Document Title	Specification of Crypto Service Manager
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	402
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R23-11

Document Change History

Date	Release	Changed by	Change Description
2023-11-23	R23-11	AUTOSAR Release Management	• Add AES Key Wrap support • Remove inconsistency in parameter types and const type modifier • Remove key location check in Csm_KeyCopy() • Editorial changes
2022-11-24	R22-11	AUTOSAR Release Management	• New Feature: Addition of the CsmCustom service for vendor customized security services. • Removal of certificate functionality (moved to KeyM and replaced by CsmCustom services) • Editorial changes
2021-11-25	R21-11	AUTOSAR Release Management	• Harmonize definition of CRYPTO_ALGOMODE between CryptoDrv and Csm • Added key format description in CSM/Crypto Driver for SHE-keys • Added Clarification on seeding and generation of random numbers in the crypto stack • removed superfluous parameter keyId in CsmJobXXX interface operations • Editorial changes

Document Change History

Date	Release	Changed by	Change Description
2020-11-30	R20-11	AUTOSAR Release Management	• New feature: Save and Restore Context for running crypto operation • New feature: KeyGetStatus and KeySetInvalid • Updated Algorithm Families • Support of Multicore • Removing inconsistency in parameter names. • Editorial changes
2019-11-28	R19-11	AUTOSAR Release Management	• Bringing return values of all services and interfaces to one line • added functionality and description of elliptic curves • Callback notification modified • Editorial changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Client-Server-Interfaces Csm<Service>_{Config} • corrected CS interfaces • removal of references to CryptoAbstractionLibrary
2017-12-08	4.3.1	AUTOSAR Release Management	• Added definition for asymmetric key formats • Error fixing and consistency improvements • Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Introduced crypto job concept • Introduced key management concept • Removed Cry_XXX functions from the Csm and introduced two new layers in the crypto stack: Crypto Interface (CryIf) and Crypto Driver (Crypto)

Document Change History

Date	Release	Changed by	Change Description
2015-07-31	4.2.2	AUTOSAR Release Management	• Changed return type from Csm_ReturnType to Std_Types in all API functions • Added detailed description of RTE interfaces • Debugging support marked as obsolete • Error fixing and consistency improvements
2014-10-31	4.2.1	AUTOSAR Release Management	• Obsolete configuration elements removed • Error fixing and consistency improvements • Editorial changes
2014-03-31	4.1.3	AUTOSAR Release Management	• Error fixing and consistency improvements • Editorial changes
2013-10-31	4.1.2	AUTOSAR Release Management	• Error fixing and consistency improvements • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Services for compression/decompression added • Services for key update added (Concept ‘CSM extension’) • Services for symmetric key generation added (Concept ‘CSM extension’) • Service state machine changed to cope with terminated users by releasing of locked resources • Production errors restructured
2011-12-22	4.0.3	AUTOSAR Administration	• Fixed issues with AUTOSAR Port Interfaces

Document Change History

Date	Release	Changed by	Change Description
2010-09-30	3.1.5	AUTOSAR Administration	• Complete Configuration parameters • Complete API specifications • Add support for secure key storage • Integration of support for key transport services • Introduction of new DET error (checking of the null pointer in getversion info).
2010-02-02	3.1.4	AUTOSAR Administration	• Initial release

Disclaimer

This work (specification and/or software implementation) 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 work.
The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.
This work 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 work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.
The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.
The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction and Functional Overview

This specification specifies the functionality, API and the configuration of the software module Crypto Service Manager (CSM) to satisfy the top-level requirements represented in the CSM Requirements Specification (SRS) [CSM_SRS].

The CSM shall provide synchronous or asynchronous services to enable a unique access to basic cryptographic functionalities for all software modules. The CSM shall provide an abstraction layer, which offers a standardized interface to higher software layers to access these functionalities.

The functionality required by a software module can be different to the functionality required by other software modules. For this reason, there shall be the possibility to configure and initialize the services provided by the CSM individually for each software module. This configuration comprises as well the selection of synchronous or asynchronous processing of the CSM services.

The construction of the CSM module follows a generic approach. Wherever a detailed specification of structures and interfaces would limit the scope of the usability of the CSM, interfaces and structures are defined in a generic way. This provides an opportunity for future extensions.

2 Acronyms and Abbreviations

Acronyms and abbreviations, which have a local scope and therefore are not contained in the AUTOSAR glossary [13], are listed in this chapter.

Abbreviation / Acronym:	***Description:***
AEAD	Authenticated Encryption with Associated Data
CDD	Complex Device Driver
CSM	Crypto Service Manager
CRYIF	Crypto Interface
CRYPTO	Crypto Driver
DET	Default Error Tracer
HSM	Hardware Security Module
HW	Hardware
SHE	Security Hardware Extension
SW	Software

2.1 Glossary of Terms

Terms:	***Description:***
Crypto Driver Object	A Crypto Driver implements one or more Crypto Driver Objects. The Crypto Driver Object can offer different crypto primitives in hardware or software. The Crypto Driver Objects of one Crypto Driver are independent of each other. There is only one workspace for each Crypto Driver Object (i.e. only one crypto primitive can be performed at the same time)
Key	A Key can be referenced by a job in the Csm. In the Crypto Driver, the key refers a specific key type.
Key Type	A key type consists of refers to key elements. The key types are typically pre-configured by the vendor of the Crypto Driver.
Key Element	Key elements are used to store data. This data can be e.g. key material or the IV needed for AES encryption. It can also be used to configure the behaviour oft he key management functions. Key elements from different keys have different memory area (both NV and RAM area).
Job	A Job is a configured 'CsmJob'. Among others, it refers to a key, a cryptographic primitive and a reference channel.
Channel	A channel is the path from a Crypto Service Manager queue via the Crypto Interface to a specific Crypto Driver Object.
Primitive	A primitive is an instance of a configured cryptographic algorithm realized in a Crypto Driver Object. Among others it refers to a functionality provided by the CSM to the application, the concrete underlining 'algorithmfamily' (e.g. AES, MD5, RSA, etc.), and a 'algorithmmode' (e.g. ECB, CBC, etc).

Operation	An operation of a crypto primitive declares what part of the crypto primitive shall be performed. There are three different operations:	
	START	Operation indicates a new request of a crypto primitive, it shall cancel all previous requests perform necessary initializations and checks if the crypto primitive can be processed.
	UPDATE	Operation indicates, that the crypto primitive expect input data. An update operation may provide intermediate results.
	FINISH	Operation indicates, that after this part all data are fed completely and the crypto primitive can finalize the calculations. A finish operation may provide final results.
	It is also possible to perform more than one operation at once by concatenating the corresponding bits of the operation_mode argument.	
Priority	The priority of a job defines the importance of it. The higher the priority (as well in value), the more immediate the job will be executed. The priority of a cryptographic job is part of the configuration.	
Processing	Indicates the kind of job processing.	
	Asynchronous	The job is not processed immediately when calling a corresponding function. Usually, the caller is informed via a callback function when the job has been finished.
	Synchronous	The job is processed immediately when calling a corresponding function. When the function returns, a result will be available.
Service	A service shall be understand as defined in the TR_Glossary document: A service is a type of operation that has a published specification of interface and behavior, involving a contract between the provider of the capability and the potential clients.	

3 Related documentation

3.1 Input Documents

[1] List of Basic Software Modules
AUTOSAR_CP_TR_BSWModuleList.pdf

[2] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture.pdf

[3] General Requirements on Basic Software Modules
AUTOSAR_CP_SRS_BSWGeneral.pdf

[4] Specification of RTE Software
AUTOSAR_CP_SWS_RTE.pdf

[5] Specification of BSW Scheduler
AUTOSAR_SWS_Scheduler.pdf

[6] Specification of ECU Configuration
AUTOSAR_CP_TPS_ECUConfiguration.pdf

[7] Specification of Memory Mapping
AUTOSAR_CP_SWS_MemoryMapping.pdf

[8] Specification of Default Error Tracer
AUTOSAR_CP_SWS_DefaultErrorTracer.doc.pdf

[9] Specification of Diagnostic Event Manager
AUTOSAR_CP_SWS_DiagnosticEventManager.pdf

[10] Specification of ECU State Manager
AUTOSAR_CP_SWS_ECUStateManager.pdf

[11] Specification of C Implementation Rules
AUTOSAR_TR_CImplementationRules.pdf

[12] Specification of Standard Types
AUTOSAR_CP_SWS_StandardTypes.pdf

[13] AUTOSAR Glossary
AUTOSAR_FO_TR_Glossary.pdf

[14] Requirements on the Crypto Stack
AUTOSAR_CP_SRS_CryptoStack.pdf

[15] Specification of the Crypto Interface
AUTOSAR_CP_SWS_CryptoInterface.pdf

[16] Specification of the Crypto Driver
AUTOSAR_CP_SWS_CryptoDriver.pdf

[17] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral.pdf

3.2 Related standards and norms

[18] IEC 7498-1 The Basic Model, IEC Norm, 1994

[19] IETF RFC5639 Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation, 2010

[20] IETF RFC6637 Elliptic Curve Cryptography (ECC) in OpenPGP, 2012

[21] IETF RFC 3394 Advanced Encryption Standard (AES) Key Wrap Algorithm, 2002

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software modules (SWS BSW General), which is also valid for Crypto Service Manager.

Thus, the specification SWS BSW General shall be considered as additional and required specification for Crypto Service Manager.

4 Constraints and Assumptions

4.1 Limitations

Some type definitions of CSM start with the Prefix “CRYPTO_” which will violate SRS_BSW_00305. This will be harmonized in release 4.3.1. Nevertheless due to the constraint [constr_1050] part 1 the ports are still consider to be compatible.

4.2 Applicability to Car Domains

n.a.

4.3 Security Implications

There is no user management in place, which prevents non-authorized access on any of CSM’s services. This means, that if any access protection is needed such must be implemented by the application and the served (by CSM) cryptographic library modules; access protection is not target of the CSM.

5 Dependencies to other Modules

[SWS_Csm_00001] ⌈The CSM shall be able to access the cryptographic interface (CRYIF), which is implemented according to the cryptographic interface specification.
⌋(SRS_CryptoStack_00082)

[SWS_Csm_00506] ⌈The CSM module shall use the interfaces of the CRYIF with the underlying Crypto Drivers (CRYPTO) to calculate the result of a cryptographic service.
⌋(SRS_CryptoStack_00082)

The incorporated cryptographic library modules or hardware extensions of the Crypto Driver provide the cryptographic routines, e.g. SHA-1, RSA, AES, Diffie-Hellman key-exchange, etc.

6 Requirements Traceability

Requirement	Description	Satisfied by
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_Csm_00646
SRS_BSW_00358	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	SWS_Csm_00646
SRS_BSW_00359	All AUTOSAR Basic Software Modules callback functions shall avoid return types other than void if possible	SWS_Csm_00970, SWS_Csm_00971
SRS_BSW_00360	AUTOSAR Basic Software Modules callback functions are allowed to have parameters	SWS_Csm_00970, SWS_Csm_00971
SRS_BSW_00373	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	SWS_Csm_00479
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_Csm_00705
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_Csm_00646
SRS_BSW_00432	Modules should have separate main processing functions for read/receive and write/transmit data path	SWS_Csm_00479
SRS_CryptoStack_00008	The Crypto Stack shall allow static configuration of keys used for cryptographic jobs	SWS_Csm_00951, SWS_Csm_00953, SWS_Csm_01012, SWS_Csm_01092
SRS_CryptoStack_00009	The Crypto Stack shall support reentrancy for all crypto services	SWS_Csm_00022
SRS_CryptoStack_00010	The Crypto Stack shall conceal symmetric keys from the users of crypto services	SWS_Csm_00959
SRS_CryptoStack_00011	The Crypto Stack shall conceal asymmetric private keys from the users of Crypto services	SWS_Csm_00959
SRS_CryptoStack_00019	The Crypto Stack shall identify random number generation as	SWS_Csm_01543

	a cryptographic primitive which can be requested to a driver	
SRS_CryptoStack_00020	The Crypto Stack shall identify symmetric encryption/decryption as a cryptographic primitive which can be requested to a driver	SWS_Csm_00984, SWS_Csm_00989
SRS_CryptoStack_00021	The Crypto Stack shall identify asymmetric encryption/decryption as a cryptographic primitive which can be requested to a driver	SWS_Csm_00984, SWS_Csm_00989
SRS_CryptoStack_00022	The Crypto Stack shall identify MAC generation/verification as a cryptographic primitive which can be requested to a driver	SWS_Csm_00982
SRS_CryptoStack_00023	The Crypto Stack shall identify asymmetric signature generation/verification as a cryptographic primitive which can be requested to a driver	SWS_Csm_00992, SWS_Csm_00996
SRS_CryptoStack_00024	The Crypto Stack shall identify hash calculation as a cryptographic primitive which can be requested to a driver	SWS_Csm_00980
SRS_CryptoStack_00026	The Crypto Stack shall provide an interface for the generation of asymmetric keys	SWS_Csm_00955
SRS_CryptoStack_00027	The Crypto Stack shall provide an interface for the generation of symmetric keys	SWS_Csm_00955
SRS_CryptoStack_00082	The CSM module specification shall specify the interface and behavior of the callback function, if the asynchronous job processing mode is selected	SWS_Csm_00001, SWS_Csm_00032, SWS_Csm_00506
SRS_CryptoStack_00084	The CSM module shall use the streaming approach for some selected services	SWS_Csm_01039
SRS_CryptoStack_00086	The CSM module shall distinguish between error types	SWS_Csm_01089, SWS_Csm_91004
SRS_CryptoStack_00087	The CSM module shall report detected development errors to the Default Error Tracer	SWS_Csm_01088, SWS_Csm_01091
SRS_CryptoStack_00090	The CSM shall provide an interface to be accessible via the RTE	SWS_Csm_00802, SWS_Csm_00803, SWS_Csm_00902, SWS_Csm_00903, SWS_Csm_00912, SWS_Csm_00922, SWS_Csm_00923, SWS_Csm_00927, SWS_Csm_00928, SWS_Csm_00930,

		SWS_Csm_00934, SWS_Csm_00935, SWS_Csm_00936, SWS_Csm_00943, SWS_Csm_00946, SWS_Csm_01042, SWS_Csm_01074, SWS_Csm_01075, SWS_Csm_01077, SWS_Csm_01078, SWS_Csm_01079, SWS_Csm_01906, SWS_Csm_01910, SWS_Csm_01915, SWS_Csm_01920, SWS_Csm_01921, SWS_Csm_01922, SWS_Csm_01923, SWS_Csm_01924, SWS_Csm_01925, SWS_Csm_01926, SWS_Csm_01927, SWS_Csm_01928, SWS_Csm_09000, SWS_Csm_91023, SWS_Csm_91045, SWS_Csm_91046, SWS_Csm_91051, SWS_Csm_91052, SWS_Csm_91053, SWS_Csm_91054, SWS_Csm_91055, SWS_Csm_91056, SWS_Csm_91057, SWS_Csm_91058, SWS_Csm_91059, SWS_Csm_91060, SWS_Csm_91062, SWS_Csm_91105
SRS_CryptoStack_00091	The CSM shall provide one Provide--Port for each configuration	SWS_Csm_00934, SWS_Csm_01042, SWS_Csm_91023, SWS_Csm_91062
SRS_CryptoStack_00095	The Crypto Driver module shall strictly separate error and status information	SWS_Csm_91043, SWS_Csm_91044
SRS_CryptoStack_00100	Synchronous Job Processing	SWS_Csm_01049
SRS_CryptoStack_00101	Asynchronous Job Processing	SWS_Csm_01049
SRS_CryptoStack_00103	The Crypto Stack shall provide an interface for the derivation of symmetric keys	SWS_Csm_00956
SRS_CryptoStack_00906	-	SWS_Csm_00947
SRS_CryptoStack_01076	-	SWS_Csm_01083
SRS_CrytptoStack_00028	-	SWS_Csm_00966, SWS_Csm_00967
SRS_CrytptoStack_00029	-	SWS_Csm_00959
SRS_Csm_00066	-	SWS_Csm_01905
SWS_BSW_00050	Check parameters passed to Initialization functions	SWS_Csm_00186
SWS_BSW_00216	-	SWS_Csm_01085

7 Functional specification

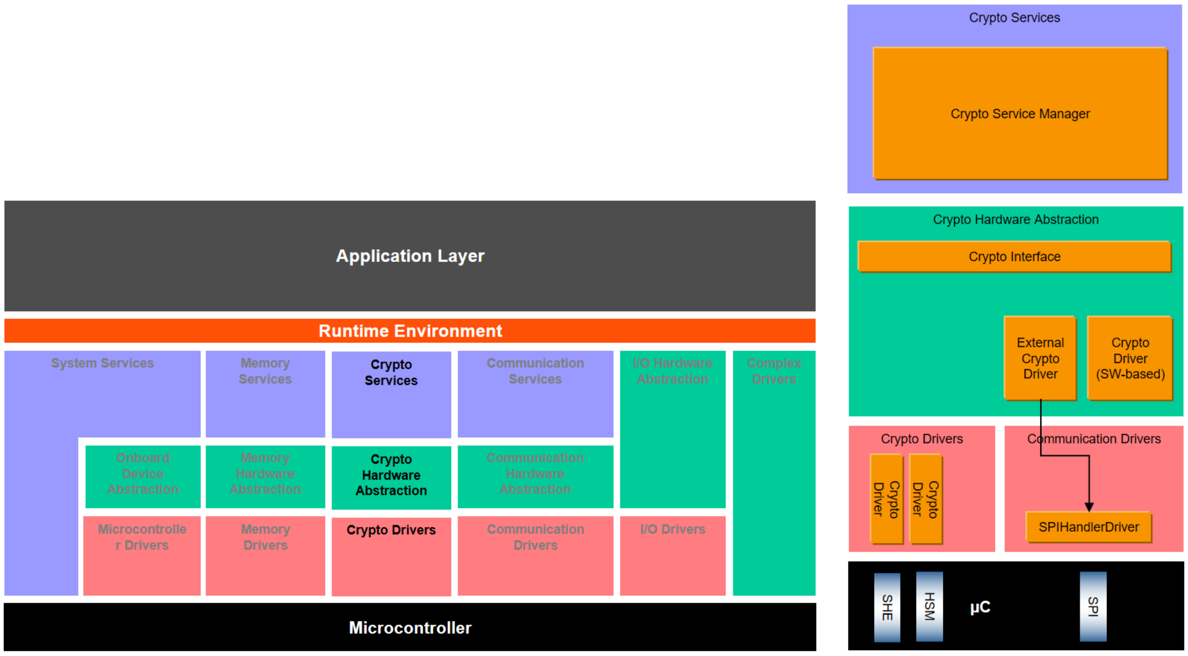


Figure 7-1 AUTOSAR Layered View with CSM

7.1 Basic Architecture Guidelines

The starting point for the description of the design of the CSM module is the AUTOSAR Layered Software Architecture (see Figure 7-1). The description of the CSM module architecture on the basis of the AUTOSAR layered software architecture shall help to understand the specification of interfaces and functionalities of the CSM module in the following sections.

The architecture of AUTOSAR consists of several layers which can be seen in Figure 7-1. The Service Layer is the highest layer of the Basic Software. Its task is to provide basic services for application and basic software modules, i.e. it offers the most relevant functionalities for application software and basic software modules.

CSM is a service that provides cryptography functionality, based on a crypto driver which relies on a software library or on a hardware module. Also, mixed setups with multiple crypto drivers are possible. The CSM accesses the different CryptoDrivers over the CRYIF.

The CSM, as a service layer, provides the interface for SW-C or BSW for cryptographic operations. The main task of the CSM is to schedule and prioritize services and to call the crypto interface (CryIf) for further operation. The CryIf schedules the requests to the crypto driver and its crypto driver object that was statically assigned to this service.
The CSM uses a static configuration of primitives (CsmPrimitives) to define a cryptographic operation. Such a primitive is then assigned to a job configuration (CsmJob) that determines further attributes like priority, asynchronous or synchronous execution and what key shall be used for the operation. It should be

noted that the key is always located in the crypto driver itself and the CSM uses only a reference to it.
The separation of the keys and primitives allows to separate the API for the cryptographic operation and the key management. This allows to let an application concentrate on the required cryptographic operation like MAC calculation and verification whereas a key manager provides the keys during a configuration setup.
The API of the CSM can roughly be divided into two categories: a direct API (mainly for key management) and a job-based API (mainly for cryptographic operations) (see Figure 7-2) [1]. The Direct API is configured through key elements, whereas the job-based API is configured in the job configuration and its associated CsmPrimitive . Job-based API functions will ignore API parameters that have an equivalent in the Crypto_JobType job structure. The direct API has a direct correspondence of the functions in the CryIf and the Crypto Driver. These functions can only be called synchronously. The CSM will pass the parameters from the application directly to the corresponding CryIf function call. The job-based API uses a job structure, the Crypto_JobType, that contains static and dynamic parameters and references to structures to provide all necessary information to the crypto driver to perform that job (see Figure 7-3). Every service that uses a job will use this structure. All necessary parameter for a service will be packed into the elements of the structure by the CSM and will then call the CryIf and this, in turn, will call the configured Crypto Driver.

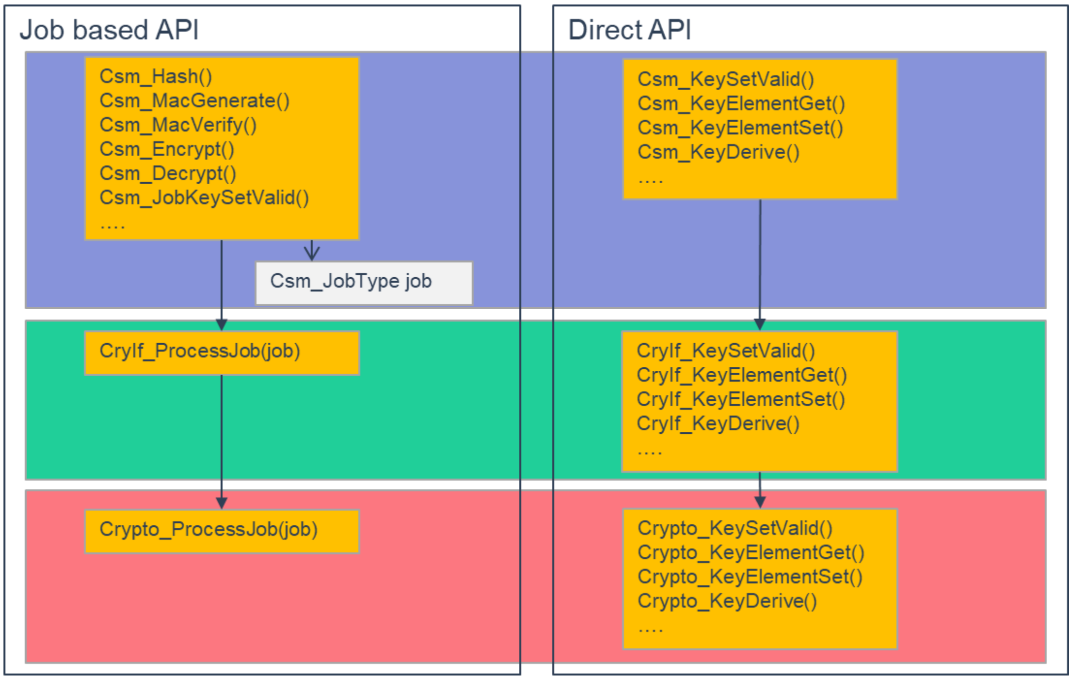


Figure 7-2 API call tree for CSM, CryIf and Crypto. Divided into job-based API and Direct API

A job can run synchronously or asynchronously depending on the static configuration. The parameters for crypto service info, crypto algorithm family and mode determine the exact cryptographic algorithm that shall be performed in the crypto driver.

[1] Historically, there are a few functions with direct synchronous API and a job based API, because the need for asynchronous execution was recognized afterwards.

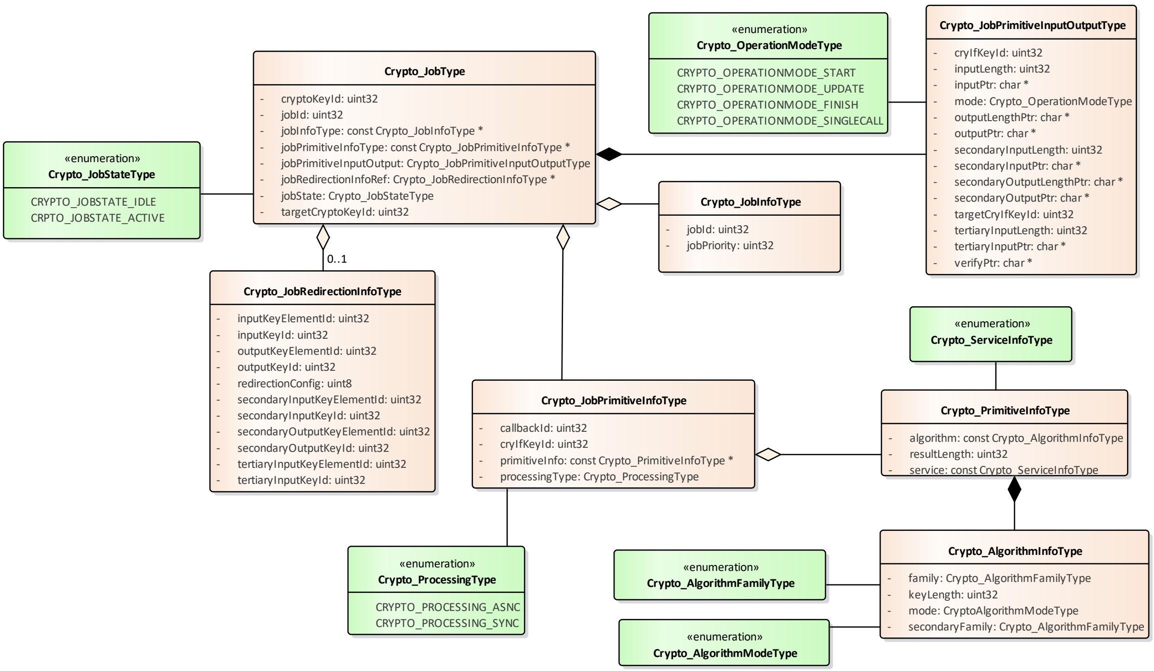

«enumeration»
Crypto_OperationModeType
CRYPTO_OPERATIONMODE_START
CRYPTO_OPERATIONMODE_UPDATE
CRYPTO_OPERATIONMODE_FINISH
CRYPTO_OPERATIONMODE_SINGLECALL
Crypto_JobPrimitiveInputOutputType
- cryIfKeyId: uint32
- inputLength: uint32
- inputPtr: char *
- mode: Crypto_OperationModeType
- outputLengthPtr: char *
- outputPtr: char *
- secondaryInputLength: uint32
- secondaryInputPtr: char *
- secondaryOutputLengthPtr: char *
- secondaryOutputPtr: char *
- targetCryIfKeyId: uint32
- tertiaryInputLength: uint32
- tertiaryInputPtr: char *
- verifyPtr: char *
Crypto_JobType
- cryptoKeyId: uint32
- jobId: uint32
- jobInfoType: const Crypto_JobInfoType *
- jobPrimitiveInfoType: const Crypto_JobPrimitiveInfoType *
- jobPrimitiveInputOutput: Crypto_JobPrimitiveInputOutputType
- jobRedirectionInfoRef: Crypto_JobRedirectionInfoType *
- jobState: Crypto_JobStateType
- targetCryptoKeyId: uint32
«enumeration»
Crypto_JobStateType
CRYPTO_JOBSTATE_IDLE
CRPTO_JOBSTATE_ACTIVE
Crypto_JobInfoType
- jobId: uint32
- jobPriority: uint32
0..1
Crypto_JobRedirectionInfoType
- inputKeyElementId: uint32
- inputKeyId: uint32
- outputKeyElementId: uint32
- outputKeyId: uint32
- redirectionConfig: uint8
- secondaryInputKeyElementId: uint32
- secondaryInputKeyId: uint32
- secondaryOutputKeyElementId: uint32
- secondaryOutputKeyId: uint32
- tertiaryInputKeyElementId: uint32
- tertiaryInputKeyId: uint32
«enumeration»
Crypto_ServiceInfoType
Crypto_JobPrimitiveInfoType
- callbackId: uint32
- cryIfKeyId: uint32
- primitiveInfo: const Crypto_PrimitiveInfoType *
- processingType: Crypto_ProcessingType
Crypto_PrimitiveInfoType
- algorithm: const Crypto_AlgorithmInfoType
- resultLength: uint32
- service: const Crypto_ServiceInfoType
«enumeration»
Crypto_ProcessingType
CRYPTO_PROCESSING_ASNC
CRYPTO_PROCESSING_SYNC
«enumeration»
Crypto_AlgorithmFamilyType
«enumeration»
Crypto_AlgorithmModeType
Crypto_AlgorithmInfoType
- family: Crypto_AlgorithmFamilyType
- keyLength: uint32
- mode: CryptoAlgorithmModeType
- secondaryFamily: Crypto_AlgorithmFamilyType

Figure 7-3 Structure of Crypto_JobType (Job) and its dependencies

⌋

[SWS_Csm_00942] ⌈ If any of the SWS items *Csm[**CsmPrimitives**]AlgorithmFamily*, *Csm[**CsmPrimitives**]AlgorithmMode* and/or *Csm[**CsmPrimitives**]SecondaryAlgorithmFamily* in the container *Csm[**CsmPrimitives**]Config* is set to CRYPTO_ALGOFAM_CUSTOM (0xFF) resp. CRYPTO_ALGOMODE_CUSTOM (0xFF), then the value of the reference to *CryptoPrimitiveAlgorithmFamilyCustom/CryptoPrimitiveAlgorithmFamilyCustomId* and/or *CryptoPrimitiveAlgorithmModeCustom/CryptoPrimitiveAlgorithmModeCustomId* defined in the Crypto Driver shall be set to either of the fields “family”, “mode” and/or “secondaryFamily” in Crypto_AlgorithmInfoType (instead of the “CUSTOM” value 0xff itself).
⌋()

7.2 General Behavior

[SWS_Csm_00941] ⌈A job is an instance of a configurated cryptographic primitive.
⌋()

[SWS_Csm_00016] ⌈ For each job just one instance shall be processed by CSM at a time.
⌋()

[SWS_Csm_00022] ⌈The CSM module shall allow parallel processing of different jobs.

