

Ensuring Safety and Security of Next Generation Automotive Software

Stephen Di Camillo, stephen.dicamillo@ldra-usa.com

Technical Marketing and Business Development Manager

Technical Marketing and Business Development Manager, LDRA

As Technical Marketing and Business Development Manager at LDRA, he works with customers worldwide in industries including automotive, aerospace, railway, medical, communications, energy, and industrial equipment. He has over 30 years of experience in embedded systems and software development, with a focus on model-based engineering and standards compliance, helping engineering organizations adopt new processes, tools, and technologies to improve the efficiency and effectiveness of system and software development and testing while maintaining compliance with new and evolving technical standards. He holds a Bachelor of Science degree from Cornell University.



Stephen Di Camillo

<https://avionicsandtesting-innovations.com/speaker/steve-di-camillo/>

<https://ldra.wistia.com/medias/4xj5ibs6lz>

<https://ldra.wistia.com/medias/3cmh3vj9yl>

LDRA introduction

Automotive software trends and challenges

The LDRA solution

Summary



Provider of Software Quality, Compliance Management & Testing Solutions

Established 1975

ISO 9001 certified company

Certified for use in safety related software development according to IEC 61508, EN 50128, ISO 26262, IEC 62304 & IEC 60880

Active participants in standards e.g. DO-178C, MISRA C/C++, CERT & ISO 26262

MISRA C & Secure C Training Courses





Professor Mike Hennell

Member of SC-205 /
WG-71 (DO-178C) formal
methods subgroup

Member of MISRA C committee
and MISRA C++ committee

Member of the working group
drafting a proposed secureC
annex for the C language
definition
(SC 22 / WG14)



Bill St Clair

Member of SC-205 /
WG-71 (DO-178C) Object Oriented
Technology subgroup

Holds U.S. patents for a portable
storage system and LDRA's embedded
requirements verification system



Andrew Banks

MISRA Committee Chair
Committee Member for Second
Edition of ISO 26262

Chair of BSI IST/15/-/26
for Software Testing

MISRA representative to BSI
IST/15 for Software and
Systems Engineering

Member of BSI IST/15-/20 for Bodies
of Knowledge and Professionalization



Aerospace



Automotive



Defense



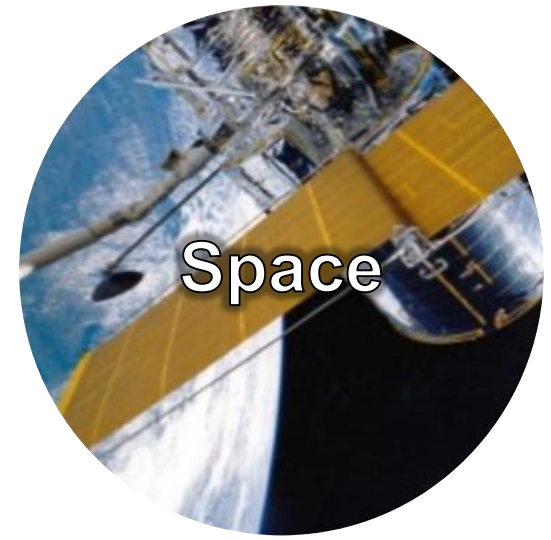
**Industrial
& Energy**



Medical



**Rail
Transportation**



Space

Automotive software trends and challenges

Innovation is causing market disruption

- Increasingly, a vehicle's DNA is its software
- Requires learning fast, deciding fast, delivering fast

Vehicles are becoming part of connected IoT solutions

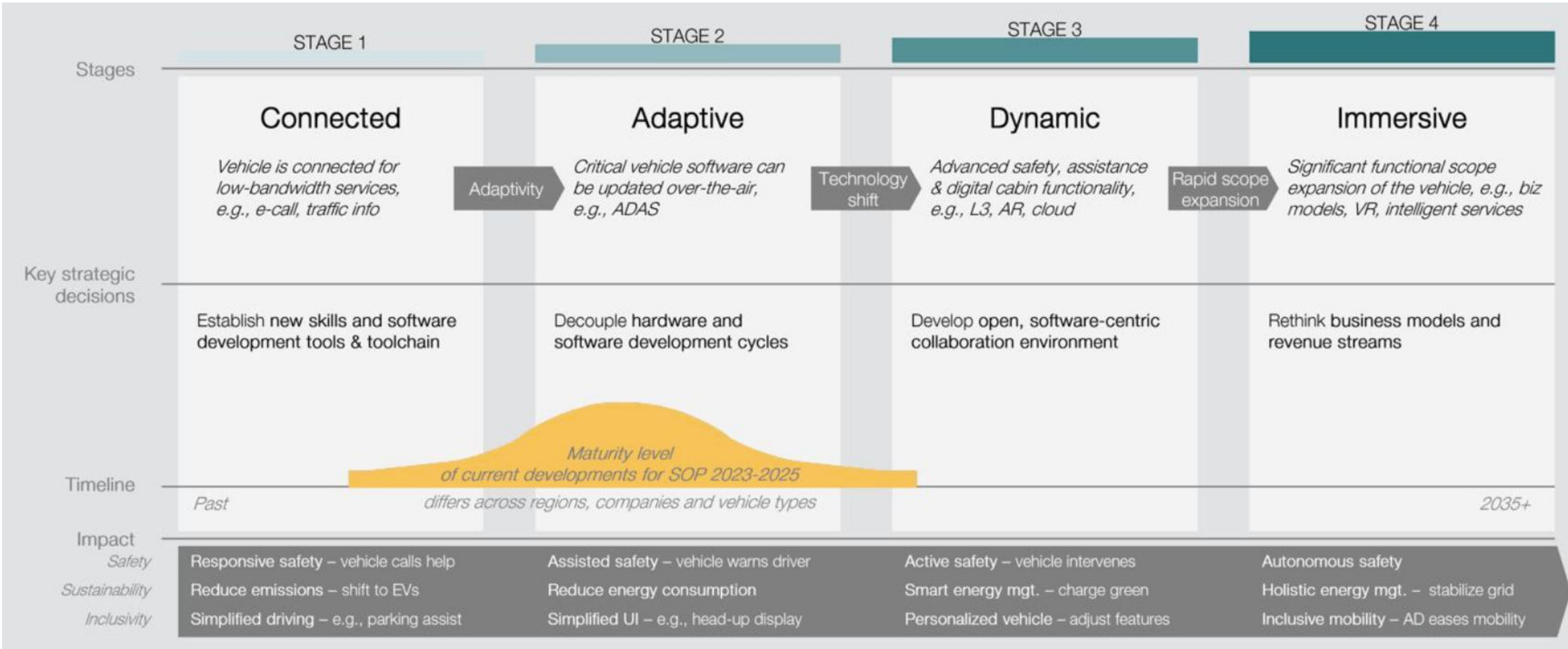
- More partners, more standards, more interfaces
- Inherently more complexity and failure modes

Vehicles are becoming autonomous

- More software, more technology, more 'intelligent'
- ADS and ADAS

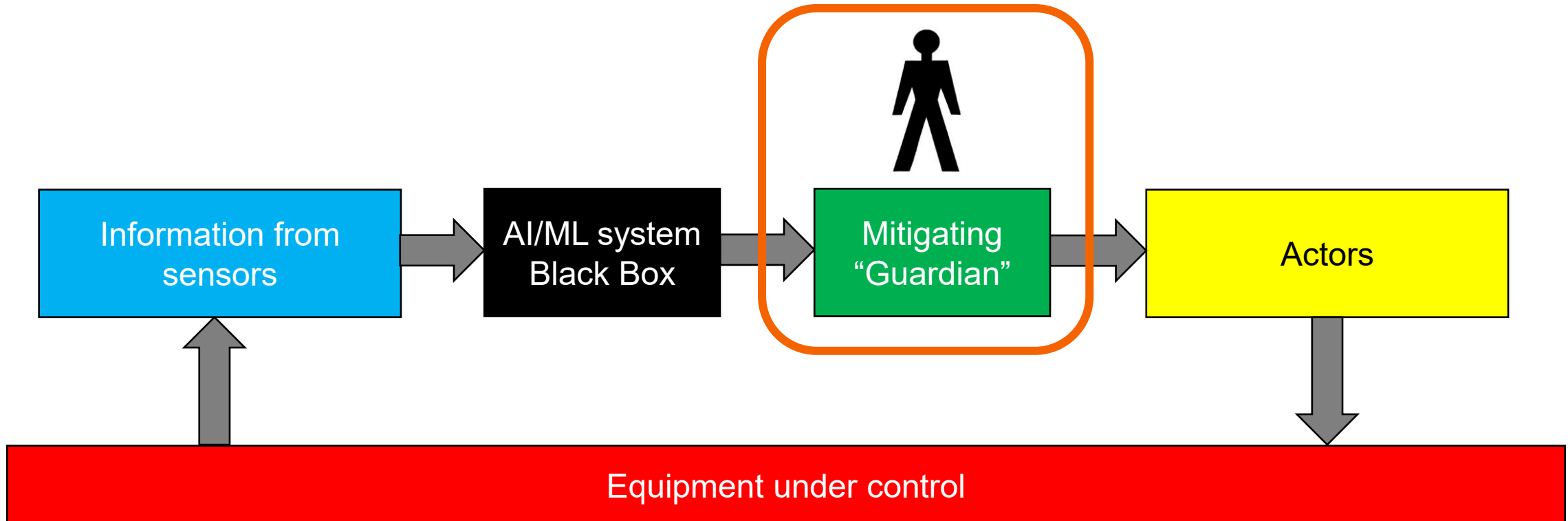


Software defined vehicle journey



Most approvals to date include some level of human intervention to provide mitigation

Human guardians can be replaced by conventional, deterministic applications developed in accordance with functional safety standards



Level	Name	DDT	Monitor Driving Environment	DDT Fallback Performance	ODD
0	No Automation	Driver	Driver	Driver	N/A
1	Driver Assistance	Driver and System	Driver	Driver	Limited
2	Partial Driving Automation	System	Driver	Driver	Limited
3	Conditional Driving Automation	System	System	Fallback Ready User becomes driver	Limited
2	High Driving Automation	System	System	System	Limited
5	Full Driving Automation	System	System	System	Unlimited

Level 3 and higher are significantly more complicated and include many more safety and security requirements

Driver Performs Part or All of the DDT

ADS (“System”) Performs the Entire DDT (While Engaged)

Functional safety is:

The absence of unreasonable risk due to hazards caused by malfunctioning behaviour of electrical and/or electronic system systems

In other words, a functionally safe system must work correctly and fail in a predictable (safe) way



Safety by [Nick Youngson](#) CC BY-SA 3.0 [Pix4free.org](#)

<https://www.fuji-setsu.co.jp/products/LDRA/Safety.html>

Cybersecurity is:

The condition in which assets are sufficiently protected against threat scenarios to items of road vehicles, their functions and their electrical or electronic components

In other words, a secure system must work correctly and ensure vulnerabilities are mitigated

Security threats can have safety implications



<https://www.fuji-setsu.co.jp/products/LDRA/CyberSecurity.html>

Compliance with safety and security standards for Next Generation Automotive Software is a difficult challenge!



Safe and secure systems and software design

HARA/TARA, safety/security analysis and traceability

Validation & verification management












Traceable change management

The LDRA solution



Meeting the Challenges of ISO 26262 & ISO 21434 Compliance



-  Objectives tracking
-  Requirements traceability
-  Coding standards compliance & vulnerability analysis
-  Data flow & control flow analysis
-  Data & control coupling analysis, and Taint analysis
-  Unit, Integration, Target testing
-  Object Code Verification (OCV)
-  Structural coverage, and Robustness & Fuzz testing
-  Timing analysis
-  Tool qualification
-  Compliance management & reporting

https://www.fuji-setsu.co.jp/products/LDRA/#LDRA_options



Assess impact of design changes
 Manage the implementation of a design change

3.4.1. Exit Sign

Requirement Updated

HLR_0210: Exit sign battery drain

The system shall provide data to summarize emergency battery drain imposed by each individual siren & exit sign, and by all such provision in the system as a whole. The battery voltage needs to be verified before exit sign.

