2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
DECEMBER 6 - 7, 2022

Accellera FS WG Update

Alessandra Nardi, Accellera FS WG Chair
Ghani Kanawati, Technical Director, ARM

accellera SYSTEMS INITIATIVE

Agenda

- The Accellera Functional Safety Working Group (FS WG)
  - Challenges and Requirements
  - Mission and the FS Standardization Landscape
  - Scope and Key Objectives
- The Accellera Functional Safety Standard
  - FMEDA process formalization
  - Conceptual Data Model (Entities and Attributes)
  - Examples (using a prototype language)
  - Validation
  - Challenges and Methodologies
- What's Next?
- Further scoping of industry requirements
  - Safety Requirements Handling
  - Verification

SYNOPSYS®

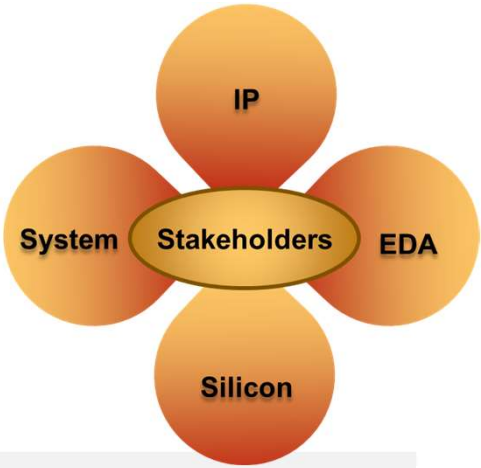
2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
DECEMBER 6 - 7, 2022

The Accellera Functional Safety Working Group

accellera SYSTEMS INITIATIVE

Accellera FS WG: History and Statistics



Industry needs and support (**What and Why**)

October 2019
PWG Formation

December 2019
F2F Kickoff

30+ companies

February 2020
WG Formation

March & October 2020
Virtual F2F

Q2 2021
White Paper

30+ companies

End-2022
White Paper on Conceptual Data model definition

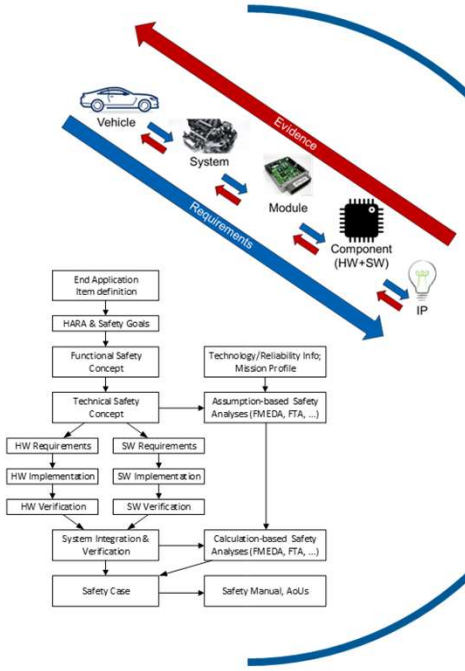
Q2 2023
Draft language (LRM) release

Standardization work (**How**)

Alessandra Nardi, Functional Safety Working Group **Chair**
Bala Chavali, Functional Safety Working Group **Vice-Chair**
Darren Galpin, Functional Safety Working Group **Secretary**

Work in Progress

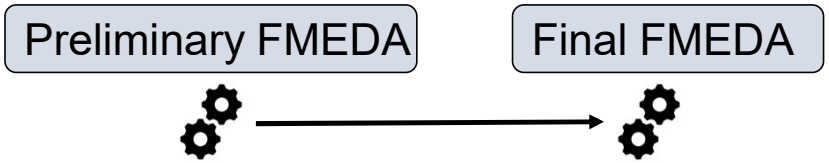
Challenges and Requirements



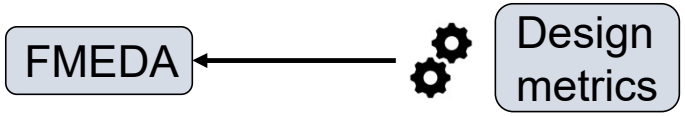
1. Exchange of same FS data across different automation tools
2. Connection between the FS data and the design information
3. Sharing of different FS data across operations/work products in the same layer
4. Exchange of FS data between suppliers and integrators
5. Traceability of information across the distributed development environment

Examples of Challenges and Requirements

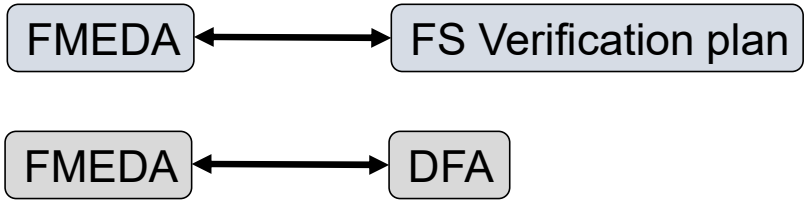
1 Exchange of same FS data across different automation tools



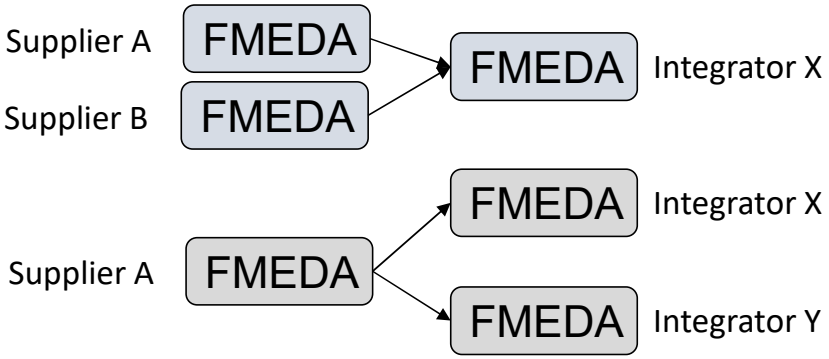
2 Connection between the FS data and the design information



3 Sharing of FS data across operations/work products in the same layer

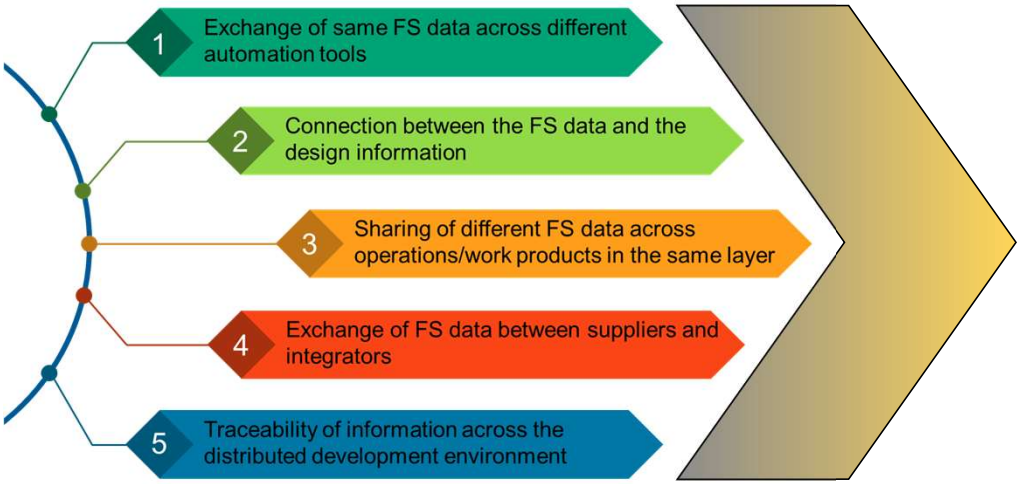


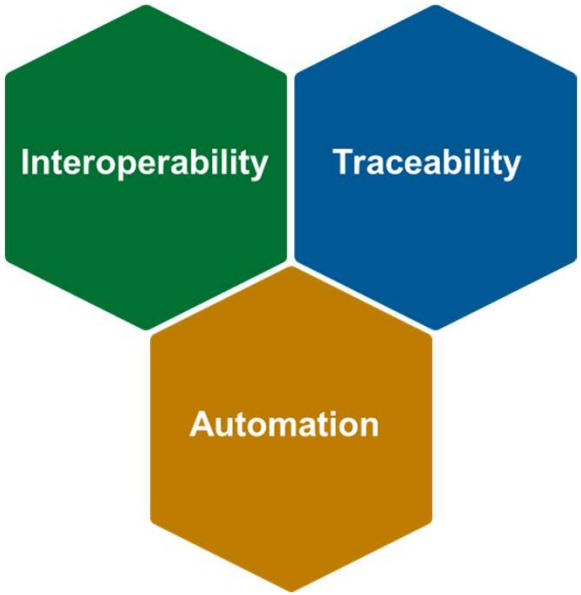
4 Exchange of FS data between suppliers and integrators