⌋(SRS_CryptoStack_00009)

[SWS_Csm_00017] ⌈If a service of the CSM module is requested and the corresponding job is in "ACTIVE" state, the job request shall call `CryIf_ProcessJob()` and pass on the return value.
⌋()

[SWS_Csm_00018] ⌈If a service of the CSM module is requested, and the CSM job needs to be queued and the queue is full, the job request shall be rejected with the return value `CRYPTO_E_BUSY`.
⌋()

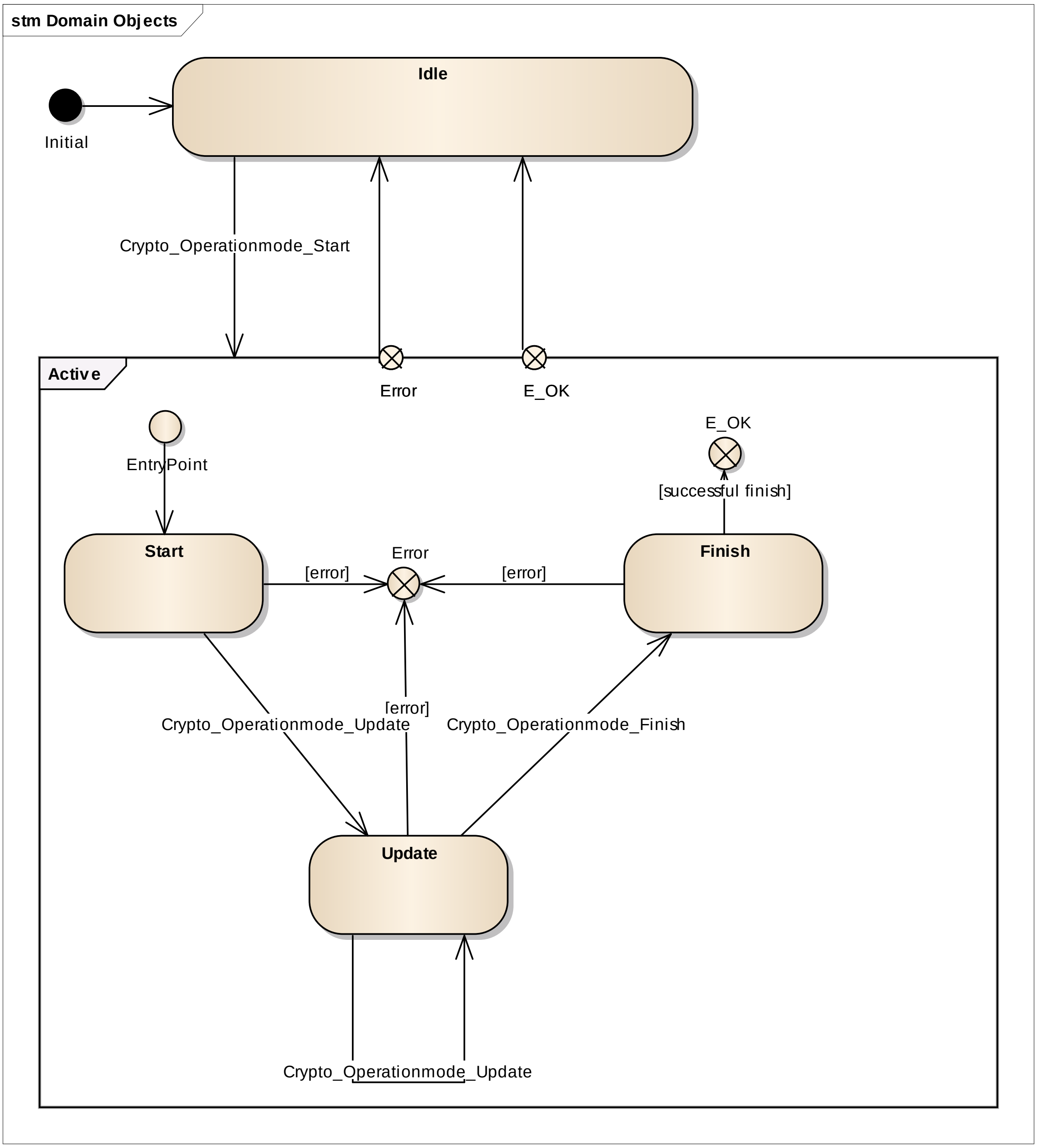
[SWS_Csm_00019] ⌈If an asynchronous interface is configured, the CSM module shall provide a main function Csm_MainFunction() which is called cyclically to control processing of the jobs via a state machine.
⌋()

7.2.1 **Normal Operation**

[SWS_Csm_01039] ⌈To unite a single call function and the streaming approach for the crypto services, there is the `mode` parameter, which determines the operation mode. This service operation is a flag field, indicating the operation mode “START”, “UPDATE” or “FINISH”. It declares explicitly what operation shall be performed. These operation modes can be mixed, and execute multiple operations at once.
The diagram in **SWS_Csm_00024** shows the state machine of a job of this design.
⌋(SRS_CryptoStack_00084)

Note: The actual transaction of the states is made in the layer, which works with these states, i.e. in the Crypto Driver.

[SWS_Csm_00024] ⌈



⌋()

[SWS_Csm_01033]⌈The CSM crypto services shall support to process multiple operation mode inputs with a single call.
⌋()

[SWS_Csm_01045]⌈If the `CRYPTO_OPERATIONMODE_START` and `CRYPTO_OPERATIONMODE_FINISH` bits are set and the `CRYPTO_OPERATIONMODE_UPDATE` is not set, the `Csm_<Service>()` function shall return with `E_NOT_OK`.
⌋()

Note: The coherent single call approach could improve the performance due to less overhead. Instead of calling the explicit API multiple times, only one call is necessary. This approach is intended to be used with small data input, which demand fast processing.
While operating with the streaming approach (“Start”, “Update”, “Finish”) the dedicated Crypto Driver Object is waiting for further input (“Update”) until the “Finish” state has been reached. No other job could be processed on this Crypto Driver instance meanwhile.Functional Requirements

7.2.1.1 Configuration

[SWS_Csm_91005] ⌈Each crypto primitive configuration shall be realized as a constant structure of type `Crypto_PrimitiveInfoType`.

⌋()

[SWS_Csm_91006] ⌈Each job primitive configuration shall be realized as a constant structure of type `Crypto_JobPrimitiveInfoType`.
⌋()

[SWS_Csm_00028] ⌈It shall be possible to create several configurations for each cryptographic primitive.
⌋()
One configuration per job per primitive is possible.

[SWS_Csm_00029] ⌈When creating a primitive configuration, it shall be possible to configure all available and allowed schemes from the underlying Crypto Driver Object.
⌋()
[SWS_Csm_00032] ⌈If the asynchronous interface is chosen, each job primitive configuration shall contain a callback function.
⌋(SRS_CryptoStack_00082)

7.2.1.2 Synchronous Job Processing

[SWS_Csm_00035] ⌈When the synchronous interface is used, the interface functions shall immediately compute the result with the help of the underlying Crypto Stack modules.
⌋()

[SWS_Csm_00037] ⌈ If a synchronous job is issued and the priority is greater than the highest priority available in the queue, the CSM shall disable processing new jobs from the queue until the next call of the main function has finished that follows after completion of the currently processed job.
⌋()

Note:
Channels may hold jobs of both asynchronous and synchronous processing type. If

so, a synchronous job might not be accepted for processing although its job's priority is higher than those of all asynchronous jobs.

Note:
As the underlying Crypto Driver can have its own queue, it can not always be ensured that the highest priority job provided by the application is processed next.

[SWS_Csm_91007] ⌈ If a synchronous job is issued and the priority is less than the highest priority available in the queue, the CSM shall return `CRYPTO_E_BUSY`.
⌋()

Note:
By pausing calls to the CSM main function with e.g. critical sections during calling the synchronous jobs, it can be ensured, that synchronous jobs can be processed in a row without having to wait for asynchronous jobs in between if they have a high enough priority. Also consider disabling queueing in the Crypto Driver Object to ensure fast processing of synchronous jobs.
If the loading of asynchronous jobs from the queue shall not be paused by synchronous jobs, the priorities of the synchronous jobs have to be smaller than the asynchronous jobs.

7.2.1.3 Asynchronous Job Processing

[SWS_Csm_00036] ⌈If the asynchronous interface is used, the interface functions shall only hand over the necessary information to the underlying Crypto Stack modules.
⌋()

[SWS_Csm_00039] ⌈The users of the CSM shall be notified when a requested cryptographic service has been processed by calling the callback function from the job primitive configuration.
⌋()

7.2.2 Design Notes

The CSM provides two services: (1) the crypto services itself and (2) key management.

7.2.2.1 CSM module startup

The `Csm_Init()` request shall not be responsible to trigger the initialization of the underlying CRYIF. It is assumed, that the underlying CRYIF will be initialized by any appropriate entity (e.g. BswM).

Software components, which are using the CSM module, shall be responsible for checking global error and status information resulting from the CSM module startup.

7.2.2.2 Crypto Services

7.2.2.2.1 Usage of the CSM crypto services

[SWS_Csm_00734]⌈CSM crypto services shall provide a `Csm_<Service>()` API.
⌋()

[SWS_Csm_00924]⌈The application shall be able to call `Csm_<Service>()` with the operation mode `CRYPTO_OPERATIONMODE_START` to initialize cryptographic computations.
⌋()

[SWS_Csm_00925]⌈The application shall be able to call Csm_<Service>() with the operation mode CRYPTO_OPERATIONMODE_UPDATE arbitrary often, but at least one time, to feed the job's crypto primitive with input data.
⌋()

[SWS_Csm_01046] ⌈The application shall be able to call `Csm_<Service>()` with the operation mode `CRYPTO_OPERATIONMODE_FINISH` to finalize cryptographic computations.
⌋()

[SWS_Csm_01055] ⌈ Only the service operations `HASH, MACGENERATE, MACVERIFY, ENCRYPT, DECRYPT, AEAD_ENCRYPT, AEAD_DECRYPT, SIGNATUREGENERATE, SIGNATUREVERIFY` shall support the operation mode `START, UPDATE` and `FINISH` as specified from the API. For all other service operations, the CSM shall set the operation mode to `CRYPTO_OPERATIONMODE_SINGLECALL`, even if the API does not provide an operation mode.
⌋()

Note:

The Csm_<Service>() will call the CryIf_ProcessJob() with a pointer to Crypto_JobType, where all the necessary information are stored to process the job.

Part of this Crypto_JobType is a Crypto_JobPrimitiveInputOutputType, where all the information about the input and output parameters depending of the service are stored. A definition of the mapping from the API parameters of Csm_<Service>() to the parameters of Crypto_JobPrimitiveInputOutputType, can be found in [SWS_Crypto_00073] of the Crypto Driver specification.

[SWS_Csm_01093] ⌈If the CSM issues either the service `CRYPTO_MACGENERATE, CRYPTO_MACVERIFY, CRYPTO_ENCRYPT, CRYPTO_DECRYPT, CRYPTO_AEADENCRYPT, CRYPTO_AEADDECRYPT, CRYPTO_RANDOMGENERATE, CRYPTO_SIGNATUREGENERATE` or `CRYPTO_SIGNATUREVERIFY` to the Crypto Interface, it needs to make sure that the element `jobPrimitiveInfo->cryIfKeyId` in the job structure of `Crypto_JobType` references to the assigned key of this job.

⌋()

Note: The CryIf is responsible to transform this ID to the corresponding key ID of the respective crypto driver.

[SWS_Csm_01094] ⌈If one of the primitive services `CRYPTO_KEYSETVALID`, `CRYPTO_KEYSETINVALID`, `CRYPTO_RANDOMSEED`, `CRYPTO_KEYGENERATE`, `CRYPTO_KEYDERIVE`, `CRYPTO_KEYEXCHANGECALCPUBVAL`, `CRYPTO_KEYEXCHANGECALCSECRET` or `CRYPTO_CUSTOM` are to be executed, the CSM shall fill in the elements of the structure `Csm_JobType->jobPrimitiveInputOutput->cryIfKeyId` and, if applicable, `Csm_JobType->jobPrimitiveInputOutput->targetCryIfKeyId` with the corresponding CryIf key ID.
⌋()

Note: The CryIf is responsible to transform these IDs to the corresponding key IDs of the respective crypto driver.

7.2.2.2.2 Queuing

The CSM may have several queues, where the jobs are lining up depending on their priority, to process multiple cryptographic requests. The path from a CSM queue via the CryIf to a Crypto Driver Object is called a *channel*. Each queue of the CSM is mapped to one channel to access the crypto primitives of the Crypto Driver Object. The size of the queue is configurable.
To optimize the hardware usage of the Crypto Driver Object, there is optionally a queue in Crypto Driver, too.

A Crypto Driver Object represents an instance of an independent crypto “device” (hardware or software, e.g. AES accelerator). There could be a channel for fast AES and CMAC calculations on an HSM for jobs with high priority, which ends on a native AES calculation service in the Crypto Driver. But it is also possible, that a Crypto Driver Object is a piece of software, e.g. for RSA calculations where users are able to encrypt, decrypt, sign or verify data.

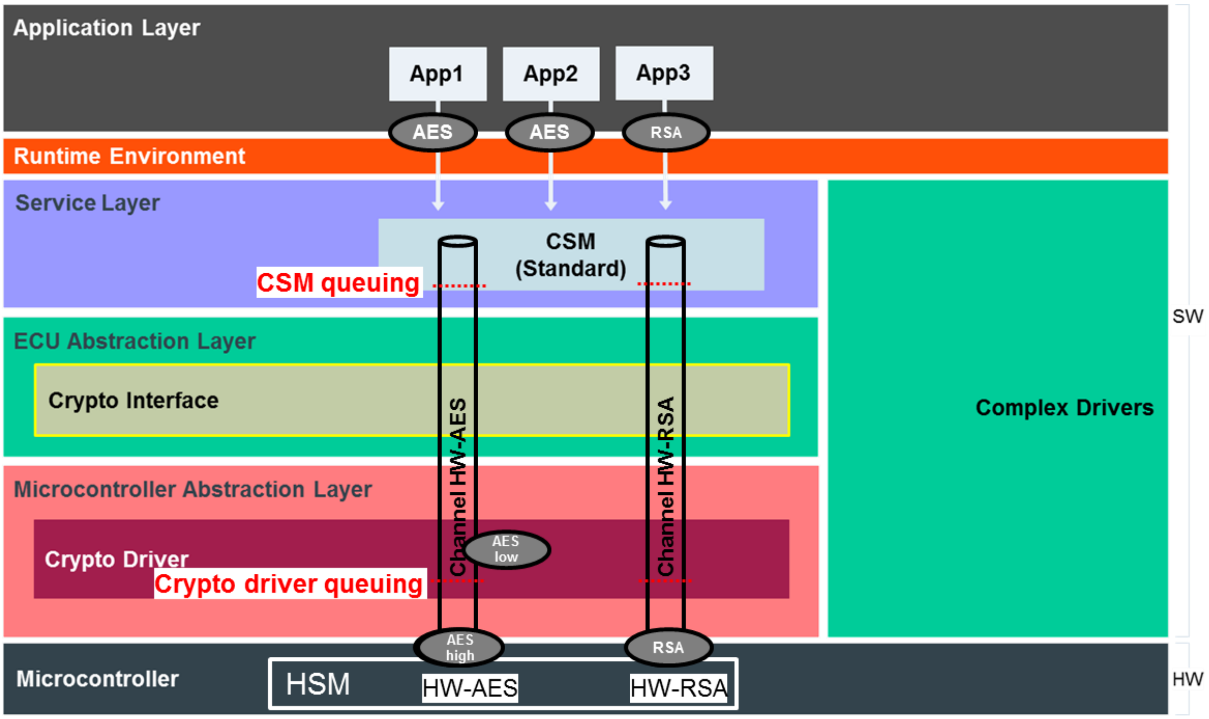


Figure 7-4 AUTOSAR Layered View with channels

Figure 7-4 illustrates an AUTOSAR Layered View with channels. In this example, there is a HSM with two Crypto Driver Objects (HW-AES and HW-RSA), each of them has an own channel. Each channel is connected to a CSM queue and a Crypto Driver Object queue.
In this case, both Crypto Driver Objects are processing a crypto job (AES-high and RSA) each, while the queue of the Crypto Driver Object contains one more job (AES-low). If the HW-AES of the HSM finished the AES-high job, AES-low job will be processed as next one.

Other scenarios with the same setup (without jobs in process or in queues) can be derived as follows:
It will be assumed, that a new job of an application calls RSA.

- If the Crypto Driver Object of the RSA is not busy, the job will be processed immediately.
- If the Crypto Driver Object of the RSA is busy, but the queue of the Crypto Driver Object is not full, the job will be listed into that queue in order of its priority. As soon as the Crypto Driver Object is free, the job with the highest priority from the Crypto Driver Object queue will be executed.
- If the Crypto Driver Object of the RSA is busy and the queue of the Crypto Driver Object is full, the job will be stored in the CSM queue in order of its priority.
- If the Crypto Driver Object of the RSA is busy and the queue of the Crypto Driver Object as well as the CSM queue are full, the CSM rejects the request.
- If the Crypto Driver Object of the RSA is active, the job is already started in the Crypto Driver and is waiting for either more data to process or the finish command.

[SWS_Csm_00940] ⌈It shall be possible to queue CSM jobs in configured `CsmQueues` in the CSM.
⌋()

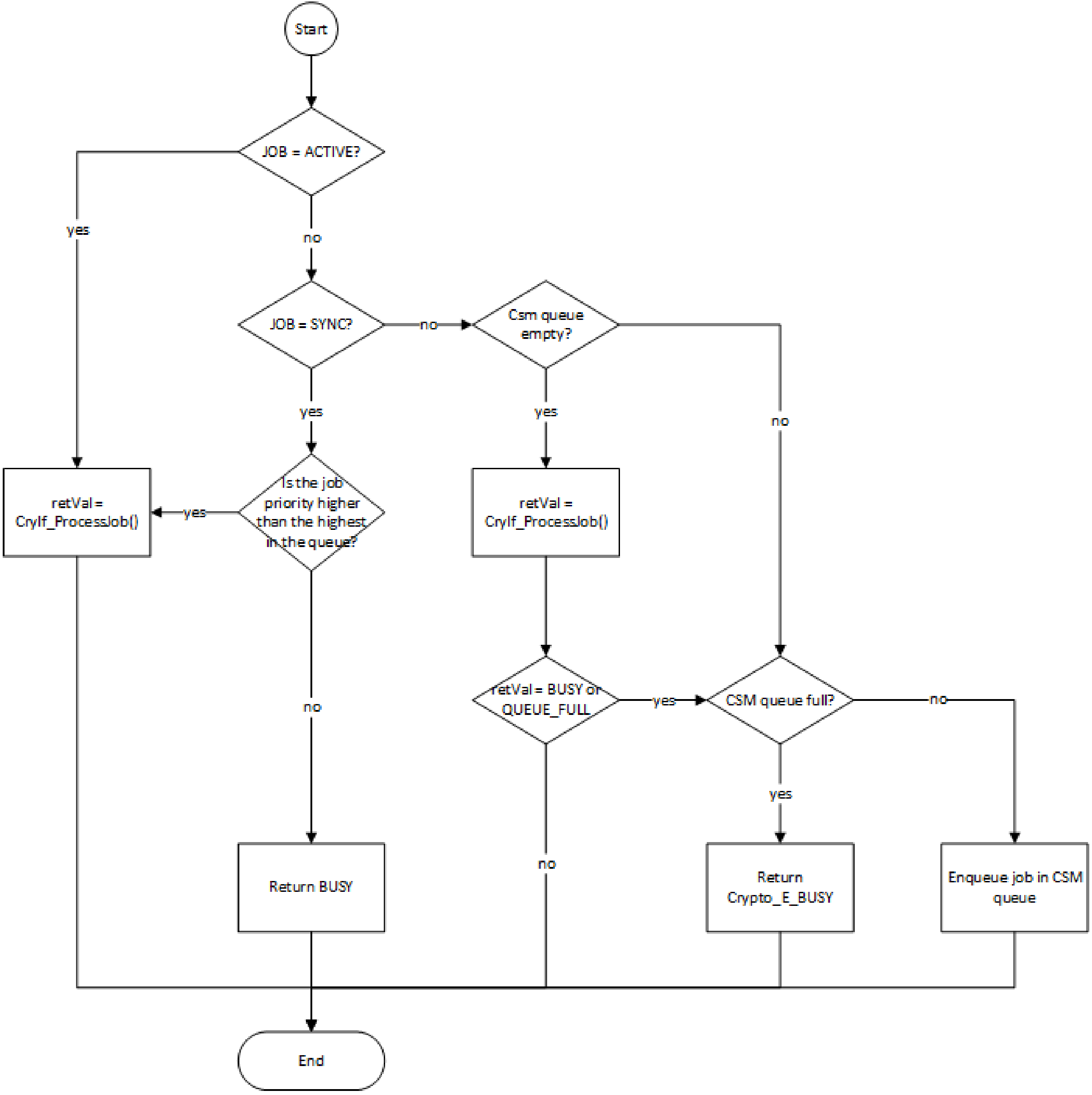
[SWS_Csm_00944] ⌈The CsmQueues shall sort the jobs according to the configured job’s priority.
⌋()
The higher the job priority value, the higher the job’s priority.

[SWS_Csm_91072] ⌈ A service operation shall only be added to the queue if the data consistency of the job structure can be guaranteed. This shall be particularly considered when services with the same jobID are added to the queue (e.g. with subsequent calls to `Csm_SignatureVerify()` and `Csm_SaveContextJob()`). If this cannot be guaranteed, the service operation shall return with `E_BUSY`.
⌋()

[SWS_Csm_91073] ⌈ ⌈ If services with the same JobID can be added to the queue, then the order of execution of these services shall correspond to the order if incoming services operation requests (“First-In-First-Out”).
⌋()

[SWS_Csm_91074] ⌈The `Csm_<Service>()` function shall behave as shown in diagram SWS_Csm_01041.
⌋()

[SWS_Csm_01041]⌈



⌋()

Synchronous job processing and queuing might not be useful. So, if synchronous job processing is chosen, the queue sizes should be “0”. However, it is also possible to use channels (including queues) with synchronous and asynchronous jobs.

The queued jobs can be passed to the CRYIF in the `Csm_MainFunction()`.

If the job has the state “active” the CSM shall assume, that the mapped cryptographic driver instance is currently processing this job and the caller wants to continue with the operation (e.g. feeding more data using “update”). The plausibility check has to be performed in the cryptographic driver instance.

7.2.2.3 Key Management

[SWS_Csm_00950] ⌈Services belonging to the key management shall provide the `Csm_<Service>()` function, only.
⌋()

[SWS_Csm_00954] ⌈A key consists of one or more key elements.
⌋()
Examples of key elements are the key material itself, an initialization vector, a seed for random number generation, or the proof of the SHE standard.

Keys, i.e. the corresponding key IDs have symbolic names given by the configuration. The Crypto Stack API uses the following key element index definition from the CSM module:

[SWS_Csm_01022] ⌈

Crypto Service:	*key element:*	*key element Name:*	*key element ID:*
MAC	Key Material	`CRYPTO_KE_MAC_KEY`	1
	Proof (SHE)	`CRYPTO_KE_MAC_PROOF`	2
	Seed	`CRYPTO_KE_KEYGENERATE_SEED`	16
Signature	Key Material	`CRYPTO_KE_SIGNATURE_KEY`	1
	ECC curve type	`CRYPTO_KE_SIGNATURE_CURVETYPE`	29
Random	Seed State	`CRYPTO_KE_RANDOM_SEED_STATE`	3
	Algorithm	`CRYPTO_KE_RANDOM_ALGORITHM`	4
Cipher/AEAD	Key Material	`CRYPTO_KE_CIPHER_KEY`	1
	Proof (SHE)	`CRYPTO_KE_CIPHER_PROOF`	2
	Init Vector	`CRYPTO_KE_CIPHER_IV`	5
	2nd Key Material	`CRYPTO_KE_CIPHER_2NDKEY`	7
Key Exchange	Base	`CRYPTO_KE_KEYEXCHANGE_BASE`	8
	Private Key	`CRYPTO_KE_KEYEXCHANGE_PRIVKEY`	9
	Own Public Key	`CRYPTO_KE_KEYEXCHANGE_OWNPUBKEY`	10
	Shared Value	`CRYPTO_KE_KEYEXCHANGE_SHAREDVALUE`	1
	Algorithm	`CRYPTO_KE_KEYEXCHANGE_ALGORITHM`	12
	ECC curve type	`CRYPTO_KE_KEYEXCHANGE_CURVETYPE`	29
Key Derivation	Password	`CRYPTO_KE_KEYDERIVATION_PASSWORD`	1
	Salt	`CRYPTO_KE_KEYDERIVATION_SALT`	13
	Iterations	`CRYPTO_KE_KEYDERIVATION_ITERATIONS`	14

	Algorithm	`CRYPTO_KE_KEYDERIVATION_ALGORITHM`	15
	ECC curve type	`CRYPTO_KE_KEYDERIVATION_CURVETYPE`	29
Key Generate	Key Material	`CRYPTO_KE_KEYGENERATE_KEY`	1
	Seed	`CRYPTO_KE_KEYGENERATE_SEED`	16
	Algorithm	`CRYPTO_KE_KEYGENERATE_ALGORITHM`	17
	ECC curve type	`CRYPTO_KE_KEYGENERATE_CURVETYPE`	29

⌋()

The key elements indices of **SWS_Csm_1022** can be extended by the vendor.

[SWS_Csm_00951] ⌈For each key element that contains cryptographic key material, the format of the provided key shall be specified in the configuration used for data exchange, e.g. for `Csm_KeyElementGet()` or `Csm_KeyElementSet()`. The key formats supported by a specific crypto driver are part of the pre-configuration information that comes along with the crypto driver.
⌋(SRS_CryptoStack_00008)

[SWS_Csm_00953] ⌈The following key formats are available:

CRYPTO_KE_FORMAT_BIN_OCTET	Key provided as octet value in binary form[1].
CRYPTO_KE_FORMAT_BIN_SHEKEYS	Combined input/output keys for SHE operation (M1+M2+M3) and (M4+M5).
CRYPTO_KE_FORMAT_BIN_IDENT_PRIVATEKEY_ PKCS8	Private key material in ASN.1 coded form (BER coding) with identification. The data is provided in binary form, not, e.g. as a BASE64 string.
CRYPTO_KE_FORMAT_BIN_IDENT_PUBLICKEY	Public key material in ASN.1 coded form (BER coding) with identification. The data is provided in binary form, not, e.g. as a BASE64 string.
CRYPTO_KE_FORMAT_BIN _RSA_PRIVATEKEY	Private key material in ASN.1 coded form (BER coding). The key material is provided in binary form, not, e.g. as a BASE64 string.
CRYPTO_KE_FORMAT_BIN _RSA_PUBLICKEY	Public key material in ASN.1 coded form (BER coding). The key material is provided in binary form, not, e.g. as a BASE64 string.

A binary Octet is the integer representation in base 256. A large value can be splitted into his factors:

$$x = x_{xLen-1} * 256^{xLen-1} + x_{xLen-2} * 256^{xLen-2} + \ldots + x_1 * 256 + x_0. \quad \text{where } 0 <= x_i < 256.$$

Let the Octet Xi have the integer value x_{xLen-i} for 1 <= i <= xLen. The octet is then

$$X = X_1 X_2 .. X_{xLen}$$

Rationale: An asymmetric key can either be provided with or without identification. The identification is used to uniquely identify the key itself that is provided, so that the key parser can check if the key material is appropriate or not. Without identification, the key material must correspond to the format that is specified for this key. Following IETF standards, the identification of a key is provided as an object identifier (OID) as part of the ASN.1 description.
⌋ (SRS_CryptoStack_00008)

[SWS_Csm_00952] ⌈Vendor specific keyElementIds should start 1000 to avoid interferences with future extended versions of the Crypto Stack.
⌋()

Note:
The key elements `CRYPTO_KE_[…]_ALGORITHM` are used to configure the behavior of the key management functions, because they are independent of jobs and therefore can not be configured like a primitive.

[SWS_Csm_01092] ⌈If a cryptographic primitive uses elliptic curve algorithm but the concrete curve parameter cannot sufficiently specified by its algorithm families and its algorithm mode, an additional key element of type CRYPTO_KE_xxxxx_CURVETYPE shall be used to provide the required information. This information is set at runtime through the key element interface. The data of the key element shall be set with its object identifier follows the format defined in [19] and [20].
⌋(SRS_CryptoStack_00008)

Example: Definition for an ECC Brainpool 160 P1 key used for signature generation.

```
const uint8 * EccKey = { 0x12, 0x23, 0x34, ... }
; // The required key value.

// According to RFC5639:
// {iso(1) identified-organization(3) teletrust(36)
algorithm(3) signatureAlgorithm(3) ecSign(2)
ecStdCurvesAndGeneration(8) ellipticCurve(1)}

brainpoolP160r1(1)
const uint8 * EccType = { 1, 3, 36, 3, 3, 2, 8, 1, 1 }
; //OID definition of ECC Brainpool 160 P1

Csm_KeyElementSet(MyEccKeyId, CRYPTO_KE_SIGNATURE_KEY, EccKey,
sizeof(EccKey) );
Csm_KeyElementSet(MyEccKeyId, CRYPTO_KE_SIGNATURE_CURVETYPE,
EccType, sizeof(EccType) );
Csm_KeySetValid(MyEccKeyId);
```

7.2.2.4 Redirection of Input and/or Output of Crypto Jobs

[SWS_Csm_91013] ⌈The input and/or output data of a job can be re-directed to a key element. Which input and output value to which key and its key element is re-directed shall be statically configured at compile time and shall not be changed at runtime.
⌋()

[SWS_Csm_91014] ⌈If an input or output value of a job is re-directed to a key element (`CsmInOutRedirectionRef` ECUC_Csm_00262 is existing) and the

corresponding input or output length value is not set to 0, the job shall not be processed and E_NOT_OK shall be returned.
⌋()

[SWS_Csm_91015] ⌈If input or output redirection is not used for a job (no CsmInOutRedirectionRef ECUC_Csm_00262 is existing), jobRedirectionInfoRef shall be set to NULL_PTR. If redirection is used (CsmInOutRedirectionRef ECUC_Csm_00262 is existing) the jobRedirectionInfoRef shall point to a structure of Crypto_JobRedirectionInfoType.
⌋()

[SWS_Csm_91016] ⌈The structure Crypto_JobRedirectionInfoType contains information which key elements shall be used for redirection. A bit field called redirectionConfig is provided that indicates which input and/or output value is redirected.
The value of redirectionConfig is a bit coded value that is used to indicate, which of the input and output buffers are redirected. If the least significant bit (Bit #0 or 0x01) of redirectionConfig is set the primary input key and its element is redirected and the value of inputKeyId and inputKeyElementId must indicate the element that is used for input buffer instead of the inputPtr and its length. If Bit #1 is set, the secondaryInputBuffer is redirected to the secondary input key is set and the key and key elements must be set, and Bit #2 is used for the tertiary input key. Bit #3 is reserved for future use.
If Bit #4 is set the outputPtr is redirected to the output key element of the output key. Bit #5 indicates the redirection of the secondary output buffer to the secondary key and its key element. If a bit is set to 0 the input or output shall not be redirected to the associated Key Element.
Example: A value of redirectionConfig of "00110001" indicates that the input should be gathered from the inputKeyElement of inputKeyId and that the output buffer and secondary output buffer shall be redirected to the outputKeyElement of outputKeyId and secondaryOutputKeyElement of secondaryOutputKeyId.
⌋()

7.2.2.5 Job context interface

The job context interface allows to save or restore the context data of the workspace for a specific crypto service from the crypto driver. This allows to store all dynamically created data within a crypto driver so that it can later be restored to continue this operation at the exact point where the context snapshot was taken.
Key element data are not affected. This means, that key elements are not part of the context data and must be set or read by the key element interface separately if necessary.

[SWS_Csm_91069] ⌈ In addition to the standard error detection described in chapter 7.4, on a call to Csm_SaveContextJob() or Csm_RestoreContextJob() the CSM shall check if the related job is currently active. If so, the operation shall

continue as specified. Otherwise (the job is in IDLE state), the function shall immediately return with E_NOT_OK.
⌋()

[SWS_Csm_91070] ⌈ On a call to Csm_SaveContextJob() or Csm_RestoreContextJob() the CSM shall check if the related job is currently actively processing data, means the CSM has scheduled an operation related to this job to the driver and is waiting for a response. In this case, the function shall immediately return with E_NOT_OK.
⌋()

[SWS_Csm_91071] ⌈ On a call to Csm_RestoreContextJob() or Csm_SaveContextJob() the CSM shall check if the job references to either of the CsmPrimitive CsmHash, CsmEncrypt, CsmEncryptAEAD, CsmDecryptAEAD, CsmDecrypt, CsmMacGenerate, CsmMacVerify, CsmSignatureGenerate or CsmSignatureVerify. If so, the operation shall continue as specified. Otherwise, the operation shall not be performed and CSM_E_SERVICE_TYPE shall be reported to the DET when CsmDevErrorDetect is true.
⌋()

7.3 Error Classification

Section 7.2 "Error Handling" of the document "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

7.3.1 Development Errors

[SWS_Csm_91004]⌈

Type of error	*Related error code*	*Error value*
API request called with invalid parameter (Nullpointer)	CSM_E_PARAM_ POINTER	0x01
Csm Configuration ID out of range	CSM_E_PARAM_HANDLE	0x04
API request called before initialization of CSM module	CSM_E_UNINIT	0x05
Initialization of CSM module failed	CSM_E_INIT_FAILED	0x07
API request called with invalid processing mode	CSM_E_PROCESSING_ MODE	0x08
Mismatch between the called API request and the service type of the job	CSM_E_SERVICE_TYPE	0x09

⌋(SRS_CryptoStack_00086)

7.3.2 Runtime Errors

[SWS_Csm_01089]⌈

Type of error	*Related error code*	*Error value*
Queue overrun	CSM_E_QUEUE_FULL	0x01

⌋(SRS_CryptoStack_00086)

7.3.3 Transient Faults

There are no transient faults.

7.3.4 Production Errors

There are no production errors.

7.3.5 Extended Production Errors

There are no extended production errors.

7.4 Error Detection

[SWS_Csm_91008] ⌈ While the CSM is not initialized and any function of the CSM API is called, except of `CSM_Init()` and `Csm_GetVersionInfo()`, the operation shall not be performed and `CSM_E_UNINIT` shall be reported to the DET when `CsmDevErrorDetect` is true.
⌋()

[SWS_Csm_91009] ⌈If a pointer to null is passed to an API function and the corresponding input or output data are not re-directed to a key element, the operation shall not be performed and `CSM_E_PARAM_POINTER` shall be reported to the DET when `CsmDevErrorDetect` is true.
⌋()

[SWS_Csm_91011] ⌈If a CSM API with a ID handle in its interface is called and the ID handle is out of range, the operation shall not be performed and CSM_E_PARAM_HANDLE shall be reported to the DET when CsmDevErrorDetect is true.
⌋()

[SWS_Csm_01091] ⌈If a CSM API with a job handle (called jobId) in its interface is called and the Crypto_ServiceInfoType of the job does not match the requested service, the operation shall not be performed and `CSM_E_SERVICE_TYPE` shall be reported to the DET when `CsmDevErrorDetect` is true.
⌋(SRS_CryptoStack_00087)

[SWS_Csm_01088] ⌈If a CSM job needs to be queued and the queue is full, the runtime error `CSM_E_QUEUE_FULL` shall be reported to the DET.
⌋(SRS_CryptoStack_00087)

Note: The indication of a queue overrun is logged as runtime error.

