135A.
- The interface between system engineering layer (ARP4754B/ED-79B) and item implementation layer (DO-178C/ED-12C and DO-254/ED-80) is passing through the MLC Engineering layer (ARP6983/ED-324) only for the items to be implemented with ML (belonging to the MLC) and not for the other items of the system architecture that are not developed using ML.
- The ML development assurance (all objectives formalized in ARP6983/ED-324) ensures that the MLC design specification* comply with the system requirements allocated to the MLC (including the requirements coming from system safety assessment and allocated to the MLC)
- The implemented SW and/or HW items in the MLC satisfy the system requirements (implemented using ML) through the satisfaction of the MLC requirements.

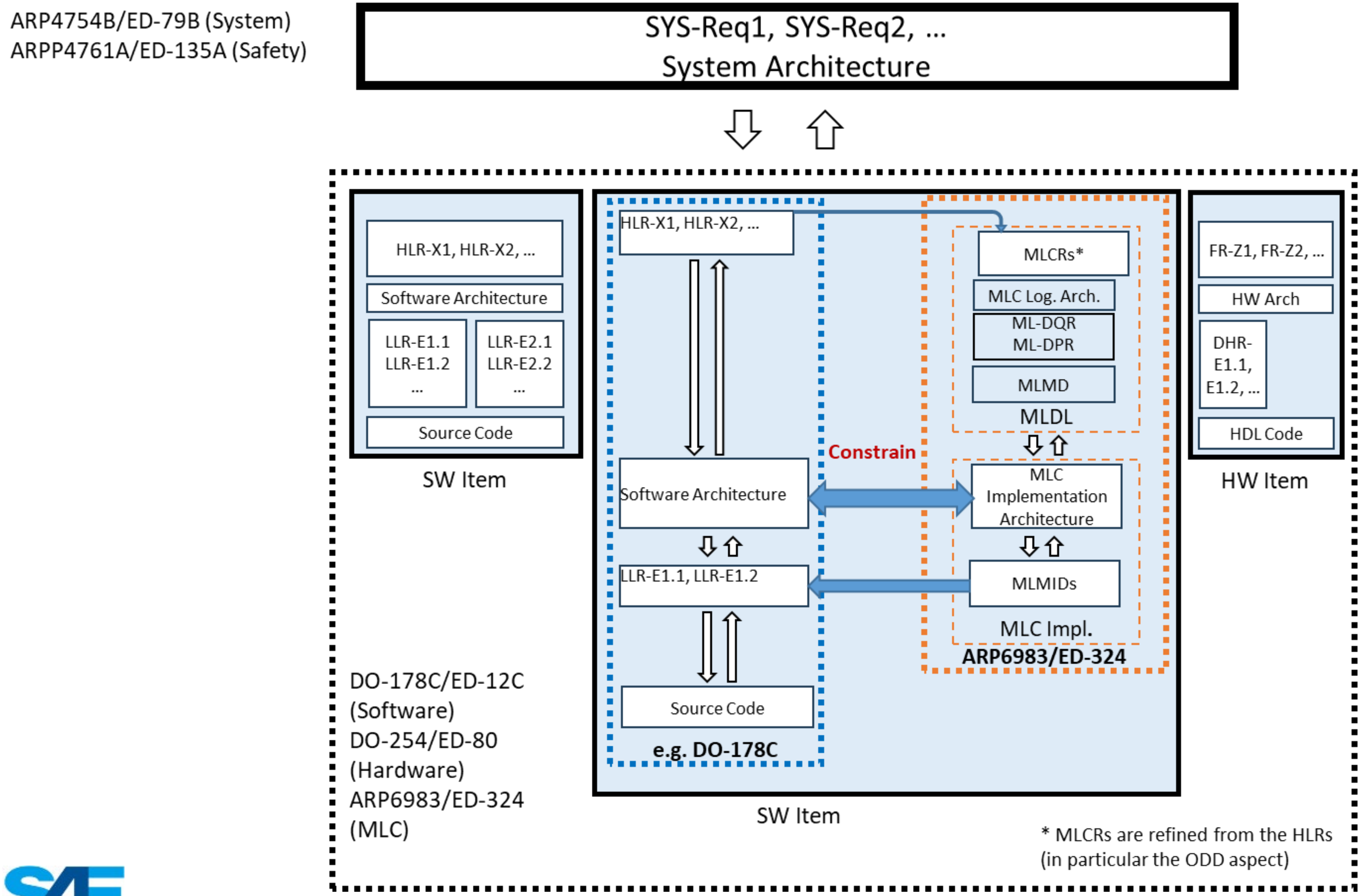
*i.e., design specification of the ML Model(s) and associated data processing passed to the SW and/or HW implementation processes.

