Document Title	Specification of Development Error Tracer
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	017
Document Classification	Standard

Document Version	3.2.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Change Description
30.09.2011	3.2.0	AUTOSAR Administration	• Clarifications related to include structure etc.
18.10.2010	3.1.0	AUTOSAR Administration	• DLT is now an optional interface of DET • Harmonized parameter error handling • Removed known limitation of Revision 4.0.1
30.11.2009	3.0.0	AUTOSAR Administration	• Changed Tracing to requirements now located in SRS_Debugging • Added a Std_ReturnType value to Det_ReportError • Harmonized configuration classes • Adopted to the changed general requirements • Legal disclaimer revised • Chapter 10.3 revised
23.06.2008	2.2.1	AUTOSAR Administration	Legal disclaimer revised
12.12.2007	2.2.0	AUTOSAR Administration	• Added API GetVersionInfo to harmonize SWS with AUTOSAR conventions • Document meta information extended • Small layout adaptations made
25.01.2007	2.1.0	AUTOSAR Administration	• Added BSW00436 to traceability matrix • Added Memmap.h • Added Chapter 11 • Legal disclaimer revised • “Advice for users” revised • “Revision Information” added
17.05.2006	2.0.0	AUTOSAR Administration	Changed to new SWS template
08.07.2005	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

This specification and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the specification.

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only.
For any other purpose, no part of the specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

1 Introduction and functional overview 5
2 Acronyms and abbreviations 6
3 Related documentation 7
3.1 Input documents 7
3.2 Related standards and norms 7
4 Constraints and assumptions 8
4.1 Limitations 8
4.2 Applicability to car domains 8
5 Dependencies to other modules 9
5.1 File structure 9
6 Requirements traceability 11
6.1 Basic Software Requirements 15
6.2 Diagnostic Requirements 21
7 Functional specification 22
7.1 Initialization 22
7.2 Error Hooks 23
7.3 Error Reporting 23
7.4 Version Information 24
7.5 Debugging support 24
7.6 Error classification 25
7.7 Error detection 25
7.8 Error notification 25
8 API specification 26
8.1 Imported types 26
8.2 Type definitions 26
8.3 Function definitions 26
8.3.1 Det_Init 26
8.3.2 Det_ReportError 26
8.3.3 Det_Start 27
8.3.4 Det_GetVersionInfo 27
8.4 Expected Interfaces 28
8.4.1 Mandatory Interfaces 28
8.4.2 Optional Interfaces 29
9 Sequence diagrams 30
9.1 Initialization 30
9.2 Error reporting 30
10 Configuration specification 32
10.1 Containers and configuration parameters 32
10.2 Variants 32
10.2.1 Variants 32

10.2.2 Det 33
10.2.3 DetGeneral 33
10.2.4 DetNotification 34
10.3 Published Information 35
11 AUTOSAR Interfaces 36
11.1 Scope of this Chapter 36
11.2 Overview 36
11.2.1 Use Case 36
11.3 Specification of the Ports and Port Interfaces 36
11.3.1 General Approach 37
11.3.2 Data Types 37
11.3.3 Port Interface 37
11.4 Definition of the Service 39
11.5 Configuration of the DET 39
12 Not applicable requirements 40

1 Introduction and functional overview

This specification describes the API of the Development Error Tracer. All detected development errors in the Basic Software are reported to this module. The API parameters allow for tracing source and kind of error:

- Module in which error has been detected
- Function in which error has been detected
- Type of error

The functionality behind the API of this module is not in scope of this specification. It is up to the software developer and software integrator to choose the optimal strategy for his specific application and testing environment. Possible functionalities could be:

- Set debugger breakpoint within error reporting API
- Count reported errors
- Log calls and passed parameters in RAM buffer
- Send reported errors via communication interface to external logger

Note: The software requirements of the Development Error Tracer are specified in the SRS Diagnostics document.

2 Acronyms and abbreviations

DET: Development Error Tracer.

3 Related documentation

3.1 Input documents

[1] List of Basic Software Modules,
AUTOSAR_TR_BSWModuleList.pdf

[2] Layered Software Architecture,
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[3] General Requirements on Basic Software Modules,
AUTOSAR_SRS_BSWGeneral.pdf

[4] Basic Software Module Description Template,
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

[5] Specification of ECU Configuration,
AUTOSAR_TPS_ECUConfiguration.pdf

[6] Requirements on Diagnostic
AUTOSAR_SRS_Diagnostic.pdf

3.2 Related standards and norms

Not applicable.

4 Constraints and assumptions

4.1 Limitations

This specification does not define the functionality behind the error reporting API.

Memory protection mechanisms of the operating system are not taken into account. It is assumed that the whole Basic Software runs in supervisor mode or the switch to supervisor mode is done by a system call within the error reporting function of the DET module.

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

5.1 File structure

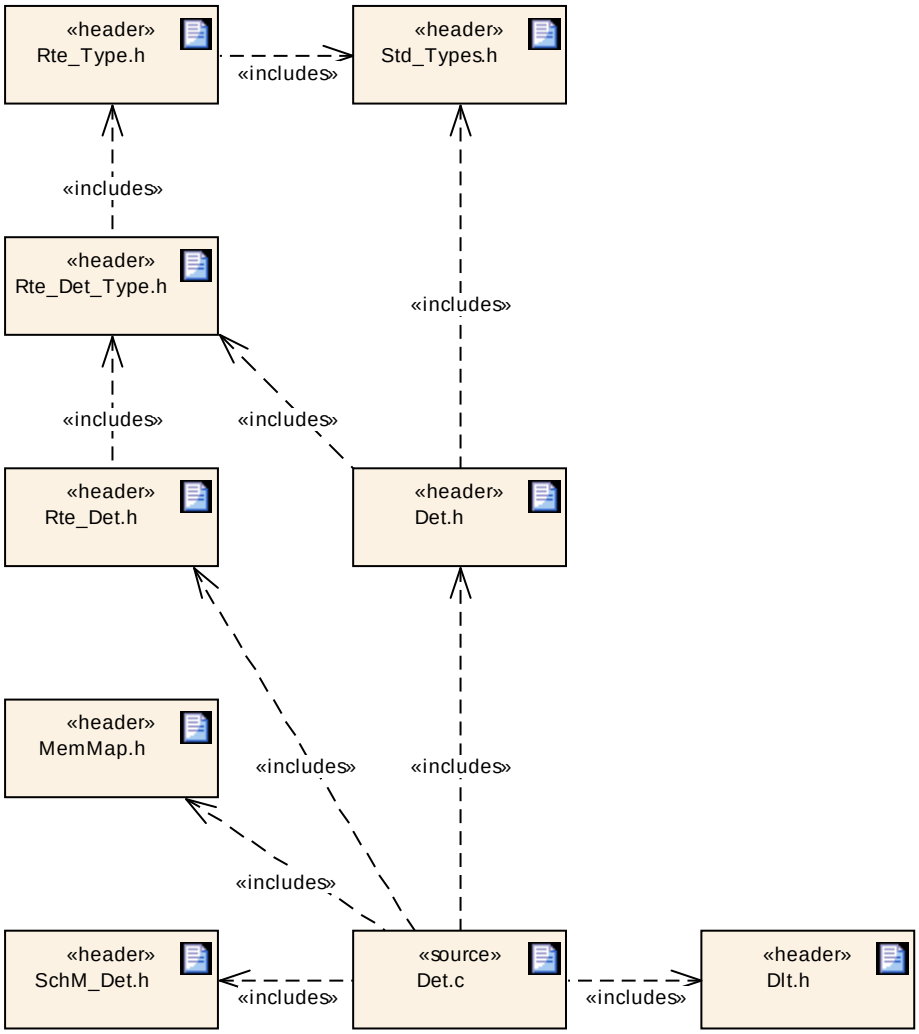

Figure 1: DET Include Structure

[DET004] ⌈The Development Error Tracer module's source code shall offer a header file Det.h, see Figure 1.⌋(BSW00300, BSW00302, BSW00321)