7.5 Multicore

In case the Crypto-Stack is distributed across several partitions, Csm shall allow calls of its <sevice> APIs in different partitions.

[SWS_Csm_91065] ⌈ CsmQueues shall be handled within the MainFunction, which is referenced via CsmQueueMainFunctionRef.
⌋()

Note:
In case the a CsmJob is not processed inside MainFunction context at all (Synchronous interfacing), the MainFunction assignment (via the respective CsmQueue) defines the partition, where the CsmJob is assigned to.

[SWS_Csm_91066] ⌈ The Csm module shall apply appropriate mechanisms to allow calls of `Csm_<Service>()` API from partitions its CsmJobs are assigned to.
⌋()

8 API Specification

8.1 Imported types

[SWS_Csm_00068] ⌈Only the standard AUTOSAR types provided by Std_Types.h shall be imported.
⌋()

8.2 Type Definitions

8.2.1 Extension to Std_ReturnType

[SWS_Csm_91043]⌈

Range	CRYPTO_E_BUSY	0x02	The service request failed because the service is still busy
	CRYPTO_E_ENTROPY_EXHAUSTED	0x04	The service request failed because the entropy of the random number generator is exhausted
	CRYPTO_E_KEY_READ_FAIL	0x06	The service request failed because read access was denied
	CRYPTO_E_KEY_WRITE_FAIL	0x07	The service request failed because the writing access failed
	CRYPTO_E_KEY_NOT_AVAILABLE	0x08	The service request failed because at least one required key element is not available.
	CRYPTO_E_KEY_NOT_VALID	0x09	The service request failed because the key is invalid.
	CRYPTO_E_KEY_SIZE_MISMATCH	0x0A	The service request failed because the key size does not match.
	CRYPTO_E_JOB_CANCELED	0x0C	The service request failed because the Job has been canceled.
	CRYPTO_E_KEY_EMPTY	0x0D	The service request failed because of uninitialized source key element.
	CRYPTO_E_CUSTOM_ERROR	0x0E	Custom processing failed.
Description	--		
Available via	Crypto_GeneralTypes.h		

⌋(SRS_CryptoStack_00095)

8.2.2 Csm_ConfigType

[SWS_Csm_01085]⌈

Name	Csm_ConfigType	
Kind	Structure	
Elements	implementation specific	
	Type	--
	Comment	The content of the configuration data structure is implementation specific.
Description	Configuration data structure of Csm module	
Available via	Csm.h	

⌋(SWS_BSW_00216)

8.2.3 Crypto_AlgorithmFamilyType

[SWS_Csm_01047]⌈

Name	Crypto_AlgorithmFamilyType		
Kind	Enumeration		
Range	CRYPTO_ALGOFAM_NOT_SET	0x00	Algorithm family is not set
	CRYPTO_ALGOFAM_SHA1	0x01	SHA1 hash
	CRYPTO_ALGOFAM_SHA2_224	0x02	SHA2-224 hash
	CRYPTO_ALGOFAM_SHA2_256	0x03	SHA2-256 hash
	CRYPTO_ALGOFAM_SHA2_384	0x04	SHA2-384 hash
	CRYPTO_ALGOFAM_SHA2_512	0x05	SHA2-512 hash
	CRYPTO_ALGOFAM_SHA2_512_224	0x06	SHA2-512/224 hash
	CRYPTO_ALGOFAM_SHA2_512_256	0x07	SHA2-512/256 hash
	CRYPTO_ALGOFAM_SHA3_224	0x08	SHA3-224 hash
	CRYPTO_ALGOFAM_SHA3_256	0x09	SHA3-256 hash

	CRYPTO_ALGOFAM_SHA3_384	0x0a	SHA3-384 hash
	CRYPTO_ALGOFAM_SHA3_512	0x0b	SHA3-512 hash
	CRYPTO_ALGOFAM_SHAKE128	0x0c	SHAKE128 hash
	CRYPTO_ALGOFAM_SHAKE256	0x0d	SHAKE256 hash
	CRYPTO_ALGOFAM_RIPEMD160	0x0e	RIPEMD hash
	CRYPTO_ALGOFAM_BLAKE_1_256	0x0f	BLAKE-1-256 hash
	CRYPTO_ALGOFAM_BLAKE_1_512	0x10	BLAKE-1-512 hash
	CRYPTO_ALGOFAM_BLAKE_2s_256	0x11	BLAKE-2s-256 hash
	CRYPTO_ALGOFAM_BLAKE_2s_512	0x12	BLAKE-2s-512 hash
	CRYPTO_ALGOFAM_3DES	0x13	3DES cipher
	CRYPTO_ALGOFAM_AES	0x14	AES cipher
	CRYPTO_ALGOFAM_CHACHA	0x15	ChaCha cipher
	CRYPTO_ALGOFAM_RSA	0x16	RSA cipher
	CRYPTO_ALGOFAM_ED25519	0x17	ED22519 elliptic curve
	CRYPTO_ALGOFAM_BRAINPOOL	0x18	Brainpool elliptic curve
	CRYPTO_ALGOFAM_ECCNIST	0x19	NIST ECC elliptic curves
	CRYPTO_ALGOFAM_RNG	0x1b	Random Number Generator
	CRYPTO_ALGOFAM_SIPHASH	0x1c	SipHash
	CRYPTO_ALGOFAM_ECCANSI	0x1e	Elliptic curve according to ANSI X9.62
	CRYPTO_ALGOFAM_ECCSEC	0x1f	Elliptic curve according to SECG
	CRYPTO_ALGOFAM_DRBG	0x20	Random number generator according to NIST SP800-90A
	CRYPTO_ALGOFAM_FIPS186	0x21	Random number generator according to FIPS 186.

	CRYPTO_ALGOFAM_PADDING_PKCS7	0x22	Cipher padding according to PKCS.7
	CRYPTO_ALGOFAM_PADDING_ONEWITHZEROS	0x23	Cipher padding mode. Fill/verify data with 0, but first bit after the data is 1. Eg. "DATA" & 0x80 & 0x00...
	CRYPTO_ALGOFAM_PBKDF2	0x24	Password-Based Key Derivation Function 2
	CRYPTO_ALGOFAM_KDFX963	0x25	ANSI X9.63 Public Key Cryptography
	CRYPTO_ALGOFAM_DH	0x26	Diffie-Hellman
	CRYPTO_ALGOFAM_SM2	0x27	SM2 elliptic curve algorithm
	CRYPTO_ALGOFAM_EEA3	0x28	Stream cipher based on [x01]
	CRYPTO_ALGOFAM_SM3	0x29	Chinese hash algorithm based on [x02]
	CRYPTO_ALGOFAM_EIA3	0x2A	Authentication algorithm [x01]
	CRYPTO_ALGOFAM_HKDF	0x2B	HMAC-based extract-and-expand key derivation function
	CRYPTO_ALGOFAM_ECDSA	0x2C	Elliptic-curve Digital Signatures
	CRYPTO_ALGOFAM_POLY1305	0x2D	MAC calculation algorithm
	CRYPTO_ALGOFAM_X25519	0x2E	Elliptic curve X25519 for ECDH
	CRYPTO_ALGOFAM_ECDH	0x2F	Elliptic-curve Diffie Hellman
	CRYPTO_ALGOFAM_CUSTOM	0xff	Custom algorithm family
Description	Enumeration of the algorithm family.		
Available via	Crypto_GeneralTypes.h		

⌋()

8.2.4 **Crypto_AlgorithmModeType**

[SWS_Csm_01048]⌈

Name	Crypto_AlgorithmModeType		
Kind	Enumeration		
Range	CRYPTO_ALGOMODE_NOT_SET	0x00	Algorithm key is not set
	CRYPTO_ALGOMODE_ECB	0x01	Blockmode: Electronic Code Book

	CRYPTO_ALGOMODE_CBC	0x02	Blockmode: Cipher Block Chaining
	CRYPTO_ALGOMODE_CFB	0x03	Blockmode: Cipher Feedback Mode
	CRYPTO_ALGOMODE_OFB	0x04	Blockmode: Output Feedback Mode
	CRYPTO_ALGOMODE_CTR	0x05	Blockmode: Counter Modex
	CRYPTO_ALGOMODE_GCM	0x06	Blockmode: Galois/Counter Mode
	CRYPTO_ALGOMODE_XTS	0x07	XEX Tweakable Block Cipher with Ciphertext Stealing
	CRYPTO_ALGOMODE_RSAES_OAEP	0x08	RSA Optimal Asymmetric Encryption Padding
	CRYPTO_ALGOMODE_RSAES_PKCS1_v1_5	0x09	RSA encryption/decryption with PKCS#1 v1.5 padding
	CRYPTO_ALGOMODE_RSASSA_PSS	0x0a	RSA Probabilistic Signature Scheme
	CRYPTO_ALGOMODE_RSASSA_PKCS1_v1_5	0x0b	RSA signature with PKCS#1 v1.5
	CRYPTO_ALGOMODE_8ROUNDS	0x0c	8 rounds (e.g. ChaCha8)
	CRYPTO_ALGOMODE_12ROUNDS	0x0d	12 rounds (e.g. ChaCha12)
	CRYPTO_ALGOMODE_20ROUNDS	0x0e	20 rounds (e.g. ChaCha20)
	CRYPTO_ALGOMODE_HMAC	0x0f	Hashed-based MAC
	CRYPTO_ALGOMODE_CMAC	0x10	Cipher-based MAC
	CRYPTO_ALGOMODE_GMAC	0x11	Galois MAC
	CRYPTO_ALGOMODE_CTRDRBG	0x12	Counter-based Deterministic Random Bit Generator
	CRYPTO_ALGOMODE_SIPHASH_2_4	0x13	Siphash-2-4
	CRYPTO_ALGOMODE_SIPHASH_4_8	0x14	Siphash-4-8
	CRYPTO_ALGOMODE_PXXXR1	0x15	ANSI R1 Curve
	CRYPTO_ALGOMODE_AESKEYWRAP	0x16	AES Key Wrap (RFC 3394)
	CRYPTO_ALGOMODE_CUSTOM	0xff	Custom algorithm mode
Description	Enumeration of the algorithm mode		
Available via	Crypto_GeneralTypes.h		

⌋()

8.2.5 Crypto_InputOutputRedirectionConfigType

[SWS_Csm_91024]⌈

Name	Crypto_InputOutputRedirectionConfigType		
Kind	Enumeration		
Range	CRYPTO_REDIRECT_CONFIG_PRIMARY_INPUT	0x01	--
	CRYPTO_REDIRECT_CONFIG_SECONDARY_INPUT	0x02	--
	CRYPTO_REDIRECT_CONFIG_TERTIARY_INPUT	0x04	--
	CRYPTO_REDIRECT_CONFIG_PRIMARY_OUTPUT	0x10	--
	CRYPTO_REDIRECT_CONFIG_SECONDARY_OUTPUT	0x20	--
Description	Defines which of the input/output parameters are re-directed to a key element. The values can be combined to define a bit field.		
Available via	Crypto_GeneralTypes.h		

⌋()

8.2.6 Crypto_JobType

[SWS_Csm_01013]⌈

Name	Crypto_JobType	
Kind	Structure	
Elements	jobId	
	Type	uint32
	Comment	Identifier for the job structure.
	jobState	
	Type	Crypto_JobStateType
	Comment	Determines the current job state.
	jobPrimitiveInputOutput	
	Type	Crypto_JobPrimitiveInputOutputType
	Comment	Structure containing input and output information depending on the job and the crypto primitive.
	jobPrimitiveInfo	
	Type	const Crypto_JobPrimitiveInfoType*
	Comment	Pointer to a structure containing further information which depends on the job and the crypto primitive.

	jobRedirectionInfoRef	
	Type	Crypto_JobRedirectionInfoType*
	Comment	Pointer to a structure containing further information on the usage of keys as input and output for jobs.
	cryptoKeyId	
	Type	uint32
	Comment	Identifier of the Crypto Driver key. The identifier shall be written by the Crypto Interface.
	targetCryptoKeyId	
	Type	uint32
	Comment	Target identifier of the Crypto Driver key. The identifier shall be written by the Crypto Interface.
	jobPriority	
	Type	const uint32
	Comment	Specifies the importance of the job (the higher, the more important).
Description	Structure which contains further information, which depends on the job and the crypto primitive.	
Available via	Crypto_GeneralTypes.h	

⌋()

8.2.7 Crypto_JobStateType

[SWS_Csm_01028]⌈

Name	Crypto_JobStateType		
Kind	Enumeration		
Range	CRYPTO_JOBSTATE_IDLE	0x00	Job is in the state "idle". This state is reached after Csm_Init() or when the "Finish" state is finished.
	CRYPTO_JOBSTATE_ACTIVE	0x01	Job is in the state "active". There was already some input or there are intermediate results. This state is reached, when the "update" or "start" operation finishes.
Description	Enumeration of the current job state.		
Available via	Crypto_GeneralTypes.h		

⌋()

8.2.8 Crypto_JobPrimitiveInputOutputType

[SWS_Csm_01009]⌈

Name	Crypto_JobPrimitiveInputOutputType	
Kind	Structure	
Elements	inputPtr	
	Type	const uint8*
	Comment	Pointer to the input data.
	inputLength	
	Type	uint32
	Comment	Contains the input length in bytes.
	secondaryInputPtr	
	Type	const uint8*
	Comment	Pointer to the secondary input data (for MacVerify, SignatureVerify).
	secondaryInputLength	
	Type	uint32
	Comment	Contains the secondary input length in bits or bytes, depending on the requested service.
	tertiaryInputPtr	
	Type	const uint8*
	Comment	Pointer to the tertiary input data (for MacVerify, SignatureVerify).
	tertiaryInputLength	
	Type	uint32
	Comment	Contains the tertiary input length in bytes.
	outputPtr	
	Type	uint8*
	Comment	Pointer to the output data.
	outputLengthPtr	
	Type	uint32*
	Comment	Holds a pointer to a memory location containing the output length in bytes.
	secondaryOutputPtr	
	Type	uint8*

	Comment	Pointer to the secondary output data.
	secondaryOutputLengthPtr	
	Type	uint32*
	Comment	Holds a pointer to a memory location containing the secondary output length in bytes.
	verifyPtr	
	Type	Crypto_VerifyResultType*
	Comment	Output pointer to a memory location holding a Crypto_VerifyResultType
	mode	
	Type	Crypto_OperationModeType
	Comment	Indicator of the mode(s)/operation(s) to be performed
	cryIfKeyId	
	Type	uint32
	Comment	Holds the CryIf key id for key operation services.
	targetCryIfKeyId	
	Type	uint32
	Comment	Holds the target CryIf key id for key operation services.
Description	Structure which contains input and output information depending on the job and the crypto primitive.	
Available via	Crypto_GeneralTypes.h	

⌋()

8.2.9 **Crypto_JobPrimitiveInfoType**

[SWS_Csm_01012]⌈

Name	Crypto_JobPrimitiveInfoType	
Kind	Structure	
Elements	callbackId	
	Type	uint32
	Comment	Internal identifier of the callback function, to be called by Csm, if the configured service is finished.
	primitiveInfo	
	Type	const Crypto_PrimitiveInfoType*

	Comment	Pointer to a structure containing further configuration of the crypto primitives
	cryIfKeyId	
	Type	uint32
	Comment	Identifier of the CryIf key.
	processingType	
	Type	Crypto_ProcessingType
	Comment	Determines the synchronous or asynchronous behavior.
Description	Structure which contains further information, which depends on the job and the crypto primitive.	
Available via	Crypto_GeneralTypes.h	

⌋(SRS_CryptoStack_00008)

8.2.10 **Crypto_ServiceInfoType**

[SWS_Csm_01031]⌈

Name	Crypto_ServiceInfoType		
Kind	Enumeration		
Range	CRYPTO_HASH	0x00	Hash Service
	CRYPTO_MACGENERATE	0x01	MacGenerate Service
	CRYPTO_MACVERIFY	0x02	MacVerify Service
	CRYPTO_ENCRYPT	0x03	Encrypt Service
	CRYPTO_DECRYPT	0x04	Decrypt Service
	CRYPTO_AEADENCRYPT	0x05	AEADEncrypt Service
	CRYPTO_AEADDECRYPT	0x06	AEADDecrypt Service
	CRYPTO_SIGNATUREGENERATE	0x07	SignatureGenerate Service
	CRYPTO_SIGNATUREVERIFY	0x08	SignatureVerify Service
	CRYPTO_RANDOMGENERATE	0x0B	RandomGenerate Service
	CRYPTO_RANDOMSEED	0x0C	RandomSeed Service
	CRYPTO_KEYGENERATE	0x0D	KeyGenerate Service
	CRYPTO_KEYDERIVE	0x0E	KeyDerive Service
	CRYPTO_ KEYEXCHANGECALCPUBVAL	0x0F	KeyExchangeCalcPubVal Service

	CRYPTO_ KEYEXCHANGECALCSECRET	0x10	KeyExchangeCalcSecret Service
	CRYPTO_KEYSETVALID	0x13	KeySetValid Service
	CRYPTO_KEYSETINVALID	0x14	KeySetInvalid Service
	CRYPTO_CUSTOM_SERVICE	0x15	Custom service job
Description	Enumeration of the kind of the service.		
Available via	Crypto_GeneralTypes.h		

⌋()

8.2.11 **Crypto_JobRedirectionInfoType**

[SWS_Csm_91026]⌈

Name	Crypto_JobRedirectionInfoType	
Kind	Structure	
Elements	redirectionConfig	
	Type	uint8
	Comment	Bit structure which indicates which buffer shall be redirected to a key element. Values from Crypto_InputOutputRedirectionConfigType can be used and combined with unary OR operation.
	inputKeyId	
	Type	uint32
	Comment	Identifier of the key which shall be used as input
	inputKeyElementId	
	Type	uint32
	Comment	Identifier of the key element which shall be used as input
	secondaryInputKeyId	
	Type	uint32
	Comment	Identifier of the key which shall be used as secondary input
	secondaryInputKeyElementId	
	Type	uint32
	Comment	Identifier of the key element which shall be used as secondary input
	tertiaryInputKeyId	
	Type	uint32

	Comment	Identifier of the key which shall be used as tertiary input
	tertiaryInputKeyElementId	
	Type	uint32
	Comment	Identifier of the key element which shall be used as tertiary input
	outputKeyId	
	Type	uint32
	Comment	Identifier of the key which shall be used as output
	outputKeyElementId	
	Type	uint32
	Comment	Identifier of the key element which shall be used as output
	secondaryOutputKeyId	
	Type	uint32
	Comment	Identifier of the key which shall be used as secondary output
	secondaryOutputKeyElementId	
	Type	uint32
	Comment	Identifier of the key element which shall be used as secondary output
Description	Structure which holds the identifiers of the keys and key elements which shall be used as input and output for a job and a bit structure which indicates which buffers shall be redirected to those key elements.	
Available via	Crypto_GeneralTypes.h	

⌋()

8.2.12 **Crypto_AlgorithmInfoType**

[SWS_Csm_01008]⌈

Name	Crypto_AlgorithmInfoType	
Kind	Structure	
Elements	family	
	Type	Crypto_AlgorithmFamilyType
	Comment	The family of the algorithm
	secondaryFamily	
	Type	Crypto_AlgorithmFamilyType

	Comment	The secondary family of the algorithm
	keyLength	
	Type	uint32
	Comment	The key length in bits to be used with that algorithm
	mode	
	Type	Crypto_AlgorithmModeType
	Comment	The operation mode to be used with that algorithm
Description	Structure which determines the exact algorithm. Note, not every algorithm needs to specify all fields. AUTOSAR shall only allow valid combinations.	
Available via	Crypto_GeneralTypes.h	

⌋()

8.2.13 **Crypto_ProcessingType**

[SWS_Csm_01049]⌈

Name	Crypto_ProcessingType		
Kind	Enumeration		
Range	CRYPTO_PROCESSING_ASYNC	0x00	Asynchronous job processing
	CRYPTO_PROCESSING_SYNC	0x01	Synchronous job processing
Description	Enumeration of the processing type.		
Available via	Crypto_GeneralTypes.h		

⌋(SRS_CryptoStack_00100, SRS_CryptoStack_00101)

8.2.14 **Crypto_PrimitiveInfoType**

[SWS_Csm_01011]⌈

Name	Crypto_PrimitiveInfoType	
Kind	Structure	
Elements	service	
	Type	const Crypto_ServiceInfoType
	Comment	Contains the enum of the used service, e.g. Encrypt
	algorithm	
	Type	const Crypto_AlgorithmInfoType

	Comment	Contains the information of the used algorithm
Description	Structure which contains basic information about the crypto primitive.	
Available via	Crypto_GeneralTypes.h	

⌋()

8.3 Function Definitions

[SWS_Csm_00478] ⌈All functions need not to be reentrant. For behavior in case of a reentrant call see **SWS_Csm_00017**.
⌋()

8.3.1 General Interface

8.3.1.1 Csm_Init

[SWS_Csm_00646]⌈

Service Name	Csm_Init	

Syntax	`void Csm_Init (`
	`  const Csm_ConfigType* configPtr`
	`)`	

Service ID [hex]	0x00	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	configPtr	Pointer to a selected configuration structure
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initializes the CSM module. In configurations, in which Csm is assigned to more than one partition (i.e. Csm_MainFunctions are mapped to partitions), Csm may provide one init function per partition.	
Available via	Csm.h	

⌋(SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414)

[SWS_Csm_00186] ⌈The Configuration pointer `configPtr` shall always have a null pointer value.
⌋(SWS_BSW_00050)

The Configuration pointer configPtr is currently not used and shall therefore be set null pointer value.

[SWS_Csm_00659] ⌈If the initialization of the CSM module fails, the CSM shall report `CSM_E_INIT_FAILED` to the DET when `CsmDevErrorDetect` is true.
⌋()

8.3.1.2 **Csm_GetVersionInfo**

[SWS_Csm_00705]⌈

Service Name	Csm_GetVersionInfo	

Syntax	`void Csm_GetVersionInfo (`
	`  Std_VersionInfoType* versioninfo`
	`)`	

Service ID [hex]	0x3b	
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.	
Available via	Csm.h	

⌋(SRS_BSW_00407)

8.3.2 **Hash Interface**

A cryptographic hash function is a deterministic procedure that takes an arbitrary block of data and returns a fixed-size bit string, the hash value, such that an accidental or intentional change to the data will change the hash value. Main properties of hash functions are that it is infeasible to find a message that has a given hash or to find two different messages with the same hash.

8.3.2.1 **Csm_Hash**

[SWS_Csm_00980]⌈

Service Name	Csm_Hash	

Syntax	`Std_ReturnType Csm_Hash (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  uint8* resultPtr,`
	`  uint32* resultLengthPtr`
	`)`	

Service ID [hex]	0x5d	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Contains the pointer to the data for which the hash shall be computed.
	data Length	Contains the number of bytes to be hashed.
Parameters (inout)	result Length Ptr	Holds a pointer to the memory location in which the output length in bytes is stored. On calling this function, this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored. If the provided length information is smaller than the total length of the hash result, the resultPtr will contain the truncated hash result.
Parameters (out)	resultPtr	Contains the pointer to the data where the hash value shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed
Description	Uses the given data to perform the hash calculation and stores the hash.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00024)

8.3.3 MAC interface

A message authentication code (MAC) is a short piece of information used to authenticate a message. A MAC algorithm accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC. The MAC value protects both a message's data integrity as well as its authenticity, by allowing verifiers (who also possess the secret key) to detect any changes to the message content.

8.3.3.1 Csm_MacGenerate

[SWS_Csm_00982]⌈

Service Name	Csm_MacGenerate	

Syntax	`Std_ReturnType Csm_MacGenerate (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  uint8* macPtr,`
	`  uint32* macLengthPtr`
	`)`	

Service ID [hex]	0x60	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Contains the pointer to the data for which the MAC shall be computed.
	data Length	Contains the number of bytes to be hashed.
Parameters (inout)	mac Length Ptr	Holds a pointer to the memory location in which the output length in bytes is stored. On calling this function, this parameter shall contain the size of the buffer provided by macPtr. When the request has finished, the actual length of the returned MAC shall be stored. If the provided length information is smaller than the total length of the MAC result, the macPtr will contain the truncated MAC result.
Parameters (out)	macPtr	Contains the pointer to the data where the MAC shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00022)

8.3.3.2 Csm_MacVerify

[SWS_Csm_01050]⌈

Service Name	Csm_MacVerify	

Syntax	`Std_ReturnType Csm_MacVerify (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  const uint8* macPtr,`
	`  uint32 macLength,`
	`  Crypto_VerifyResultType* verifyPtr`
	`)`	

Service ID [hex]	0x61	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Holds a pointer to the data for which the MAC shall be verified.
	dataLength	Contains the number of data bytes for which the MAC shall be verified.
	macPtr	Holds a pointer to the MAC to be verified.
	macLength	Contains the MAC length in BITS to be verified.
Parameters (inout)	None	
Parameters (out)	verifyPtr	Holds a pointer to the memory location, which will hold the result of the MAC verification.
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Verifies the given MAC by comparing if the MAC is generated with the given data.	
Available via	Csm.h	

⌋()

8.3.4 Cipher Interface

The cipher interfaces can be used for symmetrical and asymmentrical encryption or decryption. Furthermore, it is also possible to use these interfaces for compression and decompression, respectively.

8.3.4.1 Csm_Encrypt

[SWS_Csm_00984]⌈

Service Name	Csm_Encrypt	

Syntax	`Std_ReturnType Csm_Encrypt (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  uint8* resultPtr,`
	`  uint32* resultLengthPtr`
	`)`	

Service ID [hex]	0x5e	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Contains the pointer to the data to be encrypted.
	data Length	Contains the number of bytes to encrypt.
Parameters (inout)	result Length Ptr	Holds a pointer to the memory location in which the output length information is stored in bytes. On calling this function, this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	resultPtr	Contains the pointer to the data where the encrypted data shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Encrypts the given data and store the ciphertext in the memory location pointed by the result pointer.	

Available via	Csm.h

⌋(SRS_CryptoStack_00020, SRS_CryptoStack_00021)

In the case of block ciphers, it shall be possible to pass a `dataLength` which is not a multiple of the corresponding block size. The underlying Crypto Driver is responsible for handling these input data.

8.3.4.2 **Csm_Decrypt**

[SWS_Csm_00989]⌈

Service Name	Csm_Decrypt	

Syntax	`Std_ReturnType Csm_Decrypt (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  uint8* resultPtr,`
	`  uint32* resultLengthPtr`
	`)`	

Service ID [hex]	0x5f	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Contains the pointer to the data to be decrypted.
	data Length	Contains the number of bytes to decrypt.
Parameters (inout)	result Length Ptr	Holds a pointer to the memory location in which the output length information is stored in bytes. On calling this function, this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	resultPtr	Contains the pointer to the memory location where the decrypted data shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element

Description	Decrypts the given encrypted data and store the decrypted plaintext in the memory location pointed by the result pointer.
Available via	Csm.h

⌋(SRS_CryptoStack_00020, SRS_CryptoStack_00021)

8.3.5 **Authenticated Encryption with Associated Data (AEAD) Interface**

AEAD (also known as Authenticated Encryption) is a block cipher mode of operation which also allows integrity checks (e.g. AES-GCM).

8.3.5.1 **Csm_AEADEncrypt**

[SWS_Csm_01023]⌈

Service Name	Csm_AEADEncrypt	

Syntax	`Std_ReturnType Csm_AEADEncrypt (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* plaintextPtr,`
	`  uint32 plaintextLength,`
	`  const uint8* associatedDataPtr,`
	`  uint32 associatedDataLength,`
	`  uint8* ciphertextPtr,`
	`  uint32* ciphertextLengthPtr,`
	`  uint8* tagPtr,`
	`  uint32* tagLengthPtr`
	`)`	

Service ID [hex]	0x62	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	plaintextPtr	Contains the pointer to the data to be encrypted.
	plaintext Length	Contains the number of bytes to encrypt.
	associated DataPtr	Contains the pointer to the associated data.
	associated DataLength	Contains the number of bytes of the associated data.
Parameters (inout)	ciphertext LengthPtr	Holds a pointer to the memory location in which the output length in bytes of the ciphertext is stored. On calling this function, this parameter

		shall contain the size of the buffer in bytes provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored.
	tagLengthPtr	Holds a pointer to the memory location in which the output length in bytes of the Tag is stored. On calling this function, this parameter shall contain the size of the buffer in bytes provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	ciphertextPtr	Contains the pointer to the data where the encrypted data shall be stored.
	tagPtr	Contains the pointer to the data where the Tag shall be stored.
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Uses the given input data to perform a AEAD encryption and stores the ciphertext and the MAC in the memory locations pointed by the ciphertext pointer and Tag pointer.	
Available via	Csm.h	

⌋()

8.3.5.2 **Csm_AEADDecrypt**

[SWS_Csm_01026]⌈

Service Name	Csm_AEADDecrypt

Syntax	`Std_ReturnType Csm_AEADDecrypt (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* ciphertextPtr,`
	`  uint32 ciphertextLength,`
	`  const uint8* associatedDataPtr,`
	`  uint32 associatedDataLength,`
	`  const uint8* tagPtr,`
	`  uint32 tagLength,`
	`  uint8* plaintextPtr,`
	`  uint32* plaintextLengthPtr,`
	`  Crypto_VerifyResultType* verifyPtr`
	`)`

Service ID [hex]	0x63

Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	ciphertext Ptr	Contains the pointer to the data to be decrypted.
	ciphertext Length	Contains the number of bytes to decrypt.
	associated DataPtr	Contains the pointer to the associated data.
	associated DataLength	Contains the length in bytes of the associated data.
	tagPtr	Contains the pointer to the Tag to be verified.
	tagLength	Contains the length in bytes of the Tag to be verified.
Parameters (inout)	plaintext LengthPtr	Holds a pointer to the memory location in which the output length in bytes of the paintext is stored. On calling this function, this parameter shall contain the size of the buffer provided by plaintextPtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	plaintextPtr	Contains the pointer to the data where the decrypted data shall be stored.
	verifyPtr	Contains the pointer to the result of the verification.
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Uses the given data to perform an AEAD encryption and stores the ciphertext and the MAC in the memory locations pointed by the ciphertext pointer and Tag pointer.	
Available via	Csm.h	

⌋()

8.3.6 **Signature Interface**

A digital signature is a type of asymmetric cryptography. Digital signatures are equivalent to traditional handwritten signatures in many respects.

Digital signatures can be used to authenticate the source of messages as well as to prove integrity of signed messages. If a message is digitally signed, any change in the message after signature will invalidate the signature. Furthermore, there is no efficient way to modify a message and its signature to produce a new message with a valid signature.

8.3.6.1 **Csm_SignatureGenerate**

[SWS_Csm_00992]⌈

Service Name	Csm_SignatureGenerate	

Syntax	`Std_ReturnType Csm_SignatureGenerate (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  uint8* signaturePtr,`
	`  uint32* signatureLengthPtr`
	`)`	

Service ID [hex]	0x76	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perform.
	dataPtr	Contains the pointer to the data to be signed.
	data Length	Contains the number of bytes to sign.
Parameters (inout)	signature LengthPtr	Holds a pointer to the memory location in which the output length in bytes of the signature is stored. On calling this function, this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	signature Ptr	Contains the pointer to the data where the signature shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Uses the given data to perform the signature calculation and stores the signature in the memory location pointed by the result pointer.	

Available via	Csm.h

⌋(SRS_CryptoStack_00023)

8.3.6.2 **Csm_SignatureVerify**

[SWS_Csm_00996]⌈

Service Name	Csm_SignatureVerify	

Syntax	`Std_ReturnType Csm_SignatureVerify (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  const uint8* dataPtr,`
	`  uint32 dataLength,`
	`  const uint8* signaturePtr,`
	`  uint32 signatureLength,`
	`  Crypto_VerifyResultType* verifyPtr`
	`)`	

Service ID [hex]	0x64	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	mode	Indicates which operation mode(s) to perfom.
	dataPtr	Contains the pointer to the data to be verified.
	dataLength	Contains the number of data bytes.
	signaturePtr	Holds a pointer to the signature to be verified.
	signature Length	Contains the signature length in bytes.
Parameters (inout)	None	
Parameters (out)	verifyPtr	Holds a pointer to the memory location, which will hold the result of the signature verification.
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, a key element has the wrong size CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Verifies the given MAC by comparing if the signature is generated with the given data.	

Available via	Csm.h

⌋(SRS_CryptoStack_00023)

8.3.7 Random Interface

The random interface provides generation of random numbers. A random number can be generated either by a physical device (true random number generator), or by computational algorithms (pseudo random number generator). The randomness of pseudo random number generators can be increased by an appropriate selection of the seed.