Category 2: ML used to develop item elements from item requirements
(5')

Two Categories of Case Studies with ML



ML used to develop item elements from item requirements



The purpose of the figure is to emphasize that DO-178C cannot replace ARP6983/ED-324 but to show that DO-178C artefacts can be used by MLDL and MLC implementation processes.

- **MLC implementation architecture, in this specific case 4, has the same characteristics as the DO-178C Software Architecture.**
- **Similarly, MLMIDs as the same characteristics as DO-178C LLRs.**
- **The DO-178C verification process will address the objectives of table A5, A6 and A7**

Synthesis of the outcome of Category 2 Case Studies

The main outcome of these case studies shows that:

- The objectives of section 4 (system considerations) are still applicable in the context of the direct interface (two-way information flow) between system standards (e.g., ARP4754B/ED-79B) and item standards (e.g., DO-178C/ED-12C or DO-254/ED-80).
- The processes and activities of the ML development assurance planning (section 3) and the ML Development Lifecycle (MLDL in section 5) are agnostic of the level of engineering and can scale. For example, the same term "ML model" may be used for both an ML model implementing a system function and an ML model implementing item elements (scaling down). Same for data processing, and so on and so forth.
- Outputs from MLDL processes and activities that have the same characteristics as those of SW or AEH processes can be used to produce or be incorporated or used within the article in which they are to be integrated. For example the MLC implementation architecture in « category 2 case studies » has the same characteristics as the DO-178C Software Architecture. Similarly, MLMIDs have the same characteristics as DO-178C LLRs.
- The ARP6983 MVP1 should not preclude category 2 case studies.

Proposed definition of MLC

Previous definition of ML Constituent in Draft 5B:

> ML CONSTITUENT [ARP6983/ED-324]: A defined and bounded set of either hardware item(s) and/or software item(s) that implement one or more ML Model(s) and associated data processing that are grouped for integration purpose to support one or more subsystem function(s).

Proposed new definition of ML Constituent:

> **ML CONSTITUENT [ARP6983/ED-324]: ML Constituent (MLC) designates what is developed using ML:**

- If ML is used to develop system function(s), MLC designates the portion of the system architecture that will subsequently be implemented using ML in accordance with the guidelines of ARP6983/ED-324. This portion of the system architecture includes SW and/or HW item(s) that implement(s) ML Model(s) and associated data processing and that are grouped for integration purpose to support one or more subsystem function(s).
- If ML is used to develop item requirements, MLC designates the portion of the item architecture that will subsequently be implemented using ML in accordance with the guidelines of ARP6983/ED-324. This portion of the item architecture includes item element(s) that implement(s) ML Model(s) and associated data processing. The item can be a SW item or a HW item.

This proposed definition is a generalization of the one in Draft 5B, and it does not change or contradict its foundational concept.

Compliance (5')

Core certification objectives for traditional systems (airborne)

Legend: Black: MoC applicable with or without ML
Red: MoC no longer achievable with ML
Green: MoC specific to ML (to mitigate red ones)

Acronyms: ODD = Operational Design Domain
CCA = Common Cause Analysis
VoV = Verification of Verification

Derived from applicable regulations
- CS 25/27/29.1301, 1302, 1309, 1319
- SC-VTOL.2500, 2505, 2510
- Supported by AMC/AC 20-115D, AMC/AC 20-152A

1. Does the system perform its intended function correctly under all foreseeable operating conditions?

- **Focus:** Functional correctness
- **Standards:**
  - ARP4754B (functional requirements, system V&V)
  - DO-178C / DO-254 (software/hardware correctness)
- **Means of Compliance:**
  - Requirement validation & verification
  - Traceability from requirements to implementation
  - Testing in normal and abnormal conditions

2. Are the system safety objectives defined and satisfied?

- **Focus:** Hazard identification, categorization and mitigation
- **Standards :**
  - ARP4761A (Safety Assessments)
  - ARP4754B (flow-down of system requirements)
- **Means of Compliance:**
  - FHA, PASA, PSSA, SSA
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - DAL Allocation
  - Safety requirements verification