Links: SYS_0080

New Requirement

HLR_0211: Update Exit sign battery drain

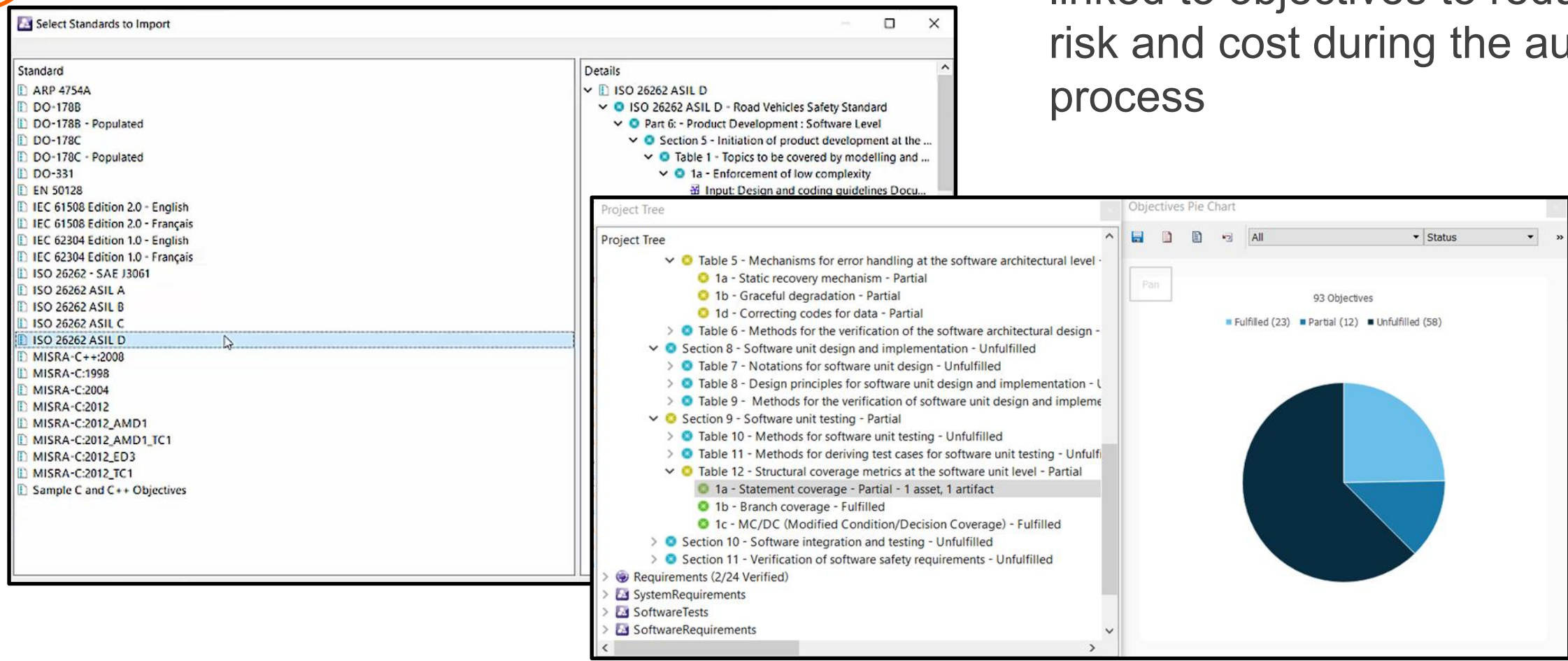
The exit sign should be updated with an indicator. The exit LED shall be made ON.

Links: SYS_0080

Number	Name	Body
HLR_0210	Exit sign battery drain	The system shall provide data to summarize emergency battery drain imposed by each individual siren & exit sign, and by all such provision in the system as a whole. The battery voltage needs to be verified before exit sign.
Upstream Impact	Number	Name Body
	SYS_0080	Sirens and Signs The Tunnel lighting system shall include facilities for the handling of emergency exit signs and sirens. These should include provision to allow the basic configuration of the system itself, in particular: 1. the number of lamps which can be positioned in a particular ceiling section, and 2. the current drain placed on emergency battery provision by each model of lamp, and by the system as a whole
Downstream Impact	Number	Name Body
	LLR_0380	System Data Query Get Lamp Minimum Lumens For each lamp type minimum lamp lumens shall be returned upon query
	LLR_0390	System Data Query Get Lamp Emergency Lumens Emergency lamp lumens shall be returned upon query
	LLR_0420	System Data Query Get Exit Sign Spacing Exit sign spacing shall be returned upon query
	LLR_0430	System Data Query Get Siren Spacing Siren spacing shall be returned upon query
	LLR_0530	Zone assign emergency cell output The zone class shall assign the emergency cell output levels to conserve emergency power supplies
	LLR_0360	System Data Query Get Lamp Power Required For each lamp type power required shall be returned upon query
LLR_0370	System Data Query Get Lamp Maximum Lumens For each lamp type maximum lamp lumens shall be returned upon query	
Modification	Content Changed	Source Mapping Changed Upstream/Downstream Changed Status Changed
	Yes	No No No

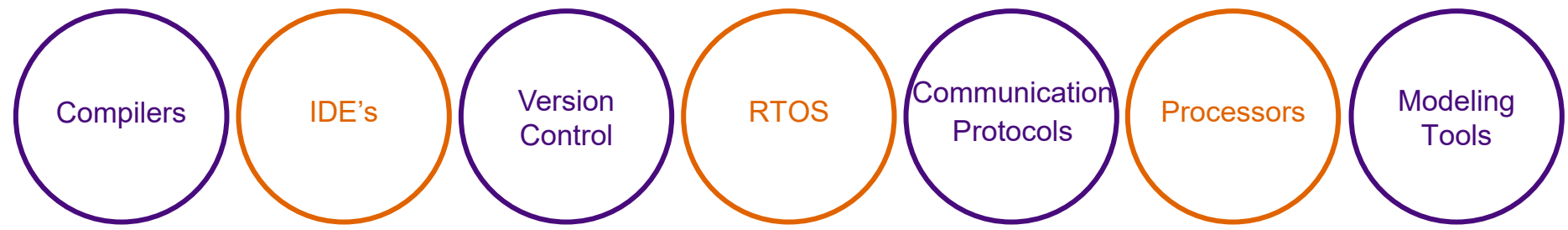
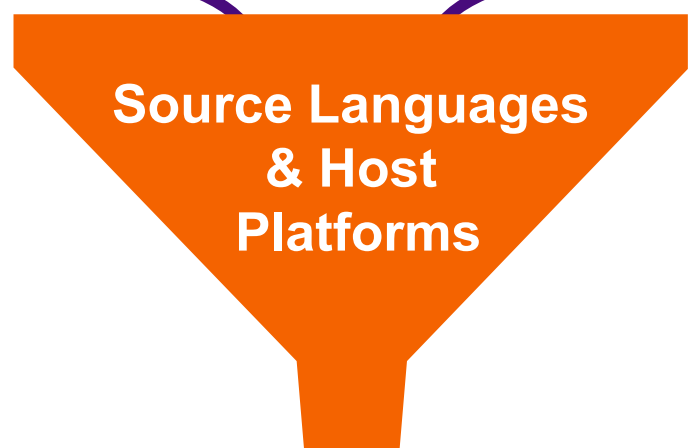


Artifacts and assets can be linked to objectives to reduce risk and cost during the audit process



The screenshot displays the LDRA software interface with three main components:

- Select Standards to Import:** A list of standards including ARP 4754A, DO-178B, DO-178C, EN 50128, IEC 61508, IEC 62304, ISO 26262, and MISRA-C. The 'ISO 26262 ASIL D' standard is highlighted.
- Details:** A tree view showing the selected standard's structure, including 'ISO 26262 ASIL D - Road Vehicles Safety Standard', 'Part 6: - Product Development : Software Level', 'Section 5 - Initiation of product development at the ...', 'Table 1 - Topics to be covered by modelling and ...', and '1a - Enforcement of low complexity'.
- Project Tree:** A detailed tree view of project objectives and artifacts. It shows various tables and sections, such as 'Table 5 - Mechanisms for error handling at the software architectural level - Partial', 'Section 8 - Software unit design and implementation - Unfulfilled', and 'Table 12 - Structural coverage metrics at the software unit level - Partial'. Specific items like '1a - Statement coverage - Partial - 1 asset, 1 artifact' are highlighted.
- Objectives Pie Chart:** A pie chart titled 'Objectives Pie Chart' showing the status of 93 objectives: 23 Fulfilled (light blue), 12 Partial (medium blue), and 58 Unfulfilled (dark blue).

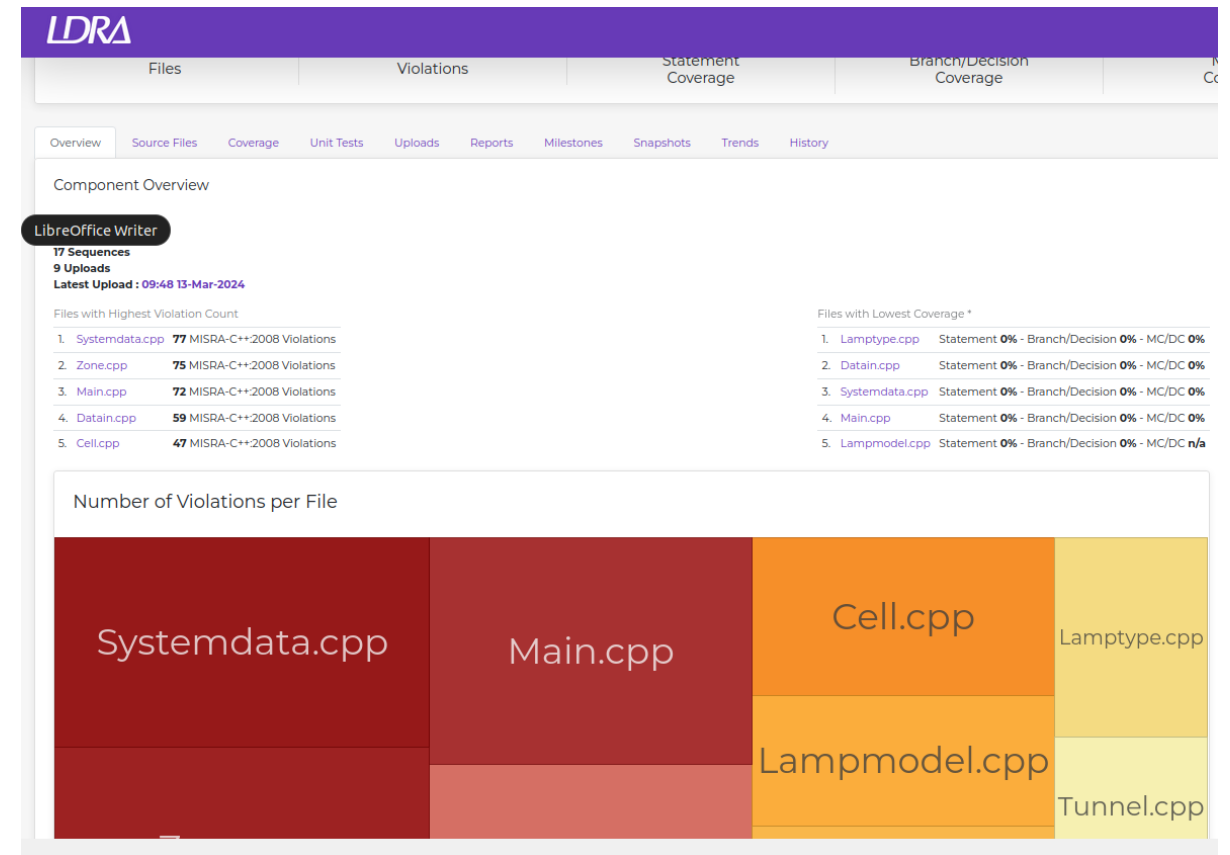




Static analysis plays a crucial role in ensuring the safety and security of software systems