Note:
How the random generators are seeded is project specific and out of scope of this specification. If applicable, proper seeding actions shall be done prior to request any random numbers.
An ECU-centralized generation of entropy is recommended, to seed random number generators. Especially if no true random number generator (TRNG) in hardware is available for generation of random numbers.

8.3.7.1 Csm_RandomGenerate

[SWS_Csm_01543]⌈

Service Name	Csm_RandomGenerate	

Syntax	`Std_ReturnType Csm_RandomGenerate (`
	`  uint32 jobId,`
	`  uint8* resultPtr,`
	`  uint32* resultLengthPtr`
	`)`	

Service ID [hex]	0x72	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	result Length Ptr	Holds a pointer to the memory location in which the result length in bytes is stored. On calling this function, this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished, the actual length of the returned random number shall be stored. If the provided length information is smaller than the total length of the random number result, the resultPtr will contain the truncated random number.
Parameters (out)	resultPtr	Holds a pointer to the memory location which will hold the result of the random number generation.

Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_ENTROPY_EXHAUSTED: Request failed, entropy of random number generator is exhausted
Description	Generate a random number and stores it in the memory location pointed by the result pointer.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00019)

To generate a random number, no streaming approach is necessary. The interface `Csm_RandomGenerate` can be called arbitrarily often to generate multiple random numbers.

[SWS_Csm_01054] ⌈ The operation mode of the Csm_RandomGenerate() function call shall be set to ”CRYPTO_OPERATIONMODE_SINGLECALL”.
⌋()

8.3.8 Key Management Interface

The following interfaces are used for key management. Basically, a key contains of one ore more key elements. A key element can be part of multiple keys. For example, this allows to derive a key element from a password with one keyId, and to use this derived key element for encryption with another keyId.

Note:
If the actual key element to be modified is directly mapped to flash memory, there could be a bigger delay when calling the key management functions (synchronous operation)

[SWS_Csm_00974] ⌈ If a key management function is called, the CSM shall disable processing new jobs from the queue until the next call of the main function.
⌋()

8.3.8.1 Key Setting Interface

8.3.8.1.1 Csm_KeyElementSet

[SWS_Csm_00957]⌈

Service Name	Csm_KeyElementSet

Syntax	`Std_ReturnType Csm_KeyElementSet (`
	`  uint32 keyId,`
	`  uint32 keyElementId,`
	`  const uint8* keyElementPtr,`
	`  uint32 keyElementLength`
	`)`

Service ID [hex]	0x78	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	keyId	Holds the identifier of the key for which a new material shall be set.
	keyElementId	Holds the identifier of the key element to be written.
	keyElementPtr	Holds the pointer to the key element bytes to be processed.
	keyElementLength	Contains the number of key element bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_WRITE_FAIL:Request failed because write access was denied CRYPTO_E_KEY_NOT_AVAILABLE: Request failed because the key is not available CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element size does not match size of provided data
Description	Sets the given key element bytes to the key identified by keyId.	
Available via	Csm.h	

⌋()

Note:
In case of error observed during `Csm_KeyElementSet()` API call, the status of the key element needs to be considered as unknown. The application could explicitly check the key content status by calling `Csm_KeyGetStatus()`.

[SWS_Csm_01002] ⌈ If no errors are detected by Csm, the service `Csm_KeyElementSet()` shall call `CryIf_KeyElementSet()`.
⌋ ( )

8.3.8.1.2 Csm_KeySetValid

[SWS_Csm_00958]⌈

Service Name	Csm_KeySetValid

Syntax	`Std_ReturnType Csm_KeySetValid (`
	`  uint32 keyId`
	`)`

Service ID [hex]	0x67	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	keyId	Holds the identifier of the key for which a new material shall be validated.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypro Driver Object is busy
Description	Sets the key state of the key identified by keyId to valid.	
Available via	Csm.h	

⌋()

[SWS_Csm_01003] ⌈ If no errors are detected by Csm, the service `Csm_KeySetValid()` shall call `CryIf_KeySetValid()`.
⌋ ( )

8.3.8.1.3 Csm_KeySetInvalid

[SWS_Csm_91075]⌈

Service Name	Csm_KeySetInvalid	
Syntax	Std_ReturnType Csm_KeySetInvalid (uint32 keyId)	
Service ID [hex]	0x85	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	keyId	Holds the identifier of the key which shall be invalidated.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypro Driver Object is busy

Description	Sets the key status to invalid. The key cannot be used any longer for cryptographic operations until it has been set to valid state again.
Available via	Csm.h

⌋()

[SWS_Csm_91050] ⌈ If no errors are detected by Csm, the service `Csm_KeySetInvalid()` shall call `CryIf_KeySetInvalid()`.
⌋ ( )

8.3.8.2 Key Status Interface

8.3.8.2.1 Csm_KeyGetStatus

[SWS_Csm_91047]⌈

Service Name	Csm_KeyGetStatus	

Syntax	`Std_ReturnType Csm_KeyGetStatus (`
	`  uint32 keyId,`
	`  Crypto_KeyStatusType* keyStatusPtr`
	`)`	

Service ID [hex]	0x83	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	keyId	Holds the identifier of the key for which the key state shall be returned.
Parameters (inout)	None	
Parameters (out)	keyStatusPtr	Contains the pointer to the data where the status of the key shall be stored.
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed
Description	Returns the key state of the key identified by keyId.	
Available via	Csm.h	

⌋()

[SWS_Csm_91048] ⌈ If no errors are detected by Csm, the service `Csm_KeyGetStatus()` shall call `CryIf_KeyGetStatus()`.
⌋()

8.3.8.3 Key Extraction Interface

8.3.8.3.1 Csm_KeyElementGet

[SWS_Csm_00959]⌈

Service Name	Csm_KeyElementGet	

Syntax	`Std_ReturnType Csm_KeyElementGet (`
	`  uint32 keyId,`
	`  uint32 keyElementId,`
	`  uint8* keyElementPtr,`
	`  uint32* keyElementLengthPtr`
	`)`	

Service ID [hex]	0x68	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	keyId	Holds the identifier of the key from which a key element shall be extracted.
	key ElementId	Holds the identifier of the key element to be extracted.
Parameters (inout)	key Element LengthPtr	Holds a pointer to the memory location in which the output buffer length in bytes is stored. On calling this function, this parameter shall contain the buffer length in bytes of the keyElementPtr. When the request has finished, the actual size of the written input bytes shall be stored.
Parameters (out)	key Element Ptr	Holds the pointer to the memory location where the key element shall be copied to.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_AVAILABLE: Request failed, the requested key element is not available CRYPTO_E_KEY_READ_FAIL: Request failed because read access was denied CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Retrieves the key element bytes from a specific key element of the key identified by the keyId and stores the key element in the memory location pointed by the key pointer.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00010, SRS_CryptoStack_00011, SRS_CrytptoStack_00029)

[SWS_Csm_01004] ⌈ If no errors are detected by Csm, the service `Csm_KeyElementGet()` shall call `CryIf_KeyElementGet()`.
⌋ ( )

The underlying Crypto Driver has to decide if and how the key element bytes are extracted.

8.3.8.4 Key Copying Interface

8.3.8.4.1 Csm_KeyElementCopy

[SWS_Csm_00969]⌈

Service Name	Csm_KeyElementCopy	

Syntax	`Std_ReturnType Csm_KeyElementCopy (`
	`  uint32 keyId,`
	`  uint32 keyElementId,`
	`  uint32 targetKeyId,`
	`  uint32 targetKeyElementId`
	`)`	

Service ID [hex]	0x71	
Sync/Async	Synchronous	
Reentrancy	Reentrant but not for the same keyId	
Parameters (in)	keyId	Holds the identifier of the key whose key element shall be the source element.
	keyElementId	Holds the identifier of the key element which shall be the source for the copy operation.
	targetKeyId	Holds the identifier of the key whose key element shall be the destination element.
	targetKey ElementId	Holds the identifier of the key element which shall be the destination for the copy operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_AVAILABLE: Request failed, the requested key element is not available CRYPTO_E_KEY_READ_FAIL: Request failed, not allowed to extract key element CRYPTO_E_KEY_WRITE_FAIL: Request failed, not allowed to write key element CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element sizes are not compatible CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	This function shall copy a key elements from one key to a target key.	
Available via	Csm.h	

⌋()

[SWS_Csm_01032] ⌈ If no errors are detected by Csm and the keyId and targetKeyId are located in different Crypto Drivers, the service Csm_KeyElementCopy() shall call CryIf_KeyElementCopy() and pass on the return value.
⌋ ( )

8.3.8.4.2 Csm_KeyCopy

[SWS_Csm_01034]⌈

Service Name	Csm_KeyCopy	

Syntax	`Std_ReturnType Csm_KeyCopy (`
	`  uint32 keyId,`
	`  uint32 targetKeyId`
	`)`	

Service ID [hex]	0x73	
Sync/Async	Synchronous	
Reentrancy	Reentrant but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key whose key element shall be the source element.
	targetKeyId	Holds the identifier of the key whose key element shall be the destination element.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_AVAILABLE: Request failed, the requested key element is not available CRYPTO_E_KEY_READ_FAIL: Request failed, not allowed to extract key element CRYPTO_E_KEY_WRITE_FAIL: Request failed, not allowed to write key element CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element sizes are not compatible CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	This function shall copy all key elements from the source key to a target key.	
Available via	Csm.h	

⌋()

[SWS_Csm_01035] ⌈ If no errors are detected by Csm, the service Csm_KeyCopy() shall call CryIf_KeyCopy() and pass on the return value.

⌋ ( )

Note: Because there are no checks by Csm and CryIf intended related to key elements during `Csm_KeyCopy` (e.g.

- sizes of target key elements are greater or equal to the sizes of the corresponding source key elements
- access rights of source and target key elements are allowing a copy between different Crypto drivers via `Crypto_KeyElementGet` and `Crypto_KeyElementGet`)

inconsistent states of target keys are possible in case of an inappropriate configuration.
Those inconsistencies have to be identified during development, and should be addressed by configuration.

8.3.8.4.3 Csm_KeyElementCopyPartial

[SWS_Csm_91025]⌈

Service Name	Csm_KeyElementCopyPartial	

Syntax	`Std_ReturnType Csm_KeyElementCopyPartial (`
	`  uint32 keyId,`
	`  uint32 keyElementId,`
	`  uint32 keyElementSourceOffset,`
	`  uint32 keyElementTargetOffset,`
	`  uint32 keyElementCopyLength,`
	`  uint32 targetKeyId,`
	`  uint32 targetKeyElementId`
	`)`	

Service ID [hex]	0x79	
Sync/Async	Synchronous	
Reentrancy	Reentrant, but not for the same keyId	
Parameters (in)	keyId	Holds the identifier of the key whose key element shall be the source element for copy operation.
	keyElementId	Holds the identifier of the key element which shall be the source for the copy operation.
	keyElement SourceOffset	This is the offset of the source key element indicating the start index of the copy operation.
	keyElement TargetOffset	This is the offset of the destination key element indicating the start index of the copy operation.
	keyElement CopyLength	Specifies the number of bytes that shall be copied.
	targetKeyId	Holds the identifier of the key whose key element shall be the destination element.

	targetKey ElementId	Holds the identifier of the key element which shall be the destination for the copy operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_AVAILABLE: Request failed, the requested key element is not available CRYPTO_E_KEY_READ_FAIL: Request failed, not allowed to extract key element CRYPTO_E_KEY_WRITE_FAIL: Request failed, not allowed to write key element CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element sizes are not compatible CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Copies a key element to another key element in the same crypto driver. The keyElementSourceOffset and keyElementCopyLength allows to copy just a part of the source key element into the destination. The offset into the target key is also specified with this function.	
Available via	Csm.h	

⌋()

Note: A Concatenation of partial keys into one key element is possible by calling Csm_KeyElementCopyPartial() multiple times and adjusting keyElementTargetOffset properly.

[SWS_Csm_91019] ⌈ If no errors are detected by Csm shall call `CryIf_KeyElementCopyPartial()` and pass on the return value.
⌋ ( )

[SWS_Csm_91020] ⌈If the current length of the target key element is greater or equal than (keyElementTargetOffset + keyElementCopyLength), the key element length remains unchanged and the target data is overwritten with the contents of the source data.
⌋ ( )

[SWS_Csm_91021] ⌈ If the current length of the target key element is lower than (`keyElementTargetOffset` + `keyElementCopyLength`) and the maximum length of the key element is greater or equal than (`keyElementTargetOffset` + `keyElementCopyLength`), then the source data shall be copied into the target key element and the length shall be set to (`keyElementTargetOffset` + `keyElementCopyLength`).
⌋ ( )

[SWS_Csm_91022] ⌈
If the maximum length of the target key element is lower than (`keyElementTargetOffset` + `keyElementCopyLength`) then the copy operation shall not be performed and the function shall return with the error code `CRYPTO_E_KEY_SIZE_MISMATCH`.
⌋ ()

8.3.8.5 Key Generation interface

The key generation interface is used to generate a key into the key element `CRYPTO_KE_KEYGENERATE_KEY` according to the algorithm defined in the key element `CRYPTO_KE_KEYGENERATE_ALGORITHM`.
The key will be generated from the random value that is located in the key element `CRYPTO_KE_KEYGENERATE_SEED`.
The random value can be generated, for example, with the function `Csm_RandomGenerate()` and must be stored in `CRYPTO_KE_KEYGENERATE_SEED` before the key generation is triggered.
It is important to check the quality of the randomness and its entropy of the seed, which depends on the used hardware, and software of a stack. The randomness has a major impact on the quality of the generated key material.

The key element with the id=`CRYPTO_KE_KEYGENERATE_ALGORITHM` contains a type from "Crypto_AlgorithmFamilyType", e.g. `CRYPTO_ALGOFAM_AES`, `CRYPTO_ALGOFAM_RSA` or `CRYPTO_ALGOFAM_ED25519`, that allows to generate an adequate key.
As a counter example, the algorithm family type `CRYPTO_ALGOFAM_SHA2_256` is not adequate because it provides no hint what key shall be generated.

For the key element `CRYPTO_KE_KEYGENERATE_KEY` the key element configuration item `CryptoKeyElement/CryptoKeyElementFormat` indicates the format of the generated key.

8.3.8.5.1 Csm_RandomSeed

[SWS_Csm_01051]⌈

Service Name	Csm_RandomSeed	

Syntax	`Std_ReturnType Csm_RandomSeed (`
	`  uint32 keyId,`
	`  const uint8* seedPtr,`
	`  uint32 seedLength`
	`)`	

Service ID [hex]	0x69	
Sync/Async	Synchronous	
Reentrancy	Reentrant, but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key for which a new seed shall be generated.

	seedPtr	Holds a pointer to the memory location which contains the data to feed the seed.
	seedLength	Contains the length of the seed in bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid"
Description	Feeds the key element CRYPTO_KE_RANDOM_SEED with a random seed.	
Available via	Csm.h	

⌋()

[SWS_Csm_01052] ⌈ If no errors are detected by Csm, the service `Csm_RandomSeed()` shall call `CryIf_RandomSeed()`.
⌋ ( )

8.3.8.5.2 Csm_KeyGenerate

[SWS_Csm_00955]⌈

Service Name	Csm_KeyGenerate	

Syntax	`Std_ReturnType Csm_KeyGenerate (`
	`  uint32 keyId`
	`)`	

Service ID [hex]	0x6a	
Sync/Async	Synchronous	
Reentrancy	Reentrant but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key for which a new material shall be generated.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid"

		CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Generates new key material and store it in the key identified by keyId.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00026, SRS_CryptoStack_00027)

[SWS_Csm_01005] ⌈ If no errors are detected by Csm, the service `Csm_KeyGenerate()` shall call `CryIf_KeyGenerate()`.
⌋ ( )

8.3.8.6 **Key Derivation Interface**

In cryptography, a key derivation function (or KDF) is a function, which derives one or more secret keys from a secret value and/or other known information such as a passphrase or cryptographic key.
Specification of input keys that are protected by hardware means can be achieved by using the Csm_KeyDeriveKey interface.

8.3.8.6.1 Csm_KeyDerive

[SWS_Csm_00956]⌈

Service Name	Csm_KeyDerive	

Syntax	`Std_ReturnType Csm_KeyDerive (`
	`  uint32 keyId,`
	`  uint32 targetKeyId`
	`)`	

Service ID [hex]	0x6b	
Sync/Async	Synchronous	
Reentrancy	Reentrant, but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key which is used for key derivation.
	targetKeyId	Holds the identifier of the key which is used to store the derived key.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_READ_FAIL: Request failed, not allowed to extract key element CRYPTO_E_KEY_WRITE_FAIL: Request failed, not allowed to write key element

		CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element sizes are not compatible CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Derives a new key by using the key elements in the given key identified by the keyId. The given key contains the key elements for the password and salt. The derived key is stored in the key element with the id 1 of the key identified by targetCryptoKeyId.	
Available via	Csm.h	

⌋(SRS_CryptoStack_00103)**Csm_KeyGenerate**

[SWS_Csm_01018] ⌈ If no errors are detected by Csm, the service `Csm_KeyDerive()` shall call `CryIf_KeyDerive()`.
⌋ ( )

[SWS_Csm_01019] ⌈ If the number of iterations for the key derivation is needed by the Crypto Driver, it shall be stored in the key element `CRYPTO_KE_KEYDERIVATION_ITERATIONS`.
⌋ ( )

8.3.8.7 Key Exchange Interface

Two users that each have a private secret can use a key exchange protocol to obtain a common secret, e.g. a key for a symmetric-key algorithm, without telling each other their private secret and without any listener being able to obtain the common secret or their private secrets

The functions `Csm_KeyExchangeCalcPubVal()` / `Csm_JobKeyExchangeCalcPubVal()` and `Csm_KeyExchangeCalcSecret()` / `Csm_JobKeyExchangeCalcSecret()` are used to support Diffie-Hellman (DH) key exchange.
This allows two partners, Alice and Bob, to generate private and public key material, to exchange public parts so that both parties can generate at the end a common shared secret. This shared secret can further be used, e.g. for symmetric data operation such as data encryption or MAC generation.
The public and private key material can either be based on prime based large number as it is used with RSA or on elliptic curve (so-called elliptic-curve diffie-hellman).

The CSM key exchange functions require a key with key elements according to [SWS_Csm_01022], in the line of Crypto Service "Key Exchange". The key elements `CRYPTO_KE_KEYEXCHANGE_BASE, CRYPTO_KE_KEYEXCHANGE_PRIVKEY` and `CRYPTO_KE_KEYEXCHANGE_OWNPUBKEY` are used to hold the public/private key material.
These values can either be pre-defined and set by `Csm_KeyElementSet()` followed by `Csm_KeySetValid()` or generated. For example, these key values can

be generated by Csm_KeyGenerate() and then copied with Csm_KeyElementCopy() to the corresponding key elements, followed by a call to Csm_KeySetValid().
In a first step, Alice will call Csm_KeyExchangeCalcPubVal() / Csm_JobKeyExchangeCalcPubVal() and send the results to Bob (exchanged data may need to be signed and/or encrypted depending on the protocol).
It should be noted, that if KeyExchangeCalcPubVal is called but no valid key material exists (key is not valid or essential key elements have length=0), the function shall generate the necessary key material and continue as normal.
If needed, Bob will put received key material from Alice into the corresponding key elements. He will also call Csm_KeyExchangeCalcPubVal() to generate his shared value that needs to be sent to Alice. Afterwards, Bob can call Csm_KeyExchangeCalcSecret() to generate the common secret. This value will be placed into the key element CRYPTO_KE_KEYEXCHANGE_SHAREDVALUE.
When Alice receives the public value from Bob, it will call KeyExchangeCalcSecret() and provides the value from Bob in the parameter of the function. The common shared secret will be generated by this function into the key element CRYPTO_KE_KEYEXCHANGE_SHAREDVALUE.
Depending on the algorithm, Bob needs to send key material to Alice to allow her to generate the common shared secret.

The key element CRYPTO_KE_KEYEXCHANGE_ALGORITHM specifies the Diffie-Hellman algorithm. The key element value is of type Crypto_AlgorithmFamily, for example CRYPTO_ALGOFAM_DH (for modulo based DH) or CRYPTO_ALGOFAM_ED25519 (for ECDH(E)).
Additional elliptic curve parameter can be specified with the additional key element CRYPTO_KE_KEYEXCHANGE_CURVE.

The other key elements have the following meaning:

	DH(E)	ECDH(E)
CRYPTO_KE_KEYEXCHANGE_BASE	Modulo	Generator point
CRYPTO_KE_KEYEXCHANGE_PRIVKEY	Local exponent	Private key
CRYPTO_KE_KEYEXCHANGE_OWNPUBKEY	Generator	Public key

8.3.8.7.1 Csm_KeyExchangeCalcPubVal

[SWS_Csm_00966]⌈

Service Name	Csm_KeyExchangeCalcPubVal

Syntax	`Std_ReturnType Csm_KeyExchangeCalcPubVal (`
	`  uint32 keyId,`
	`  uint8* publicValuePtr,`
	`  uint32* publicValueLengthPtr`
	`)`

Service ID [hex]	0x6c
Sync/Async	Synchronous

Reentrancy	Reentrant but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key which shall be used for the key exchange protocol.
Parameters (inout)	public Value LengthPtr	Holds a pointer to the memory location in which the public value length information is stored. On calling this function, this parameter shall contain the size of the buffer provided by publicValuePtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	public ValuePtr	Contains the pointer to the data where the public value shall be stored.
Return value	Std_-Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Calculates the public value of the current user for the key exchange and stores the public key in the memory location pointed by the public value pointer.	
Available via	Csm.h	

⌋(SRS_CrytptoStack_00028)

[SWS_Csm_01020] ⌈ If no errors are detected by Csm, the service `Csm_KeyExchangeCalcPubVal()` shall call `CryIf_KeyExchangeCalcPubVal()`.
⌋ ( )

8.3.8.7.2 Csm_KeyExchangeCalcSecret

[SWS_Csm_00967]⌈

Service Name	Csm_KeyExchangeCalcSecret	

Syntax	`Std_ReturnType Csm_KeyExchangeCalcSecret (`
	`  uint32 keyId,`
	`  const uint8* partnerPublicValuePtr,`
	`  uint32 partnerPublicValueLength`
	`)`	

Service ID [hex]	0x6d	
Sync/Async	Synchronous	
Reentrancy	Reentrant but not for same keyId	
Parameters (in)	keyId	Holds the identifier of the key which shall be used for the key exchange protocol.
	partnerPublic ValuePtr	Holds the pointer to the memory location which contains the partner's public value.

	partnerPublic ValueLength	Contains the length of the partner's public value in bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypto Driver Object is busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Calculates the shared secret key for the key exchange with the key material of the key identified by the keyId and the partner public key. The shared secret key is stored as a key element in the same key.	
Available via	Csm.h	

⌋(SRS_CrytptoStack_00028)

[SWS_Csm_01006] ⌈ If no errors are detected by Csm, the service `Csm_KeyExchangeCalcSecret()` shall call `CryIf_KeyExchangeCalcSecret()`.
⌋()

8.3.9 Cryptographic Primitives and Schemes

8.3.9.1 Csm_JobKeySetValid

[SWS_Csm_91027]⌈

Service Name	Csm_JobKeySetValid	

Syntax	`Std_ReturnType Csm_JobKeySetValid (`
	`  uint32 jobId`
	`)`	

Service ID [hex]	0x7a	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	None	
Parameters (out)	None	

Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypro Driver Object is busy
Description	Stores the key if necessary and sets the key state of the key identified by keyId to valid.	
Available via	Csm.h	

⌋()

8.3.9.2 Csm_JobKeySetInvalid

[SWS_Csm_91002]⌈

Service Name	Csm_JobKeySetInvalid	

Syntax	`Std_ReturnType Csm_JobKeySetInvalid (`
	`  uint32 jobId`
	`)`	

Service ID [hex]	0x84	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, Crypro Driver Object is busy
Description	Sets the key status to invalid. The key cannot be used any longer for cryptographic operations until it has been set to valid state again.	
Available via	Csm.h	

⌋()

8.3.9.3 Csm_JobRandomSeed

[SWS_Csm_91028]⌈

Service Name	Csm_JobRandomSeed

Syntax	`Std_ReturnType Csm_JobRandomSeed (`
	`  uint32 jobId,`
	`  const uint8* seedPtr,`
	`  uint32 seedLength`
	`)`	

Service ID [hex]	0x7b	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	seedPtr	Holds a pointer to the memory location which contains the data to feed the seed.
	seedLength	Contains the length of the seed in bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid"
Description	Provides a new seed for the specified key that is used for an associated random number generator.	
Available via	Csm.h	

⌋()

8.3.9.4 **Csm_JobKeyGenerate**

[SWS_Csm_91029]⌈

Service Name	Csm_JobKeyGenerate	

Syntax	`Std_ReturnType Csm_JobKeyGenerate (`
	`  uint32 jobId`
	`)`	

Service ID [hex]	0x7c	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	None	

Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Generates new key material and stores it in the key identified by keyId.	
Available via	Csm.h	

⌋()

8.3.9.5 **Csm_JobKeyDerive**

[SWS_Csm_91030]⌈

Service Name	Csm_JobKeyDerive	

Syntax	`Std_ReturnType Csm_JobKeyDerive (`
	`  uint32 jobId,`
	`  uint32 targetKeyId`
	`)`	

Service ID [hex]	0x7d	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	targetKeyId	Holds the identifier of the key which is used to store the derived key.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_READ_FAIL: Request failed, not allowed to extract key element CRYPTO_E_KEY_WRITE_FAIL: Request failed, not allowed to write key element CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_SIZE_MISMATCH: Request failed, key element sizes are not compatible CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element

Description	Derives a new key by using the key elements in the given key identified by the keyId. The given key contains the key elements for the password and salt. The derived key is stored in the key element with the id 1 of the key identified by targetCryptoKeyId.
Available via	Csm.h

⌋()

8.3.9.6 Csm_JobKeyExchangeCalcPubVal

[SWS_Csm_91031]⌈

Service Name	Csm_JobKeyExchangeCalcPubVal	

Syntax	`Std_ReturnType Csm_JobKeyExchangeCalcPubVal (`
	`  uint32 jobId,`
	`  uint8* publicValuePtr,`
	`  uint32* publicValueLengthPtr`
	`)`	

Service ID [hex]	0x7e	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	publicValueLengthPtr	Holds a pointer to the memory location in which the public value length information is stored. On calling this function, this parameter shall contain the size of the buffer provided by publicValuePtr. When the request has finished, the actual length of the returned value shall be stored.
Parameters (out)	publicValuePtr	Contains the pointer to the data where the public value shall be stored.
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Calculates the public value of the current user for the key exchange and stores the public key in the memory location pointed by the public value pointer.	
Available via	Csm.h	

⌋()

8.3.9.7 **Csm_JobKeyExchangeCalcSecret**

[SWS_Csm_91032]⌈

Service Name	Csm_JobKeyExchangeCalcSecret	

Syntax	`Std_ReturnType Csm_JobKeyExchangeCalcSecret (`
	`  uint32 jobId,`
	`  const uint8* partnerPublicValuePtr,`
	`  uint32 partnerPublicValueLength`
	`)`	

Service ID [hex]	0x7f	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
	partnerPublicValuePtr	Holds the pointer to the memory location which contains the partner's public value.
	partnerPublicValueLength	Contains the length of the partner's public value in bytes.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_KEY_NOT_VALID: Request failed, the key's state is "invalid" CRYPTO_E_KEY_EMPTY: Request failed because of uninitialized source key element
Description	Calculates the shared secret key for the key exchange with the key material of the key identified by the keyId and the partner public key. The shared secret key is stored as a key element in the same key.	
Available via	Csm.h	

⌋()

8.3.10 **Context Save and Restore**

8.3.10.1 **Csm_SaveContextJob**

[SWS_Csm_91063]⌈

Service Name	Csm_SaveContextJob

Syntax	`Std_ReturnType Csm_SaveContextJob (`
	`  uint32 jobId,`
	`  uint8* contextBufferPtr,`
	`  uint32* contextBufferLengthPtr`
	`)`	

Service ID [hex]	0x86	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job using the CSM service.
Parameters (inout)	contextBufferLengthPtr	Pointer to the buffer, where the length value is located. As input data it provides the maximum length of data available in contextBufferPtr. As output data it provides the actual length of data located in contextBufferPtr (or 0 in case of a failure)
Parameters (out)	contextBufferPtr	Pointer to the buffer in the application where the context data shall be stored to.
Return value	Std_ReturnType	E_OK: Context data successfully provided. E_NOT_OK: Context data could not be provided; values are not valid. CRYPTO_E_BUSY: Request failed, service is still busy
Description	The Crypto Driver stores the internal context of the respective crypto operation to the contextBuffer.	
Available via	Csm.h	

⌋()

[SWS_Csm_91067] ⌈ If `Csm_SaveContextJob()` is called and the job is active, CSM shall put `contextBufferPtr` to `job.PrimitiveInputOutput.outputPtr` and `contextOutputLengthPtr` into `job.PrimitiveInputOutput.outputLengthPtr`, set `job->jobPrimitiveInputOutput->mode` to `CRYPTO_OPERATIONMODE_SAVE_CONTEXT` and call `CryIf_ProcessJob()` with the corresponding `ChannelId` of this job and a pointer to the job itself. Any return value from this function call shall be provided back to the application.
⌋ ()

8.3.10.2 Csm_RestoreContextJob

[SWS_Csm_91064]⌈

Service Name	Csm_RestoreContextJob

Syntax	`Std_ReturnType Csm_RestoreContextJob (`
	`  uint32 jobId,`
	`  const uint8* contextBufferPtr,`

	uint32 contextBufferLength)	
Service ID [hex]	0x87	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant	
Parameters (in)	jobId	Holds the identifier of the job where context shall be requested from.
	contextBufferPtr	Pointer to the buffer, where the context data are located that shall be restored.
	contextBuffer Length	Provides the length of context data that are located in context BufferPtr.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Context data successfully restored. E_NOT_OK: Context data could not be restored. CRYPTO_E_BUSY: Request failed, service is still busy
Description	The Crypto Driver extracts the context data from the contextBuffer and restores the internal state so that further crypto operation of this crypto service will continue at the exact point when the context was taken.	
Available via	Csm.h	

⌋()

[SWS_Csm_91068] ⌈ If `Csm_RestoreContextJob ()` is called and the job is active, CSM shall put `contextBufferPtr` into `job.PrimitiveInputOutput.inputPtr` and `contextBufferLength` into `job.PrimitiveInputOutput.inputLength`, set `job->jobPrimitiveInputOutput->mode` to `CRYPTO_OPERATIONMODE_RESTORE_CONTEXT` and call `CryIf_ProcessJob()` with the corresponding `ChannelId` of this job and a pointer to the job itself. Any return value from this function call shall be provided back to the application.
⌋ ()

8.3.11 **Job Cancellation Interface**

8.3.11.1 **Csm_CancelJob**

[SWS_Csm_00968]⌈

Service Name	Csm_CancelJob

Syntax	Std_ReturnType Csm_CancelJob (
	  uint32 job,
	  Crypto_OperationModeType mode
	)	

Service ID [hex]	0x6f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	job	Holds the identifier of the job to be canceled
	mode	Not used, just for interface compatibility provided.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-Return-Type	E_OK: Request successful. Job removed from any queue and potentially from crypto driver hardware. E_NOT_OK: Request failed CRYPTO_E_JOB_CANCELED: Immediate cancelation not possible. The cancelation will be done at next suitable processing step and notified via a negative job's closing callback.
Description	Cancels the job processing from asynchronous or streaming jobs.	
Available via	Csm.h	

⌋()

[SWS_Csm_01086] ⌈ If development error detection for the CSM is enabled: The function `Csm_CancelJob()` shall raise the error `CSM_E_PROCESSING_MODE` and return `E_NOT_OK` if the `Csm_CancelJob()` is called for a synchronous job.
⌋ ( )

[SWS_Csm_01021] ⌈ The Csm shall remove the job from its own queue or call CryIf_CancelJob() to cancel a potential job in the driver.⌋ ( )

[SWS_Csm_01030] ⌈ In case the `CryIf_CancelJob()` returns `E_OK`, the job's closing callback CallbackNotification shall be called with a result value of `CRYPTO_E_JOB_CANCELED`.
⌋ ( )

[SWS_Csm_01087] ⌈In case the `CryIf_CancelJob()` returns `CRYPTO_E_JOB_CANCELED` (i.e. the job was not instantly cancled) the CSM shall postpone the call of the job's closing callback until the next call of `Csm_CallbackNotification()`. The result of the job's closing callback shall be `CRYPTO_E_JOB_CANCELED`.

⌋ ( )

Note: In case the crypto driver does not support an instant cancelation of the job, the application need to wait for the job's closing callback to free the buffers. The crypto driver could potentially still write to the output buffer(s).