[DET037] ⌈Det.h includes all user relevant information for the tracing of development errors.⌋()

[DET029] ⌈All type definitions of variables which shall be debugged, shall be accessible by the header file Det.h.⌋()

[DET031] ⌈The declaration of variables in the header file shall be such, that it is possible to calculate the size of the variables by C-"sizeof".⌋()

[DET033] ⌈The implementation of the DET module shall include a header file accessed by the BSW scheduler.⌋(BSW0435)

[DET042] ⌈The DET may include the header file Dlt.h⌋()

[DET006] ⌈The DET module's source code shall include the file MemMap.h in case the mapping of code and data to specific memory sections via memory mapping file is needed. ⌋(BSW00436)

6 Requirements traceability

Requirement	Satisfied by
-	DET015
-	DET037
-	DET014
-	DET035
-	DET022
-	DET052
-	[DET023]
-	DET018
-	DET041
-	DET024
-	DET031
-	DET051
-	DET020
-	DET032
-	DET026
-	DET023
-	DET030
-	DET017
-	[DET009]
-	DET102
-	DET042
-	DET025
-	DET029
-	DET039
BSW00101	DET019
BSW003	DET012
BSW00300	DET004
BSW00301	DET999
BSW00302	DET004
BSW00304	DET999
BSW00305	DET999
BSW00306	DET999
BSW00307	DET999
BSW00308	DET999
BSW00309	DET999
BSW00310	DET008, DET009, DET010, DET011
BSW00312	DET999

BSW00314	DET999
BSW00318	DET011, DET012
BSW00321	DET004
BSW00323	DET999
BSW00324	DET999
BSW00325	DET999
BSW00326	DET999
BSW00328	DET999
BSW00329	DET999
BSW00330	DET999
BSW00331	DET999
BSW00333	DET999
BSW00334	DET999
BSW00335	DET999
BSW00336	DET999
BSW00337	DET999
BSW00338	DET999
BSW00339	DET999
BSW00341	DET999
BSW00342	DET999
BSW00343	DET999
BSW00344	DET999
BSW00345	DET999
BSW00346	DET999
BSW00347	DET999
BSW00348	DET999
BSW00350	DET999
BSW00353	DET999
BSW00355	DET999
BSW00357	DET999
BSW00358	DET008
BSW00359	DET999
BSW00360	DET999
BSW00361	DET999
BSW00369	DET999
BSW00370	DET999
BSW00371	DET999
BSW00373	DET999
BSW00374	DET012
BSW00375	DET999
BSW00376	DET999
BSW00377	DET999

BSW00378	DET999
BSW00379	DET999
BSW00380	DET999
BSW00381	DET999
BSW00383	DET999
BSW00385	DET999
BSW00386	DET999
BSW00387	DET999
BSW00388	DET999
BSW00389	DET999
BSW00390	DET999
BSW00391	DET999
BSW00392	DET999
BSW00393	DET999
BSW00394	DET999
BSW00395	DET999
BSW00396	DET999
BSW00397	DET999
BSW00398	DET999
BSW00399	DET999
BSW004	DET036, DET999
BSW00400	DET999
BSW00401	DET999
BSW00404	DET999
BSW00405	DET999
BSW00406	DET999
BSW00407	DET999
BSW00409	DET999
BSW00410	DET999
BSW00411	DET028
BSW00412	DET999
BSW00413	DET999
BSW00414	DET008
BSW00415	DET999
BSW00416	DET999
BSW00417	DET999
BSW00419	DET999
BSW00420	DET999
BSW00421	DET999
BSW00422	DET999
BSW00423	DET999
BSW00424	DET999

BSW00425	DET999
BSW00426	DET999
BSW00427	DET999
BSW00428	DET999
BSW00429	DET999
BSW00431	DET999
BSW00432	DET999
BSW00433	DET999
BSW00434	DET999
BSW00436	DET006
BSW00437	DET999
BSW00438	DET999
BSW00439	DET999
BSW00440	DET999
BSW00441	DET999
BSW005	DET999
BSW006	DET999
BSW007	DET999
BSW009	DET999
BSW010	DET999
BSW04085	DET009
BSW04086	DET009
BSW04087	DET206, DET205, DET204, DET203, DET202, DET201, DET200
BSW04089	DET207
BSW04090	DET027
BSW04101	DET034
BSW0435	DET033
BSW158	DET999
BSW159	DET999
BSW160	DET999
BSW161	DET999
BSW162	DET999
BSW164	DET999
BSW167	DET999
BSW168	DET999
BSW170	DET999
BSW172	DET999

The following table lists the general basic software requirements and its related SWS requirements in the document. The specific requirements for DET are part of the diagnose mechanism and therefore part of the SRS-Diagnostic (Chapter 5.1.5 in [6]).

6.1 Basic Software Requirements

Requirement	*Satisfied by*
[BSW00300] Module naming convention [approved]	DET004
[BSW00301] Limit imported information [approved]	Not applicable (this is a development support module, no included files)
[BSW00302] Limit exported information [approved]	DET004
[BSW00304] AUTOSAR integer data types [approved]	Not applicable (Not used in this module)
[BSW00305] Self-defined data types naming convention [approved]	Not applicable (Not used in this module)
[BSW00306] Avoid direct use of compiler and platform specific keywords [approved]	Not applicable (Not used in this module)
[BSW00307] Global variables naming convention [approved]	Not applicable (Not used in this module)
[BSW00308] Definition of global data [approved]	Not applicable (Not used in this module)
[BSW00309] Global data with read-only constraint [approved]	Not applicable (Not used in this module)
[BSW00310] API naming convention [approved]	DET008 DET009 DET010 DET011
[BSW00312] Shared code shall be reentrant [approved]	Not applicable (Not used in this module)
[BSW00439] Declaration of interrupt handlers and ISRs	Not applicable (Not used in this module)
[BSW00314] Separation of interrupt frames and service routines [approved]	Not applicable (Not used in this module)
[BSW00318] Format of module version numbers [approved]	DET011 DET012
[BSW00321] Enumeration of module version numbers [approved]	DET004
[BSW00324] Do not use HIS I/O Library [approved]	Not applicable (Not used in this module)
[BSW00325] Runtime of interrupt service routines [approved]	Not applicable (Not used in this module)
[BSW00326] Transition from ISRs to OS tasks [approved]	Not applicable (Not used in this module)
[BSW00327] Error values naming convention [approved]	See Chapter 7.6
[BSW00328] Avoid duplication of code [approved]	Not applicable (Not used in this module)
[BSW00329] Avoidance of generic interfaces [approved]	Not applicable (Not used in this module)
[BSW00330] Usage of macros / inline functions instead of functions [approved]	Not applicable (Not used in this module)
[BSW00331] Separation of error and status values [approved]	Not applicable (Not used in this module)
[BSW00333] Documentation of callback function context [approved]	Not applicable (Not used in this module)