3. Is the system robust against failures and misuse?

- **Focus:** The system should behave in a safe, predictable way even when encountering faults, invalid inputs, or misuse
- **Standards:**
  - ARP4761A (CCA, FMEA)
  - ARP4754B (flow-down of system requirements)
  - DO-178C / DO-254 (robustness & coverage)
- **Means of Compliance:**
  - Fault injection testing
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - Defensive coding (error handling and boundary testing)
  - Req.-based and structural coverage

4. Is the development process commensurate with the safety criticality of the function?

- **Focus:** Process rigor based on DAL
- **Standards:**
  - ARP4754B (DAL assignment)
  - DO-178C / DO-254 (process rigor per DAL)
- **Means of Compliance:**
  - Compliance with DO-178C or DO-254 objectives for the allocated iDAL
  - Quality assurance, configuration management
  - Independence of verification for high DALs

Core certification objectives for traditional systems (airborne)

Legend
Black: MoC applicable with or without ML
Red: MoC no longer achievable with ML
Green: MoC specific to ML (to mitigate red ones)

Acronyms
ODD = Operational Design Domain
CCA = Common Cause Analysis
VoV = Verification of Verification

Derived from applicable regulations
- CS 25/27/29.1301, 1302, 1309, 1319
- SC-VTOL.2500, 2505, 2510
- Supported by AMC/AC 20-115D, AMC/AC 20-152A

1. Does the system perform its intended function correctly under all foreseeable operating conditions?

- **Focus:** Functional correctness
- **Standards:**
  - ARP4754B (functional requirements, system V&V)
  - DO-178C / DO-254 (software/hardware correctness)
- **Means of Compliance:**
  - Requirement validation & verification
  - Traceability from requirements to implementation
  - Testing in normal and abnormal conditions

2. Are the system safety objectives defined and satisfied?

- **Focus:** Hazard identification, categorization and mitigation
- **Standards :**
  - ARP4761A (Safety Assessments)
  - ARP4754B (flow-down of system requirements)
- **Means of Compliance:**
  - FHA, PASA, PSSA, SSA
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - DAL Allocation
  - Safety requirements verification

3. Is the system robust against failures and misuse?

- **Focus:** The system should behave in a safe, predictable way even when encountering faults, invalid inputs, or misuse
- **Standards:**
  - ARP4761A (CCA, FMEA)
  - ARP4754B (flow-down of system requirements)
  - DO-178C / DO-254 (robustness & coverage)
- **Means of Compliance:**
  - Fault injection testing
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - Defensive coding (error handling and boundary testing)
  - Req.-based and structural coverage

4. Is the development process commensurate with the safety criticality of the function?

- **Focus:** Process rigor based on DAL
- **Standards:**
  - ARP4754B (DAL assignment)
  - DO-178C / DO-254 (process rigor per DAL)
- **Means of Compliance:**
  - Compliance with DO-178C or DO-254 objectives for the allocated iDAL
  - Quality assurance, configuration management
  - Independence of verification for high DALs

5. Is the behavior of the system partly developed with ML (MLC) explainable and verifiable?

Example of ML Algorithm Design Specification (MLC Description lifecycle data in Draft 6)

MLC

Pre-processing requirements

1rst layer

- Recover the entries in the form of a vector of dimension 3
- Multiply this vector by the matrix:
  [[1.8514004 , 0.06981966, 1.2454928 , -0.23673481, 0.33582166, 1.144316 , -0.07928362, 0.85065633], [0.2689074 , -1.6337383 , -0.09429459, -1.5041184 , -0.8323707 , -0.19164966, 1.7850695 , -1.332562], [0.4509967 , -0.6398565 , 0.30318516, -0.5797693 , -0.28540373, -0.11622415, -0.93806374, -0.36459136]]
- Add the vector:
  [0.0436854 , 0.66848093, -0.34943226, 0.82454354, -0.941798 , -0.1605606 , 0.308173 , -0.01794778]
- For each element, apply the **ReLU** function : (x) -> (x si x >0, 0 else)
  (NB: the result of the 1st layer)