8.3.12 **Custom Service Interface**

8.3.12.1 **Custom Service**

Security related services that are not supported by the existing CSM service interfaces can be realized by the custom service. The purpose of this service is not specified and can be customized by a vendor. It is recommended to align the settings of a particular custom service in a Crypto Driver profile (see SWS Crypto Driver Chapter 7.5.2 . [16]).

[SWS_Csm_91107]⌈

Service Name	Csm_CustomService	

Syntax	`Std_ReturnType Csm_CustomService (`
	`  uint32 jobId,`
	`  Crypto_OperationModeType mode,`
	`  uint32 targetKeyId,`
	`  const uint8* inputPtr,`
	`  uint32 inputLength,`
	`  const uint8* secondaryInputPtr,`
	`  uint32 secondaryInputLength,`
	`  const uint8* tertiaryInputPtr,`
	`  uint32 tertiaryInputLength,`
	`  uint8* outputPtr,`
	`  uint32* outputLengthPtr,`
	`  uint8* secondaryOutputPtr,`
	`  uint32* secondaryOutputLengthPtr,`
	`  Crypto_VerifyResultType* verifyPtr`
	`)`	

Service ID [hex]	0x13	
Sync/Async	Depends on configuration	
Reentrancy	Reentrant for different jobId	
Parameters (in)	jobId	ID of the job which is dispatched for custom processing
	mode	Indicates which operation mode(s) to perform for the custom Job
	targetKeyId	Holds the target key id for the custom job
	inputPtr	Pointer to the input data.
	inputLength	Contains the input length in bytes.
	secondaryInputPtr	Pointer to the secondary input data.
	secondaryInputLength	Contains the secondary input length in bytes.

	tertiaryInputPtr	Pointer to the tertiary input data.
	tertiaryInputLength	Contains the tertiary input length in bytes.
Parameters (inout)	None	
Parameters (out)	outputPtr	Pointer to the output data.
	outputLengthPtr	Contains the output length in bytes.
	secondaryOutputPtr	Pointer to the secondary output data.
	secondaryOutput LengthPtr	Contains the secondary output length in bytes.
	verifyPtr	Pointer to the verification result
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: Request failed, service is still busy CRYPTO_E_CUSTOM_ERROR: Custom processing failed.
Description	Dispatches security related jobs for custom execution in a secure environment	
Available via	Csm.h	

⌋()

[SWS_Csm_91108]⌈

Service Name	Csm_CustomSync	

Syntax	`Std_ReturnType Csm_CustomSync (`
	`  uint32 dispatchId,`
	`  uint32 keyId,`
	`  uint32 keyElementId,`
	`  uint32 targetKeyId,`
	`  uint32 targetKeyElementId,`
	`  const uint8* inputPtr,`
	`  uint32 inputLength,`
	`  uint8* outputPtr,`
	`  uint32* outputLengthPtr,`
	`  uint8* secondaryOutputPtr,`
	`  uint32* secondaryOutputLengthPtr`
	`)`	

Service ID [hex]	0x88	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	dispatchId	unique id to identify the request
	keyId	key Id of the key the certificate is stored

	keyElementId	key element id
	targetKeyId	Holds the target key id
	targetKeyElementId	--
	inputPtr	Pointer to the input data.
	inputLength	Contains the input length in bytes.
Parameters (inout)	None	
Parameters (out)	outputPtr	Pointer to the output data.
	outputLengthPtr	Contains the output length in bytes.
	secondaryOutputPtr	Pointer to the secondary output data.
	secondaryOutput LengthPtr	Contains the secondary output length in bytes.
Return value	Std_ReturnType	E_OK: Request successful E_NOT_OK: Request failed CRYPTO_E_BUSY: The service request failed because the service is still busy CRYPTO_E_CUSTOM_ERROR: Custom processing failed
Description	Requests the execution of a function that is specified by the given dispatch id.	
Available via	Csm.h	

⌋()

[SWS_Csm_91110] If no errors are detected by Csm, the service `Csm_CustomSync()` shall call `CryIf_CustomSync()`.

8.3.13 **Callback Notifications**

8.3.13.1 **Csm_CallbackNotification**

[SWS_Csm_00970]⌈

Service Name	Csm_CallbackNotification

Syntax	`void Csm_CallbackNotification (`
	`  Crypto_JobType* job,`
	`  Crypto_ResultType result`
	`)`

Service ID [hex]	0x70
Sync/Async	Synchronous
Reentrancy	Reentrant

Parameters (in)	job	Holds a pointer to the job, which has finished.
	result	Contains the result of the cryptographic operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Notifies the CSM that a job has finished. This function is used by the underlying layer (CRYIF).	
Available via	Csm.h	

⌋(SRS_BSW_00359, SRS_BSW_00360)

[SWS_Csm_01044]⌈If the `CRYPTO_OPERATIONMODE_FINISH` bit is set in `job->jobPrimitiveInputOutput.mode`, the `Csm_CallbackNotification` shall call the configured callback function.

⌋()

[SWS_Csm_91017]⌈If the `CRYPTO_OPERATIONMODE_FINISH` bit is set in `job->jobPrimitiveInputOutput.mode` and `CsmProcessingMode` is set to `CRYPTO_PROCESSING_ASYNC` and `CsmJobInterfaceUsePort` is set to CRYPTO_USE_PORT_OPTIMIZED, the `CSM` shall trigger CallbackNotification service.

⌋()

8.3.14 Scheduled functions

8.3.14.1 Csm_MainFunction

[SWS_Csm_00479]⌈

Service Name	Csm_MainFunction

Syntax	`void Csm_MainFunction (`
	`  void`
	`)`

Service ID [hex]	0x01
Description	API to be called cyclically to process the requested jobs. The Csm_MainFunction shall check the queues for jobs to pass to the underlying CRYIF. Per configured Csm MainFunction instance one Csm_MainFunction_<shortName> shall be implemented. Hereby <shortName> is the short name of the CsmMainFunction configuration container in the ECU configuration.

Available via	SchM_Csm.h

⌋(SRS_BSW_00373, SRS_BSW_00432)

8.4 Expected Interfaces

8.4.1 Interfaces to Standard Software Modules

8.4.2 Mandatory Interfaces

[SWS_Csm_91100]⌈

API Function	***Header File***	***Description***
CryIf_CancelJob	CryIf.h	This interface dispatches the job cancellation function to the configured crypto driver object.
CryIf_KeyCopy	CryIf.h	This function shall copy all key elements from the source key to a target key.
CryIf_KeyElementCopy	CryIf.h	This function shall copy a key elements from one key to a target key.
CryIf_KeyElementCopyPartial	CryIf.h	Copies a key element to another key element. The keyElementOffsets and keyElementCopyLength allows to copy just parts of the source key element into the destination key element.
CryIf_KeyElementGet	CryIf.h	This function shall dispatch the get key element function to the configured crypto driver object.
CryIf_KeyElementSet	CryIf.h	This function shall dispatch the set key element function to the configured crypto driver object.
CryIf_KeyExchangeCalcSecret	CryIf.h	This function shall dispatch the key exchange common shared secret calculation function to the configured crypto driver object.
CryIf_KeyGenerate	CryIf.h	This function shall dispatch the key generate function to the configured crypto driver object.
CryIf_KeySetValid	CryIf.h	This function shall dispatch the set key valid function to the configured crypto driver object.
CryIf_ProcessJob	CryIf.h	This interface dispatches the received jobs to the configured crypto driver object.
CryIf_RandomSeed	CryIf.h	This function shall dispatch the random seed function to the configured crypto driver object.
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.

⌋()

8.4.3 Optional Interfaces

[SWS_Csm_91101]⌈

API Function	*Header File*	*Description*
Det_ReportError	Det.h	Service to report development errors.

⌋()

8.4.4 Configurable interfaces

8.4.4.1 <Csm_ApplicationCallbackNotification>

[SWS_Csm_00971]⌈

Service Name	<Csm_ApplicationCallbackNotification>	

Syntax	`void <Csm_ApplicationCallbackNotification> (`
	`  uint32 jobId,`
	`  Crypto_ResultType result`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	jobId	JobID of the operation that caused the callback
	result	Contains the result of the cryptographic operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	CSM notifies the application that a job has finished. The function name is configurable. The function name itself is derived from "{CsmJob/CsmJobPrimitive CallbackRef}/CsmCallbackFunc".	
Available via	Csm.h	

⌋(SRS_BSW_00359, SRS_BSW_00360)

[SWS_Csm_01090] ⌈`Csm_ApplicationCallbackNotification` shall be called once at the end when an asynchronous job's call has been finished, i.e. the given operation mode has been completely processed, the job has been aborted due to an error or the the job has been cancelled. Thus, if a job's call processed multiple operation modes, i.e. `CRYPTO_OPERATIONMODE_STREAMSTART` or `CRYPTO_OPERATIONMODE_SINGLECALL`, `Csm_ApplicationCallbackNotification` is called only once.

⌋()

[SWS_Csm_01095] ⌈ The CSM shall call the application callback function if the following condition is met:

```
({ecuc(Csm/CsmJobs/CsmJob.CsmProcessingMode)} == CRYPTO_PROCESSING_ASYN) &&
(CsmJob/CsmJobInterfaceUsePort == CRYPTO_USE_FNC) &&
(CsmJob/CsmJobPrimitiveCallbackRef != 0)
```

For the service interface the callback service shall be called if the asynchronous processing is configured:

```
({ecuc(Csm/CsmJobs/CsmJob.CsmProcessingMode)} == CRYPTO_PROCESSING_ASYN) &&
(CsmJob/CsmJobInterfaceUsePort != CRYPTO_USE_FNC)
```

⌋()

8.5 Service Interface

This chapter is an addition to the specification of the Csm module. Whereas the other parts of the specification define the behavior and the C-interfaces of the corresponding basic software module, this chapter formally specifies the corresponding AUTOSAR service in terms of the SWC template. The interfaces described here will be visible on the VFB and are used to generate the RTE between application software and the Csm module.

8.5.1 Client-Server-Interfaces

8.5.1.1 CsmKeyManagement_{Key}

[SWS_Csm_01905]⌈

Name	CsmKeyManagement_{Key}		
Comment	Interface to execute the key management functions.		
IsService	true		
Variation	({ecuc(Csm/CsmKeys/CsmKey.CsmKeyUsePort)} == TRUE) Key = {ecuc(Csm/CsmKeys/CsmKey.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.

	6	CRYPTO_E_KEY_READ_FAIL	The service request failed because read access was denied.
	7	CRYPTO_E_KEY_WRITE_FAIL	The service request failed because write access was denied.
	8	CRYPTO_E_KEY_NOT_AVAILABLE	The service request failed because the key is not available.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	KeyCopy	
Comment	This function shall copy all key elements from the source key to a target key.	
Mapped to API	Csm_KeyCopy	
Variation	--	
Parameters	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key whose key element shall be the destination element.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_AVAILABLE CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	KeyDerive
Comment	Derives a new key by using the key elements in the given key. The given key contains the key elements for the password and salt. The derived key is stored in the key element with the id 1 of the key identified by targetCryptoKeyId.
Mapped to API	Csm_KeyDerive
Variation	--

Parameters	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key which is used to store the derived key.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	KeyElementCopy	
Comment	This function shall copy a key elements from one key to a target key	
Mapped to API	Csm_KeyElementCopy	
Variation	--	
Parameters	keyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element which shall be the source for the copy operation.
	Variation	--
	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key whose key element shall be the destination element.
	Variation	--
	targetKeyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element which shall be the destination for the copy operation.

	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_AVAILABLE CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	KeyElementCopyPartial	
Comment	This function shall copy parts of a a key elements from one key to parts of a target key element of a target key.	
Mapped to API	Csm_KeyElementCopyPartial	
Variation	--	
Parameters	keyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element which shall be the source for the copy operation.
	Variation	--
	keyElementSourceOffset	
	Type	uint32
	Direction	IN
	Comment	This is the offset of the source key element indicating the start index of the copy operation.
	Variation	--
	keyElementTargetOffset	
	Type	uint32
	Direction	IN
	Comment	This is the offset of the destination key element indicating the start index of the copy operation.
	Variation	--
	keyElementCopyLength	
	Type	uint32
	Direction	IN

	Comment	Specifies the number of bytes that shall be copied.
	Variation	--
	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key whose key element shall be the destination element.
	Variation	--
	targetKeyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element which shall be the destination for the copy operation.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_AVAILABLE CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	KeyElementGet	
Comment	Retrieves the key element bytes from a specific key element of the key and stores the key element in the provided buffer.	
Mapped to API	Csm_KeyElementGet	
Variation	--	
Parameters	keyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element to be read.
	Variation	--
	keyElement	
	Type	Csm_KeyDataType_{Crypto}
	Direction	OUT

	Comment	Holds the data to the key element bytes to be written.
	Variation	--
	keyElementLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the number of key element bytes.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_NOT_AVAILABLE CRYPTO_E_KEY_EMPTY	

Operation	KeyElementSet	
Comment	Sets the given key element bytes to the key.	
Mapped to API	Csm_KeyElementSet	
Variation	--	
Parameters	keyElementId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key element to be written.
	Variation	--
	keyElement	
	Type	Csm_KeyDataType_{Crypto}
	Direction	IN
	Comment	Holds the data to the key element bytes to be processed.
	Variation	--
	keyElementLength	
	Type	uint32
	Direction	IN
	Comment	Contains the number of key element bytes.
	Variation	--

Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_AVAILABLE CRYPTO_E_KEY_SIZE_MISMATCH

Operation	KeyExchangeCalcPubVal	
Comment	Calculates the public value of the current user for the key exchange and stores the public key in the provided buffer	
Mapped to API	Csm_KeyExchangeCalcPubVal	
Variation	--	
Parameters	publicValue	
	Type	Csm_KeyDataType_{Crypto}
	Direction	OUT
	Comment	Contains the pointer to the memory location where the public value shall be stored.
	Variation	--
	publicValueLength	
	Type	uint32
	Direction	INOUT
	Comment	Holds a pointer to the memory location in which the public value length in bytes is stored. On calling this function, this parameter shall contain the size of the buffer in bytes provided by publicValuePtr. When the request has finished, the actual length of the returned value shall be stored.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY	

Operation	KeyExchangeCalcSecret
Comment	Calculates the shared secret key for the key exchange with the key material of the key identified by the keyId and the partner public key. The shared secret key is stored as a key element in the same key.
Mapped to API	Csm_KeyExchangeCalcSecret
Variation	--

Parameters	partnerPublicValue	
	Type	Csm_KeyDataType_{Crypto}
	Direction	IN
	Comment	Holds the pointer to the memory location containing the partner's public value
	Variation	--
	partnerPublicValueLength	
	Type	uint32
	Direction	IN
	Comment	Contains the number of bytes of the partner pulic value
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY	

Operation	KeyGenerate
Comment	Generates new key material and store it in the key identified by keyId.
Mapped to API	Csm_KeyGenerate
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY

Operation	KeyGetStatus	
Comment	Returns the key state of the key identified by keyId.	
Mapped to API	Csm_KeyGetStatus	
Variation	--	
Parameters	keyStatusPtr	
	Type	Crypto_KeyStatusType
	Direction	OUT
	Comment	Holds the status of the key referenced by the port.
	Variation	--

Possible Errors	E_OK E_NOT_OK

Operation	KeySetInvalid
Comment	Operation to set the status of the key to invalid.
Mapped to API	Csm_KeySetInvalid
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY

Operation	KeySetValid
Comment	Sets the given key element bytes to the key.
Mapped to API	Csm_KeySetValid
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY

Operation	RandomSeed	
Comment	Feeds the key element CRYPTO_KE_RANDOM_SEED with a random seed.	
Mapped to API	Csm_RandomSeed	
Variation	--	
Parameters	seed	
	Type	Csm_KeyDataType_{Crypto}
	Direction	IN
	Comment	Holds the data which shall be used for the random seed initialization.
	Variation	--
	seedLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length of the seed in bytes.
	Variation	--
Possible Errors	E_OK E_NOT_OK	

	CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY

⌋(SRS_Csm_00066)

8.5.1.2 **CsmHash_{PrimitiveCfg}**

[SWS_Csm_00946]⌈

Name	CsmHash_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the hash calculation.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.

Operation	Hash	
Comment	Streaming approach of the hash calculation.	
Mapped to API	Csm_Hash	
Variation	--	
Parameters	data	
	Type	Csm_HashDataType_{Crypto}
	Direction	IN
	Comment	Contains the data to be hashed.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data to be hashed.
	Variation	--
	result	
	Type	Csm_HashResultType_{Crypto}

	Direction	OUT
	Comment	Contains the data of the hash.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	resultLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the hash.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY	

⌋(SRS_CryptoStack_00090)

8.5.1.3 **CsmMacGenerate_{PrimitiveCfg}**

[SWS_Csm_09000]⌈

Name	CsmMacGenerate_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the MAC generation.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	MacGenerate
Comment	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.
Mapped to API	Csm_MacGenerate

Variation	--	
Parameters	data	
	Type	Csm_MacGenerateDataType_{Crypto}
	Direction	IN
	Comment	Contains the data from which a MAC shall be generated of.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data from which a MAC shall be generated of.
	Variation	--
	mac	
	Type	Csm_MacGenerateResultType_{Crypto}
	Direction	OUT
	Comment	Contains the data of the MAC.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	macLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the MAC.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.4 CsmMacVerify_{PrimitiveCfg}

[SWS_Csm_00936]⌈

Name	CsmMacVerify_{PrimitiveCfg}
Comment	Synchronous processing interface to execute the MAC verification.

IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	MacVerify	
Comment	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.	
Mapped to API	Csm_MacVerify	
Variation	--	
Parameters	data	
	Type	Csm_MacVerifyDataType_{Crypto}
	Direction	IN
	Comment	Contains the data from which a MAC shall be generated of.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data for whichs MAC shall be verified.
	Variation	--
	mac	
	Type	Csm_MacVerifyCompareType_{Crypto}
	Direction	IN
	Comment	Contains the MAC to be verified.

	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	macLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in BITS of the MAC to be verified.
	Variation	--
	verify	
	Type	Crypto_VerifyResultType
	Direction	OUT
	Comment	Contains the verification result.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.5 **CsmEncrypt_{PrimitiveCfg}**

[SWS_Csm_00947]⌈

Name	CsmEncrypt_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the encryption.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	Encrypt	
Comment	Encrypts the given data and store the ciphertext in the memory location pointed by the result pointer.	
Mapped to API	Csm_Encrypt	
Variation	--	
Parameters	data	
	Type	Csm_EncryptDataType_{Crypto}
	Direction	IN
	Comment	Contains the data to be encrypted.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data to be encrypted.
	Variation	--
	result	
	Type	Csm_EncryptResultType_{Crypto}
	Direction	OUT
	Comment	Contains the data of the cipher.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	resultLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the cipher.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00906)

8.5.1.6 CsmDecrypt_{PrimitiveCfg}

[SWS_Csm_01906]⌈

Name	CsmDecrypt_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the decryption.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	Decrypt	
Comment	Streaming approach of the decryption.	
Mapped to API	Csm_Decrypt	
Variation	--	
Parameters	data	
	Type	Csm_DecryptDataType_{Crypto}
	Direction	IN
	Comment	Contains the data to be decrypted.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data to be decrypted.
	Variation	--
	result	
	Type	Csm_DecryptResultType_{Crypto}

	Direction	OUT
	Comment	Contains the data of the decrypted plaintext.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	resultLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the decrypted plaintext.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.7 **CsmAEADEncrypt_{PrimitiveCfg}**

[SWS_Csm_01910]⌈

Name	CsmAEADEncrypt_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the AEAD encryption.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	AEADEncrypt
Comment	Streaming approach of the AEAD encryption.

Mapped to API	Csm_AEADEncrypt	
Variation	--	
Parameters	plaintext	
	Type	Csm_AEADEncryptPlaintextType_{Crypto}
	Direction	IN
	Comment	Contains the plaintext to be encrypted with AEAD.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	plaintextLength	
	Type	uint32
	Direction	IN
	Comment	This element Contains the length in bytes of the plaintext to be encrypted with AEAD.
	Variation	--
	associatedData	
	Type	Csm_AEADEncryptAssociatedDataType_{Crypto}
	Direction	IN
	Comment	Contains the data of the header (that is not part of the encryption but authentication).
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	associatedDataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data of the header.
	Variation	--
	ciphertext	
	Type	Csm_AEADEncryptCiphertextType_{Crypto}
	Direction	OUT
	Comment	Contains the data of the AEAD cipher.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	ciphertextLength	
	Type	uint32
	Direction	INOUT

	Comment	Contains the length in bytes of the data of the AEAD cipher.
	Variation	--
	tag	
	Type	Csm_AEADEncryptTagType_{Crypto}
	Direction	OUT
	Comment	Contains the data of the Tag.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	tagLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the data of the Tag.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.8 CsmAEADDecrypt_{PrimitiveCfg}

[SWS_Csm_01915]⌈

Name	CsmAEADDecrypt_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the AEAD decryption.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_ NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_ SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.

	13	CRYPTO_E_KEY_ EMPTY	The service request failed because of uninitialized source key element.

Operation	AEADDecrypt	
Comment	Streaming approach of the AEAD decryption.	
Mapped to API	Csm_AEADDecrypt	
Variation	--	
Parameters	ciphertext	
	Type	Csm_AEADDecryptCiphertextType_{Crypto}
	Direction	IN
	Comment	Contains the ciphertext to be decrypted with AEAD.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	ciphertextLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the ciphertext to be decrypted with AEAD.
	Variation	--
	associatedData	
	Type	Csm_AEADDecryptAssociatedDataType_{Crypto}
	Direction	IN
	Comment	Contains the data of the header (that is not part of the encryption but authentication) .
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	associatedDataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data of the header.
	Variation	--
	tag	
	Type	Csm_AEADDecryptTagType_{Crypto}
	Direction	IN
	Comment	Contains the data of the Tag.

	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	tagLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in BITS of the data of the Tag.
	Variation	--
	plaintext	
	Type	Csm_AEADDecryptPlaintextType_{Crypto}
	Direction	OUT
	Comment	Contains the data of the decrypted AEAD plaintext.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	plaintextLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the data of the decrypted AEAD plaintext.
	Variation	--
	verify	
	Type	Crypto_VerifyResultType
	Direction	OUT
	Comment	Contains the verification result.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.9 **CsmSignatureGenerate_{PrimitiveCfg}**

[SWS_Csm_00903]⌈

Name	CsmSignatureGenerate_{PrimitiveCfg}
Comment	Synchronous processing interface to generate a signature.

IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	SignatureGenerate	
Comment	Streaming approach of the signature generation.	
Mapped to API	Csm_SignatureGenerate	
Variation	--	
Parameters	data	
	Type	Csm_SignatureGenerateDataType_{Crypto}
	Direction	IN
	Comment	Contains the data from which the signature shall be generated.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data from which the signature shall be generated.
	Variation	--
	signature	
	Type	Csm_SignatureGenerateResultType_{Crypto}
	Direction	OUT
	Comment	Contains the signature.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}

	signatureLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the signature.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.10 CsmSignatureVerify_{PrimitiveCfg}

[SWS_Csm_00943]⌈

Name	CsmSignatureVerify_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the signature verification.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	SignatureVerify
Comment	Interface to verify a signature.
Mapped to API	Csm_SignatureVerify
Variation	--
Parameters	data

	Type	Csm_SignatureVerifyDataType_{Crypto}
	Direction	IN
	Comment	Contains the data for whichs signature shall be verified.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data for whichs signature shall be verified.
	Variation	--
	signature	
	Type	Csm_SignatureVerifyCompareType_{Crypto}
	Direction	IN
	Comment	Contains the signature to be verified.
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	signatureLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the signature to be verified.
	Variation	--
	verify	
	Type	Crypto_VerifyResultType
	Direction	OUT
	Comment	Contains the verification result.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.1.11 CsmRandomGenerate_{PrimitiveCfg}

[SWS_Csm_00902]⌈

Name	CsmRandomGenerate_{PrimitiveCfg}		
Comment	Synchronous processing interface to execute the random number generation.		
IsService	true		

Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	4	CRYPTO_E_ENTROPY_EXHAUSTED	Request failed, entropy of random number generator is exhausted.

Operation	RandomGenerate	
Comment	Synchronous processing interface to execute the random number generation.	
Mapped to API	Csm_RandomGenerate	
Variation	--	
Parameters	result	
	Type	Csm_RandomGenerateResultType_{Crypto}
	Direction	OUT
	Comment	Contains the random number
	Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}
	resultLength	
	Type	uint32
	Direction	INOUT
	Comment	Contains the length in bytes of the data of random number.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_ENTROPY_EXHAUSTED	

⌋(SRS_CryptoStack_00090)

8.5.2 Client-Server-Interfaces (DATA_REFERENCES)

8.5.2.1 CsmHash

[SWS_Csm_91051]⌈

Name	CsmHash		
Comment	Asynchronous processing interface to execute the hash calculation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	Hash	
Comment	Utilize the random seed service.	
Mapped to API	Csm_Hash	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data to be hashed.
	Variation	--
	dataLength	
	Type	uint32
	Direction	IN

	Comment	Contains the length in bytes of the data to be hashed.
	Variation	--
	resultPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the hash.
	Variation	--
	resultLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the hash.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY	

⌋(SRS_CryptoStack_00090)

8.5.2.2 **CsmMacGenerate**

[SWS_Csm_91052]⌈

Name	CsmMacGenerate		
Comment	Asynchronous processing interface to execute the MAC generation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.

	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	MacGenerate	
Comment	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.	
Mapped to API	Csm_MacGenerate	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data from which a MAC shall be generated of.
	Variation	--
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data from which a MAC shall be generated of.
	Variation	--
	macPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the MAC.
	Variation	--
	macLengthPtr	
	Type	Csm_LengthPtr

	Direction	IN
	Comment	Contains the length in bytes of the MAC.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.2.3 **CsmMacVerify**

[SWS_Csm_91053]⌈

Name	CsmMacVerify		
Comment	Asynchronous processing interface to execute the MAC verification.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	MacVerify	
Comment	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.	
Mapped to API	Csm_MacVerify	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data from which a MAC shall be generated of.
	Variation	--
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data for whichs MAC shall be verified.
	Variation	--
	macPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the MAC to be verified.
	Variation	--
	macLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in BITS of the MAC to be verified.
	Variation	--
	verifyPtr	
	Type	Csm_VerifyResultPtr
	Direction	IN
	Comment	Contains the verification result.
	Variation	--

Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY

⌋(SRS_CryptoStack_00090)

8.5.2.4 **CsmEncrypt**

[SWS_Csm_91054]⌈

Name	CsmEncrypt		
Comment	Asynchronous processing interface to execute the encryption.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	Encrypt
Comment	Encrypts the given data and stores the ciphertext in the memory location pointed by the result pointer.

Mapped to API	Csm_Encrypt	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data to be encrypted.
	Variation	--
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data to be encrypted.
	Variation	--
	resultPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the cipher.
	Variation	--
	resultLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the cipher.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.2.5 **CsmDecrypt**

[SWS_Csm_91055]⌈

Name	CsmDecrypt

Comment	Asynchronous processing interface to execute the decryption.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	Decrypt	
Comment	Decrypts the given data and stores the plaintext in the memory location pointed by the resultBuffer pointer.	
Mapped to API	Csm_Decrypt	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data to be decrypted.
	Variation	--
	dataLength	

	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data to be decrypted.
	Variation	--
	resultPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the decrypted plaintext.
	Variation	--
	resultLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the decrypted plaintext.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.2.6 **CsmAEADEncrypt**

[SWS_Csm_91056]⌈

Name	CsmAEADEncrypt		
Comment	Asynchronous processing interface to execute the AEAD encryption.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.

	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	AEADEncrypt	
Comment	Streaming approach of the AEAD encryption.	
Mapped to API	Csm_AEADEncrypt	
Variation	--	
Parameters	plaintextPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the plaintext to be encrypted with AEAD.
	Variation	--
	plaintextLength	
	Type	uint32
	Direction	IN
	Comment	This element Contains the length in bytes of the plaintext to be encrypted with AEAD.
	Variation	--
	associatedDataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data of the header (that is not part of the encryption but authentication).
	Variation	--
	associatedDataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data of the header.
	Variation	--

	ciphertextPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the AEAD cipher.
	Variation	--
	ciphertextLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the data of the AEAD cipher.
	Variation	--
	tagPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the Tag.
	Variation	--
	tagLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the data of the Tag.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

⌋(SRS_CryptoStack_00090)

8.5.2.7 CsmAEADDecrypt

[SWS_Csm_91057]⌈

Name	CsmAEADDecrypt		
Comment	Asynchronous processing interface to execute the AEAD decryption.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	AEADDecrypt	
Comment	Streaming approach of the AEAD decryption.	
Mapped to API	Csm_AEADDecrypt	
Variation	--	
Parameters	ciphertextPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the ciphertext to be decrypted with AEAD.
	Variation	--
	ciphertextLength	
	Type	uint32
	Direction	IN

	Comment	Contains the length in bytes of the ciphertext to be decrypted with AEAD.
	Variation	--
	associatedDataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data of the header (that is not part of the encryption but authentication).
	Variation	--
	associatedDataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data of the header.
	Variation	--
	tagPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data of the Tag.
	Variation	--
	tagLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in BITS of the data of the Tag.
	Variation	--
	plaintextPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the data of the decrypted AEAD plaintext.
	Variation	--
	plaintextLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN

	Comment	Contains the length in bytes of the data of the decrypted AEAD plaintext.
	Variation	--
	verifyPtr	
	Type	Csm_VerifyResultPtr
	Direction	IN
	Comment	Contains the verification result.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

⌋(SRS_CryptoStack_00090)

8.5.2.8 **CsmSignatureGenerate**

[SWS_Csm_91058]⌈

Name	CsmSignatureGenerate		
Comment	Asynchronous processing interface to generate a signature.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.

	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	SignatureGenerate	
Comment	Operation to generate a signature.	
Mapped to API	Csm_SignatureGenerate	
Variation	--	
Parameters	dataPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	References the data from which the signature shall be generated.
	Variation	--
	dataLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the data from which the signature shall be generated.
	Variation	--
	resultPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the signature.