[BSW00334] Provision of XML file [approved]	Not applicable (Not used in this module)
[BSW00335] Status values naming convention [approved]	Not applicable (Not used in this module)
[BSW00341] Microcontroller compatibility documentation [approved]	Not applicable (Not used in this module)
[BSW00342] Usage of source code and object code [approved]	Not applicable (Not used in this module)
[BSW00343] Specification and configuration of time [approved]	Not applicable (Not used in this module)
[BSW00341] Microcontroller compatibility documentation [approved]	Not applicable (Not used in this module)
[BSW00346] Basic set of module files [approved]	Not applicable (Not used in this module)
[BSW00347] Naming separation of different instances of BSW drivers [approved]	Not applicable (Not used in this module)
[BSW00441] Enumeration literals and #define naming convention	Not applicable (Not used in this module)
[BSW00350] Development error detection keyword [approved]	Not applicable (Not used in this module)
[BSW00353] Platform specific type header [approved]	Not applicable (Not used in this module)
[BSW00355] Do not redefine AUTOSAR integer data types [approved]	Not applicable (Not used in this module)
[BSW00350] Development error detection keyword [approved]	Not applicable (Not used in this module)
[BSW00358] Return type of init() functions [approved]	DET008
[BSW00359] Return type of callback functions [approved]	Not applicable (Not used in this module)
[BSW00360] Parameters of callback functions [approved]	Not applicable (Not used in this module)
[BSW00440] Function prototype for callback functions of AUTOSAR Services	Not applicable (Not used in this module)
[BSW00361] Compiler specific language extension header [approved]	Not applicable (Not used in this module)
[BSW00370] Separation of callback interface from API [approved]	Not applicable (Not used in this module)
[BSW00371] Do not pass function pointers via API [approved]	Not applicable (Not used in this module)
[BSW00373] Main processing function naming convention [approved]	Not applicable (Not used in this module)
[BSW00374] Module vendor identification [approved]	DET012
[BSW00376] Return type and parameters of main processing functions [approved]	Not applicable (Not used in this module)
[BSW00377] Module specific API return types [approved]	Not applicable (Not used in this module)
[BSW00378] AUTOSAR boolean type [approved]	Not applicable (Not used in this module)
[BSW00379] Module identification [approved]	Not applicable (Not used in this module)
[BSW00401] Documentation of multiple instances of configuration parameters [approved]	Not applicable (Not used in this module)
[BSW00408] Configuration parameter naming convention [approved]	See Chapter 10.
[BSW00410] Compiler switches shall have defined values [approved]	Not applicable (Not used in this module)
[BSW00411] Get version info keyword [approved]	DET028
[BSW00413] Accessing instances of BSW modules [approved]	Not applicable (Not used in this module)

[BSW00414] Parameter of init function [approved]	DET008
[BSW00415] User dependent include files [approved]	Not applicable (Not used in this module)
[BSW005] No hard coded horizontal interfaces within MCAL [approved]	Not applicable (Not used in this module)
[BSW006] Platform independency [approved]	Not applicable (Not used in this module)
[BSW007] HIS MISRA C [approved]	Not applicable (Not used in this module)
[BSW009] Module User Documentation [approved]	Not applicable (Not used in this module)
[BSW010] Memory resource documentation [approved]	Not applicable (Not used in this module)
[BSW158] Separation of configuration from implementation [approved]	Not applicable (Not used in this module)
[BSW160] Human-readable configuration data [approved]	Not applicable (Not used in this module)
[BSW161] Microcontroller abstraction [approved]	Not applicable (Not used in this module)
[BSW162] ECU layout abstraction [approved]	Not applicable (Not used in this module)
[BSW164] Implementation of interrupt service routines [approved]	Not applicable (Not used in this module)
[BSW172] Compatibility and documentation of scheduling strategy [approved]	Not applicable (Not used in this module)
[BSW00344] Reference to link-time configuration	Not applicable (Not used in this module)
[BSW00404] Reference to post build time configuration	Not applicable (Not used in this module)
[BSW00405] Reference to multiple configuration sets	Not applicable (Not used in this module)
[BSW00345] Pre-compile-time configuration	Not applicable (Not used in this module)
[BSW159] Tool-based configuration	Not applicable (this is a tool requirement)
[BSW167] Static configuration checking	Not applicable (Requirement on configuration tool)
[BSW171] Configurability of optional functionality	see Chapter 10
[BSW170] Data for reconfiguration of AUTOSAR SW-Components	Not applicable (Not used in this module)
[BSW00380] Separate C-Files for configuration parameters	Not applicable (Not used in this module)
[BSW00419] Separate C-Files for pre-compile time configuration parameters	Not applicable (Not used in this module)
[BSW00381] Separate configuration header file for pre-compile time parameters	Not applicable (Not used in this module)
[BSW00412] Separate H-File for configuration parameters	Not applicable (Not used in this module)
[BSW00383] List dependencies of configuration files	Not applicable (Not used in this module)
[BSW00384] List dependencies to other modules	see Chapter 8.4
[BSW00387] Specify the configuration class of callback function	Not applicable (Not used in this module)
[BSW00388] Introduce containers	Not applicable (Not used in this module)
[BSW00389] Containers shall have names	Not applicable (Not used in this module)

[BSW00390] Parameter content shall be unique within the module	Not applicable (Not used in this module)
[BSW00391] Parameter shall have unique names	Not applicable (Not used in this module)
[BSW00392] Parameters shall have a type	Not applicable (Not used in this module)
[BSW00393] Parameters shall have a range	Not applicable (Not used in this module)
[BSW00394] Specify the scope of the parameters	Not applicable (Not used in this module)
[BSW00395] List the required parameters (per parameter)	Not applicable (Not used in this module)
[BSW00396] Configuration classes	Not applicable (Not used in this module)
[BSW00397] Pre-compile-time parameters	Not applicable (Not used in this module)
[BSW00398] Link-time parameters	Not applicable (Not used in this module)
[BSW00399] Loadable Post-build time parameters	Not applicable (Not used in this module)
[BSW00400] Selectable Post-build time parameters	Not applicable (Not used in this module)
[BSW00438] Post Build Configuration Data Structure	Not applicable (Not used in this module)
[BSW00402] Published information	DET040.
[BSW00375] Notification of wake-up reason	Not applicable (Not used in this module)
[BSW101] Initialization interface	DET019
[BSW003] Version identification	DET012
[BSW004] Version check	DET036
[BSW00416] Sequence of Initialization	Not applicable (Not used in this module)
[BSW00406] Check module initialization	Not applicable (Not used in this module)
[BSW00437] NoInit--Area in RAM	Not applicable (Not used in this module)
[BSW168] Diagnostic Interface of SW components	Not applicable (Not used in this module)
[BSW00407] Function to read out published parameters	Not applicable (Not used in this module)
[BSW00423] Usage of SW-C template to describe BSW modules with AUTOSAR Interfaces	Not applicable (Not used in this module)
[BSW00424] BSW main processing function task allocation	Not applicable (Not used in this module)
[BSW00425] Trigger conditions for schedulable objects	Not applicable (Not used in this module)
[BSW00426] Exclusive areas in BSW modules	Not applicable (Not used in this module)
[BSW00427] ISR description for BSW modules	Not applicable (Not used in this module)
[BSW00428] Execution order dependencies of main processing functions	Not applicable (not related to this specification)
[BSW00429] Restricted BSW OS functionality access	Not applicable (Not used in this module)
[BSW00431] The BSW Scheduler module implements task bodies	Not applicable (Not used in this module)
[BSW00432] Modules should have separate main processing functions for read/receive and write/transmit data path	Not applicable (Not used in this module)