Objectives of Static Analysis:

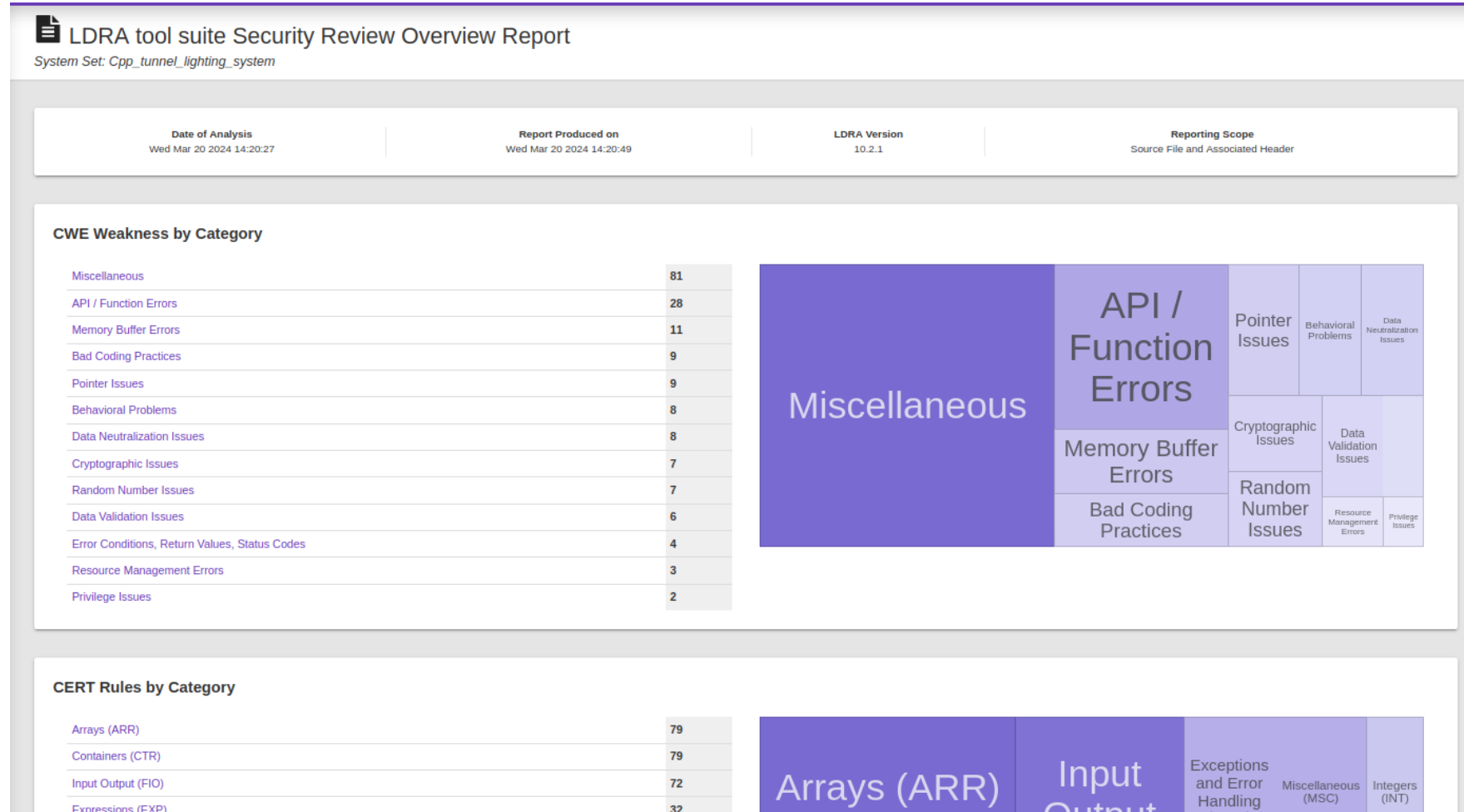
- Early Detection of Vulnerabilities
- Comprehensive Code Review
- Identification of Code Smells and Anti-Patterns
- Enforcement of Coding Standard
- Integration with CI/CD
- Scalability and Efficiency
- Regulatory Compliance



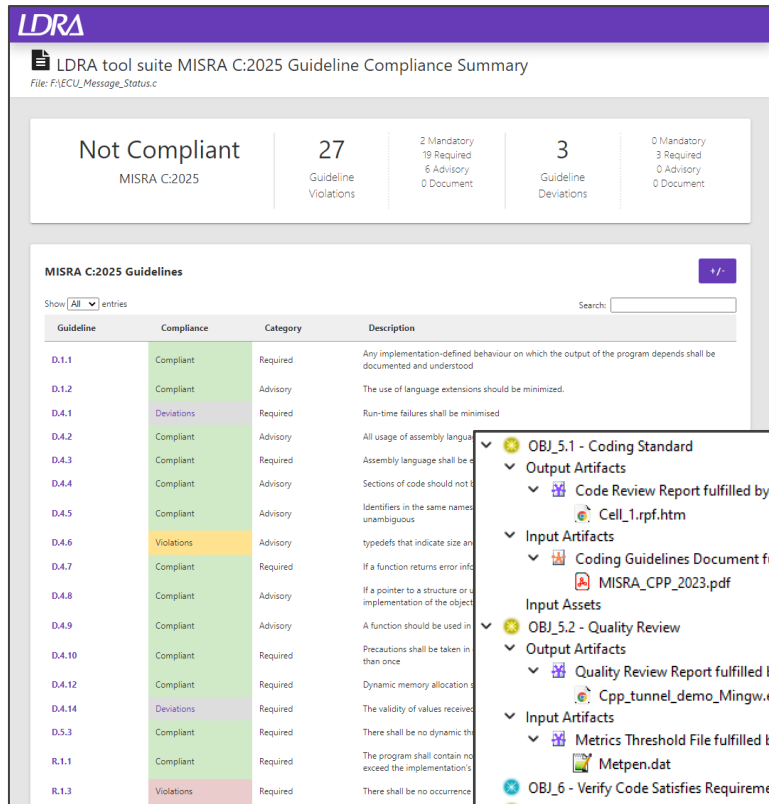
Static analysis identifies security vulnerabilities



Major Security Standards such as ISO/SAE 21434 recommend static analysis execution prior to any type of testing



Safety and security coding standards compliance

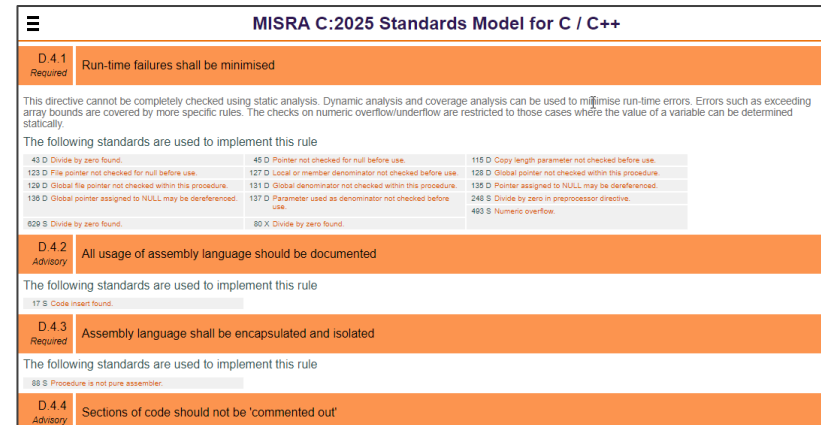


LDRA tool suite MISRA C:2025 Guideline Compliance Summary
File: F:\ECU_Message_Status.c

Not Compliant MISRA C:2025	27 Guideline Violations	2 Mandatory 19 Required 6 Advisory 0 Document	3 Guideline Deviations	0 Mandatory 3 Required 0 Advisory 0 Document
--------------------------------------	-----------------------------------	--------------------------------------------------------	----------------------------------	-------------------------------------------------------

MISRA C:2025 Guidelines

Guideline	Compliance	Category	Description
D.1.1	Compliant	Required	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood.
D.1.2	Compliant	Advisory	The use of language extensions should be minimized.
D.4.1	Deviations	Required	Run-time failures shall be minimised.
D.4.2	Compliant	Advisory	All usage of assembly language shall be documented.
D.4.3	Compliant	Required	Assembly language shall be encapsulated and isolated.
D.4.4	Compliant	Advisory	Sections of code should not be 'commented out'.
D.4.5	Compliant	Advisory	Identifiers in the same namespace shall be unambiguous.
D.4.6	Violations	Advisory	typedefs that indicate size and alignment shall be used consistently.
D.4.7	Compliant	Required	If a function returns error information, it shall be documented.
D.4.8	Compliant	Advisory	If a pointer to a structure or union is used, the implementation of the object shall be documented.
D.4.9	Compliant	Advisory	A function should be used in preference to a macro.
D.4.10	Compliant	Required	Precautions shall be taken to ensure that a resource is not used more than once.
D.4.12	Compliant	Required	Dynamic memory allocations shall be documented.
D.4.14	Deviations	Required	The validity of values received from external sources shall be documented.
D.5.3	Compliant	Required	There shall be no dynamic resizing of arrays.
R.1.1	Compliant	Required	The program shall contain no unimplemented features.
R.1.3	Violations	Required	There shall be no occurrence of undefined behaviour.



MISRA C:2025 Standards Model for C / C++

D.4.1 Required Run-time failures shall be minimised

This directive cannot be completely checked using static analysis. Dynamic analysis and coverage analysis can be used to minimise run-time errors. Errors such as exceeding array bounds are covered by more specific rules. The checks on numeric overflow/underflow are restricted to those cases where the value of a variable can be determined statically.

The following standards are used to implement this rule

43 D Divide by zero found.	45 D Pointer not checked for null before use.	115 D Copy length parameter not checked before use.
123 D File pointer not checked for null before use.	127 D Local or member denominator not checked before use.	128 D Global pointer not checked within this procedure.
129 D Global file pointer not checked within this procedure.	131 D Global denominator not checked within this procedure.	135 D Pointer assigned to NULL may be dereferenced.
136 D Global pointer assigned to NULL may be dereferenced.	137 D Parameter used as denominator not checked before use.	245 S Divide by zero in preprocessor directive.
829 S Divide by zero found.	80 X Divide by zero found.	493 S Numeric overflow.

D.4.2 Advisory All usage of assembly language should be documented

The following standards are used to implement this rule

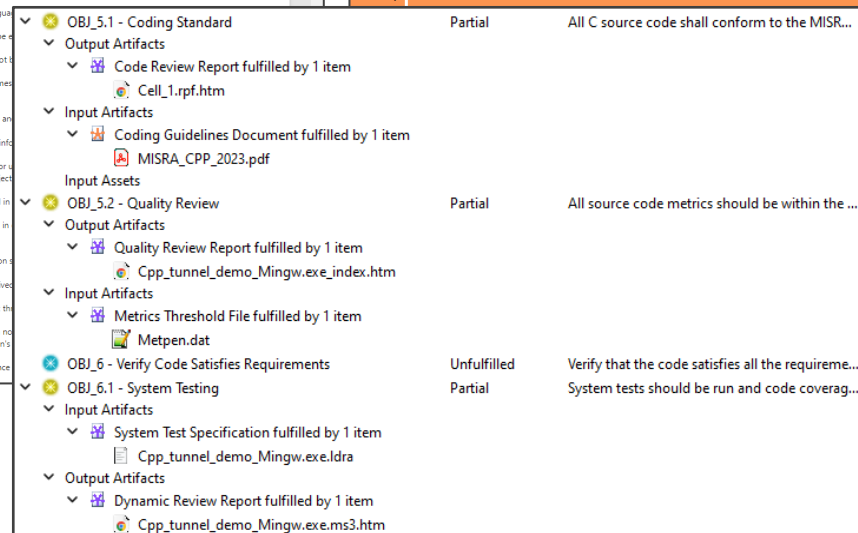
17 S Code insert found.