5 Traceability of information across the distributed development environment

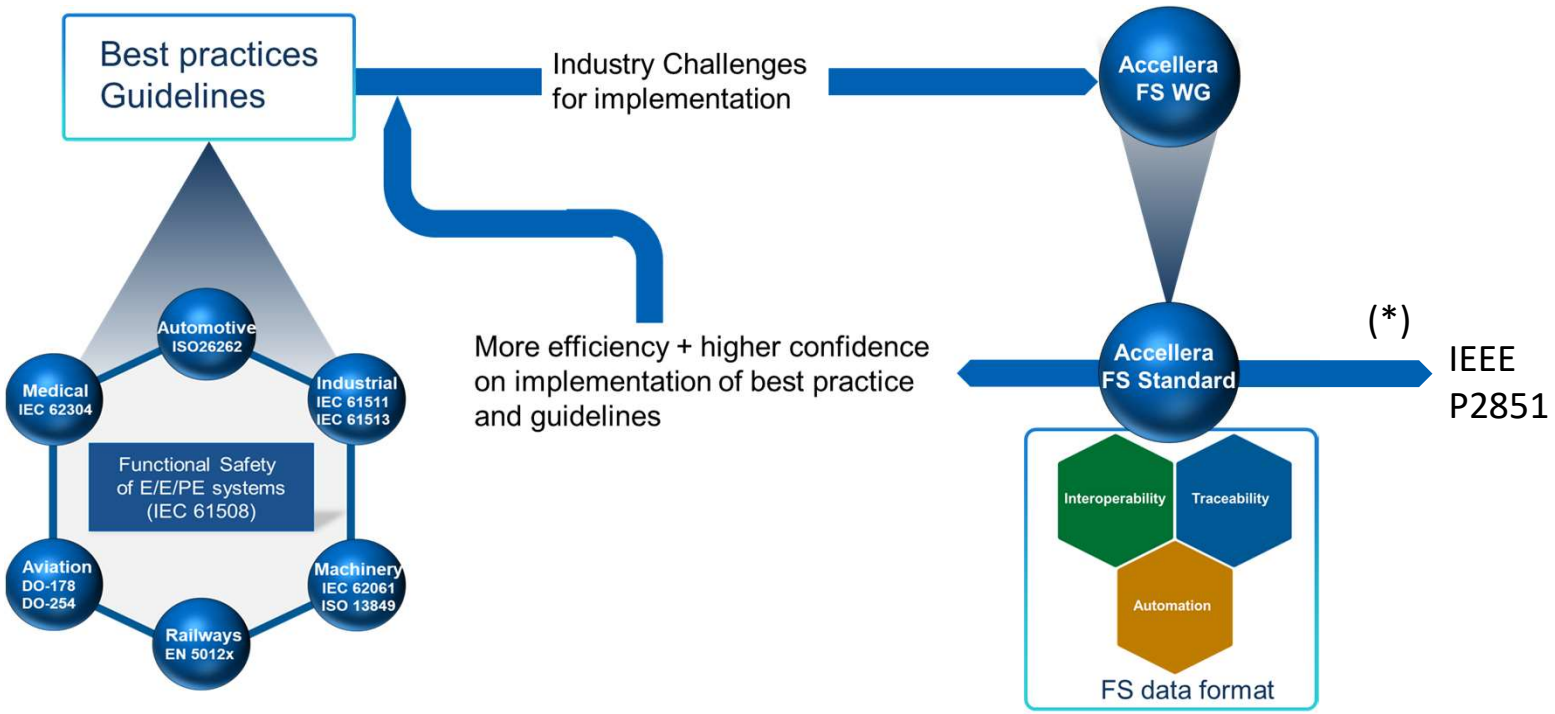
Mission of the FS WG





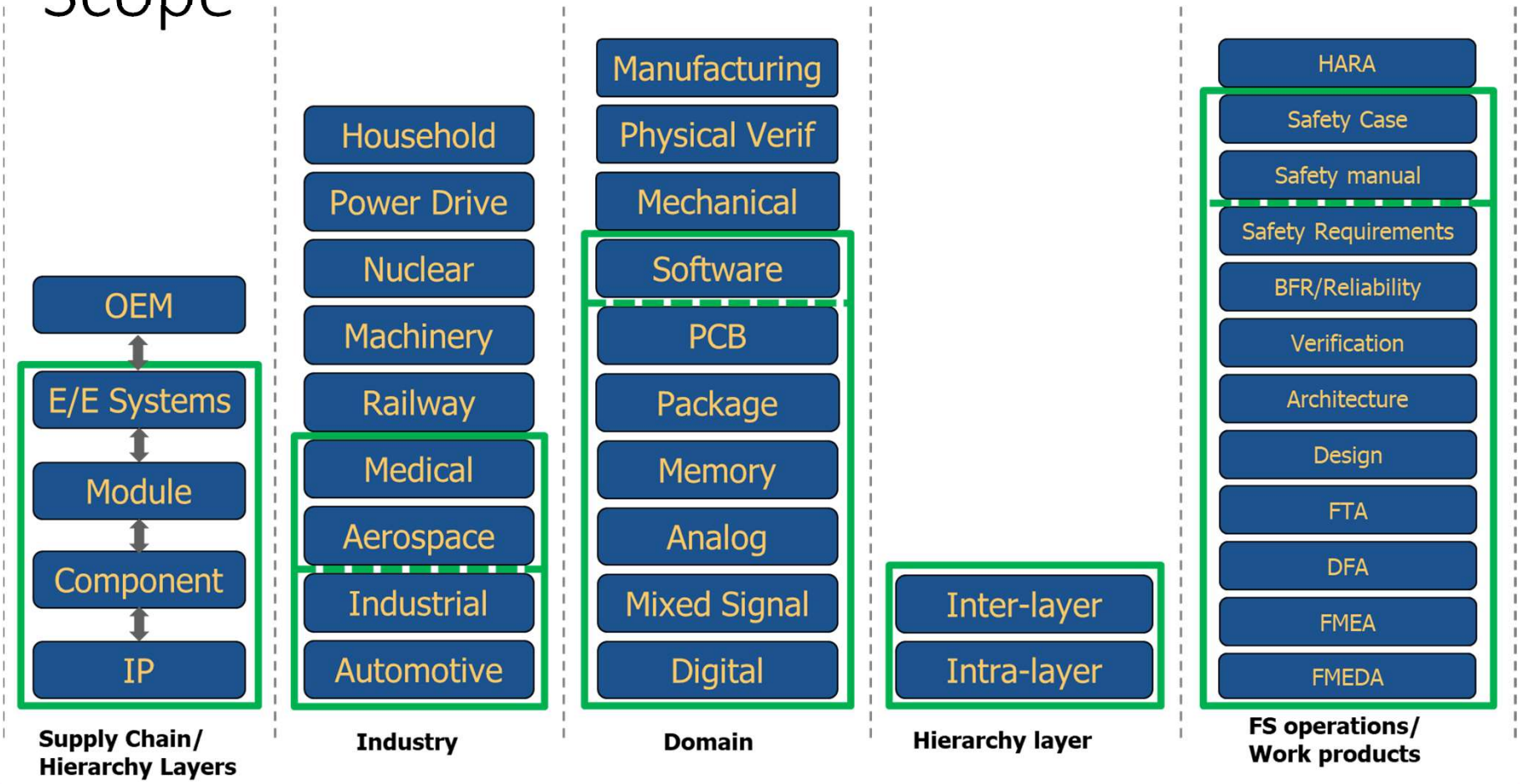
- Define a FS language to capture and propagate the functional safety data through the flow/supply chain
- Enable interoperability, traceability and automation

Mission and the FS standardization Landscape

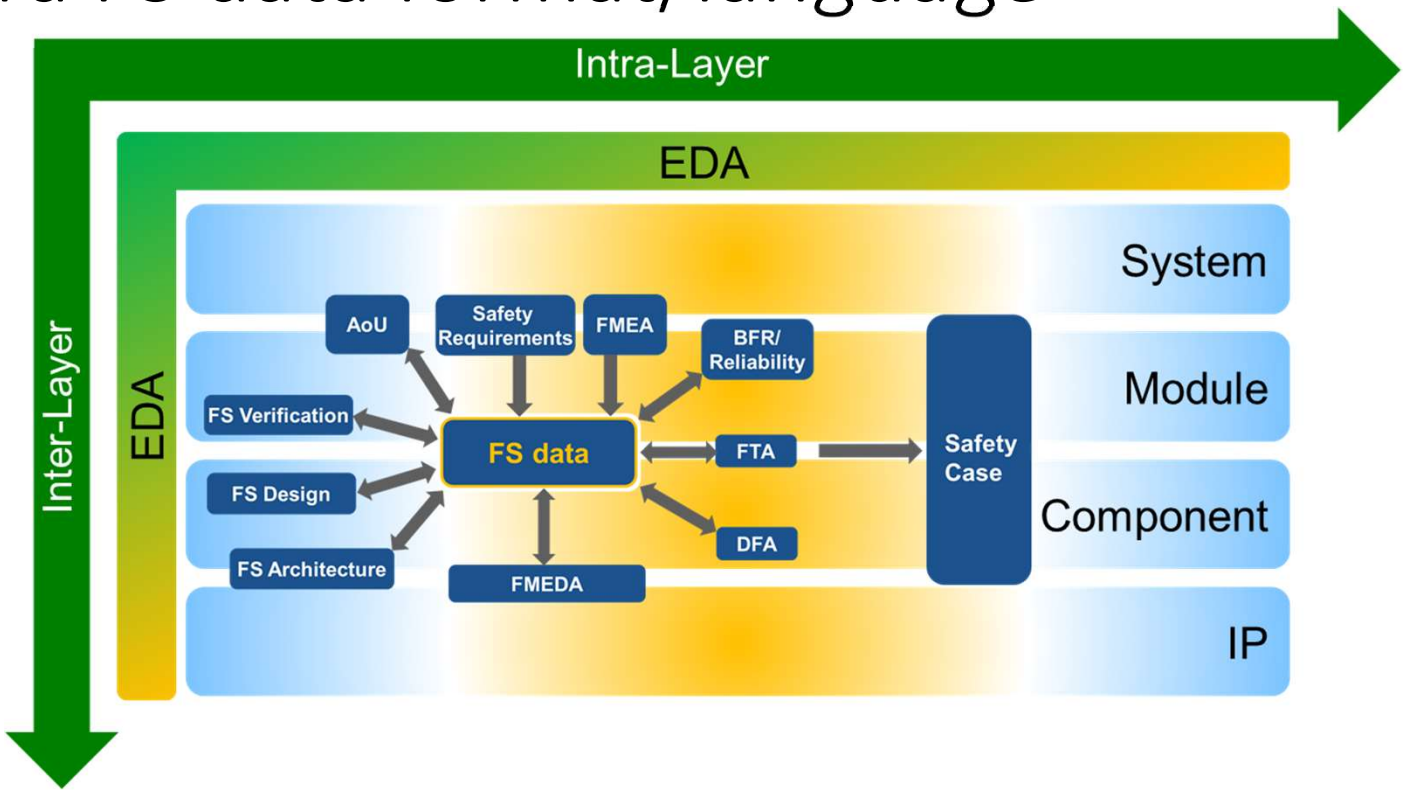


(*) Once completed and published, the Accellera FS standard is planned to be contributed to IEEE as per traditional collaboration between Accellera and IEEE

Scope

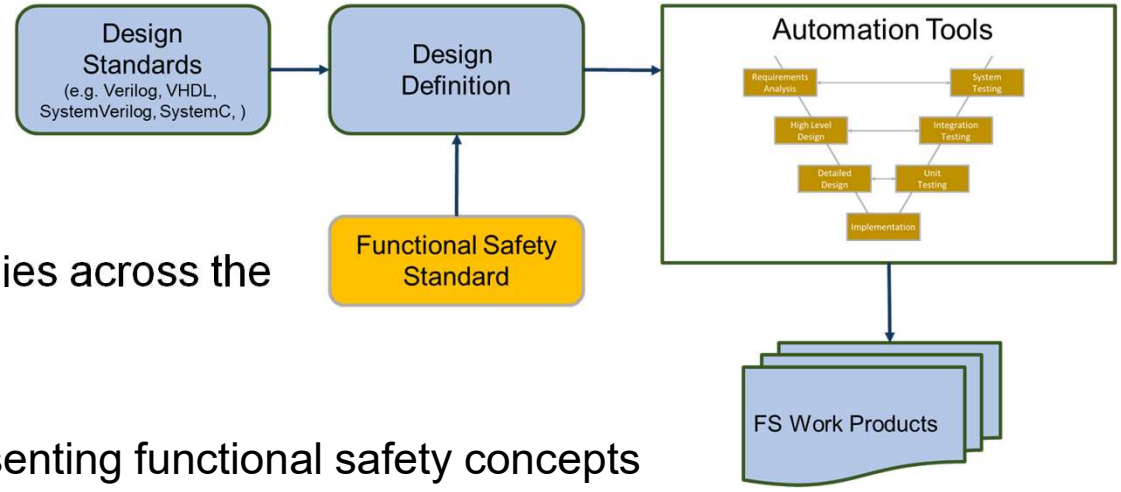

OEM
E/E Systems
Module
Component
IP
Supply Chain/ Hierarchy Layers
Household
Power Drive
Nuclear
Machinery
Railway
Medical
Aerospace
Industrial
Automotive
Industry
Manufacturing
Physical Verif
Mechanical
Software
PCB
Package
Memory
Analog
Mixed Signal
Digital
Domain
Inter-layer
Intra-layer
Hierarchy layer
HARA
Safety Case
Safety manual
Safety Requirements
BFR/Reliability
Verification
Architecture
Design
FTA
DFA
FMEA
FMEDA
FS operations/ Work products

Accellera FS data format/language



FS data = set of data needed to perform safety activities and to generate work products

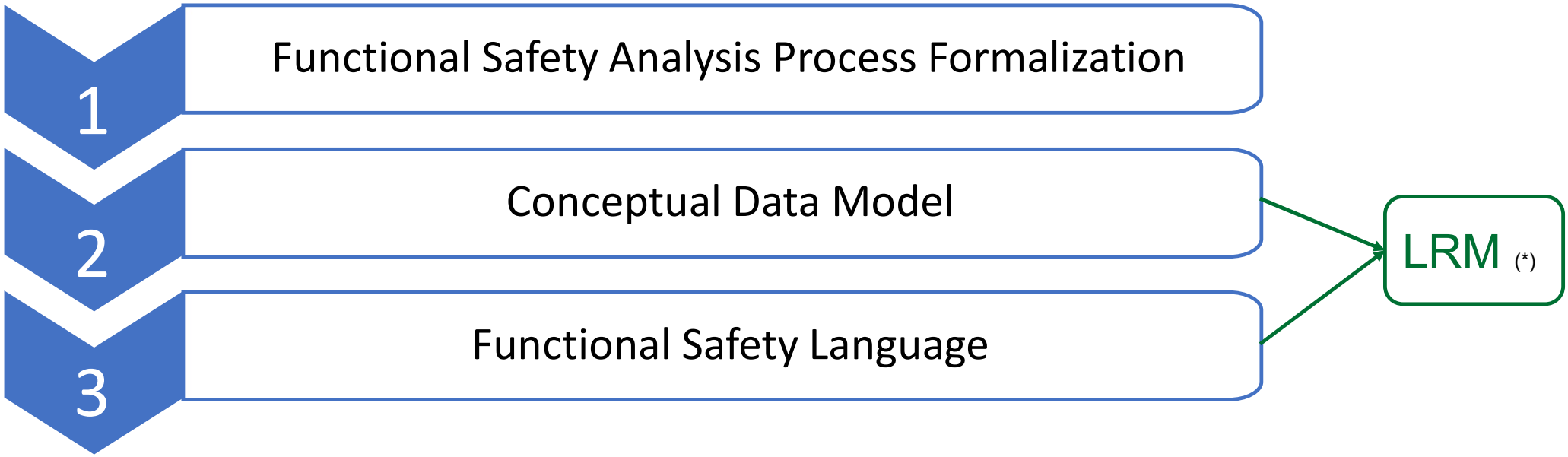
Key Objectives



- Harmonize best practices and methodologies across the industry via common language
- Enable efficient interchange of data representing functional safety concepts
  - across the diverse lifecycle development tool chain and
  - among organizations engaged in distributed development
- Be comprehensive, flexible, and scalable to minimize future perceived needs for local or proprietary customization

The data model is in addition to the existing design standards

Approach to Data Model Development



The actual exchange of information will happen through the FS Language

(*) Language Reference Manual

The conceptual data model approach

Goals:

- Define FS data
- Not to provide a reference implementation
- Systematic approach to define a language/format

Conceptual Data Model:

- Defines WHAT the system contains
- Does NOT define HOW the system should be implemented

Source: https://www.guru99.com/data-modelling-conceptual-logical.html

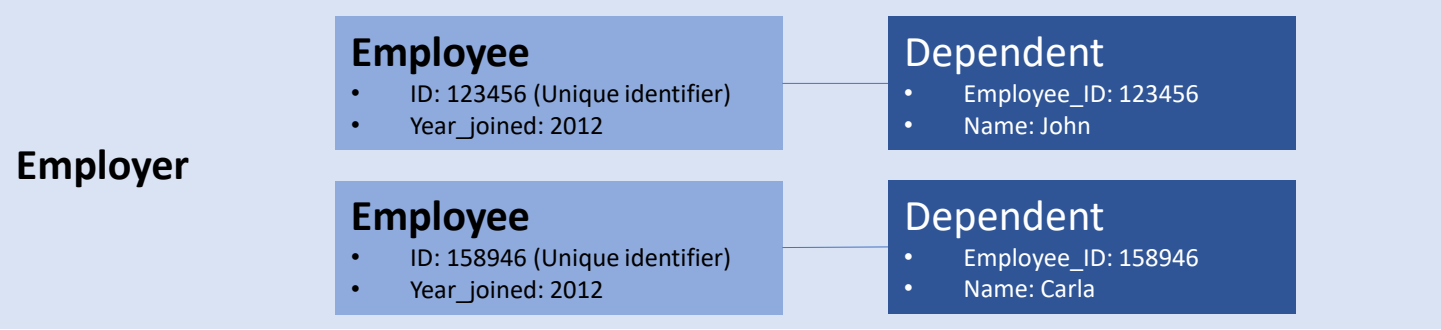
Using the Entity Relationship model

The 3 basic tenants:

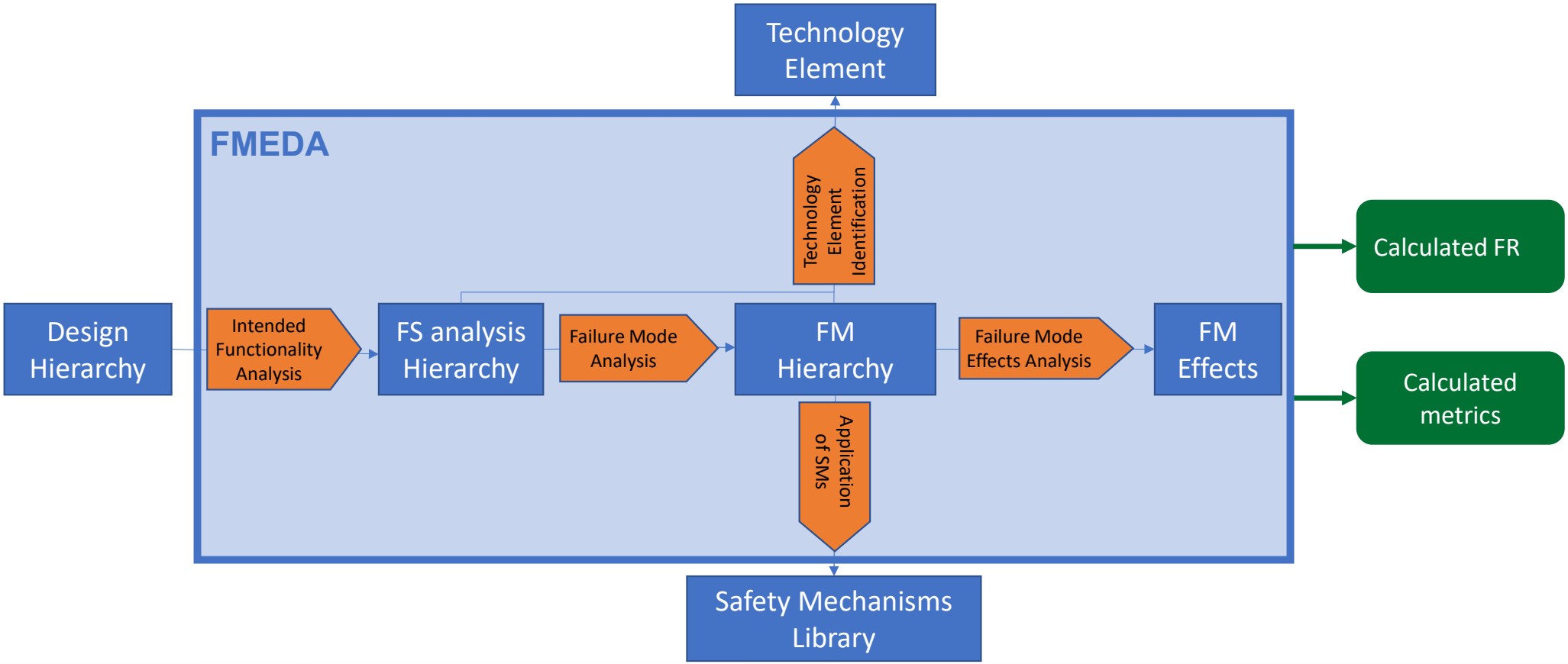
Source: https://www.guru99.com/data-modelling-conceptual-logical.html

- **Entity**: The object/data describing the system to be modeled
- **Attribute**: Characteristics or properties of an entity
- **Relationship**: Dependency or association between two entities

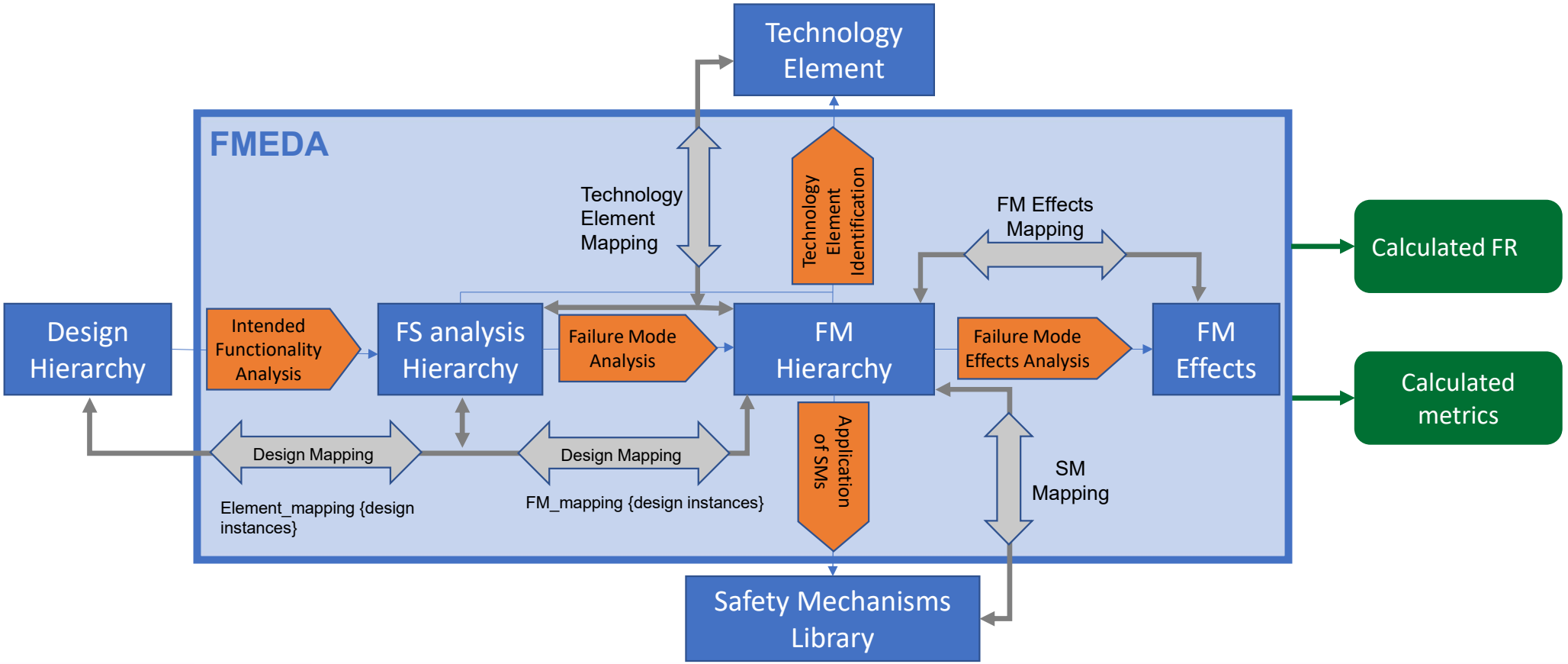
In addition, we rely on the concept **Weak entity**, which cannot be identified by its attributes alone, but only exists in the context of another entity



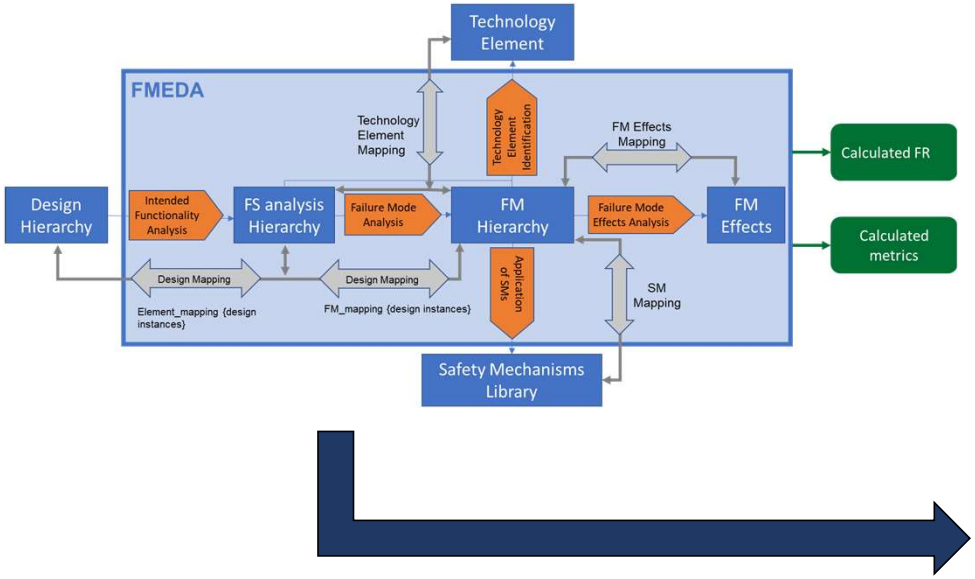
Functional Safety Analysis Process Formalization



Functional Safety Analysis Process Formalization



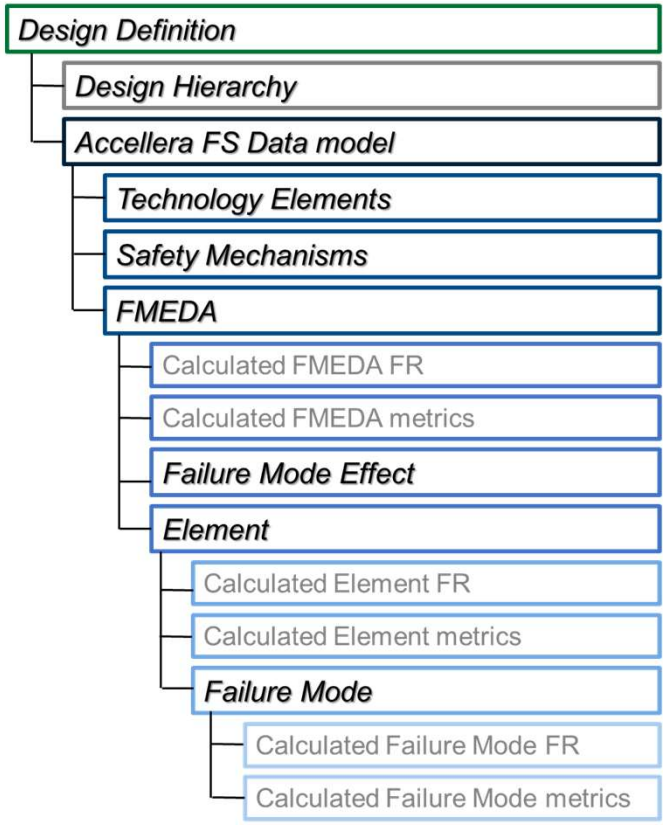
Conceptual Data Model derived from the FMEDA process