2nd Layer

- Multiply by the matrix:
  [[0.08302072 -0.34755808 0.3101814 -0.29662135 -0.16163935 0.09059773 0.16717902 0.74825156] [-0.10736815 1.0281937 -0.11446434 0.237721 0.7667631 -0.07044937 0.03289383 0.04712383] [0.62960064 0.60489106 0.40853637 0.3820236 -0.02992251 0.40818375 0.71154916 0.6853567] [0.27913433 0.57993895 -0.1088046 -1.4035206 0.05937768 0.19022994 0.33831477 0.5348654] [-0.49616838 -0.59313494 -0.13610965 -1.602424 -0.8092544 -1.4723336 -0.74360555 0.14003202] [0.9558404 -0.38831693 0.54558367 1.1563224 -0.1213242 0.96744263 -0.04928763 -0.10160095] [0.29998606 -0.5457609 0.34783924 0.2050478 -0.02698133 0.3551015 0.19448055 0.9439171] [-1.1834544 -0.73129165 -0.83786845 0.25230122 -2.1236877 -0.6258286 -0.9858019 0.1222824]]
- Add the vector:
  [-0.33079636 0.09964079 -0.06536419 -0.02690533 0.05694228 -0.24229142 -0.18812867 0.8922489]
- Apply the **ReLU** function to each element : (x) -> (x si x >0, 0 else)
  (NB: result of the 2nd layer)

3rd Layer

- Multiply by the matrix :
  [[-0.3034037 -0.11246069 -0.6560938 1.2929432 0.9277852 0.79638636 -0.6643924 0.10672873] [-1.6083356 0.25883913 -1.0975031 1.3155658 1.5284146 1.7666548 -0.9037737 1.9760858] [0.11462954 -1.0091568 -0.4328157 1.0937432 1.1971915 1.2098296 -0.91365033 1.0630316] [-1.0262014 1.3004675 -0.61168474 1.7266848 0.58779174 1.0225899 -0.14789295 0.9648] [-2.7417295 -0.02741075 -2.4579542 3.2859435 3.4163947 3.6450598 -2.7008405 3.6044834] [-0.60329115 0.45157087 0.17973106 0.9708737 0.45990798 0.10065918 0.10000039 0.97939295] [-0.09893946 -0.7206996 -0.18443836 0.5532775 0.8732048 0.31475598 -0.07816552 -0.08506049] [0.43368313 -0.47053978 0.8148482 -0.35335994 -0.3584372 -0.27395764 0.4629211 -0.24606809]]
- Add the vector :
  [1.4406947 -0.17694788 0.99267095 -0.24365164 -0.20841813 -0.2159943 1.0041207 -0.40997046]
- Apply the **ReLU** function to each element : (x) -> (x si x >0, 0 else)
  (NB: result of the 3rd layer)

- Multiply by :
  [[3.0001943] [-1.1565807] [1.3064175] [-1.6839781] [-2.2671492] [-2.198915] [1.8382062] [-2.066988]]
- Add :
  [0.9438011]
- Apply the **tanh** function to the result (=*Output of the network*)

Example source: NN and DO331 - C. Travers/A-C Kerbrat/T. Ther (Dassault Aviation) 31/01/2020

Post-processing requirements

+

Implementation constraints (performance, safety, etc.) coming from system considerations

Example of ML Algorithm Design Specification (MLC Description lifecycle data in Draft 6)

REQ ID	Type	Example of Performance Requirement (MLCR)	Example of Metric	Example of Scenario-based Performance Objective
MLCR-01	Tracking accuracy	While tracking the movement of detected objects on images containing unintended objects, the MLC shall minimize the creation of false tracks.	Rate of false tracks	In a sequence of images containing unintended objects, rolling average rate of false tracks lower or equal to <X%> over a sliding time window of <Y> system clock cycle time.
MLCR-02	Tracking consistency	While tracking the movement of detected objects on images containing partial object occultation and movement crossing, the MLC shall accurately maintain its association	Rate of false association	Rolling average rate of false association lower or equal to <X%> over a sliding time window of <Y> system clock cycle time between detected objects on a sequence of images containing partial object occultation and movement crossing.

MLC

1rst layer

- Recover the entries in the form of a vector of dimension 3
- Multiply this vector by the matrix:
  [[1.8514004 , 0.06981966, 1.2454928 , -0.23673481, 0.33582166, 1.144316 , -0.07928362, 0.85065633], [0.2689074 , -1.6337383 , -0.09429459, -1.5041184 , -0.8323707 , -0.19164966, 1.7850695 , -1.332562], [0.4509967 , -0.6398565 , 0.30318516, -0.5797693 , -0.28540373, -0.11622415, -0.93806374, -0.36459136]]
- Add the vector:
  [0.0436854 , 0.66848093, -0.34943226, 0.82454354, -0.941798 , -0.1605606 , 0.308173 , -0.01794778]
- For each element, apply the **ReLU** function : (x) -> (x si x >0, 0 else)
  (NB: the result of the 1st layer)