D.4.3 Required Assembly language shall be encapsulated and isolated

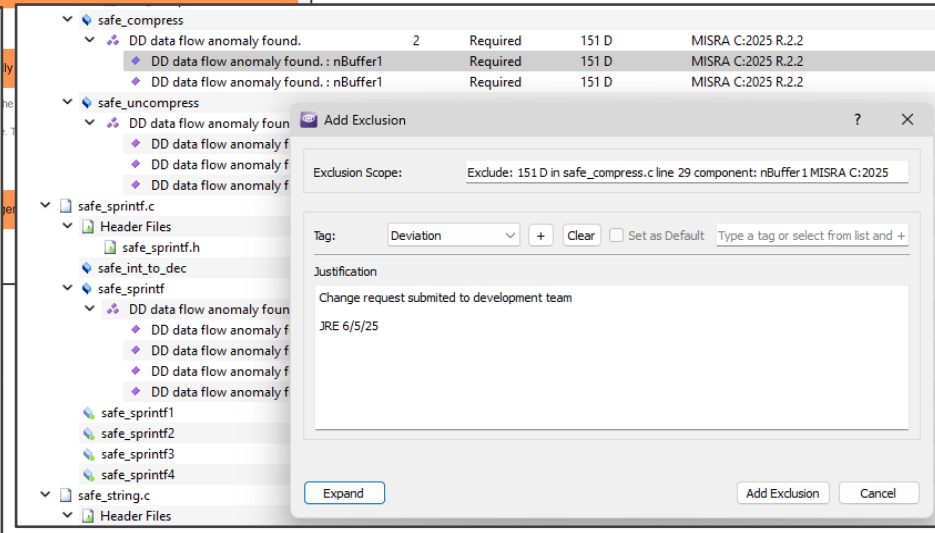
The following standards are used to implement this rule

88 S Procedure is not pure assembler.

D.4.4 Advisory Sections of code should not be 'commented out'



- OBJ_5.1 - Coding Standard (Partial) All C source code shall conform to the MISRA...
 - Output Artifacts
 - Code Review Report fulfilled by 1 item
 - Cell_1.rpf.htm
 - Input Artifacts
 - Coding Guidelines Document fulfilled by 1 item
 - MISRA_CPP_2023.pdf
 - Input Assets
 - OBJ_5.2 - Quality Review (Partial) All source code metrics should be within the ...
 - Output Artifacts
 - Quality Review Report fulfilled by 1 item
 - Cpp_tunnel_demo_Mingw.exe_index.htm
 - Input Artifacts
 - Metrics Threshold File fulfilled by 1 item
 - Metpen.dat
 - OBJ_6 - Verify Code Satisfies Requirements (Unfulfilled) Verify that the code satisfies all the requireme...
 - OBJ_6.1 - System Testing (Partial) System tests should be run and code coverag...
 - Input Artifacts
 - System Test Specification fulfilled by 1 item
 - Cpp_tunnel_demo_Mingw.exe.ldra
 - Output Artifacts
 - Dynamic Review Report fulfilled by 1 item
 - Cpp_tunnel_demo_Mingw.exe.ms3.htm



- safe_compress
 - DD data flow anomaly found. (2 Required 151 D MISRA C:2025 R.2.2)
 - DD data flow anomaly found.: nBuffer1 (Required 151 D MISRA C:2025 R.2.2)
 - DD data flow anomaly found.: nBuffer1 (Required 151 D MISRA C:2025 R.2.2)
- safe_uncompress
 - DD data flow anomaly found (Add Exclusion)
 - DD data flow anomaly f
 - DD data flow anomaly f
 - DD data flow anomaly f
- safe_sprintf.c
 - Header Files
 - safe_sprintf.h
 - safe_int_to_dec
 - safe_sprintf
 - DD data flow anomaly found
 - DD data flow anomaly f
 - DD data flow anomaly f
 - DD data flow anomaly f
 - safe_sprintf1
 - safe_sprintf2
 - safe_sprintf3
 - safe_sprintf4
- safe_string.c
- Header Files

Exclusion Scope: Exclude: 151 D in safe_compress.c line 29 component: nBuffer1 MISRA C:2025

Tag: Deviation + Clear Set as Default Type a tag or select from list and +

Justification: Change request submitted to development team
JRE 6/5/25

Expand Add Exclusion Cancel

Manage deviation, removal, justification and documentation of rules

Code complexity and quality metrics



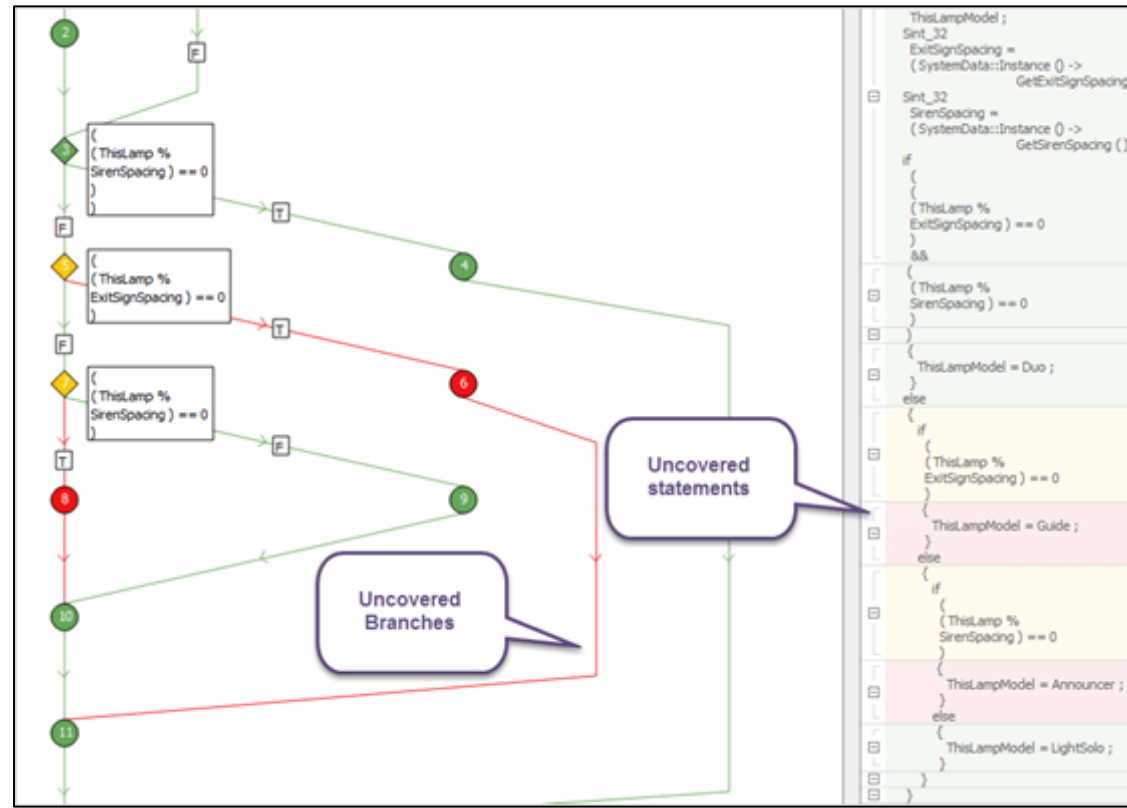
- Table A.128 - Limited size and complexity of Functions, Subroutines and Methods - Fulfilled - 1 asset, 1 artifact
 - Code review report fulfilled by 1 item
 - Mingw_c_cashregister.frm.htm
 - Design and coding guidelines document fulfilled by 1 item
 - Complexity_Guidelines from Acme_coding_standard.doc

Procedure Calls	Cyclomatic Complexity	Knots
GetOptimum	98 : (Fail)	114 : (Fail)
LzmaDec_DecodeReal	82 : (Fail)	76 : (Fail)
LzmaDec_TryDummy	61 : (Fail)	94 : (Fail)
GetOptimumFast	41 : (Fail)	38 : (Fail)
LzmaDec_DecodeToDic	30 : (Fail)	57 : (Fail)
LzmaEnc_CodeOneBlock	30 : (Fail)	32 : (Fail)
main2	23 : (Fail)	28 : (Fail)
LzmaEncProps_Normalize	20 : (Fail)	7 : (Fail)
Decode2	14 : (Fail)	15 : (Fail)
Hc4_MatchFinder_GetMatches	14 : (Fail)	9 : (Fail)
Bt4_MatchFinder_GetMatches	14 : (Fail)	9 : (Fail)
MatchFinder_Create	14 : (Fail)	8 : (Fail)
GetMatchesSpec1	13 : (Fail)	14 : (Fail)
LzmaEnc_Alloc	13 : (Fail)	8 : (Fail)
BtGetMatches	11 : (Fail)	11 : (Fail)
LzmaEnc_SetProps	11 : (Fail)	2
SkipMatchesSpec	10	13 : (Fail)
MatchFinder_ReadBlock	10	12 : (Fail)
Hc_GetMatchesSpec	10	12 : (Fail)
LzmaEnc_Init	9	32 : (Fail)

Zone.cpp

- Clarity 100%
 - Executable ref. Lines 147
 - Depth of Loop Nesting 1
 - Total LCSAJs 46
 - Unique Operands 105
 - Average Length of Basic Blocks 4.03%
 - Comments in Headers 53
- Maintainability 100%
 - Number of Procedures 7
 - Unreachable Branches 0
 - Unreachable Lines 0
 - Maximum LCSAJ Density 4
 - Unreachable LCSAJs 0
 - Total LCSAJs 46
 - Vocabulary 119
 - Cyclomatic Complexity 9
 - Knots 23
 - Essential Cyclomatic Complexity 1
 - Essential Knots 0
- Testability 100%
 - Fan Out 7
 - File Fan in 0
 - Number of Procedures 7
 - Procedure Exit Points 1
 - Number of Loops 5
 - Unreachable Branches 0
 - Unreachable Lines 0
 - Maximum LCSAJ Density 4
 - Unreachable LCSAJs 0
 - Total LCSAJs 46
 - Total Operands 286
 - Number of Basic Blocks 36
 - Executable reformatted Lines 145
 - Cyclomatic Complexity 9
 - Knots 23