FMEDA process data	Entity Type	Information Type
FMEDA	FMEDA	Object
FS Analysis Hierarchy	Element	Object
FM Hierarchy	Failure_Mode	Object
Technology Element	Technology_Element	Object
Safety Mechanism Library	Safety_Mechanism	Object
FM Effects	Failure_Mode_Effect	Object
SM Mapping	SM-FM	Relationship
FM Effects Mapping	FM-FME	Relationship
Technology Element Mapping	TE-FM	Relationship
Technology Element Mapping	TE-Element	Relationship
Design Mapping	Inside the TE-FM since there is no Design Hierarchy in the datamodel	Relationship
Design Mapping	Inside the TE-Element since there is no Design Hierarchy in the datamodel	Relationship
Calculated FR	FR_ISO26262	Weak object (*)
Calculated metrics	Metrics_ISO26262	Weak object (*)
Calculated FR	FR_IEC61508	Weak object (*)
Calculated metrics	Metrics_IEC61508	Weak object (*)

Direct traceability from the data + mapping of FMEDA process to data model

Conceptual Data Model scope and hierarchy



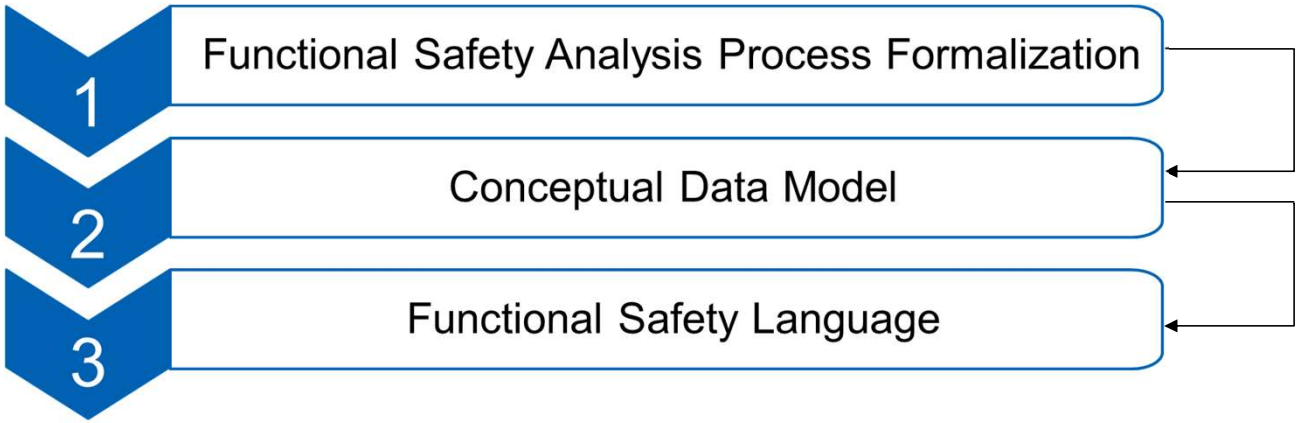
Sample Language

- Following the principle of traceability, a sample language can be derived directly from the conceptual data model with clear rules:
  - **Objects** are created and updated with **“create”** and **“set”** commands
  - **Relationships** are created with the **“assign”** commands
  - **Weak** objects are assigned a value with the command **“define”**
- Special rule stands for the Design mapping:
  - Since it connects objects in the data model to objects in the design hierarchy (not part of the data model)
  - It is described through the “-mapping” and “-exclude_mapping” options inside the design mapping relationship commands.

Conceptual Data Model + sample commands

FMEDA process data	Entity Type	Information Type	Commands
FMEDA	FMEDA	Object	create_fmeda, set_fmeda
FS Analysis Hierarchy	Element	Object	create_element, set_element
FM Hierarchy	Failure_Mode	Object	create_failure_mode, set_failure_mode
Technology Element	Technology_Element	Object	create_failure_mode, set_failure_mode
Safety Mechanism Library	Safety_Mechanism	Object	create_failure_mode, set_failure_mode
FM Effects	Failure_Mode_Effect	Object	create_failure_mode, set_failure_mode
SM Mapping	SM-FM	Relationship	assign_SM_FM
FM Effects Mapping	FM-FME	Relationship	assign_FM_FME
Technology Element Mapping	TE-FM	Relationship	assign_TE_FM
Technology Element Mapping	TE-Element	Relationship	Assign_TE_Element
Design Mapping	Inside the TE-FM since there is no Design Hierarchy in the datamodel	Relationship	assign_TE_FM –mapping {…} –exclude_mapping
Design Mapping	Inside the TE-Element since there is no Design Hierarchy in the datamodel	Relationship	assign_TE_Element –mapping {…} –exclude_mapping
Calculated FR	FR_ISO26262	Weak object (*)	define_FR_ISO26262
Calculated metrics	Metrics_ISO26262	Weak object (*)	define_metric_ISO26262
Calculated FR	FR_IEC61508	Weak object (*)	define_FR_IEC61508
Calculated metrics	Metrics_IEC61508	Weak object (*)	define_metric_IEC61508

Traceability of Data Model Development



Direct consistent mapping from process to data model objects

Direct translation from data model to language with defined rules

Traceability from:
- Requirements (FMEDA process objects and mapping) to
- Implementation of requirements (FS data model and then language commands)

R: Required
D: Derived

Detailed Conceptual Data Model

Entity	Attribute Name	Attribute Type	Default	Description	R	D
FMEDA	FMEDA_Name	String	N/A	Name (identifier) of the FMEDA of the project.	Y	N
	Type	Enumerate { assumption-based, calculation-based}	Calculation-based	Selects whether the FMEDA is assumption-based or calculation-based. This attribute is informative only. If type = calculation-based, the user can still specify the failure mode contribution through the “failure mode size attribute”.	N	N
	ASIL	Enumerate { A, B, C, D}	D	Defines the ASIL for the FMEDA (for a given Safety Goal) according to ISO26262 Used also to specify that the FMEDA is for ISO26262	N	N
	SIL	Enumerate { 1, 2, 3, 4}	1	Defines the SIL for the FMEDA according to IEC61508 Used also to specify that the FMEDA is for IEC61508	N	N
	Analysis_Type	List of Enumerate { Permanent Transient All}	All	Defines the failure types to be considered and which metrics to be calculated within the safety analysis. More than one value can be specified, e.g. Failure_Type = {Permanent} or Failure_Type = {Permanent, Transient} The value “All” implies all Failure Types are activated. Defined as “All” instead of “Both”, to allows to plan for more than just Transient and Permanent.	Y?	N
	Creator	String	N/A	Name of the company that generated the FMEDA.	N	N
	Date	Date	N/A	Date when the FMEDA was generated.	N	N
	Version	Float	N/A	Version of the FMEDA.	N	N
	Data_Model_Version	Float	N/A	Version of the data model	N	N
	Comment	String	N/A	Information which does not have a specific field in the FMEDA object.	N	N

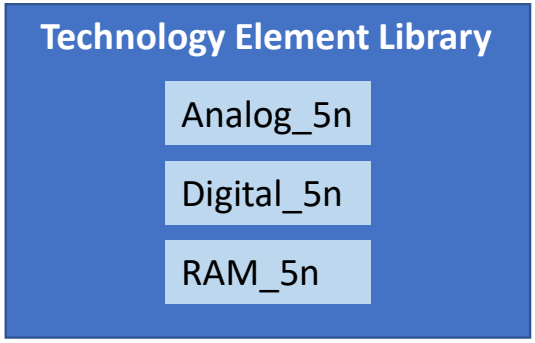
R: Required
D: Derived

Detailed Conceptual Data Model

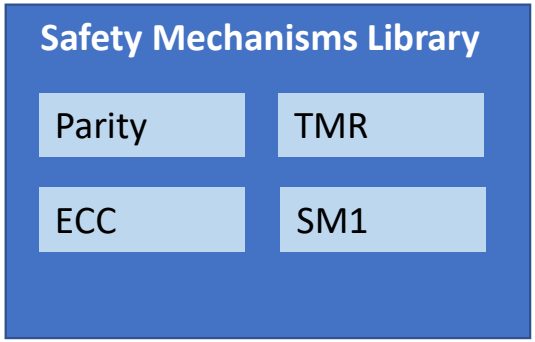
Category	Attribute Name	Attribute Type	Default	Description	R	D
Element	Element_Name	String	N/A	Name (identifier) of the Element	Y	N
	Element_Description	String	N/A	Description of the intended functionality of the Element	N	N
	Element_Type	Enum { System, Element, SubElement, Component, SubComponent, Part, SubPart}	?	Specifies the type of the Element. Element_Type = Component or SubComponent can only be defined if the analysis is for IEC61508, inferred from the FMEDA entity, whether it has ASIL or SIL defined	Y	N
	Parent_Element	String	N/A	Connects the Element to its Parent in the FS hierarchy	N	N
	FMEDA_Name	String	N/A	Connects the FS hierarchy to the FMEDA project	Y	N

Example #1 – Project Independent

- Define a Technology Element library
  - “Analog_5n” FR_permanent=3e-9
  - “Digital_5n” FR_permanent=1e-9 FR_transient=8e-9
  - “RAM_5n” FR_transient=10e-9



- Define a Safety Mechanism library
  - Parity DC_transient=70
  - ECC DC_transient=60
  - TMR DC_transient=99
  - SM1 DC_permanent=78

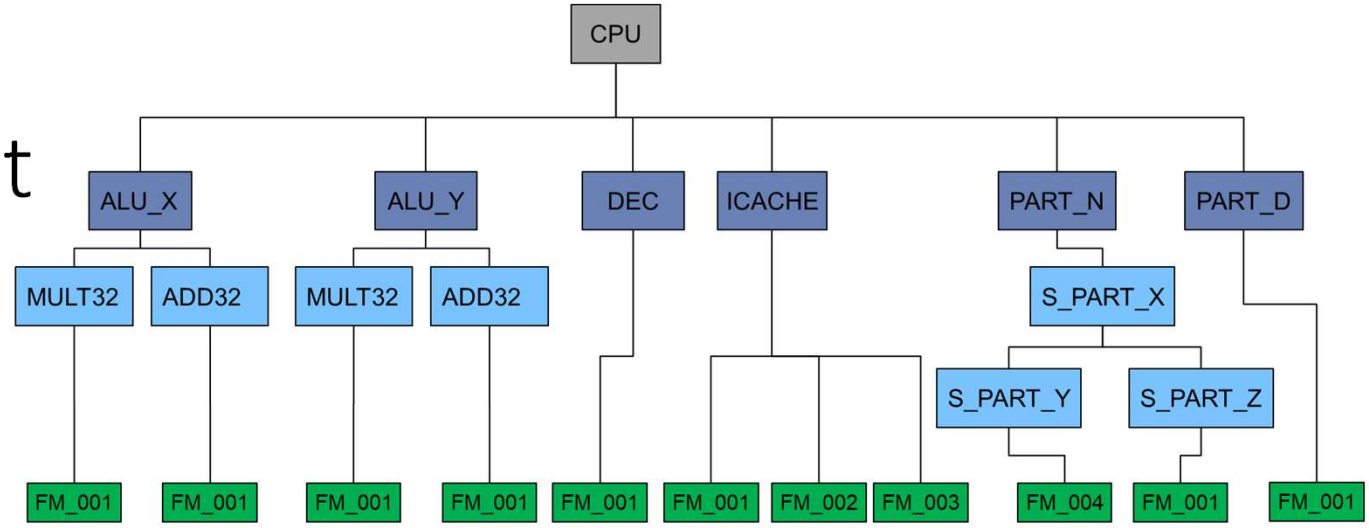


```
create_technology_element  -name “Analog_5n” –type “analog” –FR_permanent 3e-9
create_technology_element  -name “Digital_5n” –type “digital” –FR_permanent 1e-9 –FR_transient 8e-9
create_technology_element  -name “RAM_5n” –type “RAM” –FR_transient 10e-9



create_safety_mechanism –name “ECC” –DC_transient 70
create_safety_mechanism –name “ECC” –DC_transient 60
create_safety_mechanism –name “TMR” –DC_transient 99
create_safety_mechanism –name “SM1” –DC_permanent 78
```