	Variation	--
	resultLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the signature.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.2.9 CsmSignatureVerify

[SWS_Csm_91059]⌈

Name	CsmSignatureVerify		
Comment	Asynchronous processing interface to execute the signature verification.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob

Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

<table>
<tr><td><b><i>Operation</i></b></td><td colspan="2">SignatureVerify</td></tr>
<tr><td><b><i>Comment</i></b></td><td colspan="2">Operation to verify a signature.</td></tr>
<tr><td><b><i>Mapped to API</i></b></td><td colspan="2">Csm_SignatureVerify</td></tr>
<tr><td><b><i>Variation</i></b></td><td colspan="2">--</td></tr>
<tr><td rowspan="23"><b><i>Parameters</i></b></td><td colspan="2">dataPtr</td></tr>
<tr><td><b><i>Type</i></b></td><td>ConstVoidPtr</td></tr>
<tr><td><b><i>Direction</i></b></td><td>IN</td></tr>
<tr><td><b><i>Comment</i></b></td><td>References the data for which signature shall be verified.</td></tr>
<tr><td><b><i>Variation</i></b></td><td>--</td></tr>
<tr><td colspan="2">dataLength</td></tr>
<tr><td><b><i>Type</i></b></td><td>uint32</td></tr>
<tr><td><b><i>Direction</i></b></td><td>IN</td></tr>
<tr><td><b><i>Comment</i></b></td><td>Contains the length in bytes of the data for which signature shall be verified.</td></tr>
<tr><td><b><i>Variation</i></b></td><td>--</td></tr>
<tr><td colspan="2">comparePtr</td></tr>
<tr><td><b><i>Type</i></b></td><td>ConstVoidPtr</td></tr>
<tr><td><b><i>Direction</i></b></td><td>IN</td></tr>
<tr><td><b><i>Comment</i></b></td><td>References the signature to be verified.</td></tr>
<tr><td><b><i>Variation</i></b></td><td>--</td></tr>
<tr><td colspan="2">compareLength</td></tr>
<tr><td><b><i>Type</i></b></td><td>uint32</td></tr>
<tr><td><b><i>Direction</i></b></td><td>IN</td></tr>
<tr><td><b><i>Comment</i></b></td><td>Contains the length in bytes of the signature to be verified.</td></tr>
<tr><td><b><i>Variation</i></b></td><td>--</td></tr>
<tr><td colspan="2">verifyPtr</td></tr>
<tr><td><b><i>Type</i></b></td><td>Csm_VerifyResultPtr</td></tr>
<tr><td><b><i>Direction</i></b></td><td>IN</td></tr>
</table>

	Comment	Contains the verification result.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋(SRS_CryptoStack_00090)

8.5.2.10 CsmRandomGenerate

[SWS_Csm_91060]⌈

Name	CsmRandomGenerate		
Comment	Asynchronous processing interface to execute the random number generation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	4	CRYPTO_E_ENTROPY_EXHAUSTED	Request failed, entropy of random number generator is exhausted.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	RandomGenerate
Comment	Generates a random number and stores it in the memory location pointed by the resultBuffer pointer.
Mapped to API	Csm_RandomGenerate

Variation	--	
Parameters	resultPtr	
	Type	VoidPtr
	Direction	IN
	Comment	References the random number.
	Variation	--
	resultLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Contains the length in bytes of the data of random number.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_ENTROPY_EXHAUSTED	

⌋(SRS_CryptoStack_00090)

8.5.2.11 CsmCustom

[SWS_Csm_91109]⌈

Name	CsmCustom		
Comment	Asynchronous processing interface to execute custom crypto operation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	12	CRYPTO_E_JOB_ CANCELED	Request failed because the job has been canceled.
	14	CRYPTO_E_CUSTOM_ ERROR	--

Operation	CancelJob
Comment	Cancels the Job.
Mapped to API	Csm_CancelJob

Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	CustomService	
Comment	Operation to execute custom crypto job.	
Mapped to API	Csm_CustomService	
Variation	--	
Parameters	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	--
	Variation	--
	inputPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	--
	Variation	--
	inputLength	
	Type	uint32
	Direction	IN
	Comment	--
	Variation	--
	secondaryInputPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	--
	Variation	--
	secondaryInputLength	
	Type	uint32
	Direction	IN

	Comment	--
	Variation	--
	tertiaryInputPtr	
	Type	ConstVoidPtr
	Direction	IN
	Comment	--
	Variation	--
	tertiaryInputLength	
	Type	uint32
	Direction	IN
	Comment	--
	Variation	--
	outputPtr	
	Type	VoidPtr
	Direction	IN
	Comment	--
	Variation	--
	outputLengthPtr	
	Type	uint32*
	Direction	IN
	Comment	--
	Variation	--
	secondaryOutputPtr	
	Type	VoidPtr
	Direction	IN
	Comment	--
	Variation	--
	secondaryOutputLengthPtr	
	Type	uint32*
	Direction	IN
	Comment	--

	Variation	--
	verifyPtr	
	Type	Crypto_VerifyResultType*
	Direction	IN
	Comment	--
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_JOB_CANCELED CRYPTO_E_CUSTOM_ERROR	

⌋()

8.5.3 Client-Server-Interfaces (Key Management)

8.5.3.1 CsmJobKeySetValid

[SWS_Csm_91035]⌈

Name	CsmJobKeySetValid		
Comment	Interface to set a key valid.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeySetValid
Comment	Operation to set a key valid.
Mapped to API	Csm_JobKeySetValid
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY

⌋()

8.5.3.2 **CsmJobKeySetInvalid**

[SWS_Csm_91003]⌈

Name	CsmJobKeySetInvalid		
Comment	Interface to set the status of the key to invalid.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	12	CRYPTO_E_JOB_ CANCELED	Request failed because the job has been canceled.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeySetInvalid
Comment	Operation to set the status of the key to invalid.
Mapped to API	Csm_JobKeySetInvalid
Variation	--

Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY

⌋()

8.5.3.3 **CsmJobRandomSeed**

[SWS_Csm_91036]⌈

Name	CsmJobRandomSeed		
Comment	Interface to random seed operation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	RandomSeed	
Comment	Utilize the random seed service.	
Mapped to API	Csm_JobRandomSeed	
Variation	--	
Parameters	seedPtr	
	Type	ConstVoidPtr
	Direction	IN

	Comment	Holds the data which shall be used for the random seed initialization.
	Variation	--
	seedLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length of the seed in bytes.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID	

⌋()

8.5.3.4 **CsmJobKeyGenerate**

[SWS_Csm_91037]⌈

Name	CsmJobKeyGenerate		
Comment	Interface to execute key generation.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--

Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeyGenerate
Comment	Generates new key material and stores it in the key identified by keyId.
Mapped to API	Csm_JobKeyGenerate
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY

⌋()

8.5.3.5 **CsmJobKeyDerive**

[SWS_Csm_91038]⌈

Name	CsmJobKeyDerive		
Comment	Interface to execute key derive.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	6	CRYPTO_E_KEY_READ_FAIL	The service request failed because read access was denied.
	7	CRYPTO_E_KEY_WRITE_FAIL	The service request failed because write access was denied.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	10	CRYPTO_E_KEY_SIZE_MISMATCH	Request failed because the key element is not partially accessible and the provided key element length is too short or too long for that key element.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeyDerive	
Comment	Derives a new key by using the key elements in the given key. The given key contains the key elements for the password and salt. The derived key is stored in the key element with the id 1 of the key identified by targetCryptoKeyId.	
Mapped to API	Csm_JobKeyDerive	
Variation	--	
Parameters	targetKeyId	
	Type	uint32
	Direction	IN
	Comment	Holds the identifier of the key which is used to store the derived key.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_READ_FAIL CRYPTO_E_KEY_WRITE_FAIL CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_SIZE_MISMATCH CRYPTO_E_KEY_EMPTY	

⌋()

8.5.3.6 **CsmJobKeyExchangeCalcPubVal**

[SWS_Csm_91039]⌈

Name	CsmJobKeyExchangeCalcPubVal		
Comment	Interface to execute calculation of the public value for key exchange.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
	0	E_OK	Operation successful

Possible Errors	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeyExchangeCalcPubVal	
Comment	Calculates the public value of the current user for the key exchange and stores the public key in the provided buffer.	
Mapped to API	Csm_JobKeyExchangeCalcPubVal	
Variation	--	
Parameters	publicValuePtr	
	Type	VoidPtr
	Direction	IN
	Comment	Contains the pointer to the memory location where the public value shall be stored.
	Variation	--
	publicValueLengthPtr	
	Type	Csm_LengthPtr
	Direction	IN
	Comment	Holds a pointer to the memory location in which the public value length in bytes is stored. On calling this function, this parameter shall contain the size of the buffer in bytes provided by publicValuePtr. When the request has finished, the actual length of the returned value shall be stored.
	Variation	--

Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY

⌋()

8.5.3.7 **CsmJobKeyExchangeCalcSecret**

[SWS_Csm_91040]⌈

Name	CsmJobKeyExchangeCalcSecret		
Comment	Interface to execute calculation of shared secret for key exchange.		
IsService	true		
Variation	Primitive = {ecuc(Csm/CsmPrimitives.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.
	9	CRYPTO_E_KEY_NOT_VALID	Request failed, the key is not valid.
	12	CRYPTO_E_JOB_CANCELED	Request failed because the job has been canceled.
	13	CRYPTO_E_KEY_EMPTY	The service request failed because of uninitialized source key element.

Operation	CancelJob
Comment	Cancels the job.
Mapped to API	Csm_CancelJob
Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_JOB_CANCELED

Operation	KeyExchangeCalcSecret
Comment	Calculates the shared secret key for the key exchange with the key material of the key identified by the keyId and the partner public key. The shared secret key is stored as a key element in the same key.
Mapped to API	Csm_JobKeyExchangeCalcSecret
Variation	--

Parameters	partnerPublicValuePtr	
	Type	Csm_KeyDataType_{Crypto}
	Direction	IN
	Comment	Holds the pointer to the memory location containing the partner's public value.
	Variation	--
	partnerPublicValueLength	
	Type	uint32
	Direction	IN
	Comment	Contains the number of bytes of the partner public value.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY CRYPTO_E_KEY_NOT_VALID CRYPTO_E_KEY_EMPTY	

⌋()

8.5.4 Client-Server-Interface (Context Service)

8.5.4.1 CsmContextService{Job}

[SWS_Csm_91106]⌈

Name	CsmContextService_{Job}		
Comment	Interface to context data operation		
IsService	true		
Variation	{ecuc(Csm/CsmJobs/CsmJob. CsmJobServiceInterfaceContextUsePort)} == CRYPTO_USE_PORT Job = {ecuc(Csm/CsmJobs/CsmJob.SHORT-NAME)}		
Possible Errors	0	E_OK	Operation successful
	1	E_NOT_OK	Operation failed
	2	CRYPTO_E_BUSY	Request failed, service is still busy.

Operation	RestoreContextJob
Comment	Restore the job context.
Mapped to API	Csm_RestoreContextJob

Variation	--	
Parameters	contextBuffer	
	Type	ConstVoidPtr
	Direction	IN
	Comment	Provides the buffer for the context as [DATA_REFERENCE].
	Variation	--
	contextBufferLength	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the context buffer.
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY	

Operation	SaveContextJob	
Comment	Save the job context to the provided buffer.	
Mapped to API	Csm_SaveContextJob	
Variation	--	
Parameters	contextBufferPtr	
	Type	VoidPtr
	Direction	IN
	Comment	Provides the buffer for the context as [DATA_REFERENCE].
	Variation	--
	contextBufferLengthPtr	
	Type	uint32
	Direction	IN
	Comment	Contains the length in bytes of the context buffer as [DATA_ REFERENCE].
	Variation	--
Possible Errors	E_OK E_NOT_OK CRYPTO_E_BUSY	

⌋()

8.5.5 Client-Server-Interface Callbacks

8.5.5.1 CallbackNotification

[SWS_Csm_00928]⌈

Name	CallbackNotification		
Comment	Interface for the callback notification.		
IsService	true		
Variation	--		
Possible Errors	--	--	--

Operation	CallbackNotification	
Comment	Notifies the application with a return value that the job has finished.	
Mapped to API	Csm_CallbackNotification	
Variation	--	
Parameters	result	
	Type	Crypto_ResultType
	Direction	IN
	Comment	Return value that shall be returned to the application
	Variation	--
Possible Errors	--	

⌋(SRS_CryptoStack_00090)

8.5.6 Implementation Data Types

8.5.6.1 Crypto_OperationModeType

[SWS_Csm_01029]⌈

Name	Crypto_OperationModeType
Kind	Type
Derived from	uint8

Range	CRYPTO_ OPERATIONMODE_START	0x01	Operation Mode is "Start". The job's state shall be reset, i.e. previous input data and intermediate results shall be deleted.
	CRYPTO_ OPERATIONMODE_ UPDATE	0x02	Operation Mode is "Update". Used to calculate intermediate results.
	CRYPTO_ OPERATIONMODE_ STREAMSTART	0x03	Operation Mode is "Stream Start". Mixture of "Start" and "Update". Used for streaming.
	CRYPTO_ OPERATIONMODE_FINISH	0x04	Operation Mode is "Finish". The calculations shall be finalized.
	CRYPTO_ OPERATIONMODE_ SINGLECALL	0x07	Operation Mode is "Single Call". Mixture of "Start", "Update" and "Finish".
	CRYPTO_ OPERATIONMODE_SAVE_ CONTEXT	0x08	Operation mode is "Save workspace context". Context data shall be provided by the crypto driver to the application.
	CRYPTO_ OPERATIONMODE_ RESTORE_CONTEXT	0x10	Operation mode is "Restore workspace context". Application provides the context data that was previously stored and the crypto driver shall restore the internal workspace.
Description	Enumeration which operation shall be performed. This enumeration is constructed from a bit mask, where the first bit indicates "Start", the second "Update" and the third "Finish".		
Variation	--		
Available via	Rte_Csm_Type.h		

⌋()

8.5.6.2 **Crypto_VerifyResultType**

[SWS_Csm_01024]⌈

Name	Crypto_VerifyResultType		
Kind	Type		
Derived from	uint8		
Range	CRYPTO_E_ VER_OK	0x00	The result of the verification is "true", i.e. the two compared elements are identical. This return code shall be given as value "0"
	CRYPTO_E_ VER_NOT_OK	0x01	The result of the verification is "false", i.e. the two compared elements are not identical. This return code shall be given as value "1".
Description	Enumeration of the result type of verification operations.		

Variation	--
Available via	Rte_Csm_Type.h

⌋()

8.5.6.3 **Csm_KeyDataType_{Crypto}**

[SWS_Csm_00828]⌈

Name	Csm_KeyDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	max({ecuc(Csm/CsmKeys/CsmKey/CsmKeyRef->CryIfKey/CryIfKeyRef->CryptoKey/CryptoKeyTypeRef->CryptoKeyType/CryptoKeyElementRef->CryptoKeyElement/CryptoKeyElementSize) Elements		
Description	Array long enough to store any key element of the considered key		
Variation	Crypto = {ecuc(Csm/CsmKeys/CsmKey.SHORT-NAME)}		
Available via	Rte_Csm_Type.h		

⌋()

8.5.6.4 **Crypto_ResultType**

[SWS_Csm_91044]⌈

Name	Crypto_ResultType		
Kind	Type		
Derived from	uint8		
Range	E_OK	0x00	The service request is successful.
	E_NOT_OK	0x01	The service request failed.
	CRYPTO_E_BUSY	0x02	The service request failed because the service is still busy
	CRYPTO_E_ENTROPY_EXHAUSTED	0x04	The service request failed because the entropy of the random number generator is exhausted
	CRYPTO_E_KEY_READ_FAIL	0x06	The service request failed because read access was denied
	CRYPTO_E_KEY_WRITE_FAIL	0x07	The service request failed because the writing access failed
	CRYPTO_E_KEY_NOT_AVAILABLE	0x08	The service request failed because the key is not available

	CRYPTO_E_KEY_NOT_VALID	0x09	The service request failed because the key is invalid.
	CRYPTO_E_KEY_SIZE_MISMATCH	0x0A	The service request failed because the key size does not match.
	CRYPTO_E_JOB_CANCELED	0x0C	The service request failed because the Job has been canceled.
	CRYPTO_E_KEY_EMPTY	0x0D	The service request failed because of uninitialized source key element.
Description	Return for Std_ReturnType for Cryptostack.		
Variation	--		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00095)

8.5.6.5 **Csm_HashDataType_{Crypto}**

[SWS_Csm_01920]⌈

Name	Csm_HashDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmHash/CsmHashConfig/CsmHashDataMaxLength} Elements		
Description	Array long enough to store the data which shall be hashed.		
Variation	Crypto={ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.6 **Csm_HashResultType_{Crypto}**

[SWS_Csm_00912]⌈

Name	Csm_HashResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmHash/CsmHashConfig/CsmHashResultLength} Elements		
Description	Array long enough to store the data of the hash.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.7 Csm_MacGenerateDataType_{Crypto}

[SWS_Csm_00935]⌈

Name	Csm_MacGenerateDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmMacGenerate/CsmMacGenerateConfig/CsmMacGenerateDataMaxLength} Elements		
Description	Array long enough to store the data from which a MAC shall be generated.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.8 Csm_MacGenerateResultType_{Crypto}

[SWS_Csm_00927]⌈

Name	Csm_MacGenerateResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmMacGenerate/CsmMacGenerateConfig/CsmMacGenerateResultLength} Elements		
Description	Array long enough to store the data of the MAC.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.9 Csm_MacVerifyDataType_{Crypto}

[SWS_Csm_00802]⌈

Name	Csm_MacVerifyDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmMacVerify/CsmMacVerifyConfig/CsmMacVerifyDataMaxLength} Elements		
Description	Array long enough to store the data for whichs MAC shall be verified.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		

Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.10 Csm_MacVerifyCompareType_{Crypto}

[SWS_Csm_00803]⌈

Name	Csm_MacVerifyCompareType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmMacVerify/CsmMacVerifyConfig/CsmMacVerifyCompareLength}/8 Elements		
Description	Array long enough to store a MAC to be verified.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.11 Csm_EncryptDataType_{Crypto}

[SWS_Csm_01921]⌈

Name	Csm_EncryptDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmEncrypt/CsmEncryptConfig/CsmEncryptDataMaxLength} Elements		
Description	Array long enough to store the data to be encrypted.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.12 Csm_EncryptResultType_{Crypto}

[SWS_Csm_01922]⌈

Name	Csm_EncryptResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmEncrypt/CsmEncryptConfig/CsmEncryptResultMaxLength} Elements		

Description	Array long enough to store the data of the cipher.
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}
Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.13 Csm_DecryptDataType_{Crypto}

[SWS_Csm_01923]⌈

Name	Csm_DecryptDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmDecrypt/CsmDecryptConfig/CsmDecryptDataMaxLength} Elements		
Description	Array long enough to store the data to be decrypted.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.14 Csm_DecryptResultType_{Crypto}

[SWS_Csm_01924]⌈

Name	Csm_DecryptResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmDecrypt/CsmDecryptConfig/CsmDecryptResultMaxLength} Elements		
Description	Array long enough to store the data of the decrypted plaintext.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.15 Csm_AEADEncryptPlaintextType_{Crypto}

[SWS_Csm_01925]⌈

Name	Csm_AEADEncryptPlaintextType_{Crypto}		
Kind	Array	***Element type***	uint8

Size	{ecuc(Csm/CsmPrimitives/CsmAEADEncrypt/CsmAEADEncryptConfig/CsmAEADEncryptPlaintextMaxLength} Elements
Description	Array long enough to store the plaintext to be encrypted with AEAD.
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}
Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.16 Csm_AEADEncryptAssociatedDataType_{Crypto}

[SWS_Csm_01928]⌈

Name	Csm_AEADEncryptAssociatedDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmAEADEncrypt/CsmAEADEncryptConfig/CsmAEADEncryptAssociatedDataMaxLength} Elements		
Description	Array long enough to store the data of the header.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.17 Csm_AEADEncryptCiphertextType_{Crypto}

[SWS_Csm_01927]⌈

Name	Csm_AEADEncryptCiphertextType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmAEADEncrypt/CsmAEADEncryptConfig/CsmAEADEncryptCiphertextMaxLength} Elements		
Description	Array long enough to store the data of the cipher.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.18 Csm_AEADEncryptTagType_{Crypto}

[SWS_Csm_01926]⌈

Name	Csm_AEADEncryptTagType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmAEADEncrypt/CsmAEADEncryptConfig/CsmAEADEncryptTagLength)} Elements		
Description	Array long enough to store the data of the Tag.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.19 Csm_AEADDecryptCiphertextType_{Crypto}

[SWS_Csm_00922]⌈

Name	Csm_AEADDecryptCiphertextType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmAEADDecrypt/CsmAEADDecryptConfig/CsmAEADDecryptCiphertextMaxLength} Elements		
Description	Array long enough to store the ciphertext to be decrypted with AEAD.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.20 Csm_AEADDecryptAssociatedDataType_{Crypto}

[SWS_Csm_00923]⌈

Name	Csm_AEADDecryptAssociatedDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmAEADDecrypt/CsmAEADDecryptConfig/CsmAEADDecryptAssociatedDataMaxLength} Elements		
Description	Array long enough to store the data of the header.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.21 Csm_AEADDecryptTagType_{Crypto}

[SWS_Csm_01074]⌈

Name	Csm_AEADDecryptTagType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	(({ecuc(Csm/CsmPrimitives/CsmAEADDecrypt/CsmAEADDecryptConfig/CsmAEADDecryptTagLength)}+7)/8) Elements		
Description	Array long enough to store the data of the Tag.		
Variation	Crypto = {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.22 Csm_AEADDecryptPlaintextType_{Crypto}

[SWS_Csm_01075]⌈

Name	Csm_AEADDecryptPlaintextType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmAEADDecrypt/CsmAEADDecryptConfig/CsmAEADDecryptPlaintextMaxLength} Elements		
Description	Array long enough to store the data of the plaintext.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.23 Csm_SignatureGenerateDataType_{Crypto}

[SWS_Csm_01083]⌈

Name	Csm_SignatureGenerateDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmSignatureGenerate/CsmSignatureGenerateConfig/CsmSignatureGenerateDataMaxLength} Elements		
Description	Array long enough to store the data from which the signature shall be generated.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_01076)

8.5.6.24 Csm_SignatureGenerateResultType_{Crypto}

[SWS_Csm_01077]⌈

Name	Csm_SignatureGenerateResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmSignatureGenerate/CsmSignatureGenerateConfig/ CsmSignatureGenerateResultLength} Elements		
Description	Array long enough to store the signature and its length.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.25 Csm_SignatureVerifyDataType_{Crypto}

[SWS_Csm_01078]⌈

Name	Csm_SignatureVerifyDataType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmSignatureVerify/CsmSignatureVerifyConfig/CsmSignatureVerifyDataMaxLength} Elements		
Description	Array long enough to store the data for whichs signature shall be verified.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.26 Csm_SignatureVerifyCompareType_{Crypto}

[SWS_Csm_01079]⌈

Name	Csm_SignatureVerifyCompareType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmSignatureVerify/CsmSignatureVerifyConfig/CsmSignatureVerifyCompareLength} Elements		
Description	Array long enough to store a signature to be verified.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		

Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.27 Csm_RandomGenerateResultType_{Crypto}

[SWS_Csm_00930]⌈

Name	Csm_RandomGenerateResultType_{Crypto}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Csm/CsmPrimitives/CsmRandomGenerate/CsmRandomGenerateConfig/CsmRandomGenerateResultLength) Elements		
Description	Array long enough to store the data of the random number.		
Variation	Crypto= {ecuc/Csm/CsmPrimitives.SHORT-NAME}		
Available via	Rte_Csm_Type.h		

⌋(SRS_CryptoStack_00090)

8.5.6.28 Csm_LengthPtr

[SWS_Csm_91045]⌈

Name	Csm_LengthPtr
Kind	Pointer
Type	uint32*
Description	ImplementationDataType of category DATA_REFERENCE with the pointer target uint32 to be able to return asynchronously the length.
Variation	--
Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.29 Csm_VerifyResultPtr

[SWS_Csm_91046]⌈

Name	Csm_VerifyResultPtr
Kind	Pointer

Type	Crypto_VerifyResultType*
Description	ImplementationDataType of category DATA_REFERENCE with the pointer target Crypto_VerifyResultType to be able to return asynchronously the verify result.
Variation	--
Available via	Rte_Csm_Type.h

⌋(SRS_CryptoStack_00090)

8.5.6.30 Crypto_KeyStatusType

[SWS_Csm_91102]⌈

Name	Crypto_KeyStatusType		
Kind	Type		
Derived from	uint8		
Range	CRYPTO_KEYSTATUS_INVALID	0x00	The status of the key is invalid (for example after Csm_KeyElementSet the Csm_KeySetValid was not called).
	CRYPTO_KEYSTATUS_VALID	0x01	The status of the key is valid (for example the status was successfully set by the Csm_KeySetValid).
	CRYPTO_KEYSTATUS_UPDATE_IN_PROGRESS	0x02	Indicates that the NV RAM Block that is assigned to this key are currently being updated. The update operation is in progress and has not yet been completed by NVM.
Description	Enumeration for key status.		
Variation	--		
Available via	Rte_Csm_Type.h		

⌋()

8.5.7 Ports

8.5.7.1 CsmKey_{Key}

[SWS_Csm_01042]⌈

Name	CsmKey_{Key}		
Kind	ProvidedPort	***Interface***	CsmKeyManagement_{Key}
Description	Port related to a specific cryptographic key to execute the key management functions synchronously.		

Port Defined Argument Value(s)	***Type***	uint32
	Value	{ecuc(Csm/CsmKeys/CsmKey/CsmKeyId)}
Variation	{ecuc(Csm/CsmKeys/CsmKey.CsmKeyUsePort)} == TRUE Key = {ecuc(Csm/CsmKeys/CsmKey.SHORT-NAME)}	

⌋(SRS_CryptoStack_00090, SRS_CryptoStack_00091)

8.5.7.2 **CsmJob_{Job} (CRYPTO_USE_PORT)**

[SWS_Csm_91023]⌈

Name	CsmJob_{Job}		
Kind	Provided Port	***Interface***	CsmHash_{PrimitiveCfg}, CsmMacGenerate_{PrimitiveCfg}, CsmMacVerify_{PrimitiveCfg}, CsmEncrypt_{PrimitiveCfg}, CsmDecrypt_{PrimitiveCfg}, CsmAEADEncrypt_{PrimitiveCfg}, CsmAEADDecrypt_{PrimitiveCfg}, CsmSignatureGenerate_{PrimitiveCfg}, CsmSignatureVerify_{PrimitiveCfg}, CsmRandomGenerate_{PrimitiveCfg}
Description	Port related to a specific cryptographic job to execute the assigned cryptographic calculations synchronously.		
Port Defined Argument Value(s)	***Type***	uint32	
	Value	{ecuc(Csm/CsmJobs/CsmJob.CsmJobId)}	
	Type	Crypto_OperationModeType	
	Value	CRYPTO_OPERATIONMODE_SINGLECALL	

Variation	({ecuc(Csm/CsmJobs/CsmJob.CsmJobInterfaceUsePort)} == CRYPTO_USE_PORT) && ({ecuc(Csm/CsmJobs/CsmJob.CsmJobPrimitiveRef)} != NULL)
	Job = {ecuc(Csm/CsmJobs/CsmJob.SHORT-NAME)}
	Primitive = {ecuc(Csm/CsmJobs/CsmJob.CsmJobPrimitiveRef->CsmPrimitives/*.SHORT-NAME)}
	PrimitiveCfg = {ecuc(Csm/CsmPrimitives/{Primitive}/{Primitive}Config.SHORT-NAME)}		

⌋(SRS_CryptoStack_00090, SRS_CryptoStack_00091)

8.5.7.3 **CsmJob_{Job} (CRYPTO_USE_PORT_OPTIMIZED)**

[SWS_Csm_91062]⌈

Name	CsmJob_{Job}		
Kind	Provided Port	***Interface***	CsmCustom, CsmJobKeySetInvalid, CsmHash, CsmMacGenerate, CsmMacVerify, CsmEncrypt, CsmDecrypt, CsmAEADEncrypt, CsmAEADDecrypt, CsmSignatureGenerate, CsmSignatureVerify, CsmRandomGenerate, CsmJobKeyDerive, CsmJobKeyExchangeCalcPubVal, CsmJobKeyExchangeCalcSecret, CsmJobKeyGenerate, CsmJobKeySetValid, CsmJobRandomSeed

Description	Port related to a specific cryptographic job to execute the assigned cryptographic calculations asynchronously.	
Port Defined Argument Value(s)	***Type***	uint32
	Value	{ecuc(Csm/CsmJobs/CsmJob.CsmJobId)}
	Type	Crypto_OperationModeType
	Value	CRYPTO_OPERATIONMODE_SINGLECALL

Variation	({ecuc(Csm/CsmJobs/CsmJob.CsmJobInterfaceUsePort)} == CRYPTO_USE_PORT_OPTIMIZED) && ({ecuc(Csm/CsmJobs/CsmJob.CsmJobPrimitiveRef)} != NULL)
	Job = {ecuc(Csm/CsmJobs/CsmJob.SHORT-NAME)}
	Primitive = {ecuc(Csm/CsmJobs/CsmJob.CsmJobPrimitiveRef->CsmPrimitives/*.SHORT-NAME)}	

⌋(SRS_CryptoStack_00090, SRS_CryptoStack_00091)

8.5.7.4 CallbackNotification_{Job}

[SWS_Csm_00934]⌈

Name	CallbackNotification_{Job}		
Kind	RequiredPort	***Interface***	CallbackNotification
Description	Port for the callback notification.		
Variation	({ecuc(Csm/CsmJobs/CsmJob.CsmProcessingMode)}==CRYPTO_PROCESSING_ASYN)&&(CsmJob/CsmJobInterfaceUsePort!=CRYPTO_USE_FNC) Job = {ecuc(Csm/CsmJobs/CsmJob.SHORT-NAME)}		

⌋(SRS_CryptoStack_00090, SRS_CryptoStack_00091)

8.5.7.5 CsmContext_{Job}

[SWS_Csm_91105]⌈

Name	CsmContext_{Job}		
Kind	ProvidedPort	***Interface***	CsmContextService_{Job}
Description	Port related to context data operation.		
Port Defined Argument Value(s)	***Type***	uint32	
	Value	{ecuc(Csm/CsmJobs/CsmJob.CsmJobId)}	
Variation	{ecuc(Csm/CsmJobs/CsmJob. CsmJobServiceInterfaceContextUsePort)} == CRYPTO_USE_PORT Job={ecuc(Csm/CsmJobs/CsmJob.SHORT-NAME)}		

⌋(SRS_CryptoStack_00090)

9 Sequence Diagrams

The following sequence diagrams concentrate on the interaction between the CSM module and software components respectively the ECU state manager.

9.1 Asynchronous Calls

The following diagram (Sequence diagram for asynchronous call) shows a sample sequence of function calls for a request performed asynchronously. The result of the asynchronous function can be accessed after an asynchronous notification (invocation of the configured callback function).

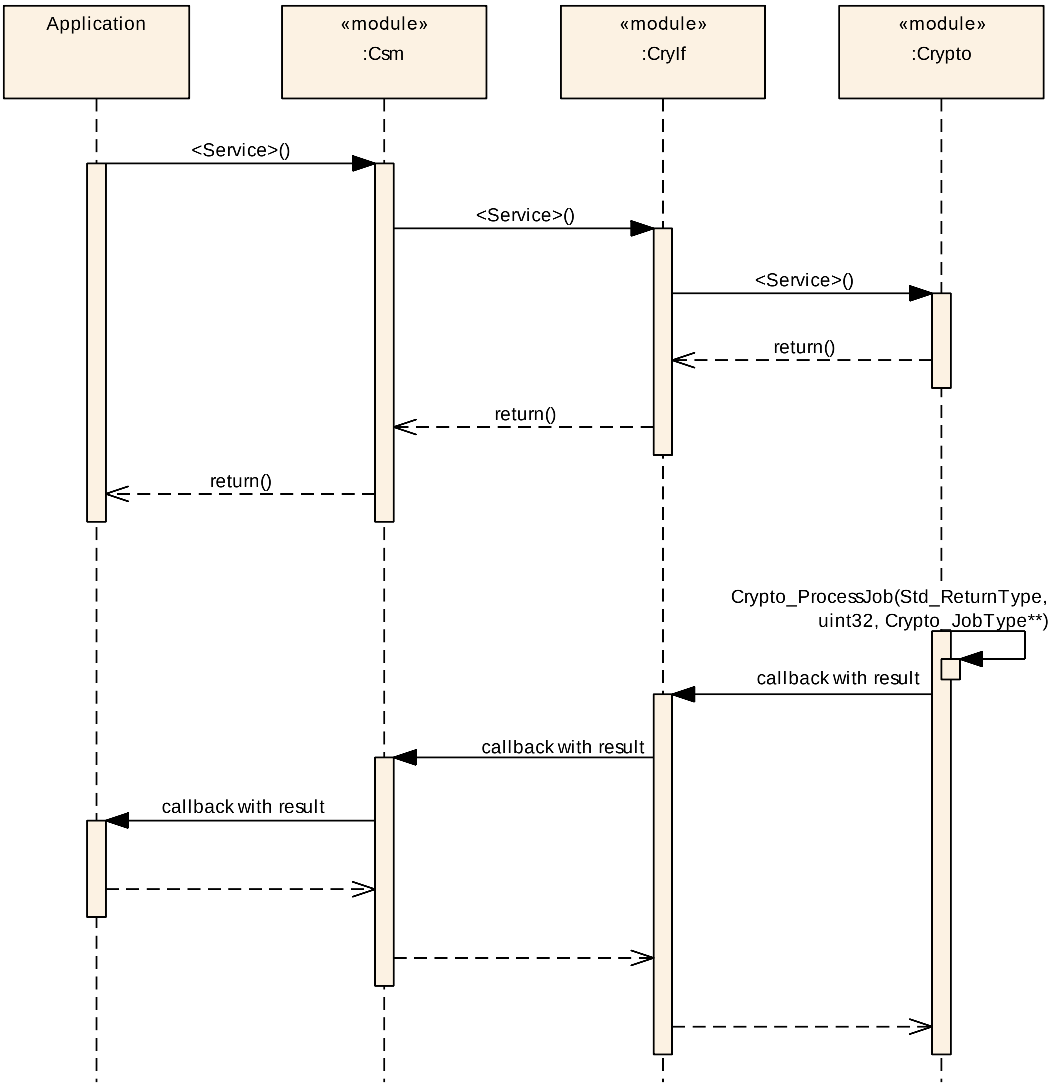


Figure 9-1 Sequence Diagram for Asynchronous Call with Callback

9.2 Synchronous Calls

The following diagram (Sequence diagram for synchronous calls) shows a sample sequence of function calls with the scheduler for a request performed synchronously.