Data & control flow visualization and analysis



Control flow graph

```

ThisLampModel ;
Sint_32
ExitSignSpacing =
(SystemData::Instance () ->
GetExitSignSpacing ())
Sint_32
SirenSpacing =
(SystemData::Instance () ->
GetSirenSpacing ())
if
{
ThisLamp %
ExitSignSpacing ) == 0
&&
{
ThisLamp %
SirenSpacing ) == 0
}
ThisLampModel = Duo ;
else
if
{
ThisLamp %
ExitSignSpacing ) == 0
}
ThisLampModel = Guide ;
else
if
{
ThisLamp %
SirenSpacing ) == 0
}
ThisLampModel = Announcer ;
else
ThisLampModel = LightSolo ;
}
    
```

```

uint32_t pulse=0U;
/* (M) DATAFLOW VIOLATION : 70 D : DU anomaly, variable value is not used. : pulse : 40F
/* See also line 17 DataFlow.c(DATAFLOW) */
uint32_t iter;

if ( cycles > 0U )
{
for ( iter=0U; iter<cycles; iter++ )
{
pulse++;
/* (M) DATAFLOW VIOLATION : 70 D : DU anomaly, variable value is not used. : pulse : 57T
/* See also line 17 DataFlow.c(DATAFLOW) */
}
}
/* (O) DATAFLOW VIOLATION : 7 D : DU data flow anomalies found. */
    
```

pulse is Defined but Unused

Variable Name	File	Procedure	Type Code	Attribute Code	Used on lines...
IDOffset	Cell.cpp	TunnelData:Cell:InitialiseCell	Global	Reference	55
LuminaireSetSize	Cell.cpp	TunnelData:Cell:InitialiseCell	Parameter	Declaration	32
ThisLamp	Cell.cpp	TunnelData:Cell:InitialiseCell	Local	Declaration	48
			Local	Reference	54 55 55 55 56
UniqueCellID	Cell.cpp	TunnelData:Cell:InitialiseCell	Parameter	Declaration	32
mLampTypeID	Cell.cpp	TunnelData:Cell:InitialiseCell	Member	Reference	55 78
			Member	Definition	54
pLampTypeIDs	Cell.cpp	TunnelData:Cell:InitialiseCell	Parameter	Declaration	32
			Parameter	Reference	54

Data flow reporting

Unit and integration testing on host/target



Table 9 - Design principles for software unit design and implementation - Unfulfilled

- 1a - One entry and one exit point in subprograms and functions - Fulfilled - 2 assets, 1 artifact
 - Code Review Report Artifact Placeholder fulfilled by 1 item
 - ISO26262_Project.frm.htm
 - System Design Specification fulfilled by 1 item
 - Software_Design_Document.doc
 - Design and coding guidelines Document fulfilled by 1 item
 - C_C++_Coding_Standards_and_Guidelines.pdf



Automated test case generation of Requirements-Based Tests and Robustness Tests

Test Case View			Variable I/O View				
Test Case	Regression P / F	Procedure	Value	Name	Type	Use	Regression Analysis
Tc 1	PASS	integer_to_ascii	I 123	value	u32	Input parameter applied through local	Assigned
Tc 2	PASS	integer_to_ascii	I 5	digits	u8	Input parameter applied through local	Assigned
Tc 3	PASS	integer_to_ascii	I 0	blanks	u8	Input parameter applied through local	Assigned
Tc 4	PASS	integer_to_ascii	O '0'	itoa_str[0]	s8	Output global	Compare + Write
Tc 5	PASS	integer_to_ascii	O '0'	itoa_str[1]	s8	Output global	Compare + Write
Tc 6	PASS	integer_to_ascii	O '1'	itoa_str[2]	s8	Output global	Compare + Write
Tc 7	PASS	integer_to_ascii	O '2'	itoa_str[3]	s8	Output global	Compare + Write
Tc 8	PASS	integer_to_ascii	O '3'	itoa_str[4]	s8	Output global	Compare + Write
Tc 9	PASS	integer_to_ascii	O '0'	itoa_str[5]	s8	Output global	Compare + Write
			O '0'	itoa_str[6]	s8	Output global	Compare + Write
			O 0	itoa_str[7]	s8	Output global	Compare + Write



Pre-configured files to use LDRA tools with a specific compiler and target out of the box

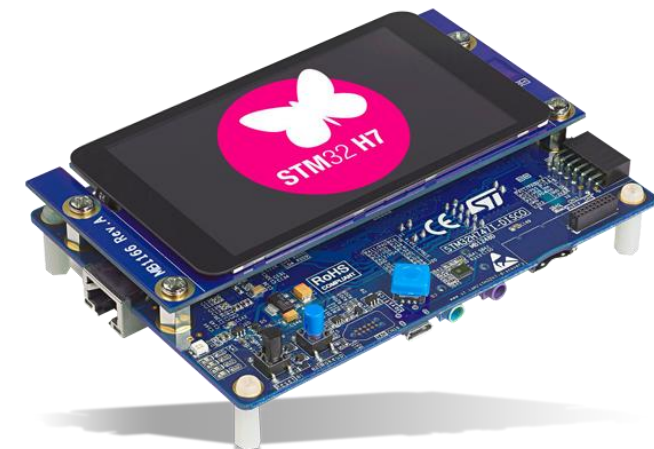
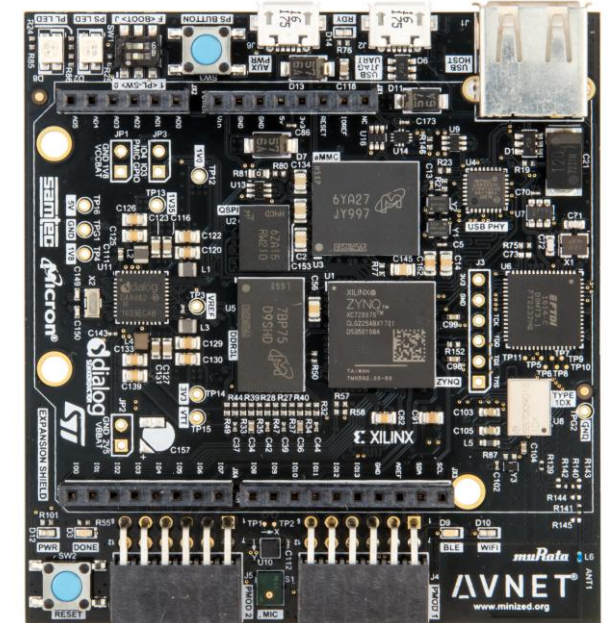
Consistent and professional look and feel

- Fully automated
- Getting Started Tutorial
- Example projects
- TBmanager project

Hundreds of combinations of compiler/target available

- New configurations can be quickly and easily created

<https://ldra.com/products/target-licence-packages-tlp/#TLP2>



Measuring structural coverage



- ✖ Table B.2 7a - Structural test coverage (entry points) 100%
 - Output Artifacts
 - Code coverage report fulfilled by 1 item
 - Cpp_tunnel_lighting_system.frm.htm
- ✖ Table B.2 7b - Structural test coverage (statements) 100%
 - Output Artifacts
 - Code coverage report fulfilled by 1 item
 - Cpp_tunnel_lighting_system.frm.htm
- ✖ Table B.2 7c - Structural test coverage (branches) 100%
 - Output Artifacts
 - Code coverage report fulfilled by 1 item
 - Cpp_tunnel_lighting_system.frm.htm
- ✖ Table B.2 7d - Structural test coverage (conditions, MC/DC)
 - Output Artifacts
 - Code coverage report fulfilled by 1 item
 - Cpp_tunnel_lighting_system.frm.htm

- Statement Coverage - 100%
 - Branch Decision Coverage - 100%
 - LCSAJ Coverage - Current
 - Branch Condition Coverage - 100%
 - Modified Condition/Decision Coverage - 67%
- ⊗ User Globals
- ⊗ sldvdemo_cruise_control2
- ⊗ sldvdemo_cruise_control2
- ⊗ sldvdemo_cruise_control2

LDRA Coverage Pass/Fail flowgraph of procedure : taskPingRun Statement: 86% Branch/Decision: 75% MC/DC: n/a

```

void
taskPingRun ( void )
{
  int32_t
  status;

  taskPingInit ();
  print ( "\n" );
  while
  (
    1
  )
  {
    print ( "\n\n" );
    taskDelay ( 5 );
    taskPongSignal ();
    status = semTake ( semPing, 5 ); /* Wait maximum of 5 ticks */
    if
    (
      status == ERROR
    )
    {
      print ( "\nTask Ping has been waiting too long" );
    }
  }
}
  
```

Source Code for Decision : (A | B) & C

Conditions in Decision : (A | B) & C

A	B	C
a		
b		
c		

Full Truth Table for Decision : (A | B) & C

Index	Conditions	Expected Outcome	Executed	MC/DC Independent Pairs Wave
1	F F F	= F	YES	
2	T F F	= F	NO	#C.
3	F T F	= F	NO	#].C.
4	T T F	= F	NO].].C.
5	F F T	= F	YES	#* .A..B..].].].
6	T F T	= T	YES	#* .A..].C..].].
7	F T T	= T	YES	#*B.....C..].
8	T T T	= T	YES	#*C..].].].

Legend: ■ Branch/Decision Covered ■ Branch/Decision not Covered

- Combined Coverage Run
 - Statement Coverage - 100%
 - Branch/Decision Coverage - 100%
 - Modified Condition / Decision Coverage - 67%
- Current Coverage Run
 - Statement Coverage - 100%

Manage coverage justifications and documentation

Manage coverage achieved through external coverage analysis

https://www.fuji-setsu.co.jp/products/LDRA/docs/178c_StructuralCoverageAnalysis.html



Data Coupling (Out from Procedure)							
Variable	Line	Type		Coupled To Parameter (Alias)	Procedure	Line	File
LampTypeID mThisLampTypeID	41	Member	-->	const LampTypeID ThisLampTypeID	TunnelData::SystemData::GetLampPowerRequired	92	Systemdata.cpp
Float_64 LumensRequired	41	Parameter	-->	const Float_64 LumensRequired	TunnelData::SystemData::GetLampPowerRequired	92	Systemdata.cpp

Data Coupling (In to Procedure)							
Variable	Procedure	Line	File	Type		Coupled To Parameter (Alias)	Line
Float_64 LumensDemand	TunnelData::Cell::SetPoweredOutputLevel	159	Cell.cpp	Array Index of Local	-->	Float_64 LumensRequired	33

Dynamic Data Flow Coverage for Data Coupling Variables						
Variable Name	File	Procedure	Type Code	Attribute Code	Used on lines...	
LumensRequired	Lamp.cpp	TunnelData::Lamp::SetLumensOutput	Parameter	Declaration	33	
			Parameter	Reference	36	39 ***** 41
mThisLampTypeID	Lamp.cpp	TunnelData::Lamp::SetLumensOutput	Member	Reference	41	