[BSW00433] Calling of main processing functions	Not applicable (Not used in this module)
[BSW00434] The Schedule Module shall provide an API for exclusive areas	Not applicable (Not used in this module)
[BSW00336] Shutdown interface	Not applicable (Not used in this module)
[BSW00337] Classification of errors	Not applicable (this module does not produce own errors)
[BSW00338] Detection and Reporting of development errors	Not applicable (this module does not produce own errors)
[BSW00369] Do not return development error codes via API	Not applicable (Not used in this module)
[BSW00339] Reporting of production relevant error status	Not applicable (Not used in this module)
[BSW00348] Standard type header	Not applicable (Not used in this module)
[BSW00357] Standard API return type	Not applicable (Not used in this module)
[BSW00421] Reporting of production relevant error events	Not applicable (Not used in this module)
[BSW00422] Debouncing of production relevant error status	Not applicable (not related to this specification)
[BSW00420] Production relevant error event rate detection	Not applicable (not related to this specification)
[BSW00417] Reporting of Error Events by Non-Basic Software	Not applicable (Not used in this module)
[BSW00323] API parameter checking	Not applicable (Not used in this module)
[BSW004] Version check	Not applicable (Not used in this module)
[BSW00409] Header files for production code error IDs	Not applicable (Not used in this module)
[BSW00385] List possible error notifications	Not applicable (Not used in this module)
[BSW00386] Configuration for detecting an error	Not applicable (Not used in this module)
[BSW00435] Module Header File Structure for the Basic Software Scheduler	DET033
[BSW00436] Module Header File Structure for the Basic Software Memory Mapping	DET006
[BSW00390] Parameter content shall be unique within the module	Not applicable (Not used in this module)
[BSW00391] Parameter shall have unique names	Not applicable (Not used in this module)
[BSW00392] Parameters shall have a type	Not applicable (Not used in this module)
[BSW00393] Parameters shall have a range	Not applicable (Not used in this module)
[BSW00394] Specify the scope of the parameters	Not applicable (Not used in this module)
[BSW00395] List the required parameters (per parameter)	Not applicable (Not used in this module)
[BSW00396] Configuration classes	Not applicable (Not used in this module)
[BSW00397] Pre-compile-time parameters	Not applicable (Not used in this module)
[BSW00398] Link-time parameters	Not applicable

	(Not used in this module)
[BSW00399] Loadable Post-build time parameters	Not applicable (Not used in this module)
[BSW00400] Selectable Post-build time parameters	Not applicable (Not used in this module)
[BSW00438] Post Build Configuration Data Structure	Not applicable (Not used in this module)
[BSW00402] Published information	DET040.
[BSW00375] Notification of wake-up reason	Not applicable (Not used in this module)
[BSW101] Initialization interface	DET019
[BSW003] Version identification	DET012
[BSW004] Version check	DET036
[BSW00416] Sequence of Initialization	Not applicable (Not used in this module)
[BSW00406] Check module initialization	Not applicable (Not used in this module)
[BSW00437] NoInit--Area in RAM	Not applicable (Not used in this module)
[BSW168] Diagnostic Interface of SW components	Not applicable (Not used in this module)
[BSW00407] Function to read out published parameters	Not applicable (Not used in this module)
[BSW00423] Usage of SW-C template to describe BSW modules with AUTOSAR Interfaces	Not applicable (Not used in this module)
[BSW00424] BSW main processing function task allocation	Not applicable (Not used in this module)
[BSW00425] Trigger conditions for schedulable objects	Not applicable (Not used in this module)
[BSW00426] Exclusive areas in BSW modules	Not applicable (Not used in this module)
[BSW00427] ISR description for BSW modules	Not applicable (Not used in this module)
[BSW00428] Execution order dependencies of main processing functions	Not applicable (not related to this specification)
[BSW00429] Restricted BSW OS functionality access	Not applicable (Not used in this module)
[BSW00431] The BSW Scheduler module implements task bodies	Not applicable (Not used in this module)
[BSW00432] Modules should have separate main processing functions for read/receive and write/transmit data path	Not applicable (Not used in this module)
[BSW00433] Calling of main processing functions	Not applicable (Not used in this module)
[BSW00434] The Schedule Module shall provide an API for exclusive areas	Not applicable (Not used in this module)
[BSW00336] Shutdown interface	Not applicable (Not used in this module)
[BSW00337] Classification of errors	Not applicable (this module does not produce own errors)
[BSW00338] Detection and Reporting of development errors	Not applicable (this module does not produce own errors)
[BSW00369] Do not return development error codes via API	Not applicable (Not used in this module)
[BSW00339] Reporting of production relevant error status	Not applicable (Not used in this module)
[BSW00348] Standard type header	Not applicable

	(Not used in this module)
[BSW00357] Standard API return type	Not applicable (Not used in this module)
[BSW00421] Reporting of production relevant error events	Not applicable (Not used in this module)
[BSW00422] Debouncing of production relevant error status	Not applicable (not related to this specification)
[BSW00420] Production relevant error event rate detection	Not applicable (not related to this specification)
[BSW00417] Reporting of Error Events by Non-Basic Software	Not applicable (Not used in this module)
[BSW00323] API parameter checking	Not applicable (Not used in this module)
[BSW004] Version check	Not applicable (Not used in this module)
[BSW00409] Header files for production code error IDs	Not applicable (Not used in this module)
[BSW00385] List possible error notifications	Not applicable (Not used in this module)
[BSW00386] Configuration for detecting an error	Not applicable (Not used in this module)
[BSW00435] Module Header File Structure for the Basic Software Scheduler	DET033
[BSW00436] Module Header File Structure for the Basic Software Memory Mapping	DET006

6.2 Diagnostic Requirements

Since there is no need to have an extra document for the DET requirements, the requirements for the DET are contained in the SRS_Diagnostic ([6], Chapter 4.1.5). The following table shows the tracing to them.

Requirement	***Satisfied by***
BSW04090 Provide a configurable list of error report receivers	[DET027
BSW04086 Provide a dedicated set of information	[DET009]
BSW04087 Provide a development error report reception service	Chapter 11, [DET205 and [DET200,[DET201, [DET202, [DET203, [DET204, [DET206
BSW04089 Fan-out of received error reports	[DET207
BSW04085 Provide an interface to receive error reports	Chapter 8, [DET009]
BSW04101 Notification interface for log & trace (DLT)	[DET034

7 Functional specification

The Development Error Tracer provides functionality to support error detection and tracing of errors during the development of Software Components and other Basic Software Modules. For this purpose the Development Error Tracer receives and evaluates error messages from these components and modules.

Due to the always specific (non generic!) requirements regarding functionality in error cases there is no explicit specification of the DET implementation, except:

- A configurable list of error hooks will be executed in case of an error report.
- Interfaces will be provided to report errors, allow optional error recovery after reset, to handle optional error recovery information and to retrieve version information.

7.1 Initialization

[DET019] ⌈The DET shall provide the initialization function Det_Init (see DET008).⌋(BSW00101)

[DET020] ⌈Each call of the Det_Init function shall be used to set the Development Error Tracer to a defined initial status (e.g. by removing optional error recovery information).⌋()

Note: DET020 is not testable without knowledge about the non specified functionality and the probably used optional error recovery information.

Note: The usage and meaning of error recovery information is optional and not specified.

[DET025] ⌈The Development Error Tracer shall provide the function Det_Start (see DET010).⌋()

Note: The Development Error Tracer's environment can use the function Det_Start to trigger the Development Error Tracer module for instance (if needed) in case of completed NVRAM initialization for persistent error storage.

Note: In case the Development Error Tracer does not require a startup call the Det_Start function can be empty.

Note: The integrator can decide by configuration of the EcuM, when Det_Init will be called.

Note: The integrator can decide by configuration of the EcuM or ModeM, when and whether Det_Start will be called.

7.2 Error Hooks

[DET207] ⌈To support debugging and error tracing during development, the Development Error Tracer provides functionality for notification of received error reports. Therefore so called error hooks are configurable. The error hooks will be used to forward error notifications. If at least one error hook has been configured, the Development Error Tracer will notify each received error report by calling the configured error hook(s). Configuration of error hooks is done by the AUTOSAR configuration methods described in chapter 10.⌋(BSW04089)

[DET027] ⌈The Development Error Tracer shall be configurable by a list of Error_Hooks.⌋(BSW04090)

[DET035] ⌈Each Error_Hook shall be called with the same set of parameters as the function Det_ReportError.⌋()

7.3 Error Reporting

DET013: The DET shall provide the error report function Det_ReportError (see DET009).

Note: Det_ReportError serves to notify development errors.