2nd Layer

- Multiply by the matrix:
  [[0.08302072 -0.34755808 0.3101814 -0.29662135 -0.16163935 0.09059773 0.16717902 0.74825156] [-0.10736815 1.0281937 -0.11446434 0.237721 0.7667631 -0.07044937 0.03289383 0.04712383] [0.62960064 0.60489106 0.40853637 0.3820236 -0.02992251 0.40818375 0.71154916 0.6853567] [0.27913433 0.57993895 -0.1088046 -1.4035206 0.05937768 0.19022994 0.33831477 0.5348654] [-0.49616838 -0.59313494 -0.13610965 -1.602424 -0.8092544 -1.4723336 -0.74360555 0.14003202] [0.9558404 -0.38831693 0.54558367 1.1563224 -0.1213242 0.96744263 -0.04928763 -0.10160095] [0.29998606 -0.5457609 0.34783924 0.2050478 -0.02698133 0.3551015 0.19448055 0.9439171] [-1.1834544 -0.73129165 -0.83786845 0.25230122 -2.1236877 -0.6258286 -0.9858019 0.1222824]]
- Add the vector:
  [-0.33079636 0.09964079 -0.06536419 -0.02690533 0.05694228 -0.24229142 -0.18812867 0.8922489]
- Apply the **ReLU** function to each element : (x) -> (x si x >0, 0 else)
  (NB: result of the 2nd layer)

3rd Layer

- Multiply by the matrix :
  [[-0.3034037 -0.11246069 -0.6560938 1.2929432 0.9277852 0.79638636 -0.6643924 0.10672873] [-1.6083356 0.25883913 -1.0975031 1.3155658 1.5284146 1.7666548 -0.9037737 1.9760858] [0.11462954 -1.0091568 -0.4328157 1.0937432 1.1971915 1.2098296 -0.91365033 1.0630316] [-1.0262014 1.3004675 -0.61168474 1.7266848 0.58779174 1.0225899 -0.14789295 0.9648] [-2.7417295 -0.02741075 -2.4579542 3.2859435 3.4163947 3.6450598 -2.7008405 3.6044834] [-0.60329115 0.45157087 0.17973106 0.9708737 0.45990798 0.10065918 0.10000039 0.97939295] [-0.09893946 -0.7206996 -0.18443836 0.5532775 0.8732048 0.31475598 -0.07816552 -0.08506049] [0.43368313 -0.47053978 0.8148482 -0.35335994 -0.3584372 -0.27395764 0.4629211 -0.24606809]]
- Add the vector :
  [1.4406947 -0.17694788 0.99267095 -0.24365164 -0.20841813 -0.2159943 1.0041207 -0.40997046]
- Apply the **ReLU** function to each element : (x) -> (x si x >0, 0 else)
  (NB: result of the 3rd layer)

- Multiply by :
  [[3.0001943] [-1.1565807] [1.3064175] [-1.6839781] [-2.2671492] [-2.198915] [1.8382062] [-2.066988]]
- Add :
  [0.9438011]
- Apply the **tanh** function to the result (=*Output of the network*)

Core certification objectives for systems partly developed with ML (airborne)

Derived from applicable regulations
- CS 25/27/29.1301, 1302, 1309, 1319
- SC-VTOL.2500, 2505, 2510
- Supported by AMC/AC 20-115D, AMC/AC 20-152A

Legend
Black: MoC applicable with or without ML
Red: MoC no longer achievable with ML
Green: MoC specific to ML (to mitigate red ones)

Acronyms
ODD = Operational Design Domain
CCA = Common Cause Analysis
VoV = Verification of Verification

1. Does the system perform its intended function correctly under all foreseeable operating conditions?

- **Focus:** Functional correctness
- **Standards:**
  - ARP4754B (functional requirements, system V&V)
  - Future ED-324/ARP6983 (ML)
  - DO-178C / DO-254 (software/hardware correctness)
- **Means of Compliance:**
  - Requirement validation & verification
  - Traceability from requirements to implementation
  - Testing in normal and abnormal conditions
  - Learning assurance (ODD, ML model generalization, stability, robustness, and ML development explainability)
  - Two-steps verification (training time and implementation time)