data access not exercised



US 20230048792A1



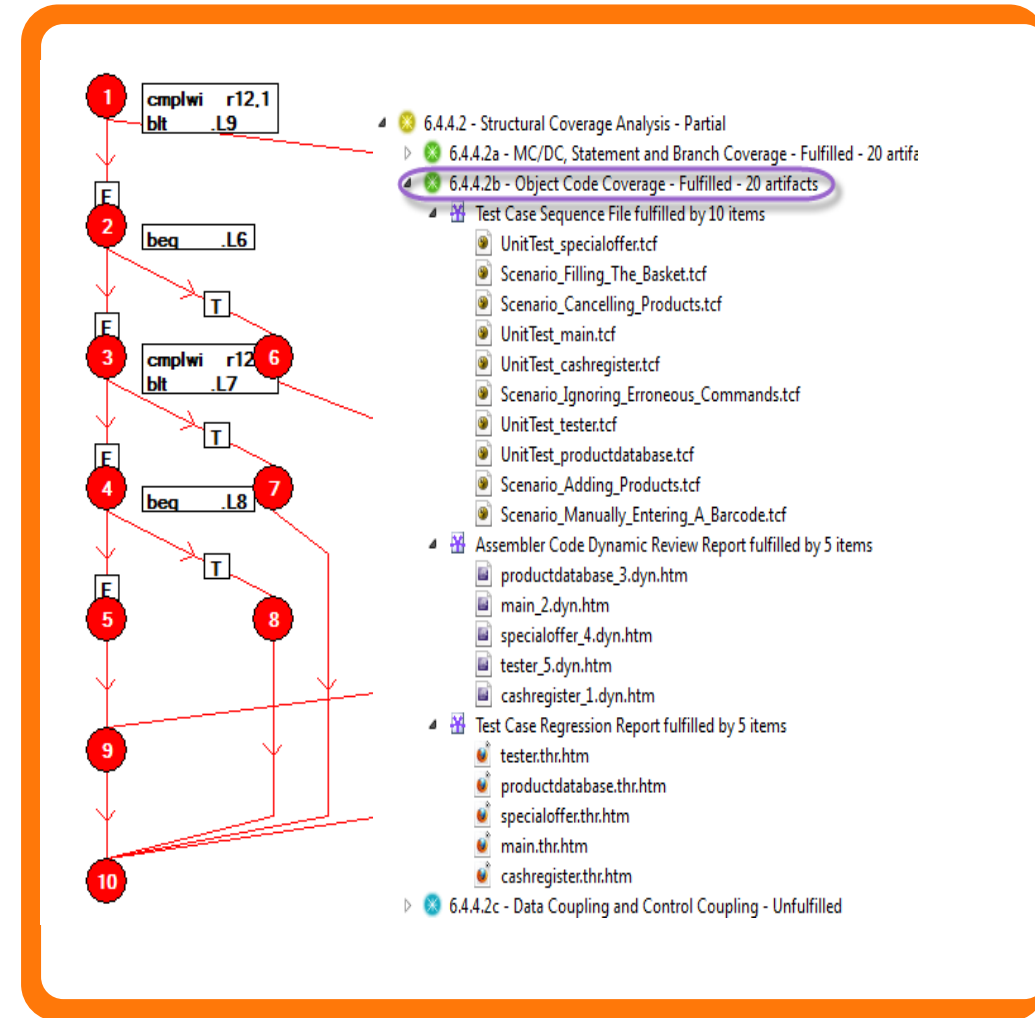
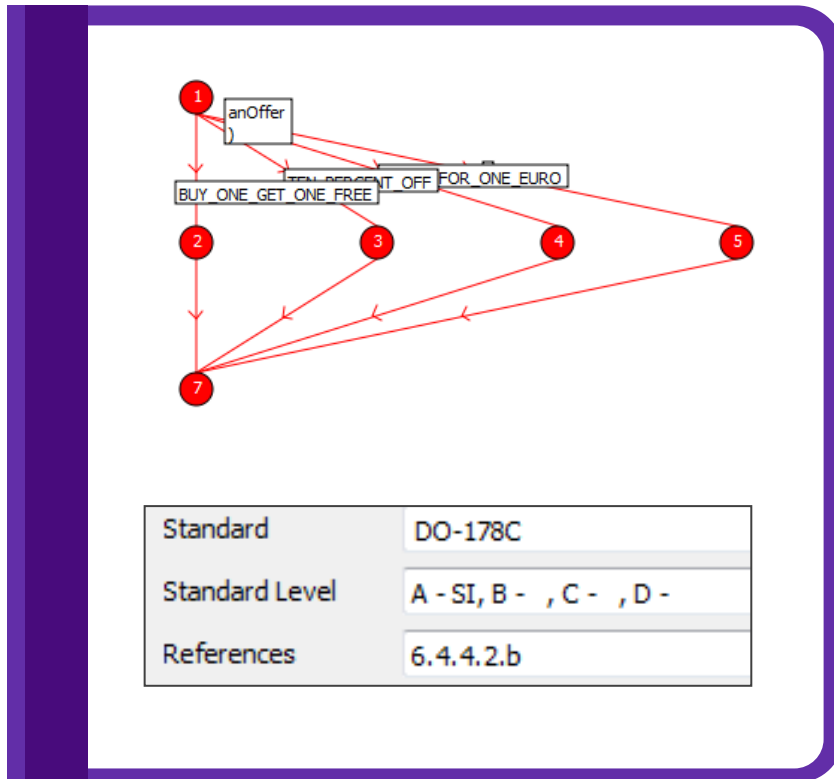
(10) International Publication Number
WO 2022/256103 A1



US011474927B1

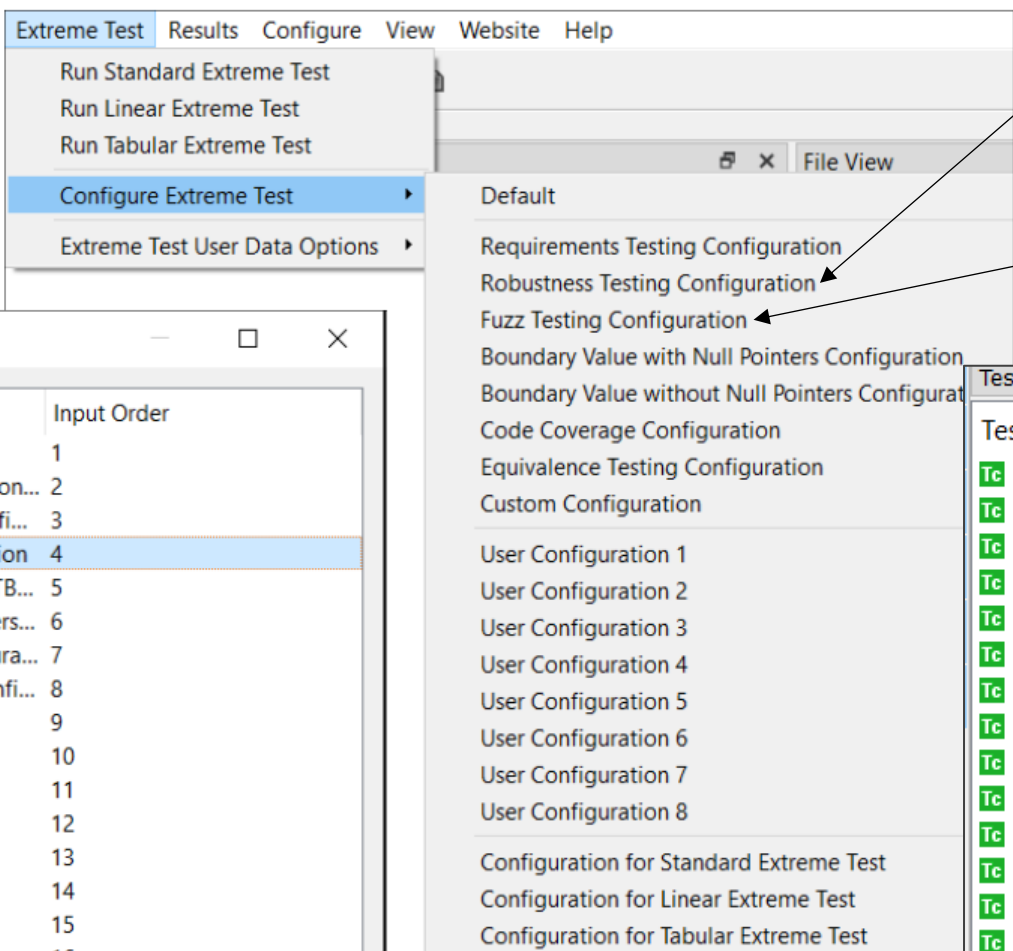
<https://www.fuji-setsu.co.jp/products/LDRA/resources.html#DCCC>

Object code verification



<https://www.fuji-setsu.co.jp/products/LDRA/resources.html#OCV>

Automatic test case generation/robustness & fuzz testing



Extreme Test Results Configure View Website Help

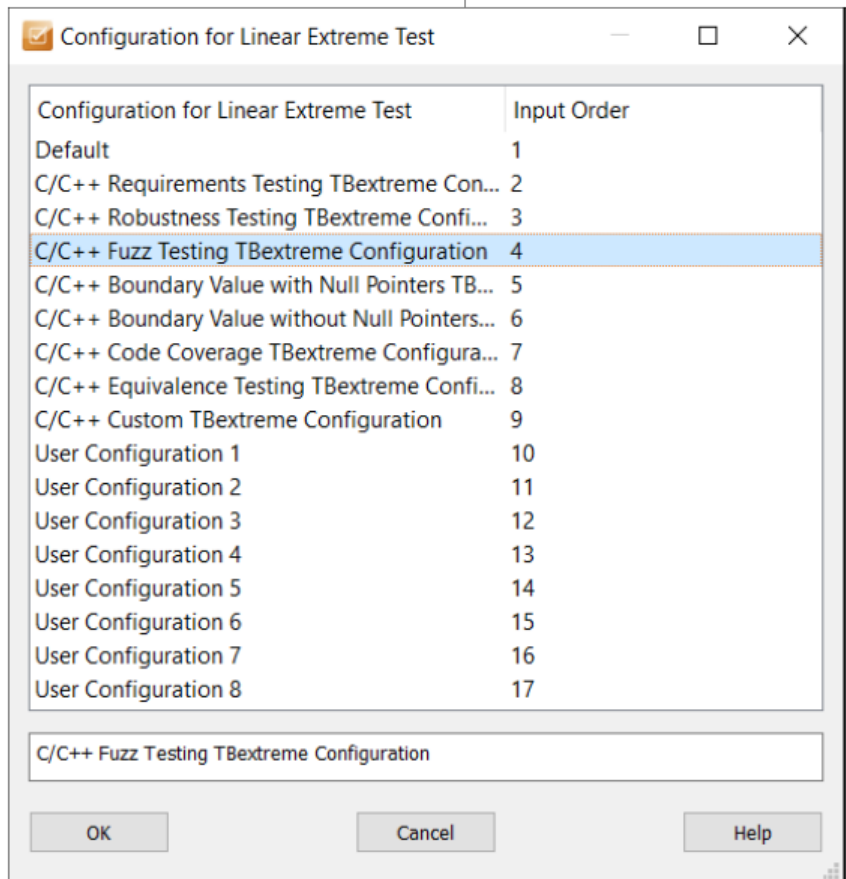
- Run Standard Extreme Test
- Run Linear Extreme Test
- Run Tabular Extreme Test
- Configure Extreme Test**
- Extreme Test User Data Options

File View

- Default
- Requirements Testing Configuration
- Robustness Testing Configuration
- Fuzz Testing Configuration**
- Boundary Value with Null Pointers Configuration
- Boundary Value without Null Pointers Configuration
- Code Coverage Configuration
- Equivalence Testing Configuration
- Custom Configuration
- User Configuration 1
- User Configuration 2
- User Configuration 3
- User Configuration 4
- User Configuration 5
- User Configuration 6
- User Configuration 7
- User Configuration 8
- Configuration for Standard Extreme Test
- Configuration for Linear Extreme Test
- Configuration for Tabular Extreme Test

Automates robustness testing to verify system resilience against unexpected inputs and boundary values.

Performs fuzz testing by injecting malformed data to expose vulnerabilities and bugs.