[DET024] ⌈If the Development Error Tracer has not been initialized before Det_ReportError is called, Det_ReportError shall return immediately without any other action (no Error_Hook shall be used, no implementer specific function shall be performed and no development error shall be reported). ⌋()

Note: The Development Error Tracer shall never return an error indication (even in case of calling the un-initialized module).

[DET014] ⌈The error report function Det_ReportError shall call immediately all configured Error_Hooks (see DET027).⌋()

[DET018] ⌈The Development Error Tracer shall execute the list of configured Error_Hooks in the order given by the configuration.⌋()

[DET017] ⌈Each called Error_Hook shall get the parameters completely and unchanged that were received by Det_ReportError (see DET009).⌋()

[DET015] ⌈Optional implementation specific functionality shall only be performed after all configured Error_Hooks (see DET027) have been called.⌋()

[DET034] ⌈Each call of the Det_ReportError function shall be forwarded to the DLT module, if this is available/configured.⌋(BSW04101)

[DET039] ⌈The Det_ReportError function shall be reentrant.⌋()

[DET026] ⌈Recursive calls of Det_ReportError are not allowed.⌋()

Note: Such recursive call could happen in case of calling an un-initialized module via an Error_Hook and would lead to a stack overflow.

7.4 Version Information

[DET022] ⌈The Development Error Tracer shall provide the function Det_GetVersionInfo to allow retrieval of version information.⌋()

[DET012] ⌈The function Det_GetVersionInfo shall return the version information of this module including Module Id, Vendor Id and Vendor specific version number according to BSW00318, BSW00321, BSW003 and BSW004 (see [4]).⌋(BSW00318, BSW00374, BSW003)

[DET023] ⌈The Development Error Tracer shall not deliver the version information if the received version info pointer is NULL and shall raise the Error DET_E_PARAM_POINTER and return without any further action.⌋()

[DET036] ⌈The DET module shall avoid the integration of incompatible files by the following pre-processor checks for included (external) header files
<MODULENAME>_AR_RELEASE_MAJOR_VERSION
<MODULENAME>_AR_RELEASE_MINOR_VERSION.⌋(BSW004)

7.5 Debugging support

[DET030] ⌈Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable.⌋()

[DET032] ⌈Variables available for debugging shall be described in the respective Basic Software Module Description⌋()

7.6 Error classification

The DET is only used for development errors. Production errors are not handled by the DET. Since DET uses configurable functions that cannot be null, no errors can occur during Det_ReportError. However the function Det_GetVersionInfo might be called with a NULL-Pointer (see **[DET023]**). Therefore there is only one development error that can be thrown by the DET.

Type or error	*Relevance*	*Related error code*	*Value [hex]*
Det_GetVersionInfo called with null parameter pointer	Development	DET_E_PARAM_POINTER	0x01

7.7 Error detection

The DET is used for error detection. The call of Det_ReportError will cause calls to all configured callback functions see parameter DetErrorHook in **DET005_Conf :**. If no function is configured development error detection is disabled.

[DET050] ⌈The DET is used for error detection. The call of Det_ReportError will cause calls to all configured callback functions see parameter DetErrorHook in **DET005_Conf :**. If no function is configured development error detection is disabled.⌋()

Note: In case no Error_Hooks are configured no additional functions are called. However the forwarding to DLT (see DET0340 and DET006_Conf) is still active if configured.

[DET051] ⌈The DET shall detect the error DET_E_PARAM_POINTER in the function getVersionInfo by comparing the provided structure pointer to NULL.⌋()

7.8 Error notification

Since DET is used only during development, there are no production error notifications.

[DET052] ⌈The DET shall notify the error DET_E_PARAM_POINTER to all functions configured in DetErrorHook.⌋()

8 API specification

8.1 Imported types

This section lists all imported types used by the API. Even if the DET does not require new types, some RTE or Component types can be used within the configuration of the hook functions. Therefore the DET also has the standardized include structure (see BSW00447) for modules with service interfaces:

[DET102]

⌈

-

Module	Imported Type
Std_Types	Std_ReturnType
	Std_VersionInfoType

- Rte_Det_Type: Used RTE-Types (if used in configured functions)⌋()

8.2 Type definitions

No module specific types are required.

8.3 Function definitions

8.3.1 Det_Init

[DET008]

⌈

Service name:	Det_Init
Syntax:	`void Det_Init( void )`
Service ID[hex]:	0x00
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service to initialize the Development Error Tracer.

⌋(BSW00310, BSW00358, BSW00414)

8.3.2 Det_ReportError

[DET009]

⌜

Service name:	Det_ReportError	
Syntax:	Std_ReturnType Det_ReportError(uint16 ModuleId, uint8 InstanceId, uint8 ApiId, uint8 ErrorId)	
Service ID[hex]:	0x01	
Sync/Async:	Depending on implemented functionality: 1. Breakpoint set: no return 2. Internal error counting/logging in RAM: synchronous 3. External error logging via communication interface: asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	ModuleId	Module ID of calling module.
	InstanceId	The identifier of the index based instance of a module, starting from 0, If the module is a single instance module it shall pass 0 as the InstanceId.
	ApiId	ID of API service in which error is detected (defined in SWS of calling module)
	ErrorId	ID of detected development error (defined in SWS of calling module).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	returns always E_OK (is required for services)
Description:	Service to report development errors.	

⌟(BSW00310, BSW04086, BSW04085)

8.3.3 Det_Start

[DET010]

⌜

Service name:	Det_Start
Syntax:	void Det_Start(void)
Service ID[hex]:	0x02
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service to start the Development Error Tracer.

⌟(BSW00310)

8.3.4 Det_GetVersionInfo

[DET011] ⌜

Service name:	Det_GetVersionInfo

Syntax:	`void Det_GetVersionInfo( Std_VersionInfoType* versioninfo )`	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	versioninfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

」(BSW00310, BSW00318)

[DET028] 「The function Det_GetVersionInfo will be available only if it is enabled by the pre-processor switch DetVersionInfoApi.」(BSW00411)

8.4 Expected Interfaces

This chapter specifies all required interfaces of other modules. Figure 2 shows the interfaces of the DET. The only expected interface is optional; it is the Dlt_Det (see Section 8.4.2).

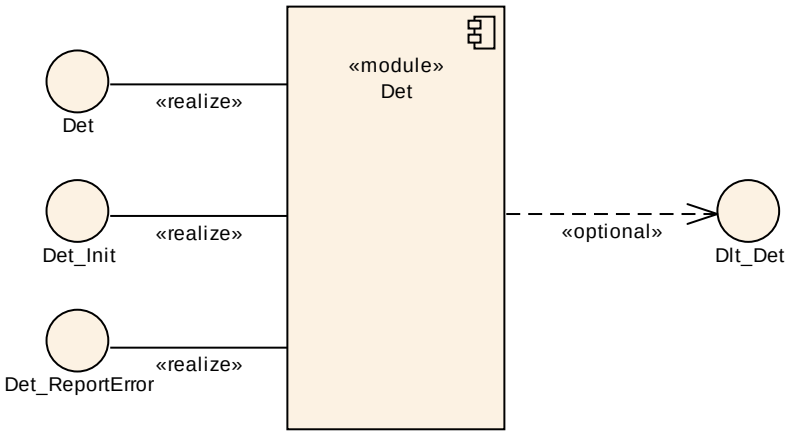

Figure 2: Interfaces of DET

8.4.1 Mandatory Interfaces

There is no mandatory expected interface, but all <User_ErrorHooks> APIs that are used and are configured as callbacks have to be included.

API function	Description

Note: The Name of the user API will not be specified, <User_ErrorHook> is a synonym only.

Note: A list of User_ErrorHook can be defined.

8.4.2 Optional Interfaces

This chapter defines the interfaces that is required to fulfill an optional functionality of the Development Error Tracer.

API function	Description
Dlt_DetForwardErrorTrace	Service to forward error reports from Det to Dlt.