2. Are the system safety objectives defined and satisfied?

- **Focus:** Hazard identification, categorization and mitigation
- **Standards :**
  - ARP4761A (Safety Assessments)
  - ARP4754B (flow-down of system requirements)
  - Future ED-324/ARP6983 (ML)
- **Means of Compliance:**
  - FHA, PASA, PSSA, SSA
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - DAL Allocation
  - Safety requirements verification
  - QSA (allocation and refinement of performance requirement to the MLC/MLM)
  - Runtime monitoring

3. Is the system robust against failures and misuse?

- **Focus:** The system should behave in a safe, predictable way even when encountering faults, invalid inputs, or misuse
- **Standards:**
  - ARP4761A (CCA, FMEA)
  - ARP4754B (flow-down of system requirements)
  - Future ED-324/ARP6983 (ML)
  - DO-178C / DO-254 (robustness & coverage)
- **Means of Compliance:**
  - Fault injection testing
  - Defensive design techniques (e.g., fail-safe, redundancy)
  - Defensive coding (error handling and boundary testing)
  - Req.-based and structural coverage
  - Outliers detection, Novelties prevention (ODD), singularities management
  - Scenario-based testing (corner cases)
  - VoV based on ODD coverage
  - VoV based on explainability methods
  - Runtime monitoring

4. Is the development process commensurate with the safety criticality of the function?

- **Focus:** Process rigor based on DAL
- **Standards:**
  - ARP4754B (DAL assignment)
  - Future ED-324/ARP6983 (ML)
  - DO-178C / DO-254 (process rigor per DAL)
- **Means of Compliance:**
  - Compliance with AERP6983/ED-324 objectives for the allocated iDAL
  - Compliance with DO-178C or DO-254 objectives for the allocated iDAL
  - Quality assurance, configuration management
  - Independence of verification for high DALs

5. Is the behavior of the system partly developed with ML (MLC) explainable and verifiable?

- **Focus:** Learning and implementation correctness, and MLC robustness
- **Regulations and standards:**
  - Future EASA Part AI (on-going RMT.0742) / Future FAA regulation / Other regulation ; Future ED-324/ARP6983 (ML)
- **Means of Compliance:** **c.f. ABOVE in green**

THALES
Building a future we can all t

ATM / ANS Regulatory Context
Annex Overview

Fly AI Forum 22-23 April 2025

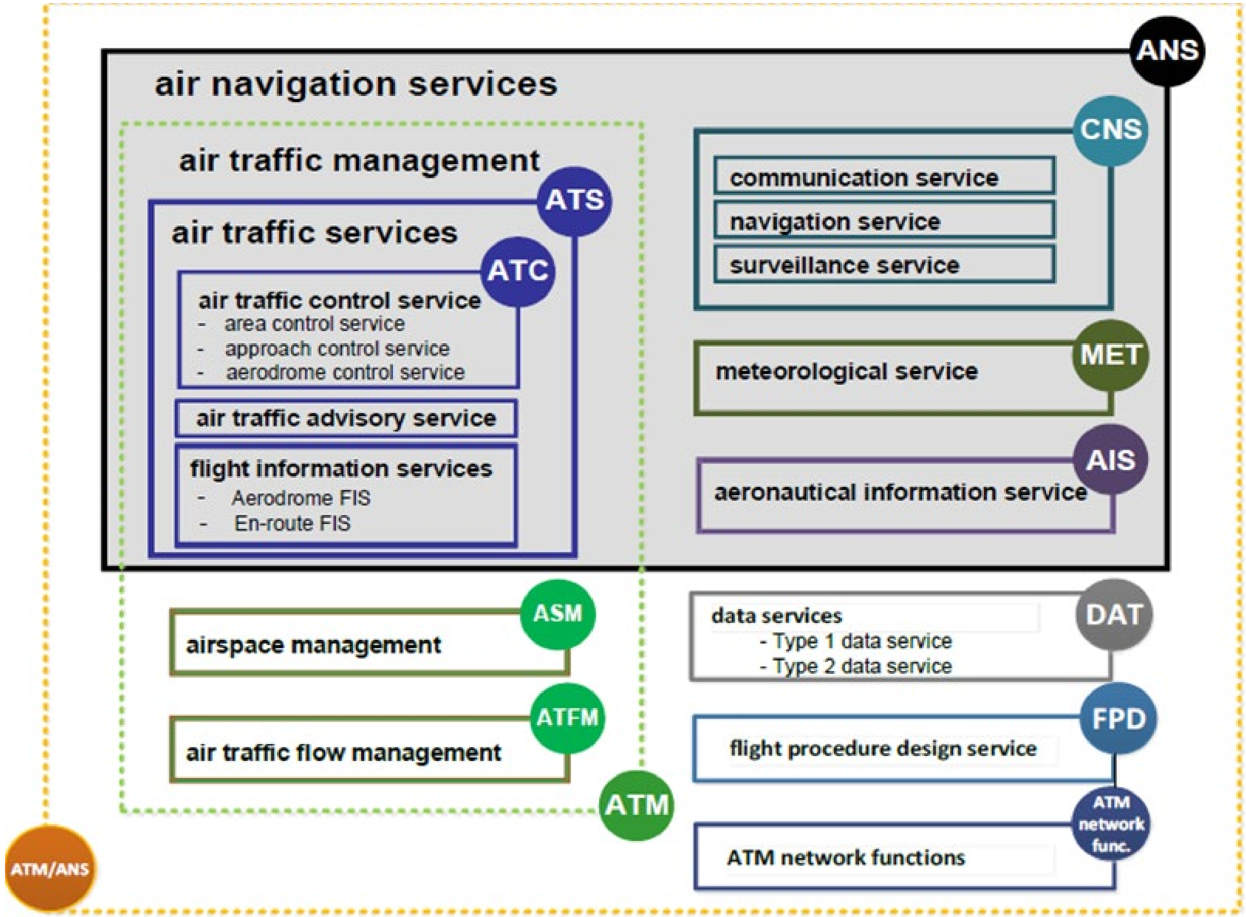
Joseph Machrouh (Thales LAS/SIAM)
Corinne Gingins (SkyGuide)
François Triboulet (EASA)
Laurent Plateaux (DGAC)
Loic Eyglunent (DGAC)
www.thalesgroup.com