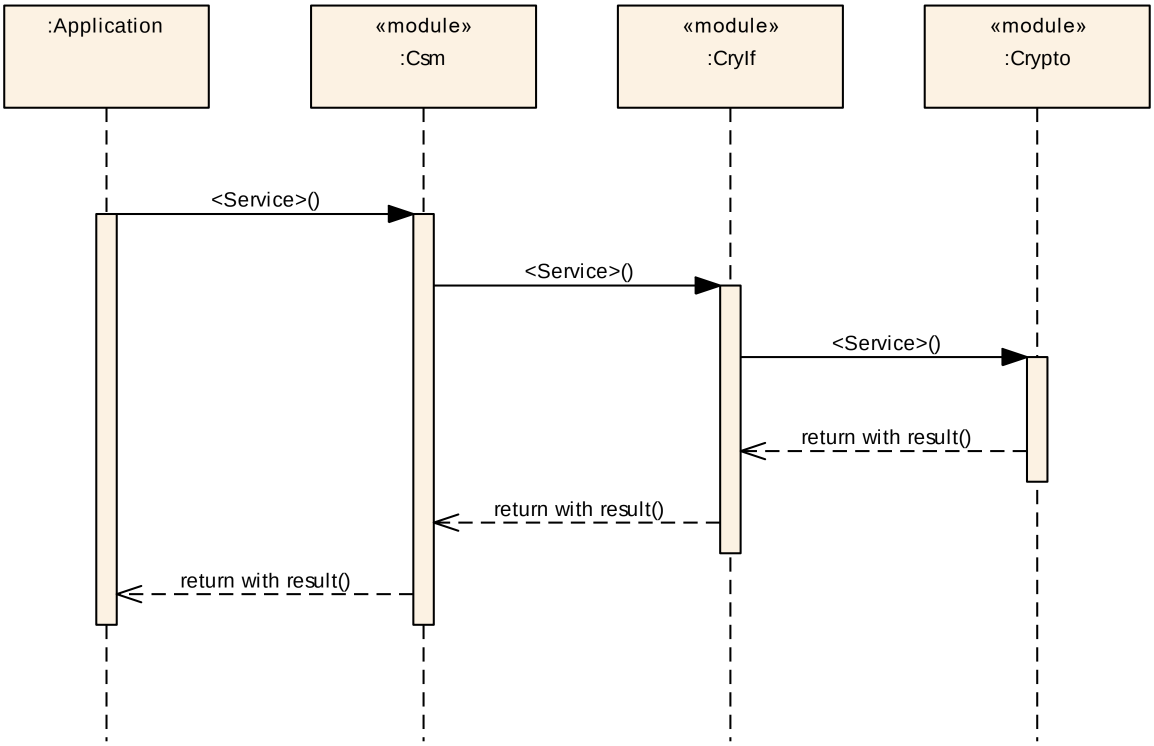


Figure 9-2 Sequence Diagram for Synchronous Call

10 Configuration

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification.

Chapter 10.2 specifies the structure (containers) and the parameters of the module CSM.

Chapter 10.3 specifies published information of the module CSM.

10.1 How to Read this Chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in *SWS_BSWGeneral.*

10.2 Containers and Configuration Parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 7 and Chapter 8.

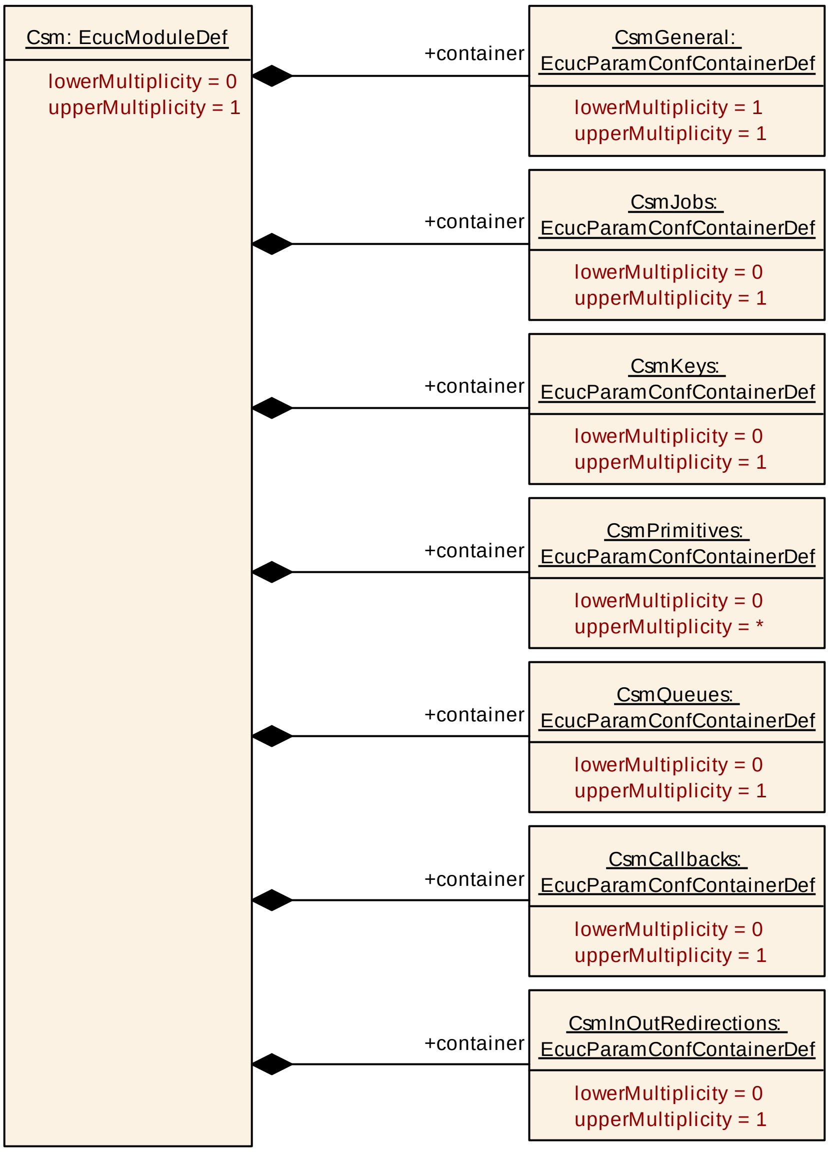


Figure 10-1 Crypto Service Manager Layout

10.2.1 Csm

SWS Item	[ECUC_Csm_00818]
Module Name	Csm
Description	Configuration of the Csm (CryptoServiceManager) module.
Post-Build Variant Support	false
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers

Container Name	Multiplicity	Scope / Dependency
CsmCallbacks	0..1	Container for callback function configurations
CsmGeneral	1	Container for common configuration options.
CsmInOutRedirections	0..1	Configuration for CSM redirection configurations
CsmJobs	0..1	Container for configuration of CSM jobs.
CsmKeys	0..1	Container for CSM key configurations.
CsmMainFunction	0..*	Each element of this container defines one instance of Csm_Main Function. For each partition, where the Csm module shall be instantiated, at least one MainFunction instance needs to be configured.
CsmPrimitives	0..*	Container for configuration of CsmPrimitives
CsmQueues	0..1	Container for CSM queue configurations

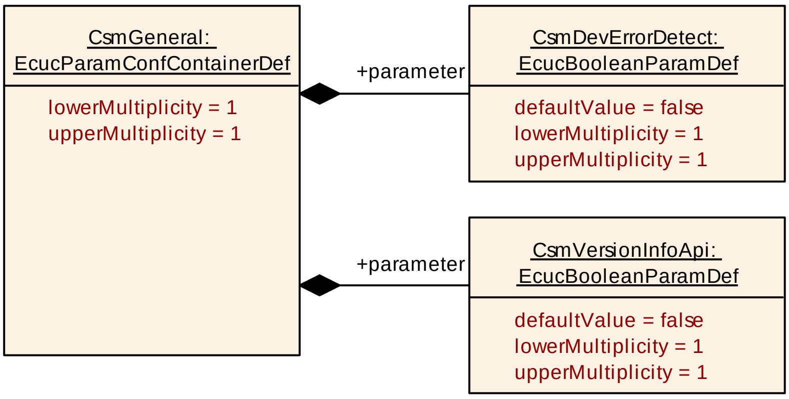


Figure 10-2 Crypto Service Manager General Layout

10.2.2 CsmGeneral

SWS Item	[ECUC_Csm_00002]		
Container Name	CsmGeneral		
Parent Container	Csm		
Description	Container for common configuration options.		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Configuration Parameters

SWS Item	[ECUC_Csm_00001]		
Parameter Name	CsmDevErrorDetect		
Parent Container	CsmGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00003]		
Parameter Name	CsmVersionInfoApi		
Parent Container	CsmGeneral		
Description	Pre-processor switch to enable and disable availability of the API Csm_Get VersionInfo(). True: API Csm_GetVersionInfo() is available. False: API Csm_ GetVersionInfo() is not available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

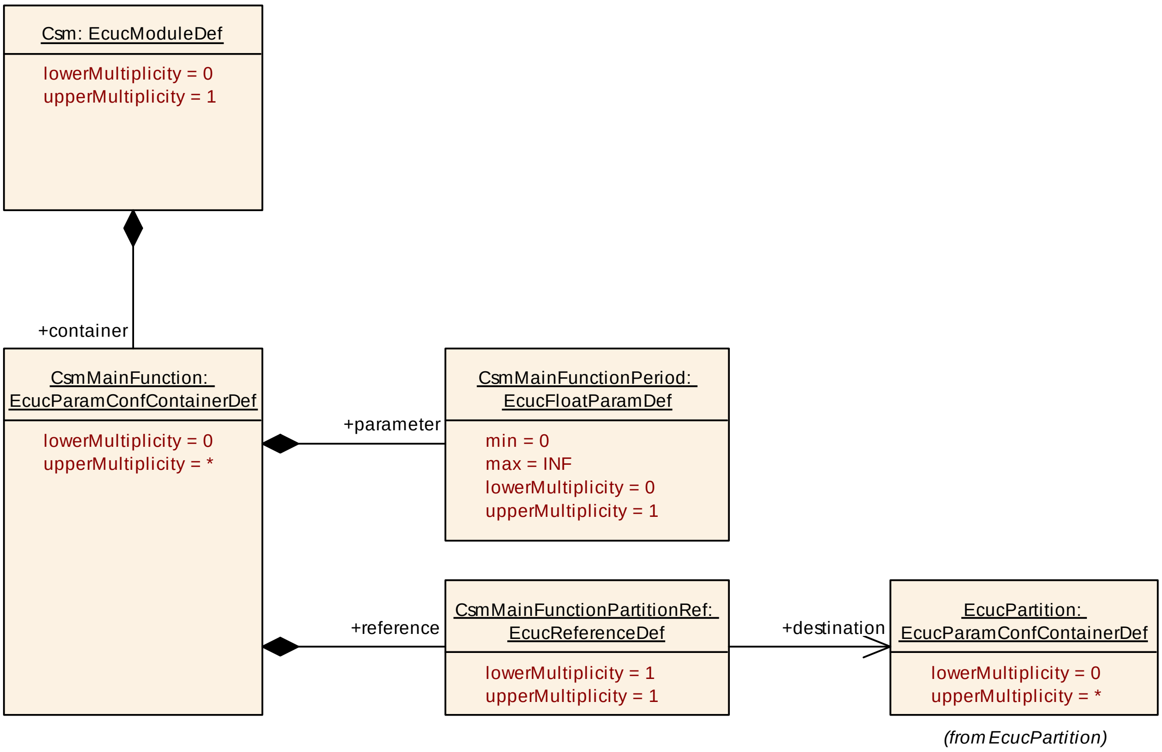


Figure 10-3 Crypto Service Manager MainFunction Layout

10.2.3 CsmMainFunction

SWS Item	[ECUC_Csm_00279]
Container Name	CsmMainFunction

Parent Container	Csm
Description	Each element of this container defines one instance of Csm_MainFunction. For each partition, where the Csm module shall be instantiated, at least one MainFunction instance needs to be configured.
Configuration Parameters	

SWS Item	[ECUC_Csm_00113]		
Parameter Name	CsmMainFunctionPeriod		
Parent Container	CsmMainFunction		
Description	Specifies the period of main function Csm_MainFunction in seconds.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	]0 .. INF[		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00280]		
Parameter Name	CsmMainFunctionPartitionRef		
Parent Container	CsmMainFunction		
Description	Reference to EcucPartition, where the according CsmMainFunction instance is assigned to.		
Multiplicity	1		
Type	Reference to EcucPartition		
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

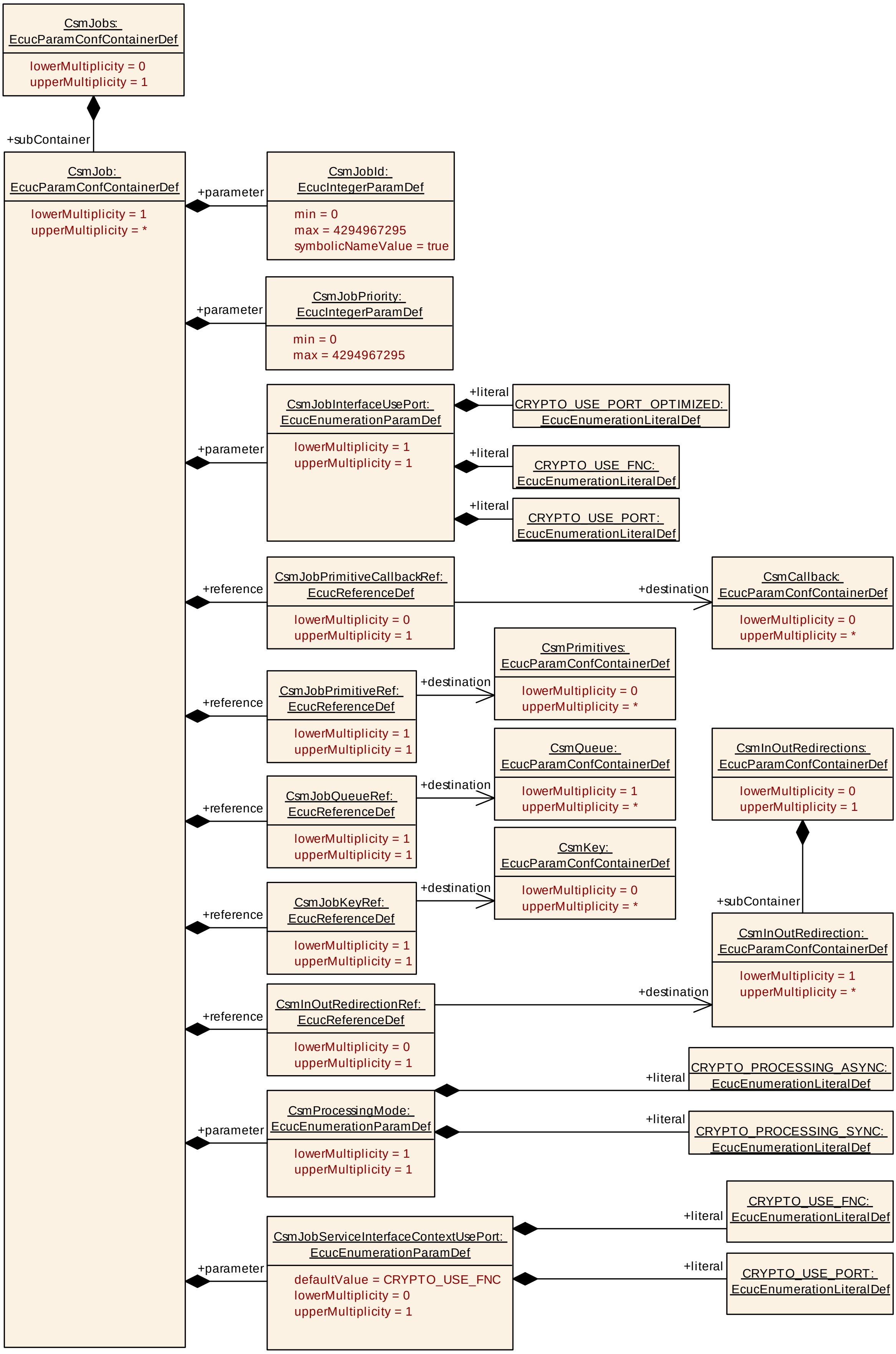

CsmJobs: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJob: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+parameter
CsmJobId: EcucIntegerParamDef
min = 0
max = 4294967295
symbolicNameValue = true
+parameter
CsmJobPriority: EcucIntegerParamDef
min = 0
max = 4294967295
+parameter
CsmJobInterfaceUsePort: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_USE_PORT_OPTIMIZED: EcucEnumerationLiteralDef
+literal
CRYPTO_USE_FNC: EcucEnumerationLiteralDef
+literal
CRYPTO_USE_PORT: EcucEnumerationLiteralDef
+reference
CsmJobPrimitiveCallbackRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
CsmCallback EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmJobPrimitiveRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmPrimitives: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmJobQueueRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmQueue: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
CsmInOutRedirections: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+reference
CsmJobKeyRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmKey: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+subContainer
CsmInOutRedirection: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+reference
CsmInOutRedirectionRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
+literal
CRYPTO_PROCESSING_ASYNC: EcucEnumerationLiteralDef
+parameter
CsmProcessingMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_PROCESSING_SYNC: EcucEnumerationLiteralDef
+literal
CRYPTO_USE_FNC: EcucEnumerationLiteralDef
+parameter
CsmJobServiceInterfaceContextUsePort: EcucEnumerationParamDef
defaultValue = CRYPTO_USE_FNC
lowerMultiplicity = 0
upperMultiplicity = 1
+literal
CRYPTO_USE_PORT: EcucEnumerationLiteralDef

Figure 10-4 CsmJobs Layout

10.2.4 CsmJobs

SWS Item	[ECUC_Csm_00112]		
Container Name	CsmJobs		
Parent Container	Csm		
Description	Container for configuration of CSM jobs.		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Configuration Parameters			

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJob	1..*	Container for configuration of CSM job. The container name serves as a symbolic name for the identifier of a job configuration.

10.2.5 CsmJob

SWS Item	[ECUC_Csm_00118]		
Container Name	CsmJob		
Parent Container	CsmJobs		
Description	Container for configuration of CSM job. The container name serves as a symbolic name for the identifier of a job configuration.		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	[ECUC_Csm_00119]
Parameter Name	CsmJobId
Parent Container	CsmJob
Description	Identifier of the CSM job. The set of actually configured identifiers shall be consecutive and gapless.

Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00275]		
Parameter Name	CsmJobInterfaceUsePort		
Parent Container	CsmJob		
Description	Does the job need RTE interfaces?		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_USE_FNC	Port is not used.	
	CRYPTO_USE_PORT	Port is used.	
	CRYPTO_USE_PORT_OPTIMIZED	DATA_REFERENCE is used.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00120]
Parameter Name	CsmJobPriority
Parent Container	CsmJob

Description	Priority of the job. The higher the value, the higher the job's priority.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00327]		
Parameter Name	CsmJobServiceInterfaceContextUsePort		
Parent Container	CsmJob		
Description	Does the job need RTE interfaces for context operations		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_USE_FNC	Port is not used.	
	CRYPTO_USE_PORT	Port is used for this operation.	
Default value	CRYPTO_USE_FNC		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00276]		
Parameter Name	CsmProcessingMode		
Parent Container	CsmJob		
Description	Determines how the interface shall be used for that job. Synchronous processing returns with the result while asynchronous processing returns without processing the job. The caller will be notified by the corresponding callback.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_PROCESSING_ASYNC	--	
	CRYPTO_PROCESSING_SYNC	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00263]		
Parameter Name	CsmInOutRedirectionRef		
Parent Container	CsmJob		
Description	This parameter refers to the used redirection.		
Multiplicity	0..1		
Type	Reference to CsmInOutRedirection		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00126]		
Parameter Name	CsmJobKeyRef		
Parent Container	CsmJob		
Description	This parameter refers to the key which shall be used for the CsmPrimitive. It's possible to use a CsmKey for different jobs		
Multiplicity	1		
Type	Reference to CsmKey		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: A dummy key shall be referenced if the CsmPrimitive doesn't require a key (e.g. for Hash calculation).		

SWS Item	[ECUC_Csm_00123]		
Parameter Name	CsmJobPrimitiveCallbackRef		
Parent Container	CsmJob		
Description	This parameter refers to the used CsmCallback. The referred Csm Callback is called when the crypto job has been finished.		
Multiplicity	0..1		
Type	Reference to CsmCallback		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00122]		
Parameter Name	CsmJobPrimitiveRef		
Parent Container	CsmJob		
Description	This parameter refers to the used CsmPrimitive. Different jobs may refer to one CsmPrimitive. The referred CsmPrimitive provides detailed information on the actual cryptographic routine.		
Multiplicity	1		
Type	Reference to CsmPrimitives		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00125]		
Parameter Name	CsmJobQueueRef		
Parent Container	CsmJob		
Description	This parameter refers to the queue. The queue is used if the underlying crypto driver object is busy. The queue refers also to the channel which is used.		
Multiplicity	1		
Type	Reference to CsmQueue		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

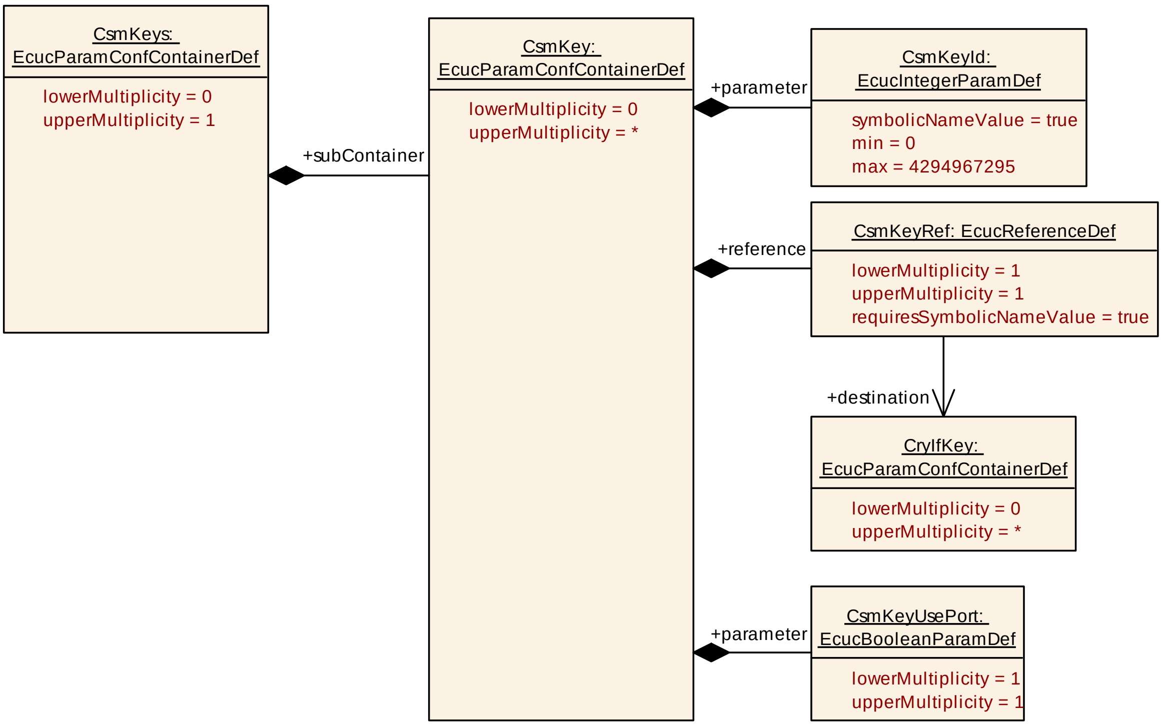


Figure 10-5 Crypto Service Manager Keys Layout

10.2.6 CsmKeys

SWS Item	[ECUC_Csm_00005]
Container Name	CsmKeys
Parent Container	Csm
Description	Container for CSM key configurations.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**

CsmKey	0..*	Container for configuration of a CSM key. The container name serves as a symbolic name for the identifier of a key configuration.

10.2.7 **CsmKey**

SWS Item	[ECUC_Csm_00014]
Container Name	CsmKey
Parent Container	CsmKeys
Description	Container for configuration of a CSM key. The container name serves as a symbolic name for the identifier of a key configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00015]		
Parameter Name	CsmKeyId		
Parent Container	CsmKey		
Description	Identifier of the CsmKey. The set of actually configured identifiers shall be consecutive and gapless.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00127]
Parameter Name	CsmKeyUsePort

Parent Container	CsmKey		
Description	Does the key need RTE interfaces? True: RTE interfaces used for this key False: No RTE interfaces used for this key		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00016]		
Parameter Name	CsmKeyRef		
Parent Container	CsmKey		
Description	This parameter refers to the used CryIfKey. The underlying CryIfKey refers to a specific CryptoKey in the Crypto Driver.		
Multiplicity	1		
Type	Symbolic name reference to CryIfKey		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

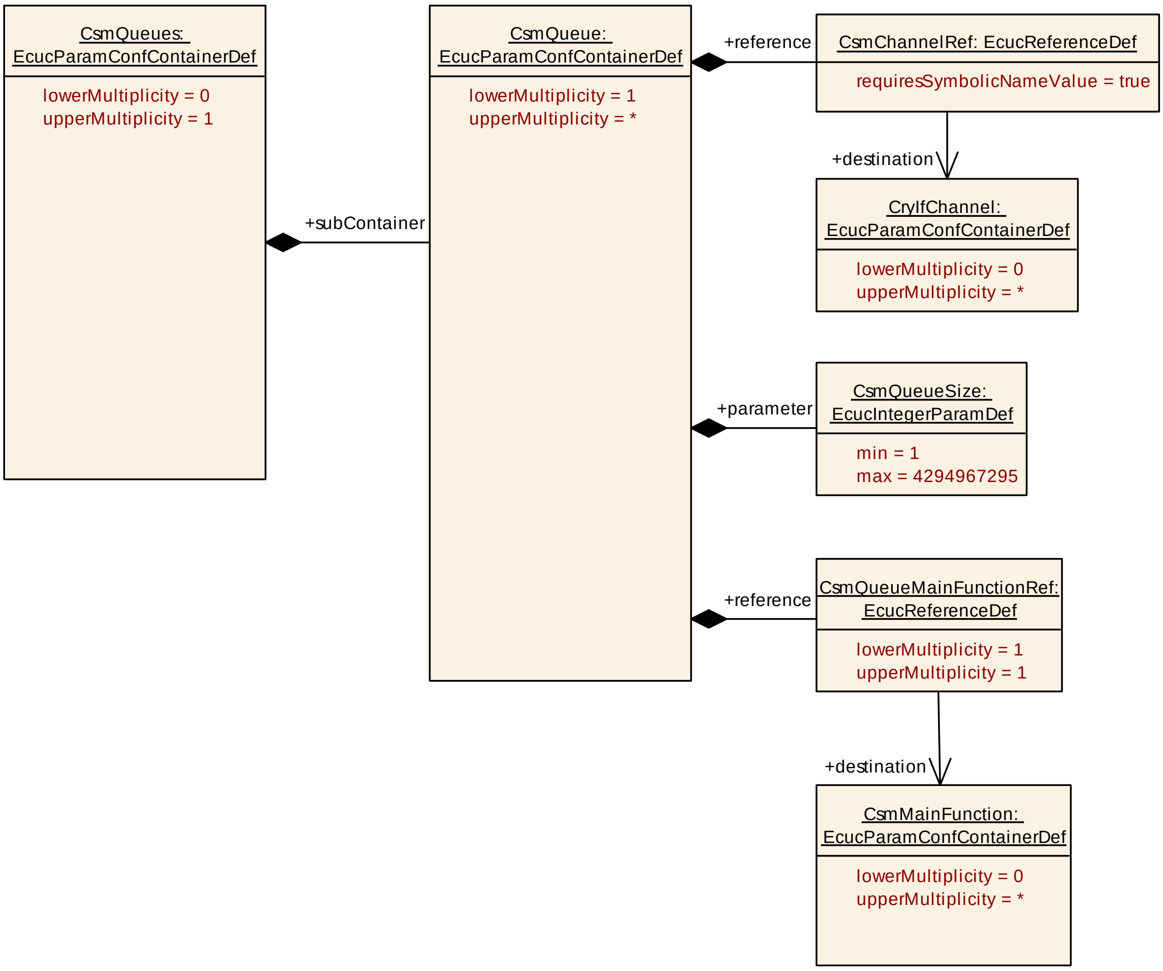


Figure 10-6 Crypto Service Manager Queues Layout

10.2.8 **CsmQueues**

SWS Item	[ECUC_Csm_00007]
Container Name	CsmQueues
Parent Container	Csm
Description	Container for CSM queue configurations
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**

CsmQueue	1..*	Container for configuration of a CSM queue. A queue has two tasks: 1. queue jobs which cannot be processed since the underlying hardware is busy and 2. refer to channel which shall be used

10.2.9 **CsmQueue**

SWS Item	[ECUC_Csm_00032]
Container Name	CsmQueue
Parent Container	CsmQueues
Description	Container for configuration of a CSM queue. A queue has two tasks: 1. queue jobs which cannot be processed since the underlying hardware is busy and 2. refer to channel which shall be used
Configuration Parameters	

SWS Item	[ECUC_Csm_00034]		
Parameter Name	CsmQueueSize		
Parent Container	CsmQueue		
Description	Size of the CsmQueue. If jobs cannot be processed by the underlying hardware since the hardware is busy, the jobs stay in the prioritized queue. If the queue is full, the next job will be rejected.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00033]		
Parameter Name	CsmChannelRef		
Parent Container	CsmQueue		
Description	Refers to the underlying Crypto Interface channel.		
Multiplicity	1		
Type	Symbolic name reference to CryIfChannel		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00281]		
Parameter Name	CsmQueueMainFunctionRef		
Parent Container	CsmQueue		
Description	Reference to CsmMainFunction, where the according CsmQueue is assigned to.		
Multiplicity	1		
Type	Reference to CsmMainFunction		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

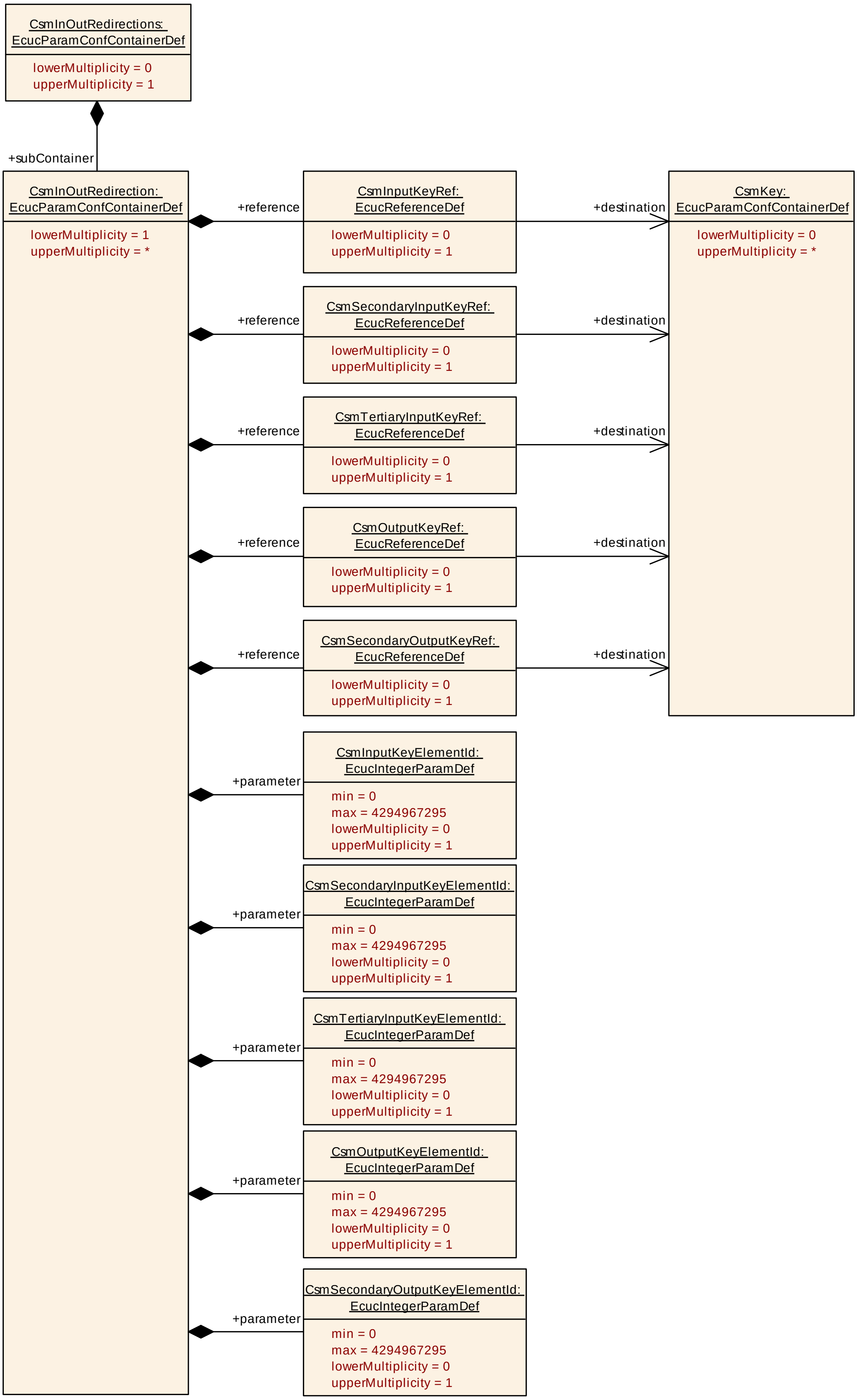

CsmInOutRedirections: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmInOutRedirection: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+reference
CsmInputKeyRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
CsmKey: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmSecondaryInputKeyRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
+reference
CsmTertiaryInputKeyRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
+reference
CsmOutputKeyRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
+reference
CsmSecondaryOutputKeyRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
+destination
+parameter
CsmInputKeyElementId: EcucIntegerParamDef
min = 0
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmSecondaryInputKeyElementId: EcucIntegerParamDef
min = 0
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmTertiaryInputKeyElementId: EcucIntegerParamDef
min = 0
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmOutputKeyElementId: EcucIntegerParamDef
min = 0
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmSecondaryOutputKeyElementId: EcucIntegerParamDef
min = 0
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-7 Crypto Service Manager CsmInOutRedirections Layout

10.2.10 CsmInOutRedirections

SWS Item	[ECUC_Csm_00262]
Container Name	CsmInOutRedirections
Parent Container	Csm
Description	Configuration for CSM redirection configurations
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmInOut-Redirection	1..*	Container for configuration of a CSM redirection. A redirection let a CSM job use a specific key element as input or/and output.