Example Sample Language

Defining the TE and SM libraries

Example #1 – Project Dependent



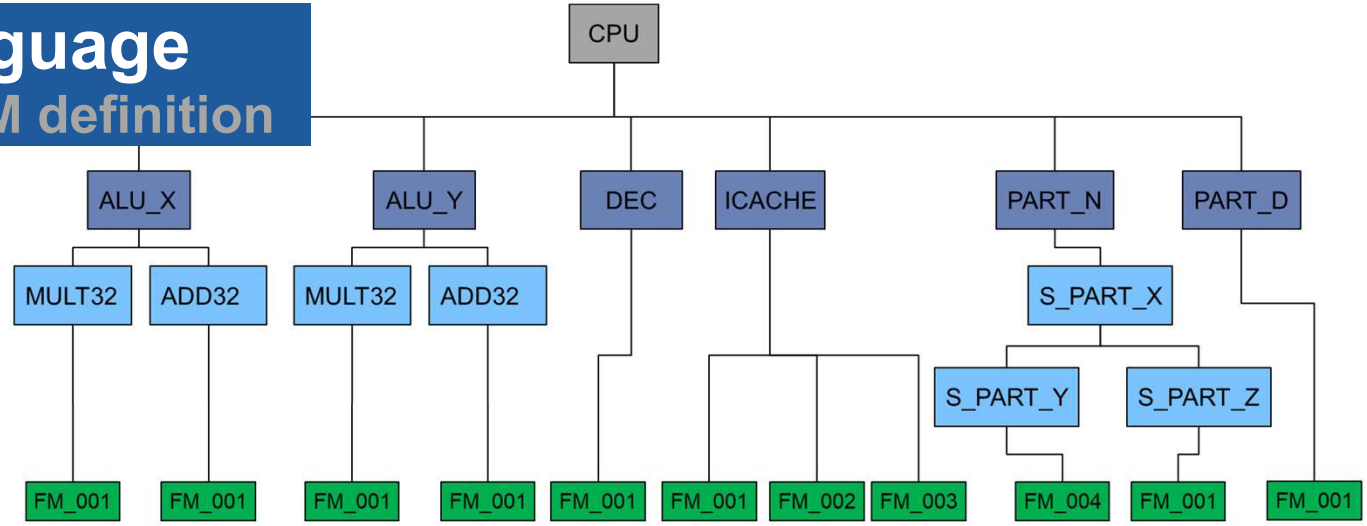
Top	Part	Subpart	Subpart	FM
CPU	ALU_X	MULT32		FM_001
		ADD32		FM_001
	ALU_Y	MULT32		FM_001
		ADD32		FM_001
	DEC			FM_001
	ICACHE			FM_001
				FM_002
				FM_003
	PARTN	S_PART_X	S_PART_Y	FM_004
			S_PART_Z	FM_001
	PARTD			FM_001

OR

Top	Part	Subpart	FM
CPU	ALU_X	MULT32	FM_001
		ADD32	FM_001
	ALU_Y	MULT32	FM_001
		ADD32	FM_001
	DEC		FM_001
	ICACHE		FM_001
			FM_002
			FM_003
	PARTN	S_PART_X.S_PART_Y	FM_004
		S_PART_X.S_PART_Z	FM_001
	PARTD		FM_001

Example Sample Language

FMEDA, FS Hierarchy and FM definition

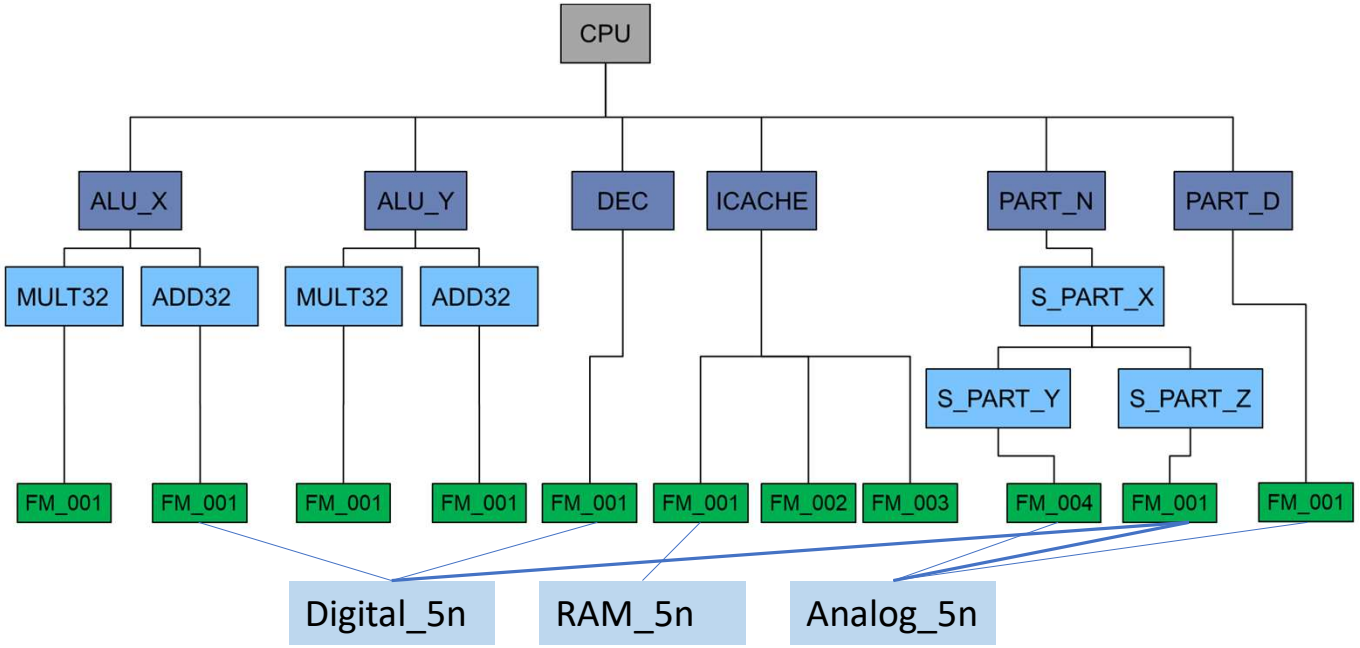



```
create_fmeda -name “CPU_FMEDA” -type “assumption”

create_element -name “ALU_X” -type part -fmeda “CPU_FMEDA”
create_element -name “ALU_Y” -type part -fmeda “CPU_FMEDA”
create_element -name “DEC” -type part -fmeda “CPU_FMEDA”
create_element -name “ICACHE” -type part -fmeda “CPU_FMEDA”
create_element -name “PARTN” -type part -fmeda “CPU_FMEDA”
create_element -name “PARTD” -type part -fmeda “CPU_FMEDA”
```

```
create_element -name “MULT32” -type subpart -parent_element “ALU_X” -fmeda “CPU_FMEDA”
create_element -name “ADDER” -type subpart -parent_element “ALU_X” -fmeda “CPU_FMEDA”

create_element -name “MULT32” -type subpart -parent_element “ALU_Y” -fmeda “CPU_FMEDA”
create_element -name “ADDER” -type subpart -parent_element “ALU_Y” -fmeda “CPU_FMEDA”

create_element -name “S_PART_X” -type subpart -parent_element “PARTN” -fmeda “CPU_FMEDA”

create_element -name “S_PART_Y” -type subpart -parent_element “PARTN.S_PART_X” -fmeda
“CPU_FMEDA”
create_element -name “S_PART_Z” -type subpart -parent_element “PARTN.S_PART_X” -fmeda
“CPU_FMEDA”

create_failure_mode -name “FM_001” -parent_element “ALU_X.MULT32” -fmeda  “CPU_FMEDA”
create_failure_mode -name “FM_001” -parent_element “ALU_X.ADDER” -fmeda  “CPU_FMEDA”
```

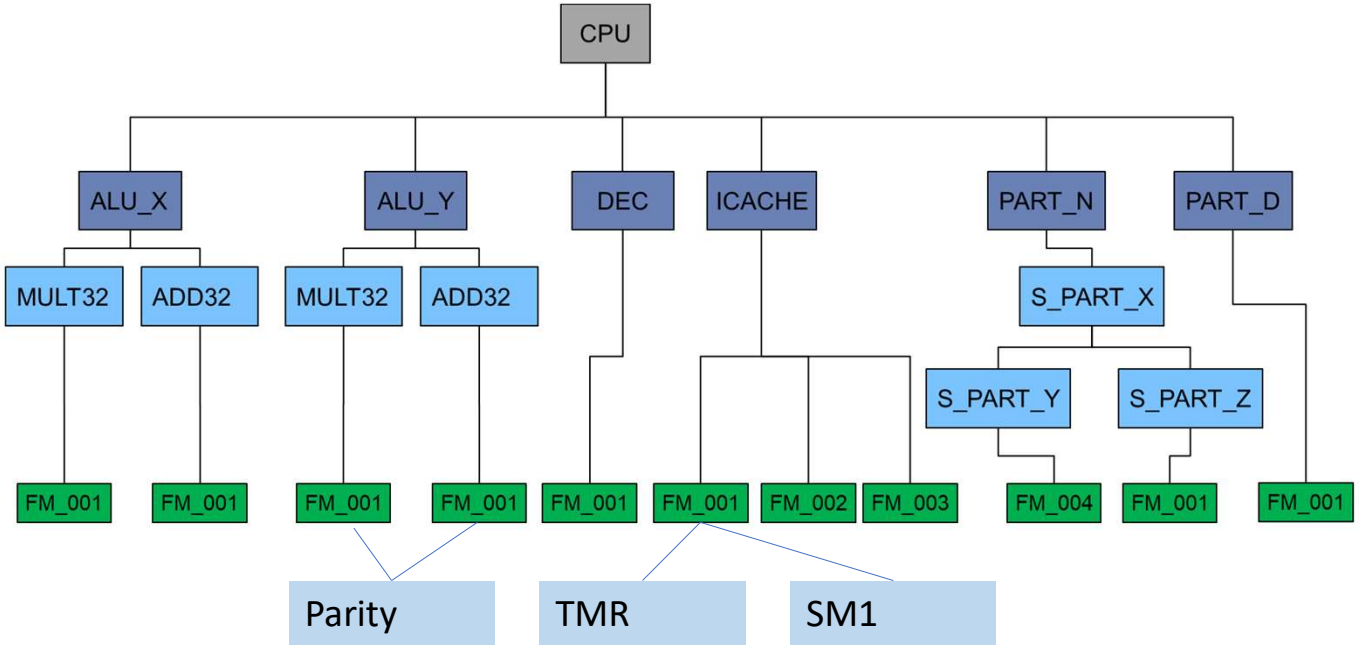
```
create_failure_mode -name “FM_001” -parent_element “ALU_Y.MULT32” -fmeda  “CPU_FMEDA”
create_failure_mode -name “FM_001” -parent_element “ALU_Y.ADDER” -fmeda  “CPU_FMEDA”

create_failure_mode -name “FM_001” -parent_element “DEC” -fmeda  “CPU_FMEDA”

create_failure_mode -name “FM_001” -parent_element “ICACHE” -fmeda  “CPU_FMEDA”
create_failure_mode -name “FM_002” -parent_element “ICACHE” -fmeda  “CPU_FMEDA”
create_failure_mode -name “FM_003” -parent_element “ICACHE” -fmeda  “CPU_FMEDA”

create_failure_mode -name “FM_004” -parent_element “PARTN.S_PART_X.S_PART_Y” -fmeda
“CPU_FMEDA”
create_failure_mode -name “FM_001” -parent_element “PARTN.S_PART_X.S_PART_Z” -fmeda
“CPU_FMEDA”

create_failure_mode -name “FM_001” -parent_element “PARTD” -fmeda  “CPU_FMEDA”
```

Example Sample Language
FM-TE Mapping




```
assign_TE_FM –TE_name “Analog_5n” –FM_name “FM_001” -parent_element “PARTD” -fmeda  “CPU_FMEDA” –FM_size_permanent 10
assign_TE_FM –TE_name “Digital_5n” –FM_name “FM_001” -parent_element “ALU_X.MULT32” -fmeda  “CPU_FMEDA” –FM_size_permanent 35
…
assign_TE_FM –TE_name “Digital_5n” –FM_name “FM_001” -parent_element “DEC” -fmeda  “CPU_FMEDA” –FM_size_permanent 10 –FM_size_transient 20
assign_TE_FM –TE_name “RAM_5n” –FM_name “FM_001” -parent_element “ICACHE” -fmeda  “CPU_FMEDA” –FM_size_transient 10

assign_TE_FM –TE_name “Analog_5n” –FM_name “FM_004” -parent_element “PARTN.S_PART_X.S_PART_Y” -fmeda  “CPU_FMEDA”
assign_TE_FM –TE_name “Analog_5n” –FM_name “FM_001” -parent_element “PARTN.S_PART_X.S_PART_Z” -fmeda  “CPU_FMEDA” –FM_size_permanent 5
assign_TE_FM –TE_name “Digital_5n” –FM_name “FM_001” -parent_element “PARTN.S_PART_X.S_PART_Z” -fmeda  “CPU_FMEDA” –FM_size_permanent 5
```