9 Sequence diagrams

9.1 Initialization

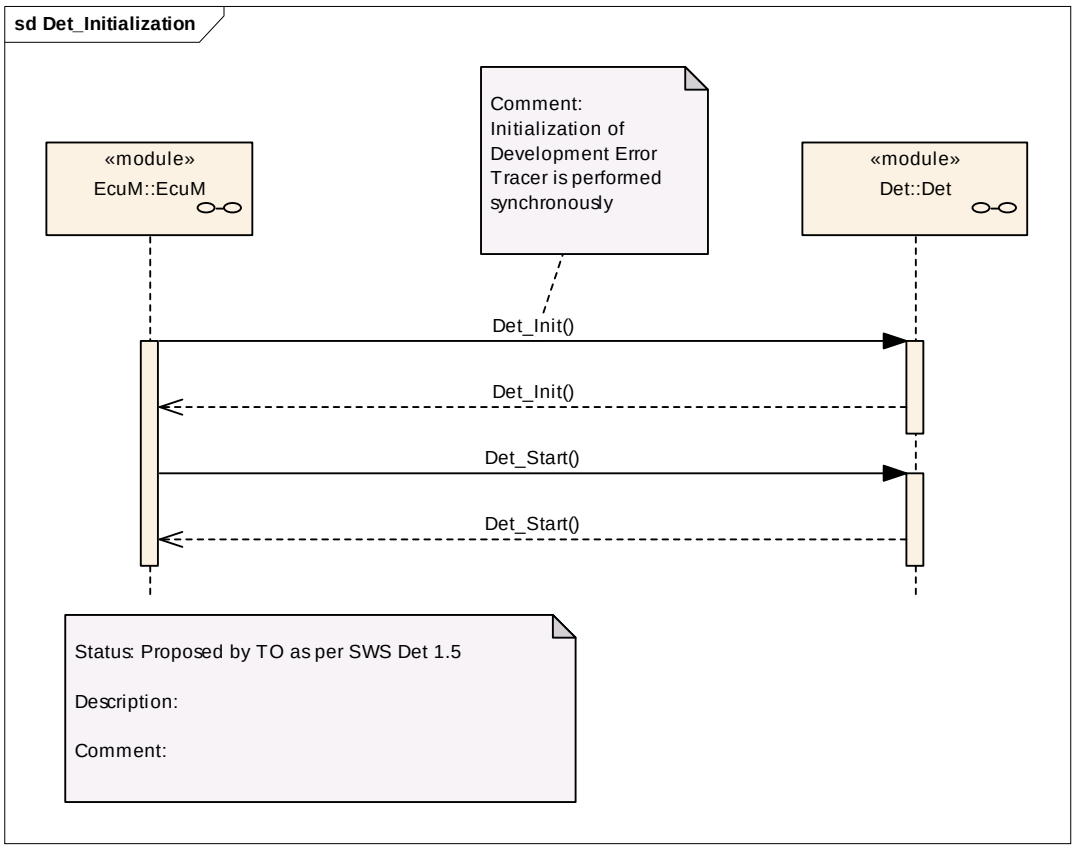

Figure 3: Initialization and start of DET

9.2 Error reporting

Precondition: Initialization as described in chapter 9.1.

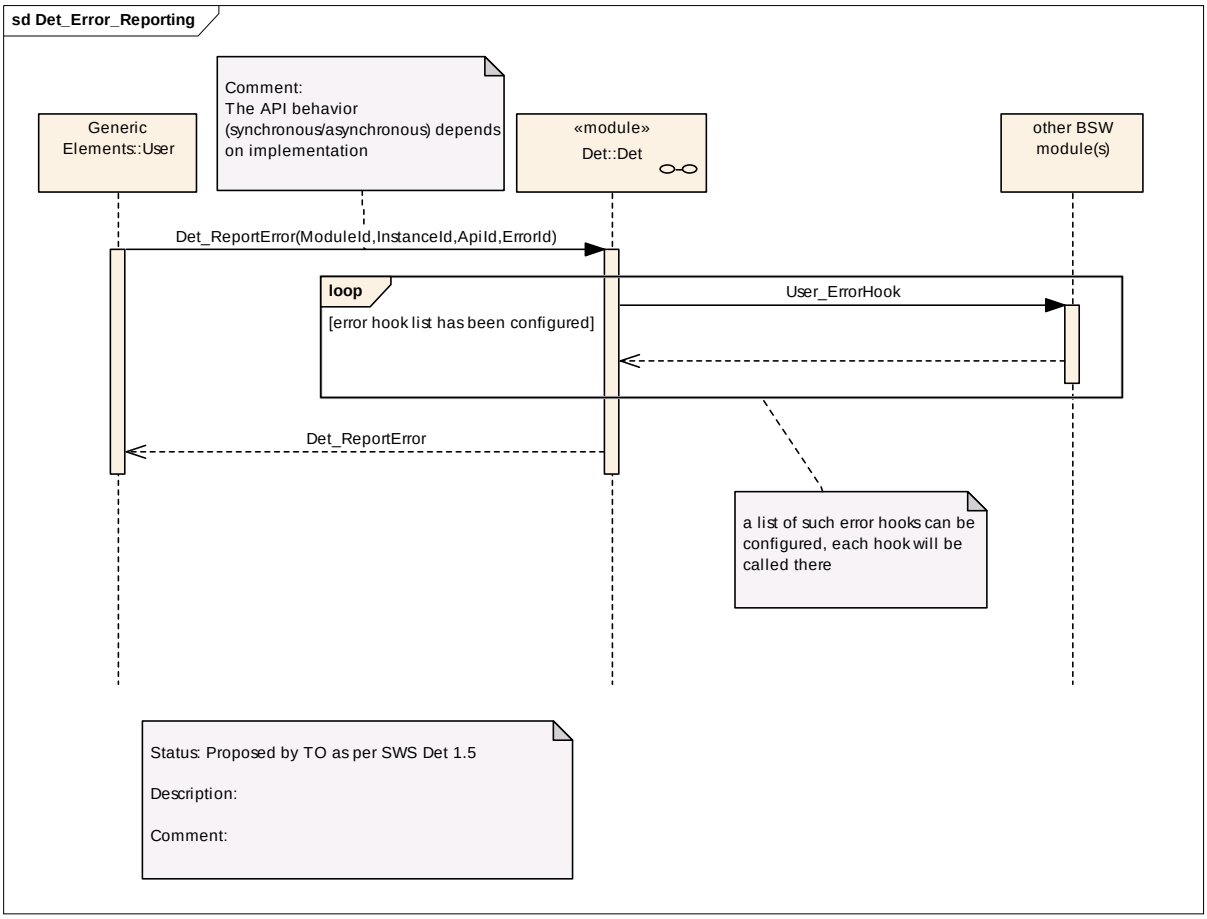

Figure 4: Call of Det_ReportError and user defined hooks

10 Configuration specification

10.1 Containers and configuration parameters

Configuration parameters define the variability of the generic part(s) of an implementation of a module. This means that only generic or configurable module implementation can be adapted to the environment (software/hardware) in use during system and/or ECU configuration.

10.2 Variants

Variants describe sets of configuration parameters. E.g., variant 1: only pre-compile time configuration parameters; variant 2: mix of pre-compile- and post build time-configuration parameters (see [5]).

10.2.1 Variants

The following variants are available in AUTOSAR modules:

- VARIANT-PRE-COMPILE: Only parameters with "Pre-compile time" configuration are allowed in this variant,
- VARIANT-LINK-TIME: Only parameters with "Pre-compile time" and "Link time" are allowed in this variant, and
- VARIANT-POST-BUILD: Parameters with "Pre-compile time", "Link time" and "Post-build time" are allowed in this variant.