Background: ATM/ANS regulatory context

> **EU 2017/373 lays down common requirements for providers of air traffic management/air navigation services and other air traffic management network functions and their oversight.**

> **EU 2017/373 distinguishes between**

1. Air Traffic Services (ATS) and
2. Other than Air Traffic Services (non ATS)



ATS vs. non ATS responsibilities

ATS

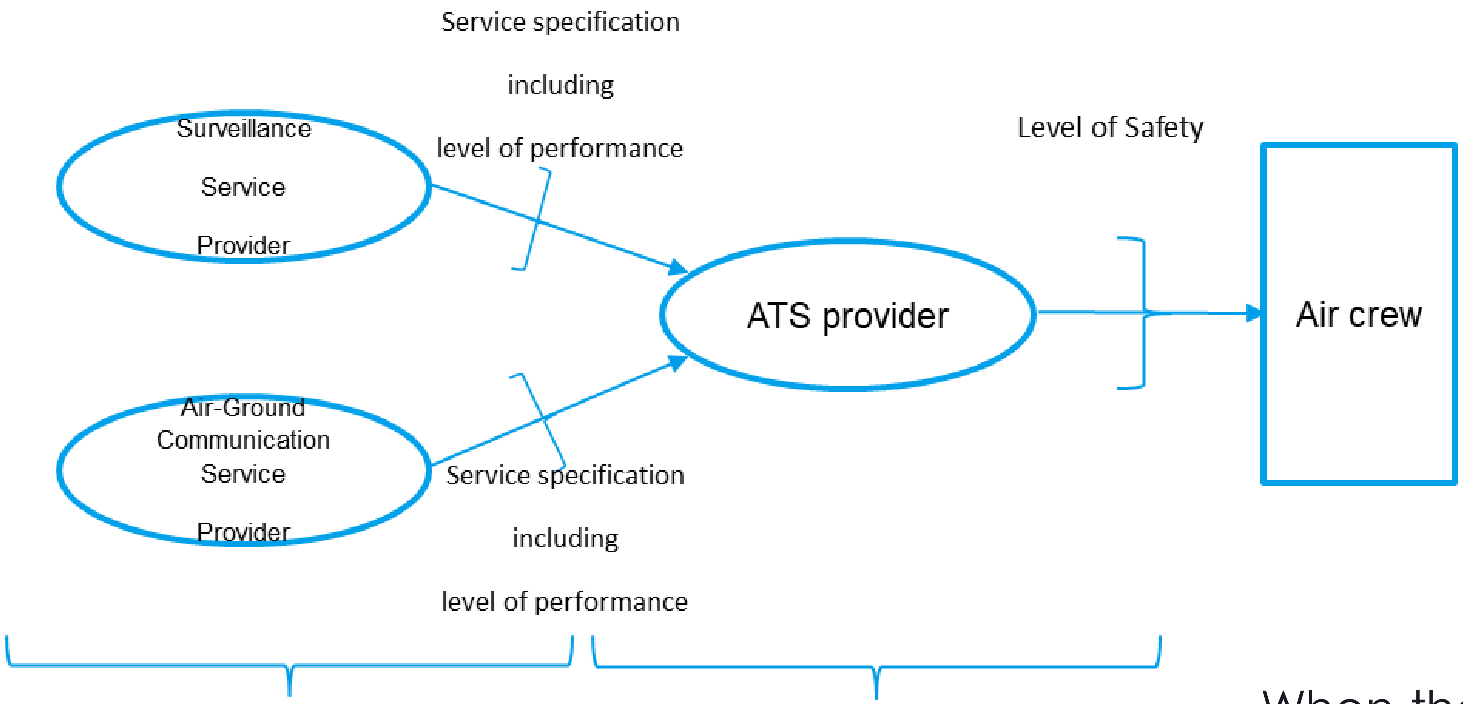
- ensure the safety of flights during the different phases of a flight.
- monitor the airspace situation
- provide clearances to improve the efficiency of the aviation network
- prevent any kind of conflicts, in the air and on the ground.