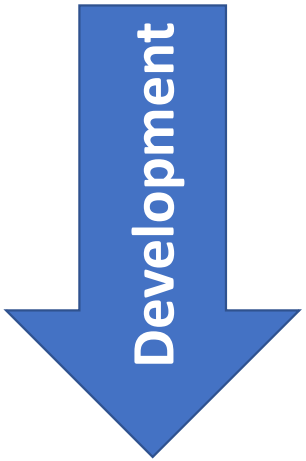
Example Sample Language
SM-FM Mapping




```
assign_SM_FM –SM_name “Parity” –FM_name “FM_001” -parent_element “ALU_X.MULT32” -fmeda  “CPU_FMEDA”
assign_SM_FM –SM_name “Parity” –FM_name “FM_001” -parent_element “ALU_X.ADD32” -fmeda  “CPU_FMEDA”


assign_SM_FM –SM_name “TMR” –FM_name “FM_001” -parent_element “ICACHE” -fmeda  “CPU_FMEDA”
assign_SM_FM –SM_name “SM1” –FM_name “FM_001” -parent_element “ICACHE” -fmeda  “CPU_FMEDA”
```

Validation

FMEDA process + Methodologies/Use Cases
(Requirements)



Conceptual Data Model + Language
(Implementation)

Validation of the Proposed Data Model
+ another proposal

FMEDA examples

Challenges

- Harmonization of the FMEDA process
- Agreement on the conceptual data model: top-down or bottom-up?
- Requirements and Use Cases
  - General: simplicity vs complexity (and flexibility)
  - Inputs and Outputs // Use cases
- Methodology
  - Handling Safety Mechanisms // Use cases and priority schema
  - Hierarchical FMEDAs (and integration)
- Language
  - Required for FMEDA vs required by the data model (the use of defaults)
  - Atomic commands vs split commands

Handling of Safety Mechanisms

SM: Safety Mechanism
FM: Failure Mode
DC: Diagnostic Coverage

Scope/Entity	Description	Attribute
SM Library	SM in isolation	DC_Perm DC_Trans
SM-FM	SM applied to a FM	DC_Perm DC_Trans
FM	Multiple SM applied to a FM	DC_Total_Perm DC_Total_Trans DC_Aggregation_method DC_expert

Handling of Safety Mechanisms

SM: Safety Mechanism
FM: Failure Mode
DC: Diagnostic Coverage

Scope/Entity	Description	Attribute
SM Library	SM in isolation	DC_Perm DC_Trans
SM-FM	SM applied to a FM	DC_Perm DC_Trans
FM	Multiple SM applied to a FM	DC_Total_Perm DC_Total_Trans DC_Aggregation_method DC_expert

SM1: DC_Perm=90%, DC_Trans=60%
SM2: DC_Perm=30%, DC_Trans=60%
SM3: DC_Perm=60%, DC_Trans=90%

SM1-FM1: DC_Perm=80%, DC_Trans=50%
SM2-FM1: DC_Perm=30%, DC_Trans=60%
SM3-FM1: DC_Perm=45%, DC_Trans=75%

FM1: Aggregate_Method=Max
FM1: DC_Total_Perm=80%
FM1: DC_Total_Trans=75%

Handling of Safety Mechanisms

SM: Safety Mechanism
FM: Failure Mode
DC: Diagnostic Coverage

Scope/Entity	Description	Estimated	Measured
SM Library	SM in isolation	DC_Perm DC_Trans	N/A
SM-FM	SM applied to a FM	DC_Perm_est DC_Trans_est	DC_Perm_meas DC_Trans_meas
FM	Multiple SM applied to a FM	DC_Total_Perm_est DC_Total_Trans_est DC_Aggregation_method DC_expert	DC_Total_Perm_meas DC_Total_Trans DC_Aggregation_method DC_expert

?

?

Do we care about this use case?

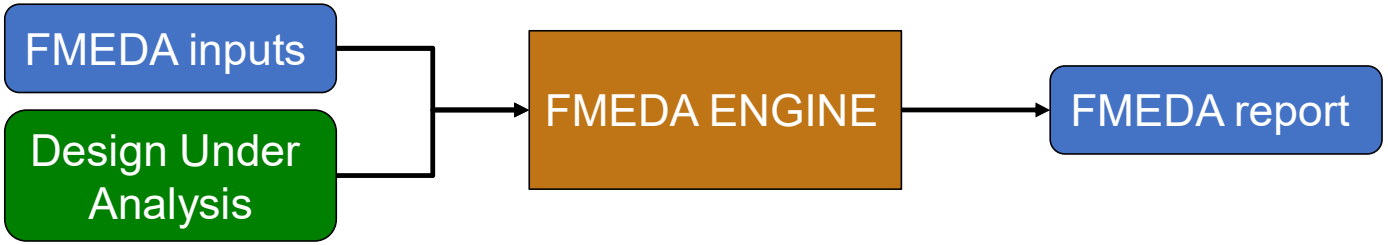
Simplicity or Complexity?

Scope/Entity	Description	Estimated	Measured
SM Library	SM in isolation	DC_Perm DC_Trans	N/A
SM-FM	SM applied to a FM	DC_Perm_est ? DC_Trans_est	? DC_Perm_meas DC_Trans_meas
FM	Multiple SM applied to a FM	DC_Total_Perm_est DC_Total_Trans_est DC_Aggregation_method DC_expert	DC_Total_Perm_meas DC_Total_Trans DC_Aggregation_method DC_expert

Do we care about this use case?

…and we could also have added a DC_aggregation for Permanent and DC_Aggregation_Transient

Two important use cases



- **Authoring/recalculating an FMEDA**
  - IP provider share FMEDA to integrator that will harden the IP
  - The integrator might also want to configure the IP
  - The data exchange focuses on the inputs to enable FMEDA calculation and update

- **Exchange/auditing an FMEDA report**
  - IP provider share FMEDA to integrator that will not modify it
  - Some input data used to calculate the metrics (e.g. Failure Mode size) might not be shared
  - The data exchange focuses on FMEDA reports (read and integrate)

- **And everything in between….configurability!**
- **FMEDA reports includes inputs and outputs….what goes into the language???**

What's Next?

- Wrapping up version 1.0
- Working on the White Paper to include the conceptual data model…stay tuned
- Looking for feedback
  - F2F on December 7 (open to the community)
  - F2F on December 8 (Accellera members/working session)
  - After the White Paper is published
- Finalize the language and publish the LRM (2023)

Questions ?

accellera
SYSTEMS INITIATIVE

SYNOPSYS®

2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

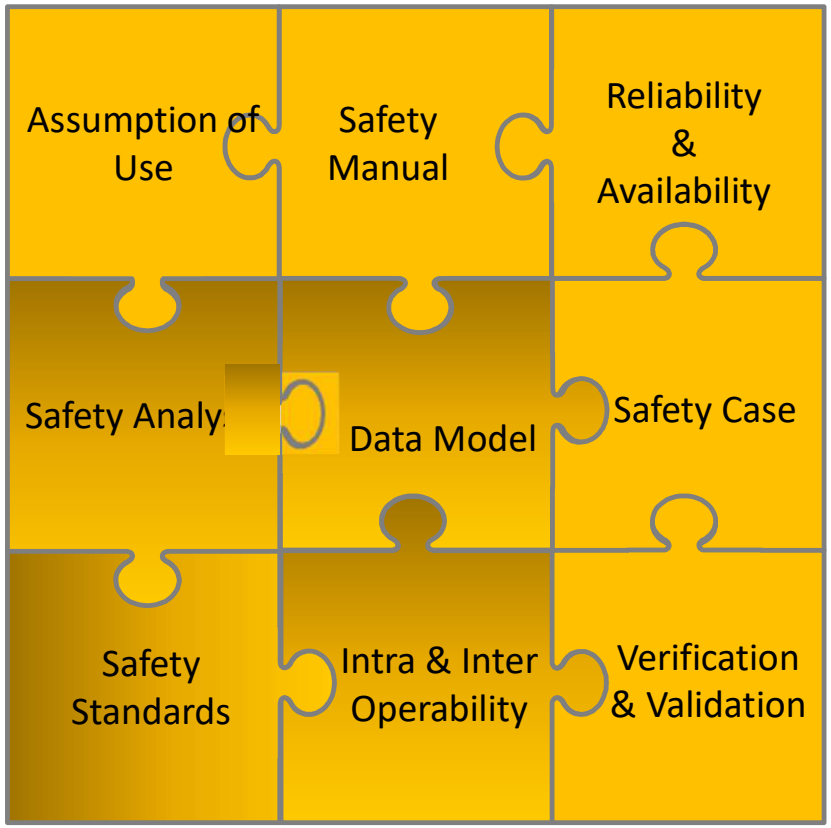
MUNICH, GERMANY
DECEMBER 6 - 7, 2022

Future Work

Ghani Kanawati, Technical Director of Functional Safety, ARM

accellera SYSTEMS INITIATIVE

Topics for Future Investigation



Future Topics

Safety Requirements and AoUs

Safety Verification

Work in Progress

Inadequate requirements & AoU: Exploring the root cause?

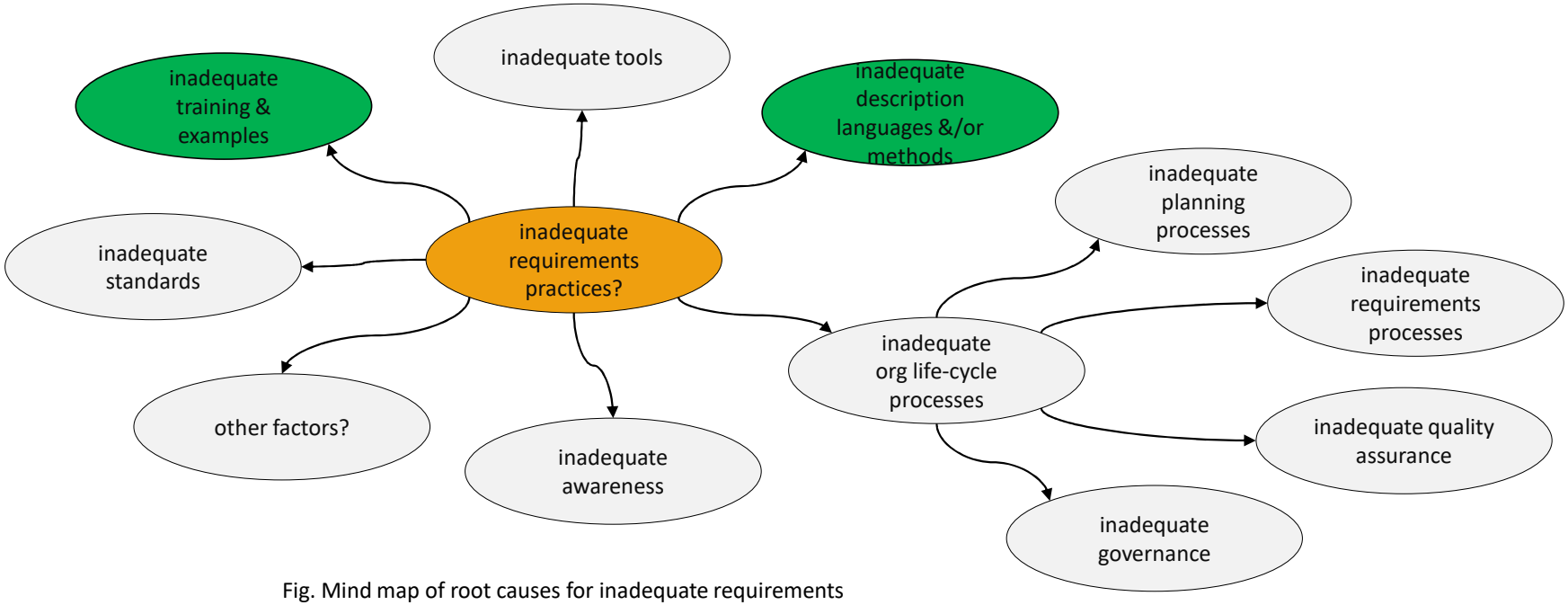


Fig. Mind map of root causes for inadequate requirements

- Mix of root causes. Mix may vary across orgs. Inadequacies in org life-cycle processes dominate
- Accellera needs to closely examine ROI of efforts here, i.e. we could spend a lot of time addressing one of the root causes but the needle does not shift in industry

What is the problem?