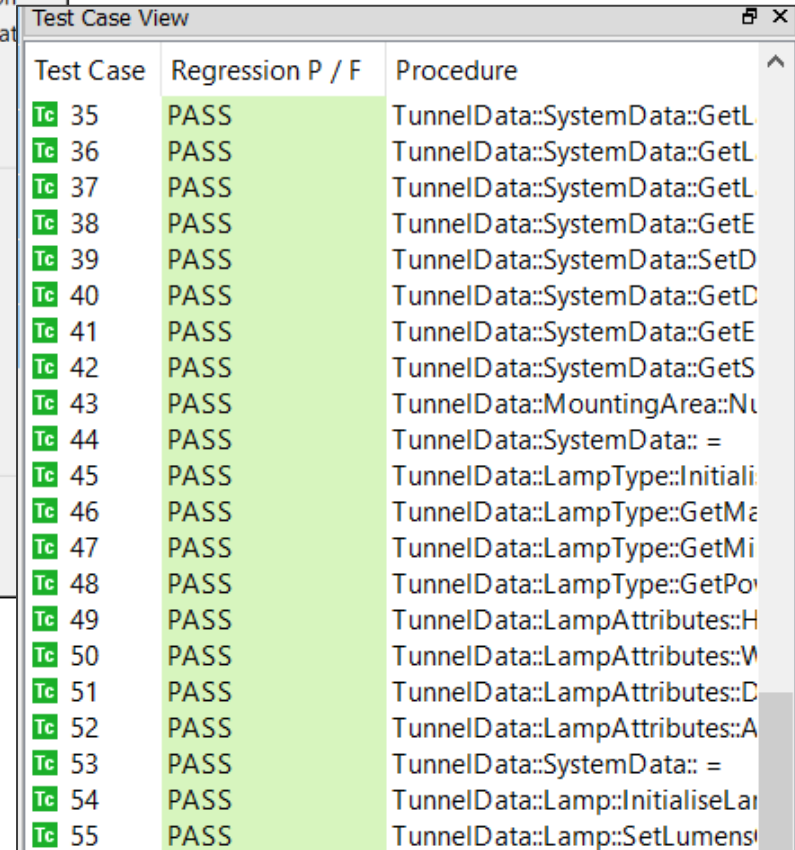


Configuration for Linear Extreme Test

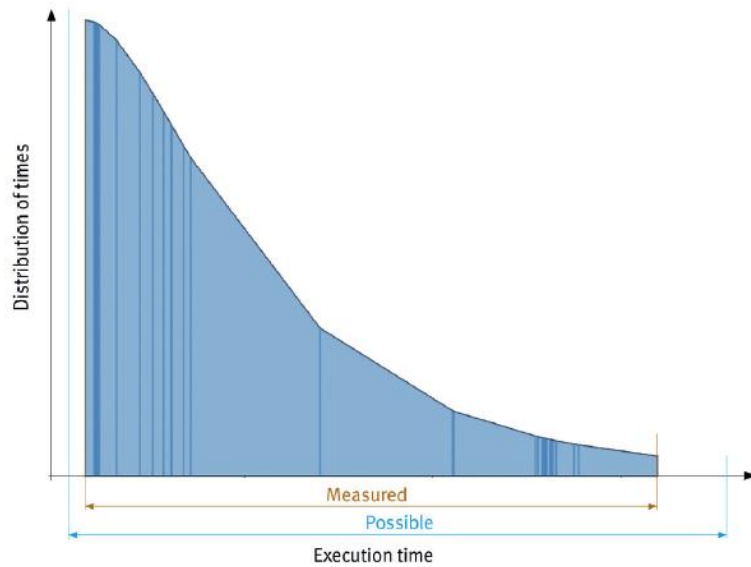
Configuration for Linear Extreme Test	Input Order
Default	1
C/C++ Requirements Testing TBextreme Con...	2
C/C++ Robustness Testing TBextreme Confi...	3
C/C++ Fuzz Testing TBextreme Configuration	4
C/C++ Boundary Value with Null Pointers TB...	5
C/C++ Boundary Value without Null Pointers...	6
C/C++ Code Coverage TBextreme Configura...	7
C/C++ Equivalence Testing TBextreme Confi...	8
C/C++ Custom TBextreme Configuration	9
User Configuration 1	10
User Configuration 2	11
User Configuration 3	12
User Configuration 4	13
User Configuration 5	14
User Configuration 6	15
User Configuration 7	16
User Configuration 8	17

C/C++ Fuzz Testing TBextreme Configuration

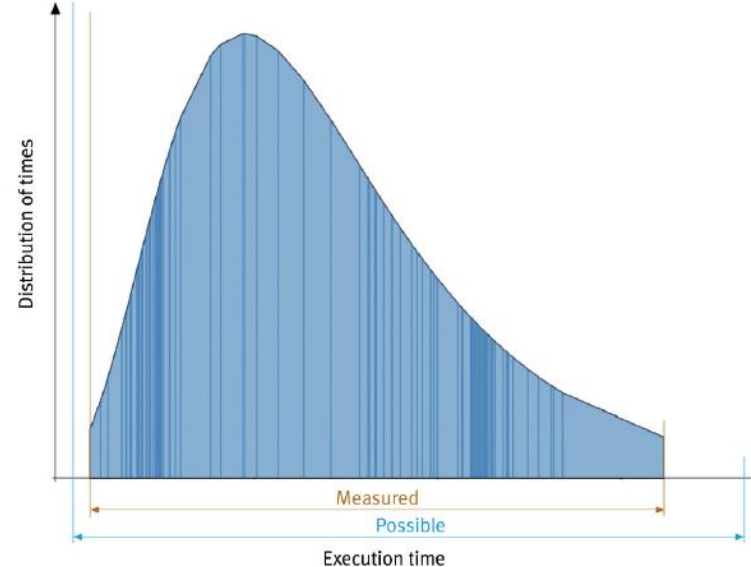
OK Cancel Help



Test Case	Regression P / F	Procedure
Tc 35	PASS	TunnelData::SystemData::GetL
Tc 36	PASS	TunnelData::SystemData::GetL
Tc 37	PASS	TunnelData::SystemData::GetL
Tc 38	PASS	TunnelData::SystemData::GetE
Tc 39	PASS	TunnelData::SystemData::SetD
Tc 40	PASS	TunnelData::SystemData::GetD
Tc 41	PASS	TunnelData::SystemData::GetE
Tc 42	PASS	TunnelData::SystemData::GetS
Tc 43	PASS	TunnelData::MountingArea::Nu
Tc 44	PASS	TunnelData::SystemData:: =
Tc 45	PASS	TunnelData::LampType::Initiali
Tc 46	PASS	TunnelData::LampType::GetMa
Tc 47	PASS	TunnelData::LampType::GetMi
Tc 48	PASS	TunnelData::LampType::GetPo
Tc 49	PASS	TunnelData::LampAttributes::H
Tc 50	PASS	TunnelData::LampAttributes::V
Tc 51	PASS	TunnelData::LampAttributes::D
Tc 52	PASS	TunnelData::LampAttributes::A
Tc 53	PASS	TunnelData::SystemData:: =
Tc 54	PASS	TunnelData::Lamp::InitialiseLar
Tc 55	PASS	TunnelData::Lamp::SetLumens



Timing Summary	
Number of Tests	100
Best-Case Execution Time	57321953.100000
Worst-Case Execution Time	338863903.400000
Minimal Observed Execution Time	63691059 - Test Case 1 (Rep 1)
Maximal Observed Execution Time	308058094 - Test Case 1 (Rep 2)
Mean Observed Execution Time	115596348



Timing Summary	
Number of Tests	100
Best-Case Execution Time	59658020.100000
Worst-Case Execution Time	495044356.400000
Minimal Observed Execution Time	66286689 - Test Case 1 (Rep 63)
Maximal Observed Execution Time	450040324 - Test Case 1 (Rep 38)
Mean Observed Execution Time	228836793

Run many unit/Integration test cycles

Generate interference across shared resources

Visualize the distribution of execution times



Static analysis can be used to satisfy code review objectives for both safety and security in parallel

Dynamic analysis such as structural coverage, robustness testing, data coupling analysis, and WCET measurements can also satisfy objectives for safety and security

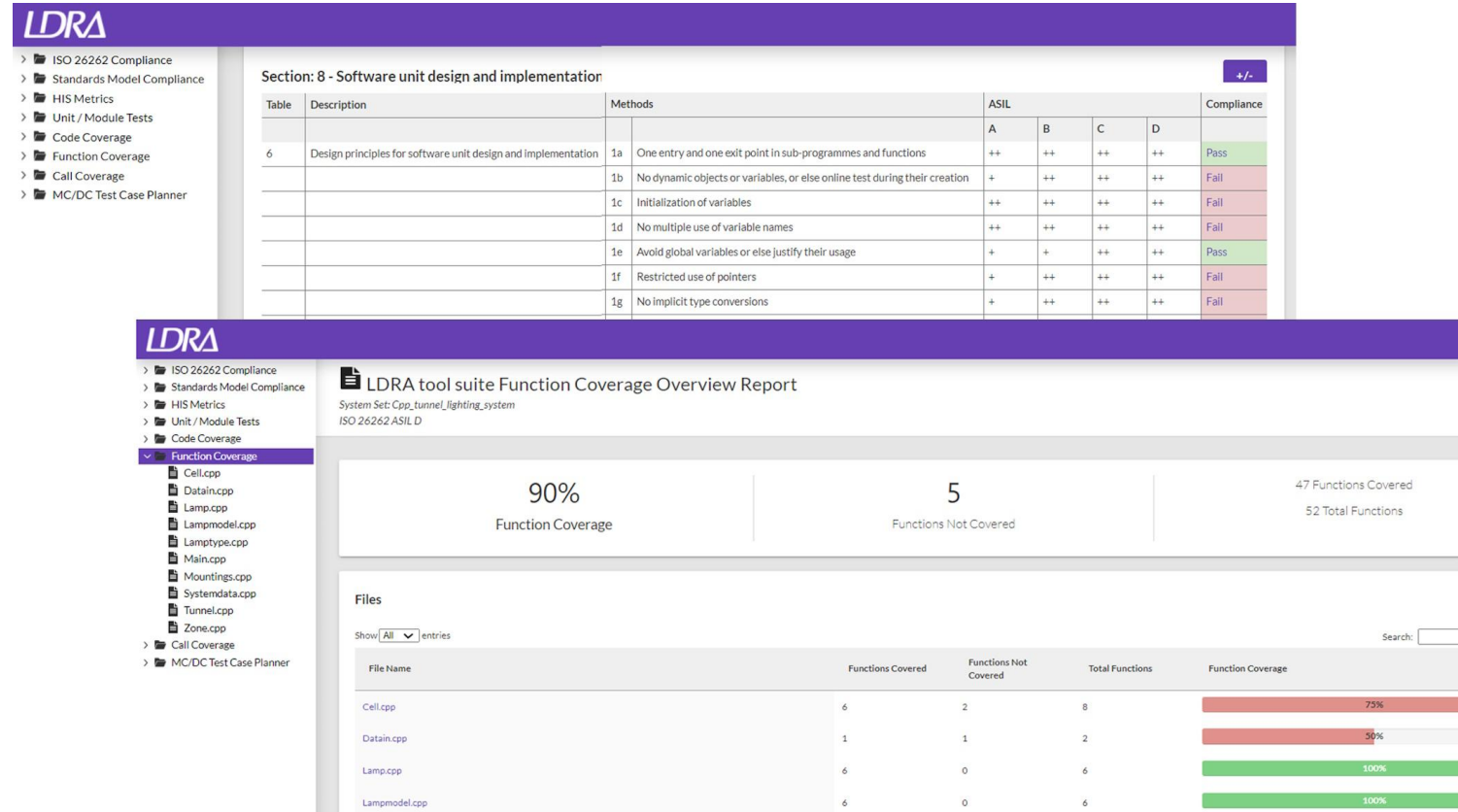
Issue	Count	Category	Severity	Count	Standard
Local or member denominator not checked before use.	3	Rule	127 D		CWE-3.4 129,369
Local or member denominator not checked before use. : SirenSpacing		Rule	127 D		CWE-3.4 129,369
Local or member denominator not checked before use. : SirenSpacing		Rule	127 D		CWE-3.4 129,369
Local or member denominator not checked before use. : ExitSignSpacing		Rule	127 D		CWE-3.4 129,369

Issue	Count	Category	Severity	Count	Standard
Remainder of % op could be negative.	4	Document	584 S		MISRA C:2023 D.1.1
Local or member denominator not checked before use.	3	Required	127 D		MISRA C:2023 D.4.1,D.4.14
No brackets to then/else. (analysed w. MISRA-C++:2008)	4	Required	12 S		MISRA C:2023 R.15.6



Publish analysis and test results as indexed reports

Reports fulfill safety and security standards objectives



The screenshot displays two reports from the LDRA tool suite. The top report is titled "Section: 8 - Software unit design and implementation" and contains a table of compliance results. The bottom report is titled "LDRA tool suite Function Coverage Overview Report" and shows a summary of function coverage for a system set.