Concretely, they can directly interact with aircraft crews and provide information or guidance

Non ATS

- provide supporting services to ATS providers.
- For example, surveillance service, air-ground communication service or aeronautical information service
- None of them directly provides a service to the crews

Interfaces ATS non ATS



Non-ATS providers provide a service without the knowledge of the final safety performance which depends on ATS provider’s functional system and operational procedures

ATS provider applies operational procedures and rely on non-ATS provider to ensure a proper level of safety.

When the non ATS service specification evolves, this necessitates coordination with ATS provider, and this may trigger safety assessment activities at the latter.

ATS vs. non ATS: applicable regulatory requirements

ATS

- Certified considering the level of safety they can ensure
- SMS
- Changes to functional system supported by safety assessment to demonstrate maintenance of the safety level

Non ATS

- Certified considering their adherence to their service specification including service performance.
- No SMS
- Changes to functional system supported by safety support assessment to demonstrate adherence to service specification

ARP-6983|ED-324 Usability in ATM/ANS Context

To maintain readability and simplicity, no distinction is made between ATS and non-ATS in the core text of the ARP/ Standard, i.e. without considering non-ATS provider specificities.

However, the ARP6983|ED-324 can not be used "as is" by non ATS services

ATS non-ATS Annex aims at clarifying how to apply the standard for both ATS and non-ATS provider.

ATS non-ATS Annex Content Summary

General considerations on how to interpret the safety-related terms of this standard from a non-ATS provider regulatory standpoint

chapters necessitating additional or more accurate explanations are supplemented to fit with non-ATS provider concepts and ensure adequate understanding by non-ATS provider's systems manufacturers in applying this standard.

Safety related terms requiring interpretation

- Safety-critical
- Safety risk
- Safety process
- Safety objectives/criteria
- Acceptable level of safety
- Safety assessment
- Safety requirements
- Derived requirements
- Safety impact/safety events
- Functional hazard assessment (FHA)
- Preliminary System Safety Assessment (PSSA)/System Safety Assessment

Interpretation of safety related terms: examples

- **Safety-critical**: for non-ATS provider, this term highlights the need to accurately control the satisfaction of the specification and all associated performances as any deviation may have consequences on the ATS providers using this service.

- **Safety risk**: no safety risk is to be considered and managed by non-ATS provider, all requirements and design choices are derived from business decisions and regulatory requirements and are further formalized in the service specification. As such when this standard refers to "safety risk", non-ATS provider or non-ATS provider equipment manufacturers should read "risk of not meeting service specification".

- **Safety process**: should read safety assessment process for ATS providers and safety support assessment process for non ATS providers.

THALES
Building a future we can all trust

Contact

Fateh Kaakai

Safety Expert and AI Assurance Researcher

M: +33 (0) 6 65 45 98 34

fateh.kaakai@thalesgroup.com

Joseph Machrouh

Safety Expert & New technologies R&T lead

M: +33 (0) 6 37 58 19 02

joseph.machrouh@thalesgroup.com

THALES

Building a future we can all trust

Thank you

www.thalesgroup.com