Problem1: Inadequate practices

- Completeness
- Unclear
- Inconsistencies
- Unambiguous
- Does not meet intended behavior?
- Dependencies
- Lack of appropriate processes followed (proper training planning, practices)
- Safety requirements are not evaluated in a complete and a systematic way

Problem 2: Derive complete accurate IC level requirements

- Challenges of interpreting and deriving the IC requirements
- What additional attributes are needed to address the interpretation/derivation from system requirements
- Missing attributes in the requirements to enable derivation

Problem 3: How do we know that the application/module Concept (Functional/Technical) map to IC level the requirements

Standardization of Requirements and AoU

What it is:

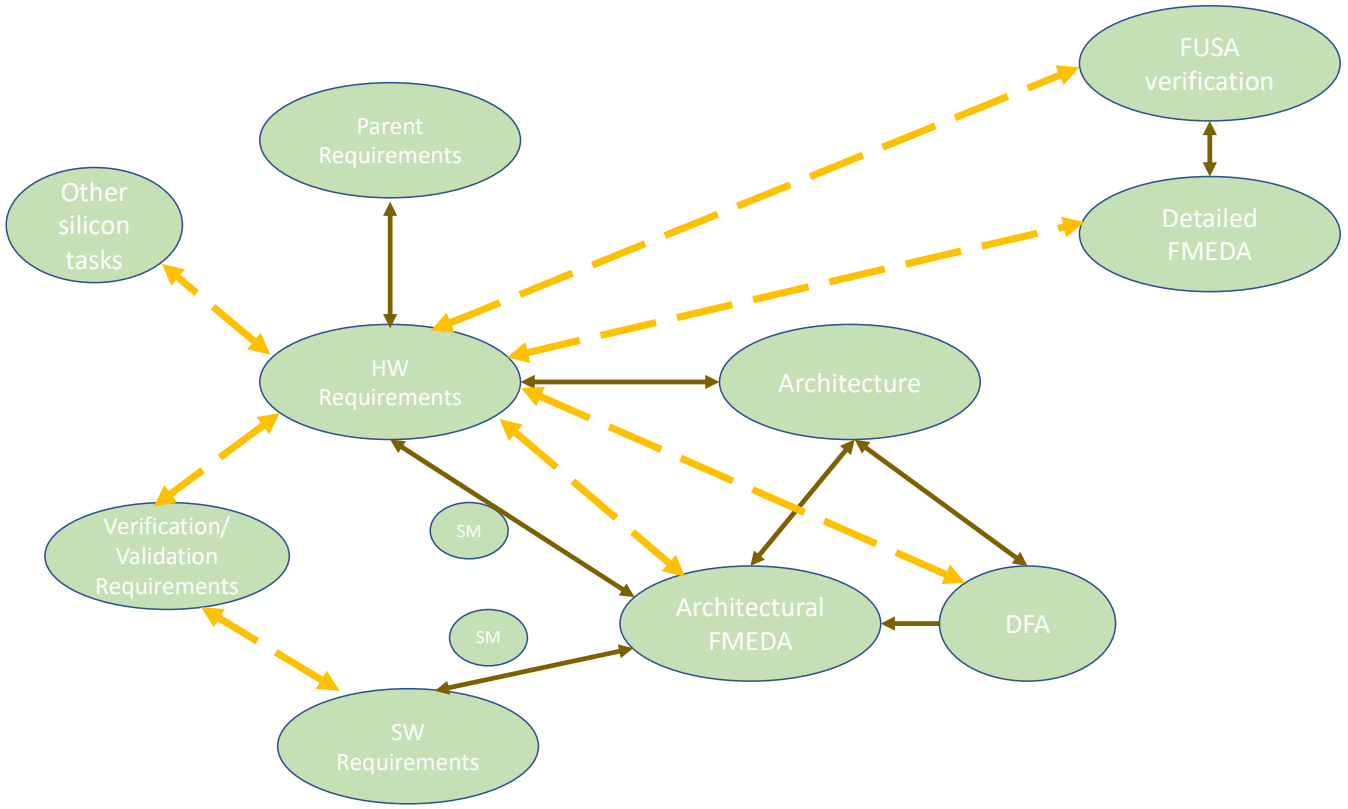
- Define criteria of a good “Requirement”
- Additional attribute from safety perspective
- Attributes (one or more) to enable traceability; ex: parent child relationship
- Identify constraints and assumptions to satisfy the requirement
- How requirements are linked to the Data model
- Item-to-IP
- Functional Req
- Technical Requirements
- HW and SW requirements (how we derive the HW and SW requirements)

What it is not:

- Standard for writing requirements; there are many out there and there is no intention yet to develop another
- Traceability – to enable impact and analysis

Scope: Define Requirements “Attributes” to support requirement types
Attributes to enable traceability of any requirements and its corresponding engineering activities

Requirement Interact With Other FUSA Work Products

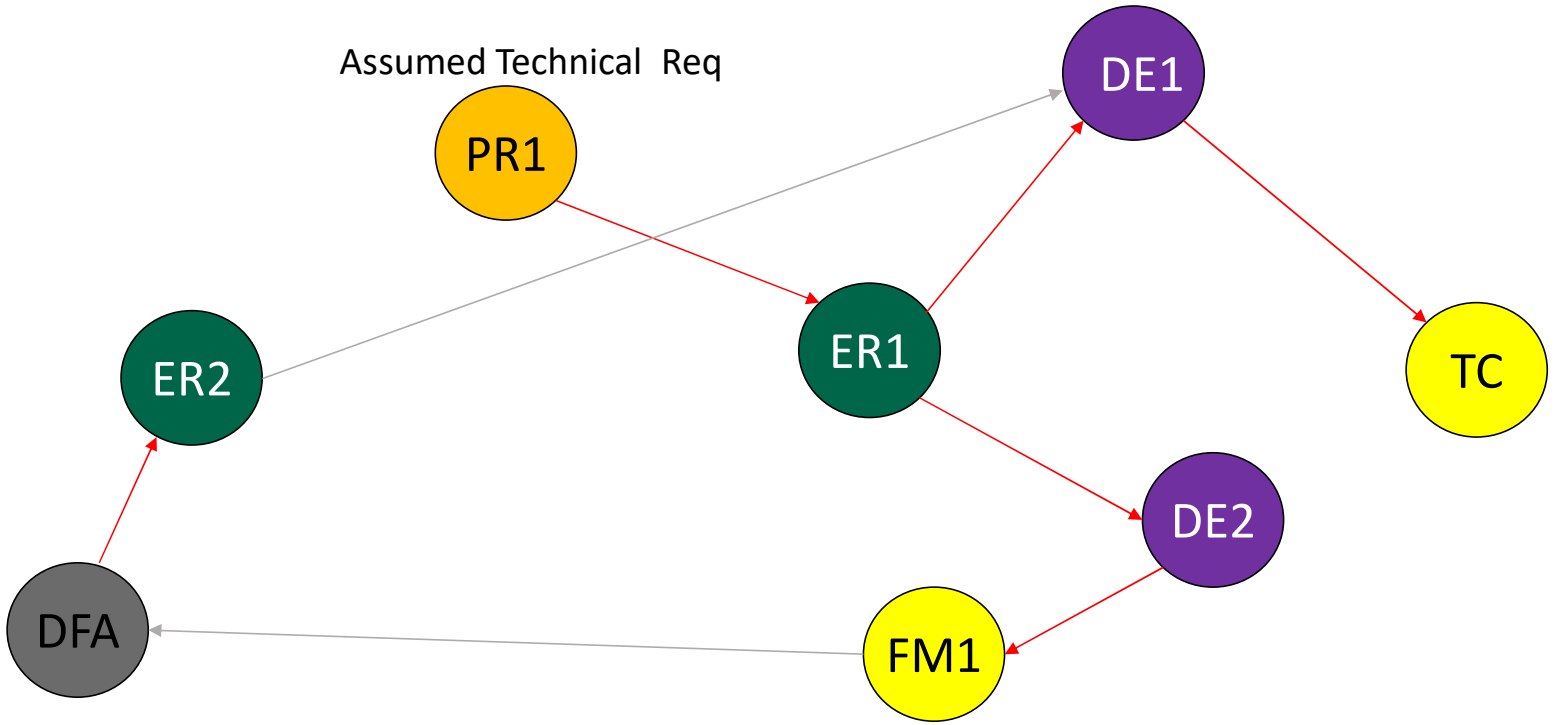


Example 1 Interconnect

Symbol	Requirement Type	Description
PR1	Product Requirement	Data Protection on Cache
ER1	Architecture Req	Dual lock step shared RAM
ER2	Architecture Req	Logical isolation of Dual lock step (primary, secondary)
DE1	uArch Requirement	Agent shared RAM
DE2	uArch Requirement	Temporal diversity
FM1	Architecture FMEA	Transaction failure
FM2	Architecture FMEA	Message failure
DFA	DFA	CCF