Table	Description	Methods	ASIL				Compliance
			A	B	C	D	
6	Design principles for software unit design and implementation	1a	++	++	++	++	Pass
		1b	+	++	++	++	Fail
		1c	++	++	++	++	Fail
		1d	++	++	++	++	Fail
		1e	+	+	++	++	Pass
		1f	+	++	++	++	Fail
		1g	+	++	++	++	Fail

LDRA tool suite Function Coverage Overview Report
System Set: Cpp_tunnel_lighting_system
ISO 26262 ASIL D

90% Function Coverage | 5 Functions Not Covered | 47 Functions Covered / 52 Total Functions

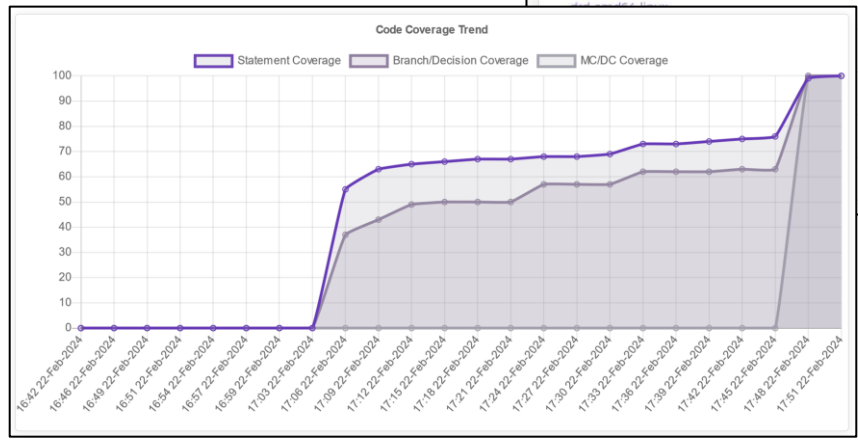
File Name	Functions Covered	Functions Not Covered	Total Functions	Function Coverage
Cell.cpp	6	2	8	75%
Datain.cpp	1	1	2	50%
Lamp.cpp	6	0	6	100%
Lampmodel.cpp	6	0	6	100%



Efficiently manage large sets of software certification artifacts

Leverage LDRA's proven analysis capabilities

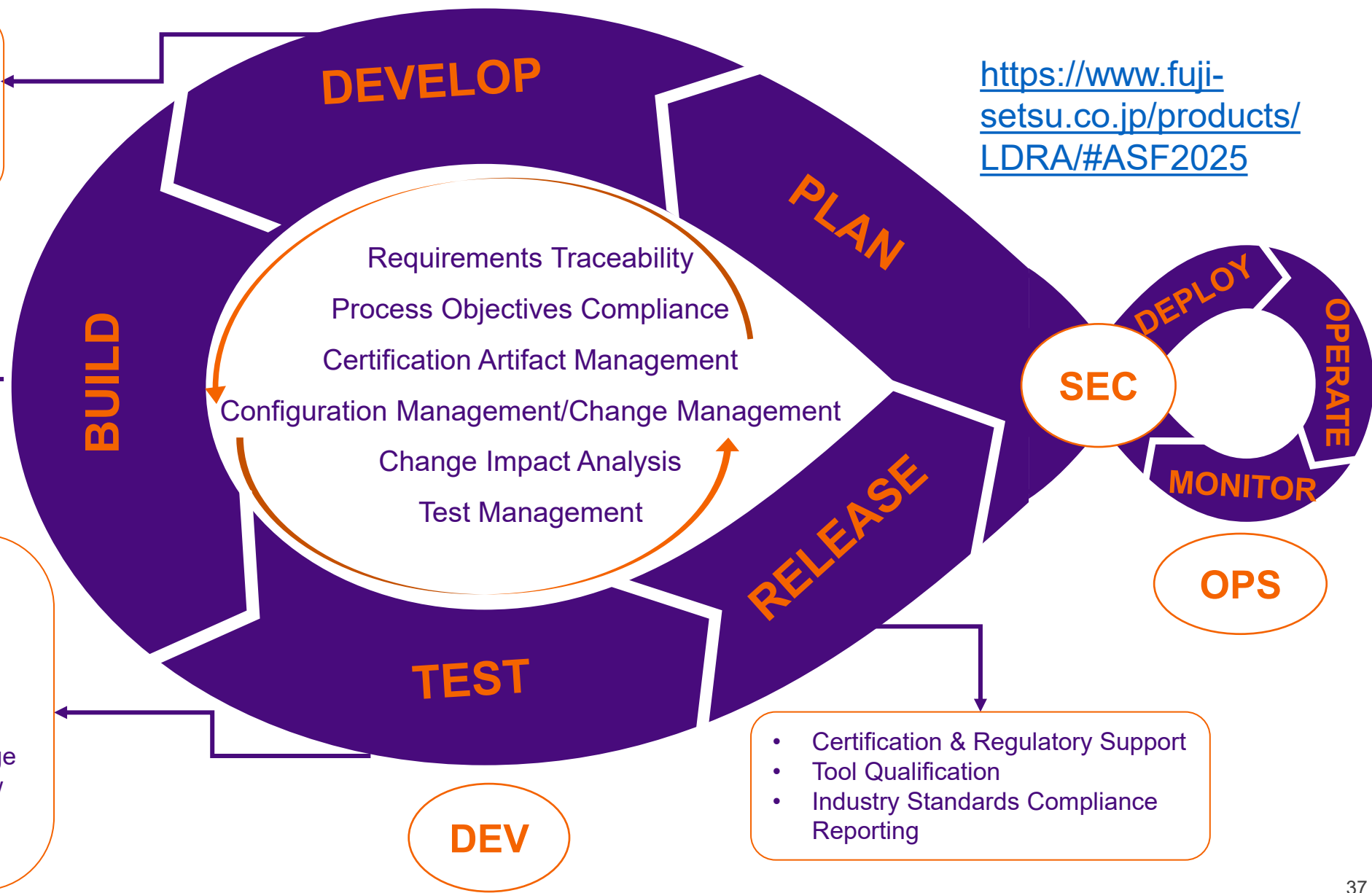
Component	Files	Violations	Statement	Branch/Decision	MC/DC	TC Total	TC Passing	Uploads
Cachegrind-amd64-Linux	4	3842 (+457)	15% (+5)	11% (+3)	4%	84 (+37)	77 (+33)	11
Calgrind-amd64-linux	14	406 (-66)	n/a	n/a	n/a	n/a	n/a	4
cg_merge	1	0	9% (+8)	8% (+7)	0%	8 (+4)	1	3
	14	3378	n/a	n/a	n/a	n/a	n/a	1
	0	n/a	n/a	n/a	n/a	n/a	n/a	0
	0	n/a	n/a	n/a	n/a	n/a	n/a	0
	0	n/a	n/a	n/a	n/a	n/a	n/a	0
	1	45	n/a	n/a	n/a	n/a	n/a	1



- Coding Standards Compliance
- Code Quality Analysis
- Taint/Vulnerability Analysis
- Data & Control Flow Analysis
- Software Quality Reporting

<https://www.fuji-setsu.co.jp/products/LDRA/#ASF2025>

- Target Testing
- Robustness Testing/Fuzzing
- Requirements-Based Testing
- Timing Analysis
- Structural Coverage Analysis
- Source/Object Code Coverage
- Function Coverage & Call Coverage
- Dynamic Data Flow & Control Flow Coverage
- Software Verification Reporting



- Certification & Regulatory Support
- Tool Qualification
- Industry Standards Compliance Reporting



ZERTIFIKAT ◆ CERTIFICATE ◆ CERTIFICADO ◆ CERTIFICAT




Product Service

CERTIFICATE

No. Z10 084753 0006 Rev. 01

Holder of Certificate: LDRA Ltd.
Portside, Monks Ferry
Wirral, Merseyside CH41 5LH
UNITED KINGDOM

Certification Mark: 

Product: Software Tool for Safety Related Development

Model(s): LDRA tool suite
LDRArules
LDRAcover
LDRAunit
LDRAlite

Parameters: The certified tools fulfils the requirements for support tools classified T2 according to IEC 61508-3 and EN 50128. The tools are qualified to be used in safety-related software development according to IEC 61508, EN 50128 and ISO 26262. It is suitably validated for use in safety-related development according to IEC 62304. The test report is a mandatory part of this certificate.

Tested according to: IEC 61508-3:2010
IEC 62304:2006
IEC 62304:2006/AMD1:2015
ISO 26262-8:2018
EN 50128:2011/AMD2:2020

The product was tested on a voluntary basis and complies with the essential requirements. The certification mark shown above can be affixed on the product. It is not permitted to alter the certification mark in any way. In addition the certification holder must not transfer the certificate to third parties. This certificate is valid until the listed date, unless it is cancelled earlier. All applicable requirements of the Testing, Certification, Validation and Verification Regulations of TÜV SÜD Group have to be complied. For details see: www.tuvsud.com/ps-cert

Test report no.: LW85043C

Valid until: 2029-04-18

Date, 2024-04-23


 (Christian Dirmeier)

Page 1 of 1
TÜV SÜD Product Service GmbH • Certification Body • Ridlerstraße 65 • 80339 Munich • Germany






CERTIFICATE NO FS/71/220/23/1048 **PAGE 1/1**

ZERTIFIKAT NR. (GEGEN)

LICENCE HOLDER & MANUFACTURER
GENEHMIGTER/PROFIZIENZIERER

LDRA Ltd
Portside, Monks Ferry,
Wirral, CH41 5LH,
United Kingdom

PROJECT NO/ID
PROJEKT-NR./ID

M15F-AU05

LICENSED TEST MARK
GENEHMIGTES PROFIZIENZIECHEN



CERT. REPORT NO.
ZERTIFIKATBERICHT NR.

M15F0010
is an integral part of this certificate.
ist ein integral Bestandteil dieses Zertifikats.

Certified product(s)
Zertifizierte Produkt(e)

LDRA tool suite
LDRAunit
LDRArules
LDRAcover
LDRAlite

Version 10.0.3 and 10.1.0

Tested according to
Geprüft nach

IEC 61508-3:2010, clause 7.4.4
ISO 26262-8:2018, clause 11.4.9
EN 50128:2011
IEC 60880:2006
IEC 62304:2015

Technical Data and Parameter
Technische Daten und Parameter

Suitable for development of safety-related software acc. to:

- ISO 26262 up to ASIL D,
- IEC 61508 up to SIL 4, class T2 tool
- EN 50128 up to SIL 4, class T2 tool
- IEC 62304 up to SW safety Class C
- IEC 60880 for category A functions

The certificate is based on voluntary tests. The compliance of the certified product against the requirements of above listed functional safety standards was evaluated. Any changes to the design, components or processing may reduce applicability of some parts of the certification to retain the certification. All applicable requirements of the testing, and certification regulations of SGS-TÜV Saar GmbH have to be complied, see www.sgs-tuv-saar.com/en/ps-cert

Certification Body
for Functional Safety &
Cyber Security
SGS-TÜV Saar GmbH
Zertifizierungsstelle für Funktionale Sicherheit &
Cyber Sicherheit



Munich, Mar 08, 2023


 Gudrun Neumann

Reference to
SGS Certificates
Datenbank



SGS-TÜV Saar GmbH, Hoffmanns-Str.
61079 Mönchengladbach, Germany
Website: www.sgs-tuv-saar.com
© 2023, Hoffmanns.com

Challenges of certifying next-generation automotive software

- Rapid evolution of SDV technology
- Safety and security concerns
- Growth in software content, including safety and security functions

Overcoming the challenges with LDRA

- Flexible and scalable proven in use solutions
- Designed to meet the highest safety and security standards
- Supports DevSecOps and continuous integration platforms



Contact Us



ldra.com



info@ldra.com

Follow Us



LDRA Limited



LDRA



FUJI SETSUBI



LDRA スタンダード認証支援テストツール

<https://www.fuji-setsu.co.jp/products/LDRA/>



富士設備工業(株)電子機器事業部
<https://www.fuji-setsu.co.jp>