[DET101] 「For DET only the VARIANT-PRE-COMPILE is relevant.」()

The Parameters of DET are described in the following sub-sections. Figure 5 gives an overview over them.

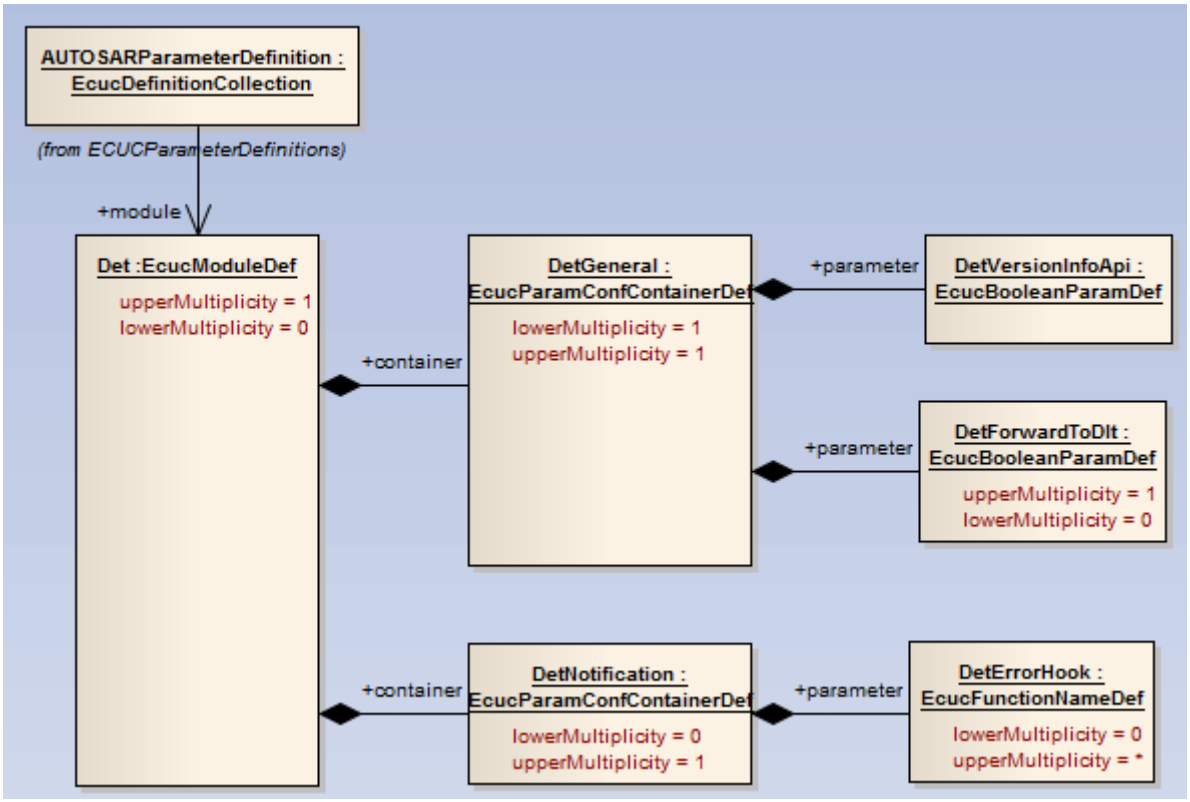

Figure 5: Parameters of DET

10.2.2 Det

SWS Item	**DET001_Conf :**
Module Name	**Det**
Module Description	Det configuration includes the functions to be called at notification. On one side the application functions are specified and in general it can be decided whether Dlt shall be called at each call of Det.

Included Containers		
Container Name	Multiplicity	Scope / Dependency
DetGeneral	1	Generic configuration parameters of the Det module.
DetNotification	0..1	Configuration of the notification functions.

10.2.3 DetGeneral

SWS Item	**DET002_Conf :**
Container Name	DetGeneral
Description	Generic configuration parameters of the Det module.
Configuration Parameters	

SWS Item	**DET006_Conf :**

Name	DetForwardToDlt {DET_FORWARD_TO_DLT}		
Description	Only if the parameter is present and set to true, the Det requires the Dlt interface and forwards it's call to the function Dlt_DetForwardErrorTrace. In this case the optional interface to Dlt_Det is required.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

SWS Item	**DET003_Conf :**		
Name	DetVersionInfoApi {DET_VERSION_INFO_API}		
Description	Pre-processor switch to enable / disable the API to read out the modules version information. true: Version info API enabled. false: Version info API disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

No Included Containers

10.2.4 DetNotification

SWS Item	**DET004_Conf :**
Container Name	DetNotification
Description	Configuration of the notification functions.
Configuration Parameters	

SWS Item	**DET005_Conf :**		
Name	DetErrorHook {DET_ERROR_HOOK}		
Description	Optional list of functions to be called by the Development Error Tracer in context of each call of Det_ReportError. The type of these functions shall be identical the type of Det_ReportError itself: Std_ReturnType (*f)(uint16, uint8, uint8, uint8).		
Multiplicity	0..*		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

No Included Containers

10.3 Published Information

[DET208] 「The standardized common published parameters as required by BSW00402 in the SRS General on Basic Software Modules [3] shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules [1].」()

Additional module-specific published parameters are listed below if applicable.

11 AUTOSAR Interfaces

11.1 Scope of this Chapter

This chapter defines the AUTOSAR Interfaces of the Development Error Tracer Service (DET).

The definitions in this section are interpreted to be in ARPackage AUTOSAR/Services/DET.

11.2 Overview

The Development Error Tracer BSW module was originally developed for supporting the other BSW modules during development phase. It is however possible to also use the Development Error Tracer to perform error tracing during development of SW-C's.

11.2.1 Use Case

On each ECU, there is one instance of the Development Error Tracer Service and several Atomic Software Component instances named “clients” which interact with the Development Error Tracer Service.

Each component instance may report a development error to the Development Error Tracer Service, and these errors can then be analyzed with debugging tools. The behavior of the Development Error Tracer is not standardized (except the optional forwarding of error reports to the DLT as described in chapter 7.3). Thus, what happens when using DET Service might differ on different ECUs or different Development Error Tracer implementations? The Development Error Tracer might for example do one of the following:

- Set debugger breakpoint within error reporting API
- Count reported errors
- Log calls and passed parameters in RAM buffer
- Send reported errors via communication interface to external logger

Note: For debugging and tracing purpose e.g. in an Error Hook, the use of additional functionalities offered by “Debugging Module” and “Log and Trace Module” should be considered.

11.3 Specification of the Ports and Port Interfaces

This chapter specifies the ports and port interfaces which are needed in order to operate the Development Error Tracer functionality over the VFB.

Each AUTOSAR SW-C which uses the service must contain “service ports” in its own SW-C description which will be typed by the same interfaces and which has to be

connected to the ports of the Development Error Tracer, so that the RTE, the appropriate IDs and the required symbols can be generated.

11.3.1 General Approach

The client-server paradigm is used since more than one parameter has to be transferred.

In order to reuse the C API already defined in the Development Error Tracer BSW module, the Development Error Tracer service uses the same argument names as in the C API, even though the names can not directly be mapped into the SW-C world. "Module ID" can preferably be interpreted as either a component or runnable entity but this is the decision of the implementer of the SW-C.

The Development Error Tracer service needs a "Module ID" as first argument for the C-function.

In order to keep the client code independent from the configuration of number of clients, the "Module IDs" are not passed from the clients to Development Error Tracer but are modeled as "port defined argument values" of the Provide ports on the Development Error Tracer side. As a consequence, the "Module IDs" will not show up as arguments in the operation of the client-server interface. As a further consequence for this approach, there will be separate ports for each "Module ID" both on the client side as well as on the server side.

The Module ID type is of range 0…65535. Values in the range of 0...255 are reserved for Basic Software Modules, all others can be used for application software components.