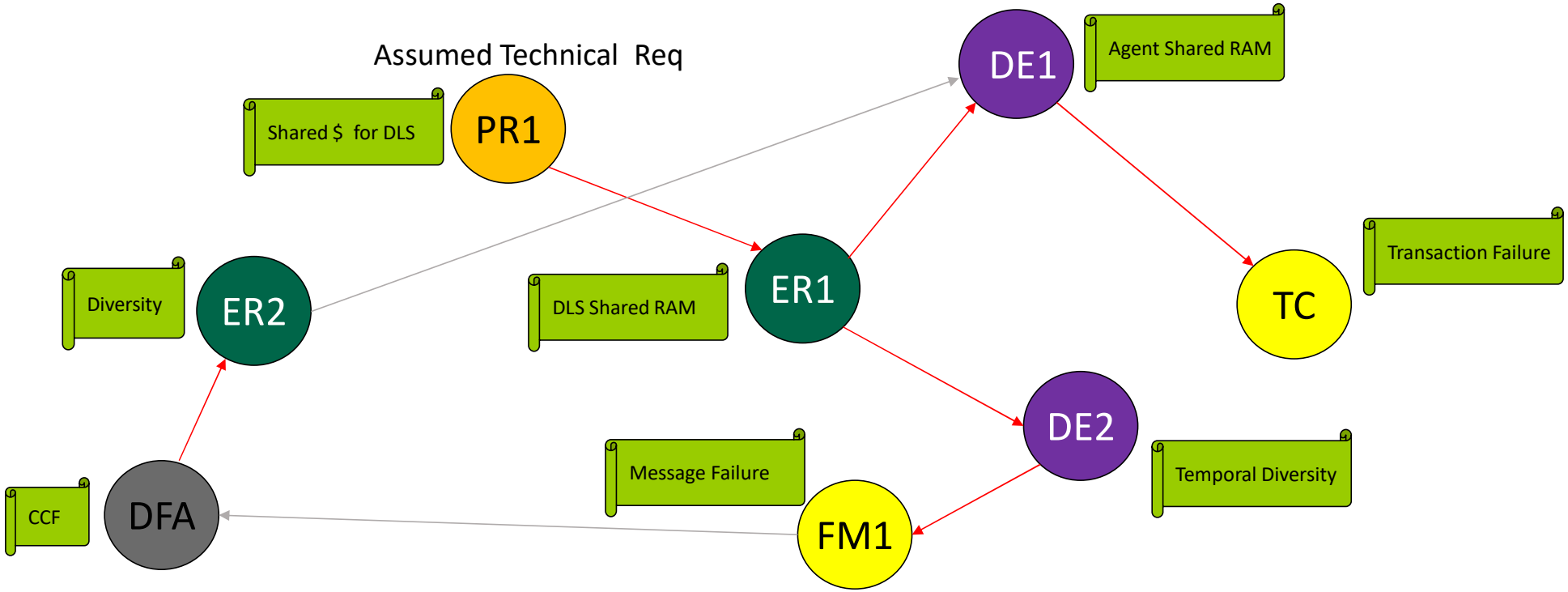
Example-1 Interconnect

PR: Product Req
ER: Engineering Req
DE: Design Req
TC: Test content Req



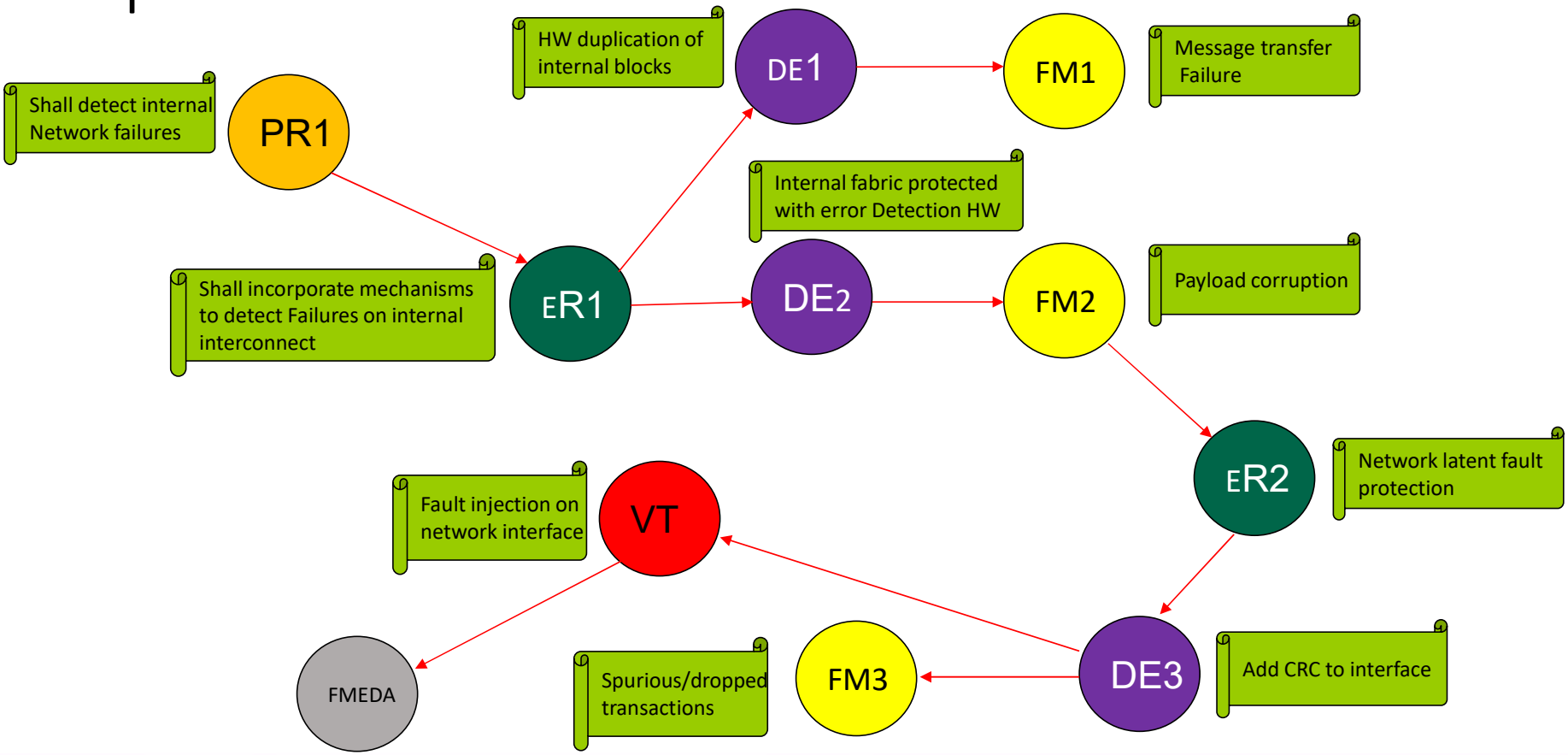
Example-1 Interconnect

PR: Product Req
ER: Engineering Req
DE: Design Req
TC: Test content Req

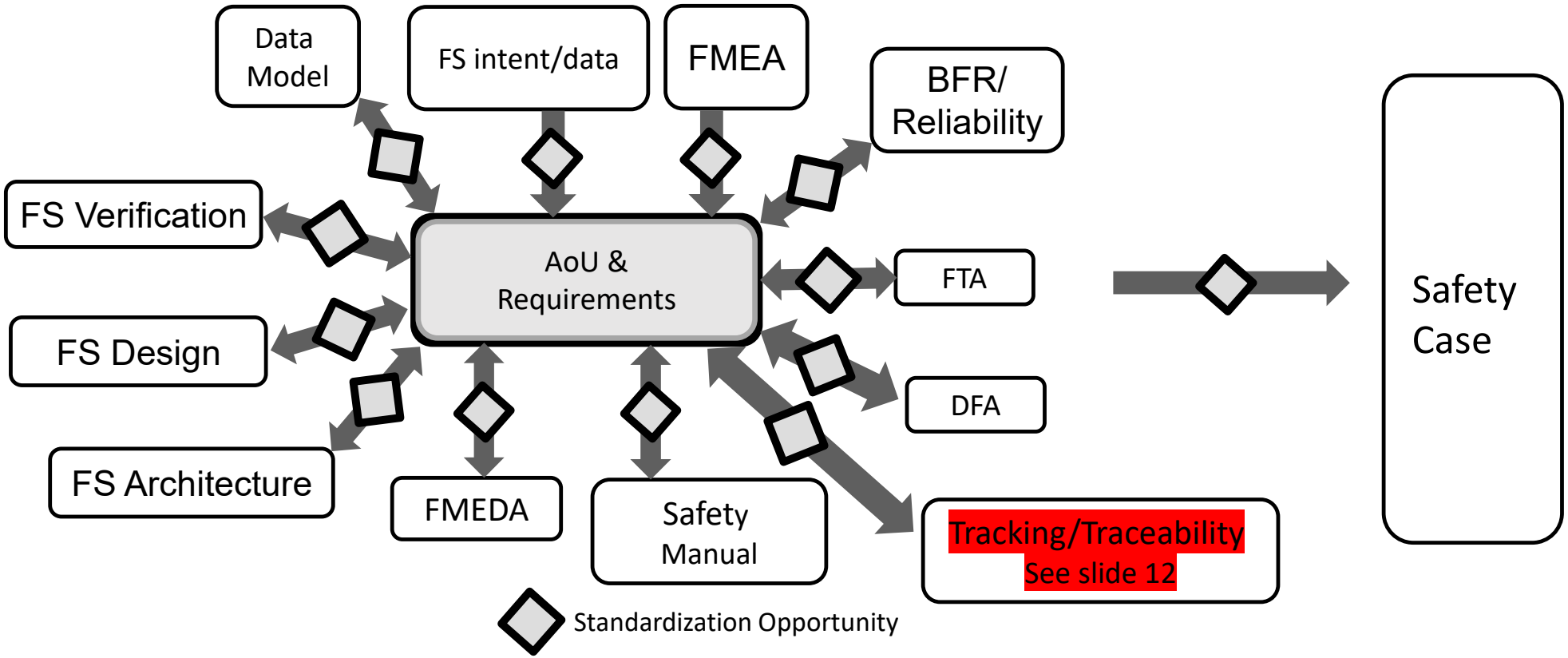


Example-2 Fabric

VT: Validation requirement



Capture the Intra-Layer Operations/Data/Workproducts



Tracking and Traceability

- All requirements have to be tracked, and will have link to verification, validation and design engineering related tasks
- Same standardization applies for those
- Without tracking and fulfilling all requirements we cannot release device to production

Attributes of Safety Requirements/AoUs

ISO 26262 requirements checklist
6.4.1 Appropriate combination of natural language, semi-formal, formal notation per ASIL?
6.4.2.1 Unambiguously identifiable
6.6.2.2 Inherited ASIL
6.4.2.3 Allocated to element
6.4.2.4 a) unambiguous
6.4.2.4 b) comprehensible
6.4.2.4 c) atomic (singular)
6.4.2.4 d) internally consistent
6.4.2.4 e) feasible and achievable
6.4.2.4 f) verifiable
6.4.2.4 g) necessary
6.4.2.4 h) implementation free Note: Potential conflict for TSR given they are defined as *"requirement derived for implementation of associated FSR"*
6.4.2.4 i) complete
6.4.2.4 j) conforming
6.4.2.5 a) unique unchanging id
6.4.2.5 b) status
6.4.2.5 c) ASIL

accellera SYSTEMS INITIATIVE

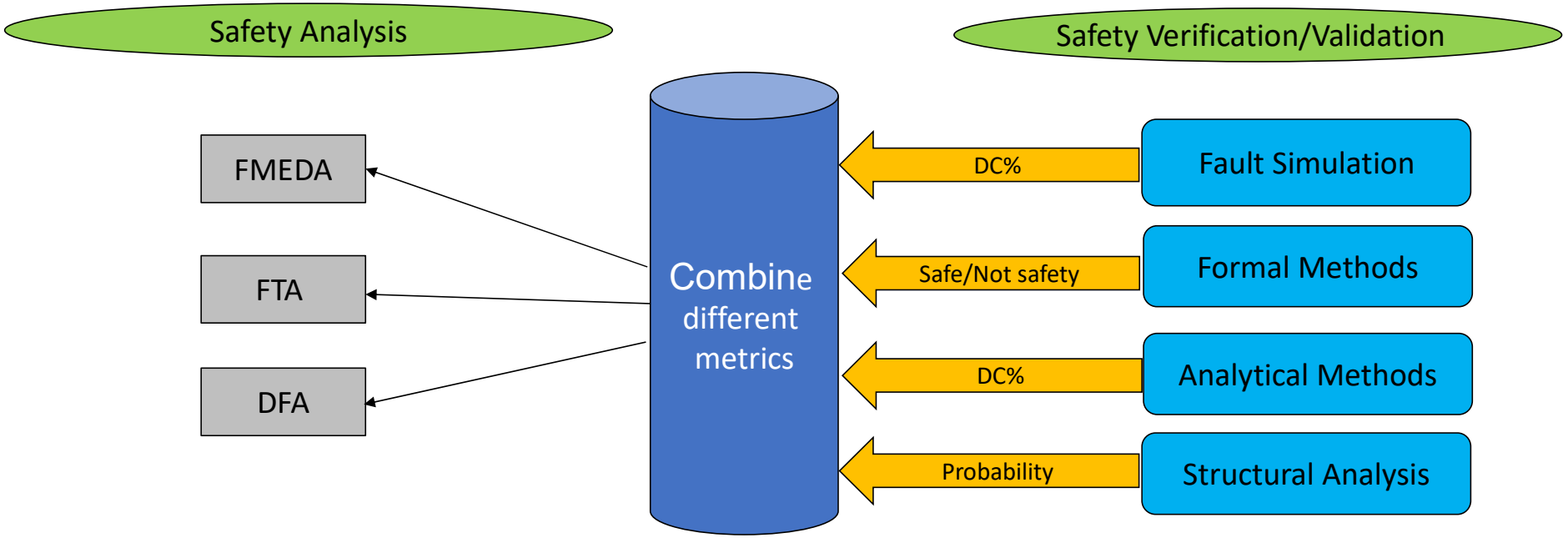
Recommended Attributes (WIP)

Attribute	More Information
Unique Identifier	
FTTI, FDTI, FRTI	Range of values (max, min)
Parent Requirement(FSR, TSR, SG)	
Version of the requirement	
Hierarchy Group(SF, FSR, TSR, System, HW , SW, Verification, etc)	If Hierarchy is applicable. Need to define the type for other safety standards
Module Identifier	
For HW: Type of HW requirements (Analog, Digital, memory, other technology)	
Assumed Diagnostic requirement (Safety features)	
Systematic/Random	
Safety related/non safety related	
Description	Describe the function
Additional information about the requirement	
Recommended verification tasks/link to tasks for traceability	What is expected ?
Type of requirement (FUNCTIONAL, NON-FUNCTIONAL)	Make sure that the word “Type” is in the context of what it is defined

Executive Summary

- Enough evidence that the Safety Requirements and AoU work group is needed
- Still work to be done to identify the interfaces with other FUSA work-products
- Examples shown earlier are only to demonstrate the intra-layer interdependencies which were not meant to list all the inter dependencies.
- The FS WG recommends to continue the effort by expanding the scope (identifying all the interfaces between Requirements and other work-products (FMEDA, DFA, Architecture FMEA, Safety Verification, FTA, Architecture and Design).
- More volunteers are welcomed to participate in the WG.

Safety Verification/Validation



- Identify how the Data Model should support the different verification methods
  - What are the changes that are needed in the data model to support these methods

Verification problems

- Verifying safety mechanisms and failure modes
  - Normal functional verification needs to inject a fault (a failure mode) to test a safety mechanism to hit the standard coverage metrics. Can we export this coverage for use elsewhere?
  - Have fault injection campaigns, can we record the results at an IP level to pass up to a system level?
  - May run statistical fault injection campaign at system level – if the statistical sample selects a fault already tested at an IP level, reuse that.

  - All of the above can leverage the FM and SM information in the database, but needs to extend this to identify the signals where the failure modes can be sampled, and where the failure can be observed. Also potentially need to record time of flight information for fault observation.

Analog/mixed simulation FMEDA

- Current proposal has focussed just on digital designs. But can do an FMEDA on analog and mixed-signal designs as well.
- Can have a fault which is observed in the digital part and detected in the analog, and vice versa.
- New IEEE P2427 standard for analog fault simulation which includes fault models and weighting schemes

- **Proposals[1]**
- Extend failure_mode definition to mark whether it is digital or analog
- If analog, have a fault model scheme which you can select from. Default to the models used in the IEEE P2427 annex (which includes user defined).
- If analog, take weighting schemes from IEEE P2427 annex, allow selection between them or user defined
- Extend safety_mechanism definition to mark whether it is digital or analog
- If analog, have an alternative set of analog enums for the type field

Questions ?

accellera
SYSTEMS INITIATIVE