11.3.2 Data Types

[DET200] ⌈For the port interface of the Development Error Tracer service uint8 and uint16 are required and refer to the AUTOSAR data types. ⌋(BSW04087)

11.3.3 Port Interface

[DET201] ⌈There is only one operation used as service from Development Error Tracer. In C-style, it looks as follows:

```
Std_ReturnType Det_ReportError
(
   uint16 ModuleId,
   uint8  InstanceId,
   uint8  ApiId,
   uint8  ErrorId
```

)⌋(BSW04087)

[DET202] ⌈That operation looks like the following in Port Interface pseudo code.

```
ClientServerInterface DETService
{
```

```
  isService = true;
  return ReportError ( IN uint8 InstanceId,
                       IN uint8 ApiId,
                       IN uint8 ErrorId);
};
```
」(BSW04087)

[DET203] 「Note that the returned values is always true (E_OK), since a Std_ReturnType is required for all services」(BSW04087)

Compared to the API, the "`Det_`" prefix in the name is not required, because the name given here will show up in the XML not globally but as part of an interface description.

Note that the argument InstanceId can not be used for component instance since only the first argument (in this case ModuleId) can be used in "port defined argument value". Instead, multiple instantiations of a module are converted into multiple "Module IDs".

It is up to the implementer of a SW-C to decide about the semantics of the arguments InstanceId, ApiId and ErrorId since these arguments are not standardized by AUTOSAR (at least not above the RTE). However, the ApiId typically corresponds to the operations that can report an error, and ErrorId corresponds to the type of error that is reported. Both ApiId and ErrorId are numbered 0x00..0xFF without specific order.

Figure 6 shows how AUTOSAR Software components (single or multiple instances) are connected via service ports to the DET.

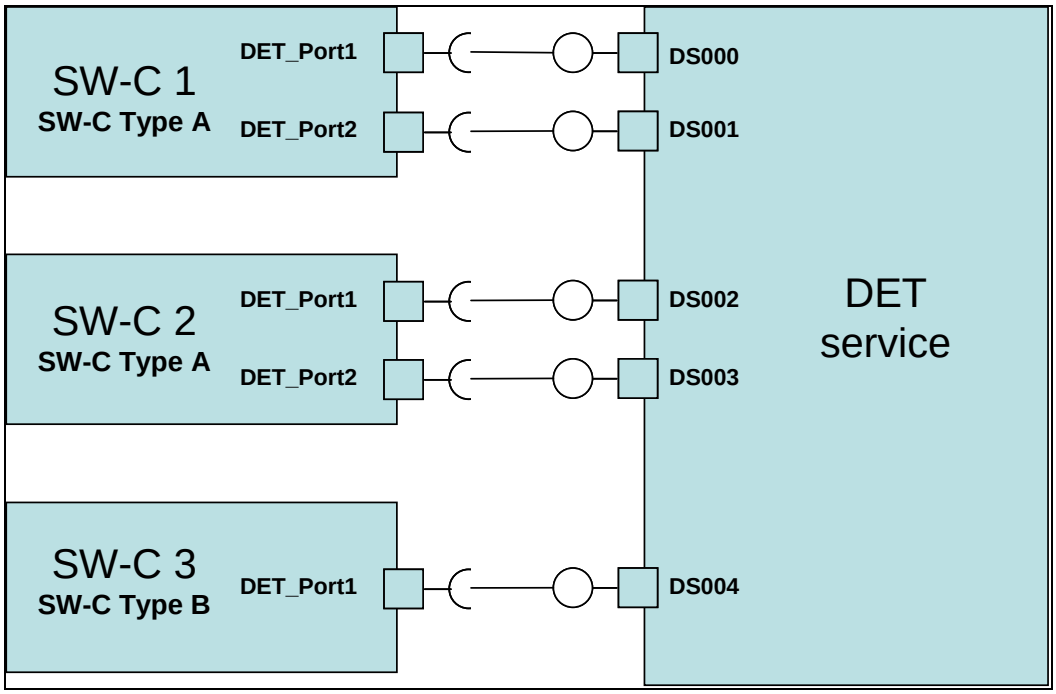

Figure 6: Example of SW-Cs connected to the DET via service ports. On the left side, there are two instances of component SW-C Type A and one instance of component SW-C Type B.

On the DET side, there is one port per "Module ID" providing the service of the interface `DETService` described above. Each client has one port for requiring the

service for each "Module ID" associated with that Software Component (SWC or BSW-module).

Typically one port is created for each SW-C. It is also possible to create a port for each runnable entity inside a SW-C.

There is no naming convention for the ports providing the services. In this example they are named: `DS000, DS001, … , DSnnn`

[DET041] ⌈The port values shall be converted into an `uint16`, with values between `0x1000` and `0xFFFF`, in order to not interfere with BSW Module IDs.⌋()

11.4 Definition of the Service

[DET204] ⌈The Provide Ports have a certain relation to the internal behavior of the DET: With each call, the "Module ID" is passed as an additional argument by the RTE to the C-function which implements the associated runnable entity (feature “port defined argument value”).⌋(BSW04087)

[DET205] ⌈

```
Service DET
{
  // "nnn" below is number of ports using the service
  ProvidePort DETService DS000;
  ...
  ProvidePort DETService DSnnn;

  InternalBehavior
  {
    RunnableEntity ReportError
      symbol "Det_ReportError"
      canbeInvokedConcurrently = TRUE

    DS000.ReportError->ReportError
    PortArgument {port=DS000, value.type=uint16, value.value=0x1000}
      …
    DSnnn.ReportError->ReportError
    PortArgument {port=DSnnn, value.type=uint16,
    value.value=(0x1000 + nnn)}
  };
```

`};`⌋(BSW04087)

11.5 Configuration of the DET

[DET206] ⌈The "Module IDs" of the DET service are modeled as “port defined argument values”. Thus the configuration of those values is part of the RTE configuration. Pre-compile configuration can be done by changing the XML specification for the ports on the client (SW-C) or service (i.e. DET) side.⌋(BSW04087)

12 Not applicable requirements

[DET999] 「These requirements are not applicable to this specification.」(BSW00301, BSW00304, BSW00305, BSW00306, BSW00307, BSW00308, BSW00309, BSW00312, BSW00439, BSW00314, BSW00324, BSW00325, BSW00326, BSW00328, BSW00329, BSW00330, BSW00331, BSW00333, BSW00334, BSW00335, BSW00341, BSW00342, BSW00343, BSW00341, BSW00346, BSW00347, BSW00441, BSW00350, BSW00353, BSW00355, BSW00350, BSW00359, BSW00360, BSW00440, BSW00361, BSW00370, BSW00371, BSW00373, BSW00376, BSW00377, BSW00378, BSW00379, BSW00401, BSW00410, BSW00413, BSW00415, BSW005, BSW006, BSW007, BSW009, BSW010, BSW158, BSW160, BSW161, BSW162, BSW164, BSW172, BSW00344, BSW00404, BSW00405, BSW00345, BSW159, BSW167, BSW170, BSW00380, BSW00419, BSW00381, BSW00412, BSW00383, BSW00387, BSW00388, BSW00389, BSW00390, BSW00391, BSW00392, BSW00393, BSW00394, BSW00395, BSW00396, BSW00397, BSW00398, BSW00399, BSW00400, BSW00438, BSW00375, BSW00416, BSW00406, BSW00437, BSW168, BSW00407, BSW00423, BSW00424, BSW00425, BSW00426, BSW00427, BSW00428, BSW00429, BSW00431, BSW00432, BSW00433, BSW00434, BSW00336, BSW00337, BSW00338, BSW00369, BSW00339, BSW00348, BSW00357, BSW00421, BSW00422, BSW00420, BSW00417, BSW00323, BSW004, BSW00409, BSW00385, BSW00386)