10.2.11 CsmInOutRedirection

SWS Item	[ECUC_Csm_00264]
Container Name	CsmInOutRedirection
Parent Container	CsmInOutRedirections
Description	Container for configuration of a CSM redirection. A redirection let a CSM job use a specific key element as input or/and output.
Configuration Parameters	

SWS Item	[ECUC_Csm_00266]	
Parameter Name	CsmInputKeyElementId	
Parent Container	CsmInOutRedirection	
Description	Identifier of the key element used as input	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	0 .. 4294967295	
Default value	--	

Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00272]		
Parameter Name	CsmOutputKeyElementId		
Parent Container	CsmInOutRedirection		
Description	Identifier of the key element used as output.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00269]
Parameter Name	CsmSecondaryInputKeyElementId
Parent Container	CsmInOutRedirection
Description	Identifier of the key element used as secondary input.

Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00274]		
Parameter Name	CsmSecondaryOutputKeyElementId		
Parent Container	CsmInOutRedirection		
Description	Identifier of the key element used as secondary output.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00270]		
Parameter Name	CsmTertiaryInputKeyElementId		
Parent Container	CsmInOutRedirection		
Description	Identifier of the key element used as tertiary input.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00265]		
Parameter Name	CsmInputKeyRef		
Parent Container	CsmInOutRedirection		
Description	This parameter refers to the key used as input.		
Multiplicity	0..1		
Type	Reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00271]		
Parameter Name	CsmOutputKeyRef		
Parent Container	CsmInOutRedirection		
Description	This parameter refers to the key used as output.		
Multiplicity	0..1		
Type	Reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00267]		
Parameter Name	CsmSecondaryInputKeyRef		
Parent Container	CsmInOutRedirection		
Description	This parameter refers to the key used as secondary input.		
Multiplicity	0..1		
Type	Reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00273]		
Parameter Name	CsmSecondaryOutputKeyRef		
Parent Container	CsmInOutRedirection		
Description	This parameter refers to the key used as secondary output.		
Multiplicity	0..1		
Type	Reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00268]		
Parameter Name	CsmTertiaryInputKeyRef		
Parent Container	CsmInOutRedirection		
Description	This parameter refers to the key used as tertiary input.		
Multiplicity	0..1		
Type	Reference to CsmKey		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

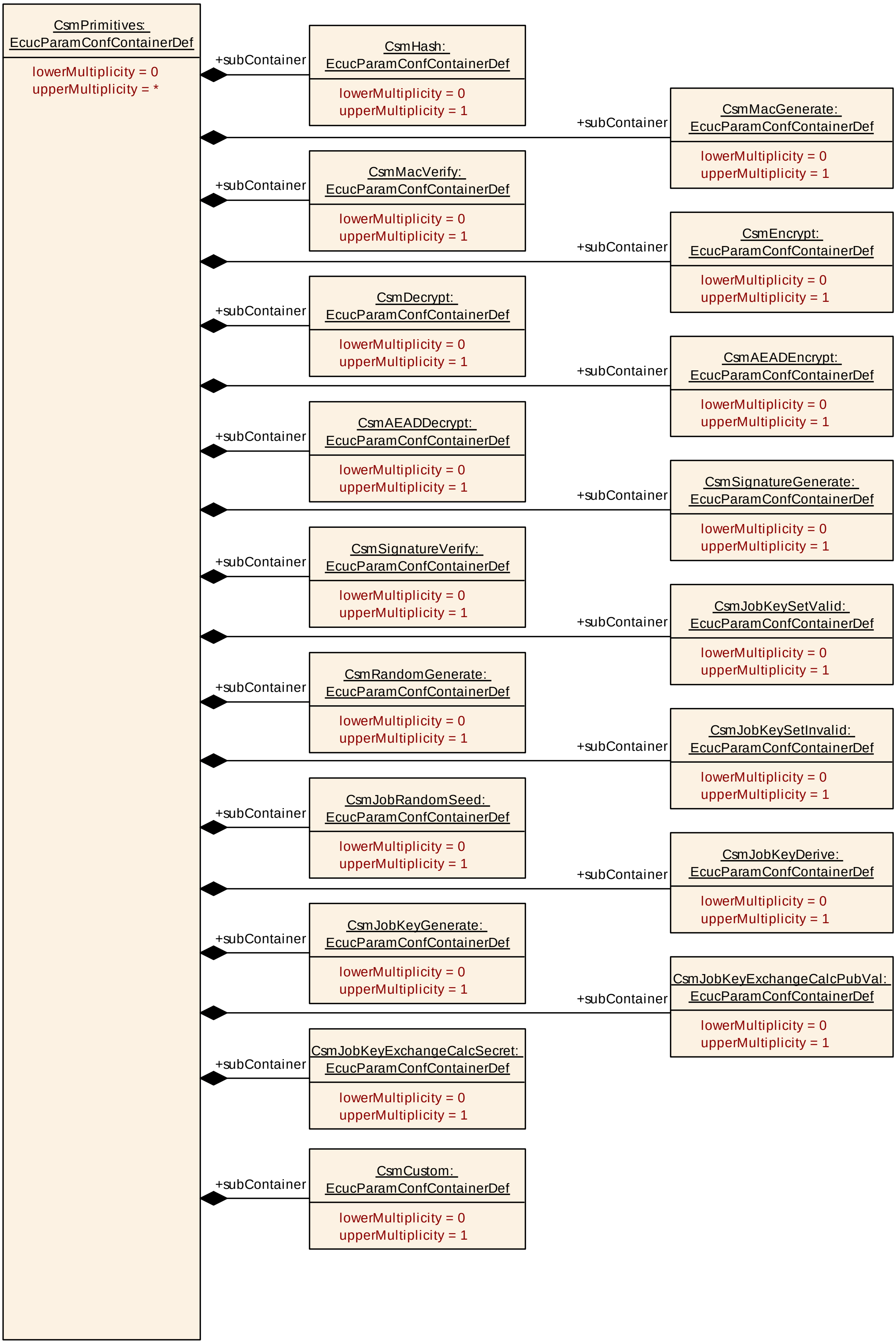


Figure 10-8 CsmPrimitives Layout

10.2.12 CsmPrimitives

SWS Item	[ECUC_Csm_00006]
Container Name	CsmPrimitives
Parent Container	Csm
Description	Container for configuration of CsmPrimitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmAEADDecrypt	0..1	Configuration of AEAD decryption primitives
CsmAEADEncrypt	0..1	Configuration of AEAD encryption primitives
CsmCustom	0..1	Container for configuration of a CSM custom crypto primitive.
CsmDecrypt	0..1	Configurations of Decryption primitives
CsmEncrypt	0..1	Configurations of Encryption primitives
CsmHash	0..1	Container for Hash Configurations
CsmJobKeyDerive	0..1	Configurations of KeyDerive primitives
CsmJobKeyExchangeCalc-PubVal	0..1	Configurations of KeyExchangeCalcPubVal primitives
CsmJobKeyExchangeCalc-Secret	0..1	Configurations of KeyExchangeCalcSecret primitives
CsmJobKeyGenerate	0..1	Configurations of KeyGenerate primitives
CsmJobKeySetInvalid	0..1	Configurations of KeySetInvalid primitives
CsmJobKeySetValid	0..1	Configurations of KeySetValid primitives
CsmJobRandomSeed	0..1	Configurations of RandomSeed primitives
CsmMacGenerate	0..1	Configurations of MacGenerate primitives
CsmMacVerify	0..1	Configurations of MacVerify primitives
CsmRandomGenerate	0..1	Configurations of RandomGenerate primitives
CsmSignatureGenerate	0..1	Configurations of SignatureGenerate primitives
CsmSignatureVerify	0..1	Configurations of SignatureVerify primitives

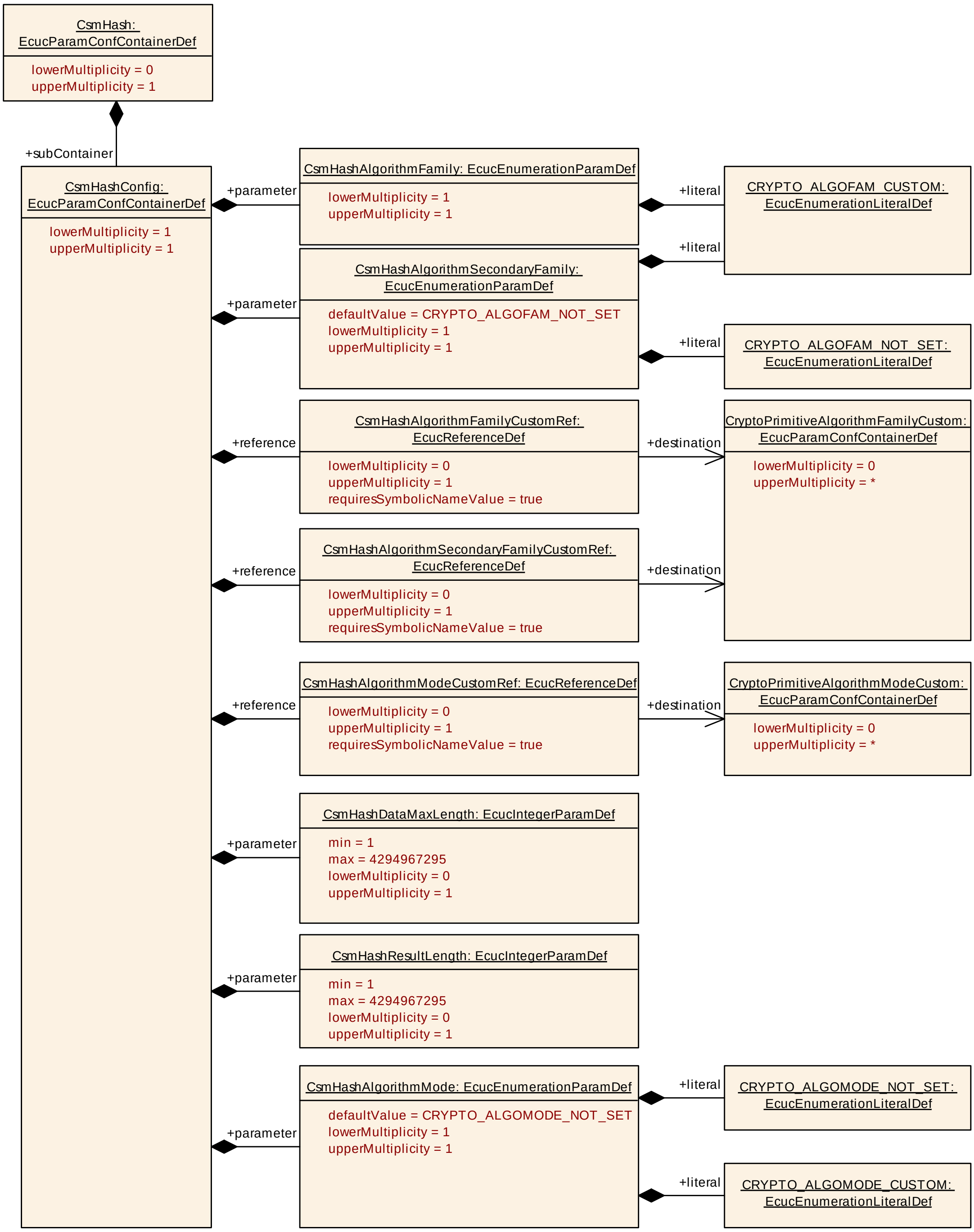

CsmHash: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmHashConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmHashAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
+parameter
CsmHashAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+reference
CsmHashAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmHashAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmHashAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmHashDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmHashResultLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmHashAlgorithmMode: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOMODE_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef

Figure 10-9 CsmHash Layout

10.2.13 CsmHash

SWS Item	[ECUC_Csm_00021]
Container Name	CsmHash
Parent Container	CsmPrimitives

Description	Container for Hash Configurations
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmHashConfig	1	Container for configuration of a CSM hash. The container name serves as a symbolic name for the identifier of a key configuration.

10.2.14 CsmHashConfig

SWS Item	[ECUC_Csm_00036]
Container Name	CsmHashConfig
Parent Container	CsmHash
Description	Container for configuration of a CSM hash. The container name serves as a symbolic name for the identifier of a key configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00038]	
Parameter Name	CsmHashAlgorithmFamily	
Parent Container	CsmHashConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_BLAKE_1_256	--
	CRYPTO_ALGOFAM_BLAKE_1_512	--
	CRYPTO_ALGOFAM_BLAKE_2s_256	--
	CRYPTO_ALGOFAM_BLAKE_2s_512	--
	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_RIPEMD160	--
	CRYPTO_ALGOFAM_SHA1	--

	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
	CRYPTO_ALGOFAM_SM3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00131]	
Parameter Name	CsmHashAlgorithmMode	
Parent Container	CsmHashConfig	
Description	Determines the algorithm mode used for the crypto service	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOMODE_CUSTOM	--
	CRYPTO_ALGOMODE_NOT_SET	--
Default value	CRYPTO_ALGOMODE_NOT_SET	

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00181]		
Parameter Name	CsmHashAlgorithmSecondaryFamily		
Parent Container	CsmHashConfig		
Description	Determines the algorithm family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00040]
Parameter Name	CsmHashDataMaxLength
Parent Container	CsmHashConfig
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.

Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00130]		
Parameter Name	CsmHashResultLength		
Parent Container	CsmHashConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.

SWS Item	[ECUC_Csm_00282]		
Parameter Name	CsmHashAlgorithmFamilyCustomRef		
Parent Container	CsmHashConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmHashAlgorithm Family is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00284]		
Parameter Name	CsmHashAlgorithmModeCustomRef		
Parent Container	CsmHashConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmHashAlgorithm Mode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00283]		
Parameter Name	CsmHashAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmHashConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmHashSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

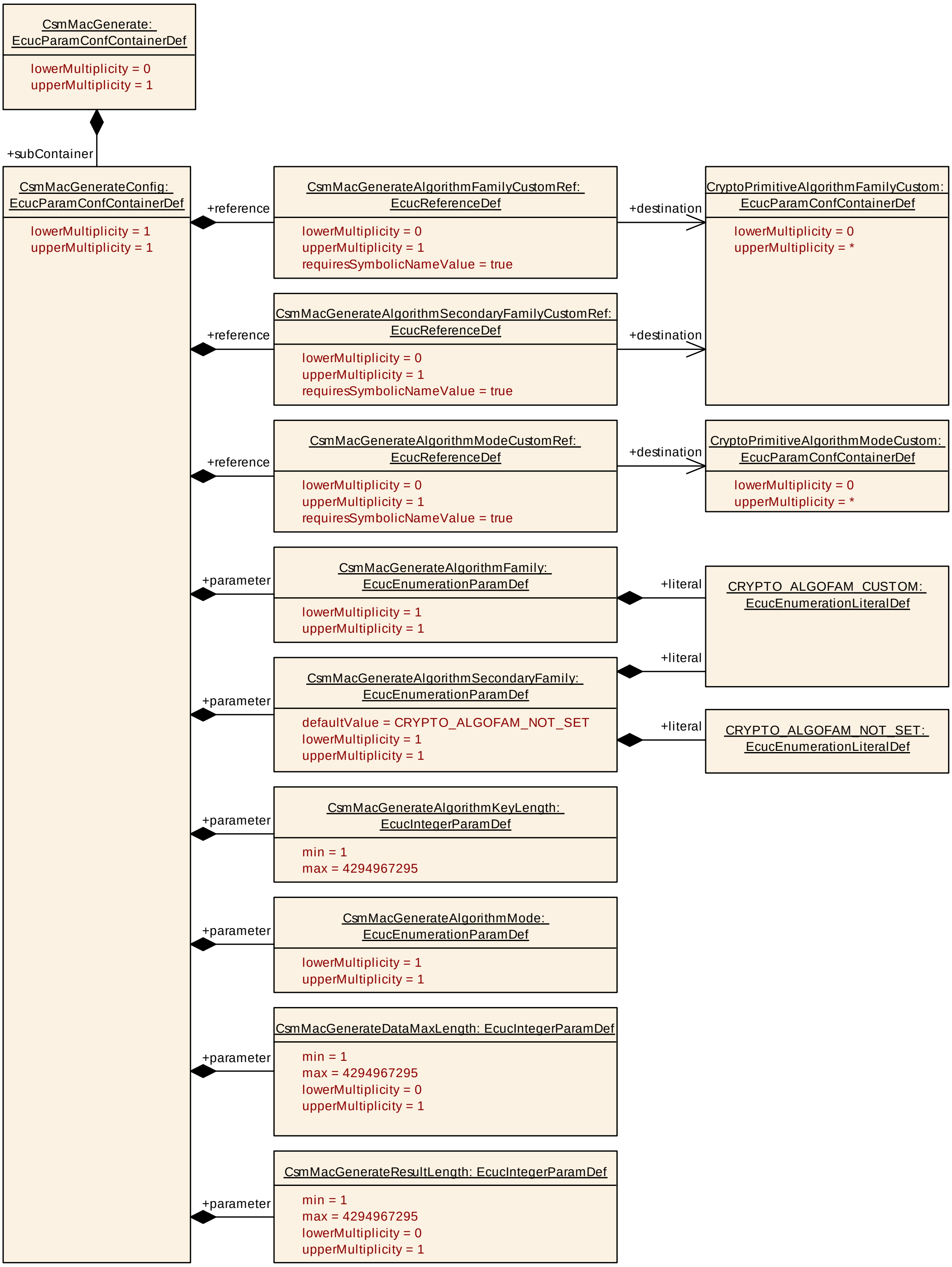

CsmMacGenerate: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmMacGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+reference
CsmMacGenerateAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmMacGenerateAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmMacGenerateAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmMacGenerateAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
+parameter
CsmMacGenerateAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+parameter
CsmMacGenerateAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
+parameter
CsmMacGenerateAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmMacGenerateDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmMacGenerateResultLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-10 CsmMacGenerate Layout

10.2.15 CsmMacGenerate

SWS Item	[ECUC_Csm_00022]

Container Name	CsmMacGenerate
Parent Container	CsmPrimitives
Description	Configurations of MacGenerate primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmMacGenerateConfig	1	Container for configuration of a CSM mac generation interface. The container name serves as a symbolic name for the identifier of a MAC generation interface.

10.2.16 CsmMacGenerateConfig

SWS Item	[ECUC_Csm_00041]
Container Name	CsmMacGenerateConfig
Parent Container	CsmMacGenerate
Description	Container for configuration of a CSM mac generation interface. The container name serves as a symbolic name for the identifier of a MAC generation interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00188]	
Parameter Name	CsmMacGenerateAlgorithmFamily	
Parent Container	CsmMacGenerateConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--
	CRYPTO_ALGOFAM_BLAKE_1_256	--
	CRYPTO_ALGOFAM_BLAKE_1_512	--
	CRYPTO_ALGOFAM_BLAKE_2s_256	--

	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_EEA3	--	
	CRYPTO_ALGOFAM_EIA3	--	
	CRYPTO_ALGOFAM_POLY1305	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_RNG	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
	CRYPTO_ALGOFAM_SIPHASH	--	
	CRYPTO_ALGOFAM_SM3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00044]		
Parameter Name	CsmMacGenerateAlgorithmKeyLength		
Parent Container	CsmMacGenerateConfig		
Description	Size of the MAC key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00189]	
Parameter Name	CsmMacGenerateAlgorithmMode	
Parent Container	CsmMacGenerateConfig	
Description	Determines the algorithm mode used for the crypto service	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOMODE_CMAC	--
	CRYPTO_ALGOMODE_CTRDRBG	--
	CRYPTO_ALGOMODE_CUSTOM	--
	CRYPTO_ALGOMODE_GMAC	--
	CRYPTO_ALGOMODE_HMAC	--
	CRYPTO_ALGOMODE_NOT_SET	--
	CRYPTO_ALGOMODE_SIPHASH_2_4	--

	CRYPTO_ALGOMODE_SIPHASH_4_8	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00134]		
Parameter Name	CsmMacGenerateAlgorithmSecondaryFamily		
Parent Container	CsmMacGenerateConfig		
Description	Determines the secondary algorithm family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00137]
Parameter Name	CsmMacGenerateDataMaxLength
Parent Container	CsmMacGenerateConfig

Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00138]		
Parameter Name	CsmMacGenerateResultLength		
Parent Container	CsmMacGenerateConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00285]		
Parameter Name	CsmMacGenerateAlgorithmFamilyCustomRef		
Parent Container	CsmMacGenerateConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmMacGenerate AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00286]		
Parameter Name	CsmMacGenerateAlgorithmModeCustomRef		
Parent Container	CsmMacGenerateConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This reference shall only be present if CsmMacGenerate AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.

SWS Item	[ECUC_Csm_00287]		
Parameter Name	CsmMacGenerateAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmMacGenerateConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmMacGenerate SecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

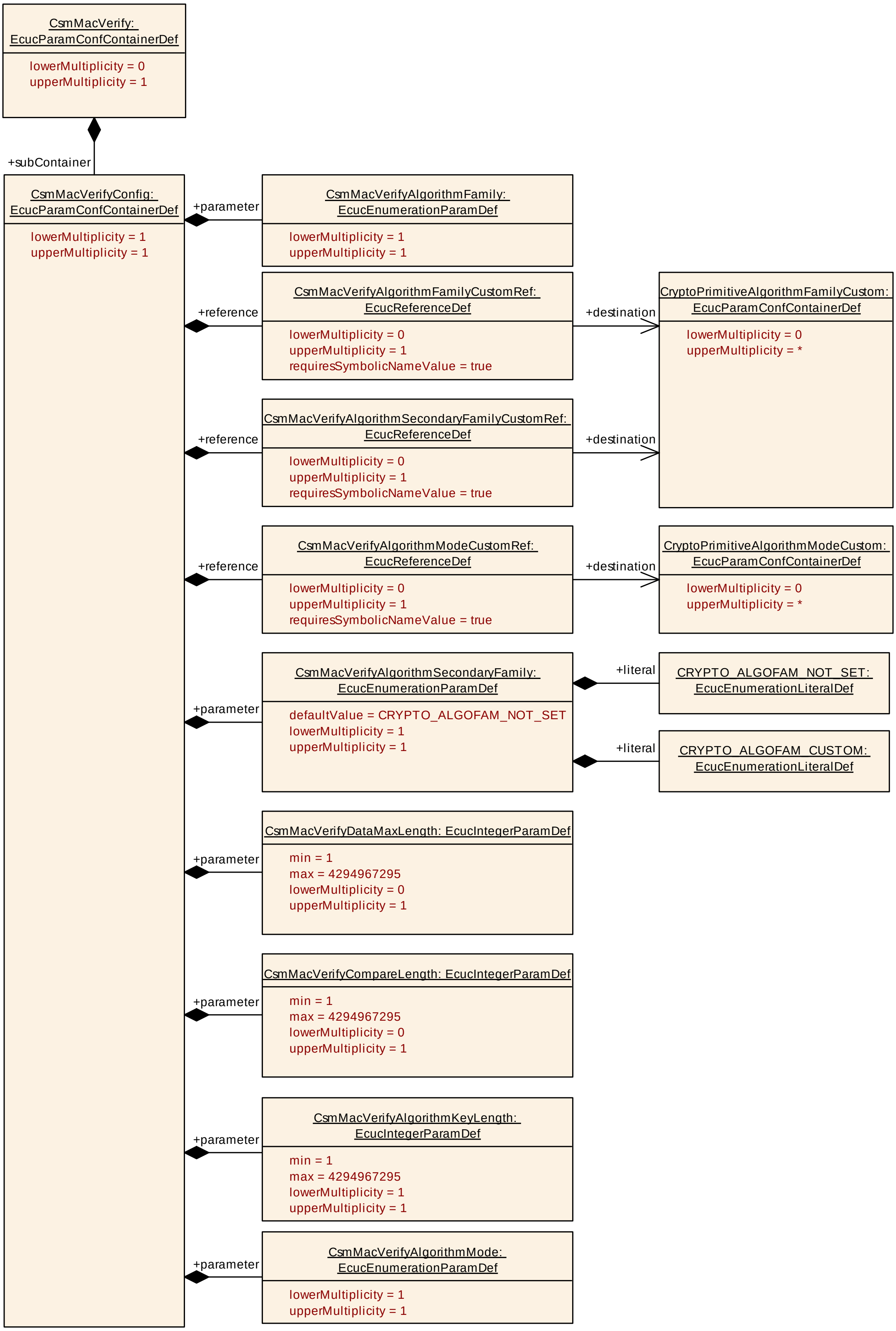

CsmMacVerify: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmMacVerifyConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmMacVerifyAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+reference
CsmMacVerifyAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmMacVerifyAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmMacVerifyAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmMacVerifyAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+parameter
CsmMacVerifyDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmMacVerifyCompareLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmMacVerifyAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmMacVerifyAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1

Figure 10-11 CsmMacVerify Layout

10.2.17 CsmMacVerify

SWS Item	[ECUC_Csm_00023]
Container Name	CsmMacVerify
Parent Container	CsmPrimitives
Description	Configurations of MacVerify primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmMac-VerifyConfig	1	Container for configuration of a CSM MAC verification interface. The container name serves as a symbolic name for the identifier of a MAC generation interface

10.2.18 CsmMacVerifyConfig

SWS Item	[ECUC_Csm_00049]
Container Name	CsmMacVerifyConfig
Parent Container	CsmMacVerify
Description	Container for configuration of a CSM MAC verification interface. The container name serves as a symbolic name for the identifier of a MAC generation interface
Configuration Parameters	

SWS Item	[ECUC_Csm_00051]	
Parameter Name	CsmMacVerifyAlgorithmFamily	
Parent Container	CsmMacVerifyConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--

	CRYPTO_ALGOFAM_BLAKE_1_256	--	
	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_EEA3	--	
	CRYPTO_ALGOFAM_EIA3	--	
	CRYPTO_ALGOFAM_POLY1305	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_RNG	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
	CRYPTO_ALGOFAM_SIPHASH	--	
	CRYPTO_ALGOFAM_SM3	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00193]		
Parameter Name	CsmMacVerifyAlgorithmKeyLength		
Parent Container	CsmMacVerifyConfig		
Description	Size of the MAC key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00195]	
Parameter Name	CsmMacVerifyAlgorithmMode	
Parent Container	CsmMacVerifyConfig	
Description	Determines the algorithm mode used for the crypto service	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOMODE_CMAC	--
	CRYPTO_ALGOMODE_CTRDRBG	--
	CRYPTO_ALGOMODE_CUSTOM	--
	CRYPTO_ALGOMODE_GMAC	--
	CRYPTO_ALGOMODE_HMAC	--

	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_SIPHASH_2_4	--	
	CRYPTO_ALGOMODE_SIPHASH_4_8	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00140]		
Parameter Name	CsmMacVerifyAlgorithmSecondaryFamily		
Parent Container	CsmMacVerifyConfig		
Description	Determines the secondary algorithm family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00142]
Parameter Name	CsmMacVerifyCompareLength

Parent Container	CsmMacVerifyConfig		
Description	Size of the input MAC buffer in BITS for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00056]		
Parameter Name	CsmMacVerifyDataMaxLength		
Parent Container	CsmMacVerifyConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00288]		
Parameter Name	CsmMacVerifyAlgorithmFamilyCustomRef		
Parent Container	CsmMacVerifyConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmMacVerify AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00289]		
Parameter Name	CsmMacVerifyAlgorithmModeCustomRef		
Parent Container	CsmMacVerifyConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmMacVerifyAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00290]		
Parameter Name	CsmMacVerifyAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmMacVerifyConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmMacVerifySecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

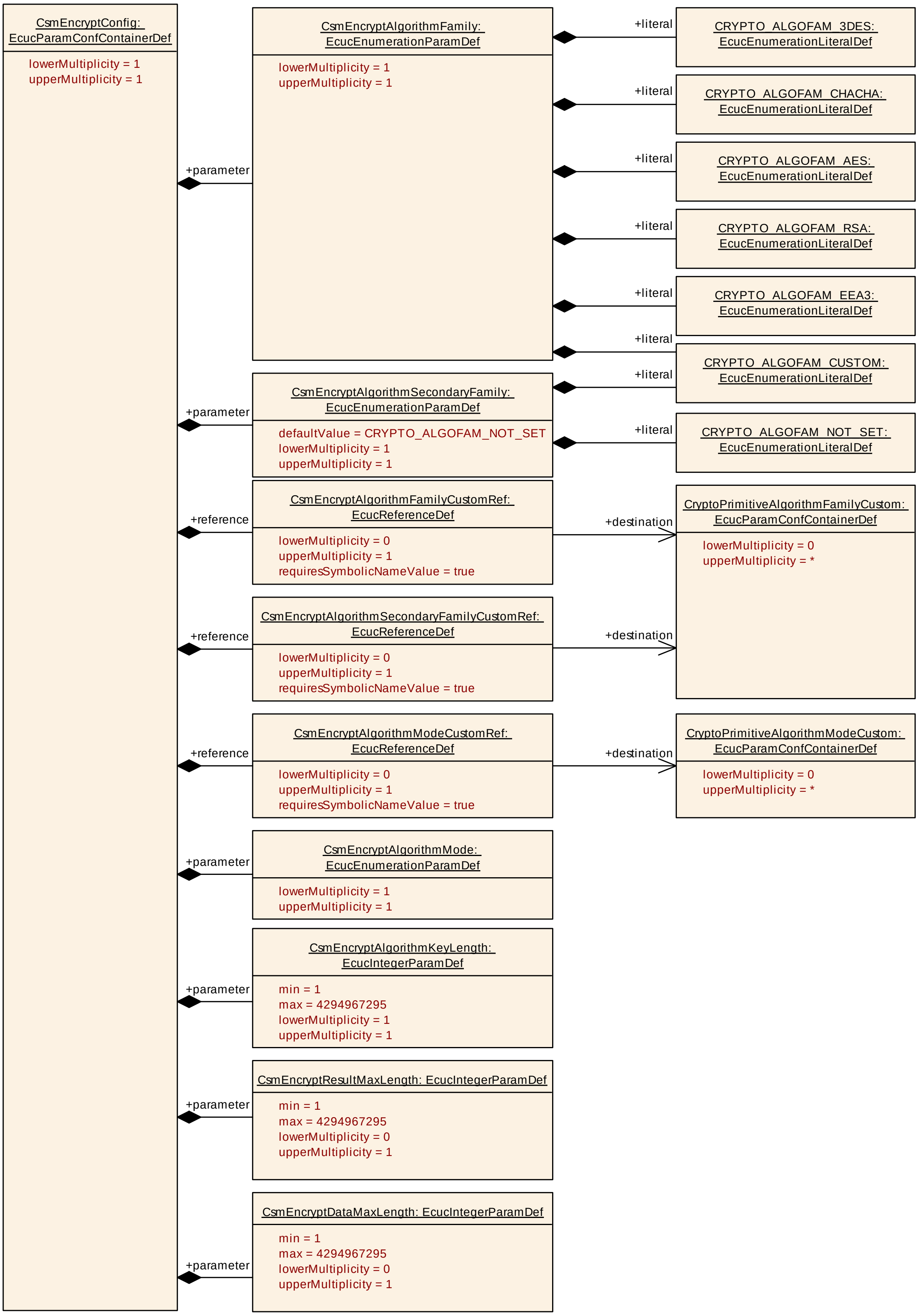

CsmEncryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmEncryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_RSA: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_EEA3: EcucEnumerationLiteralDef
+literal
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+parameter
CsmEncryptAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+reference
CsmEncryptAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmEncryptAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmEncryptAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmEncryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmEncryptAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmEncryptResultMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmEncryptDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-12 CsmEncrypt Layout

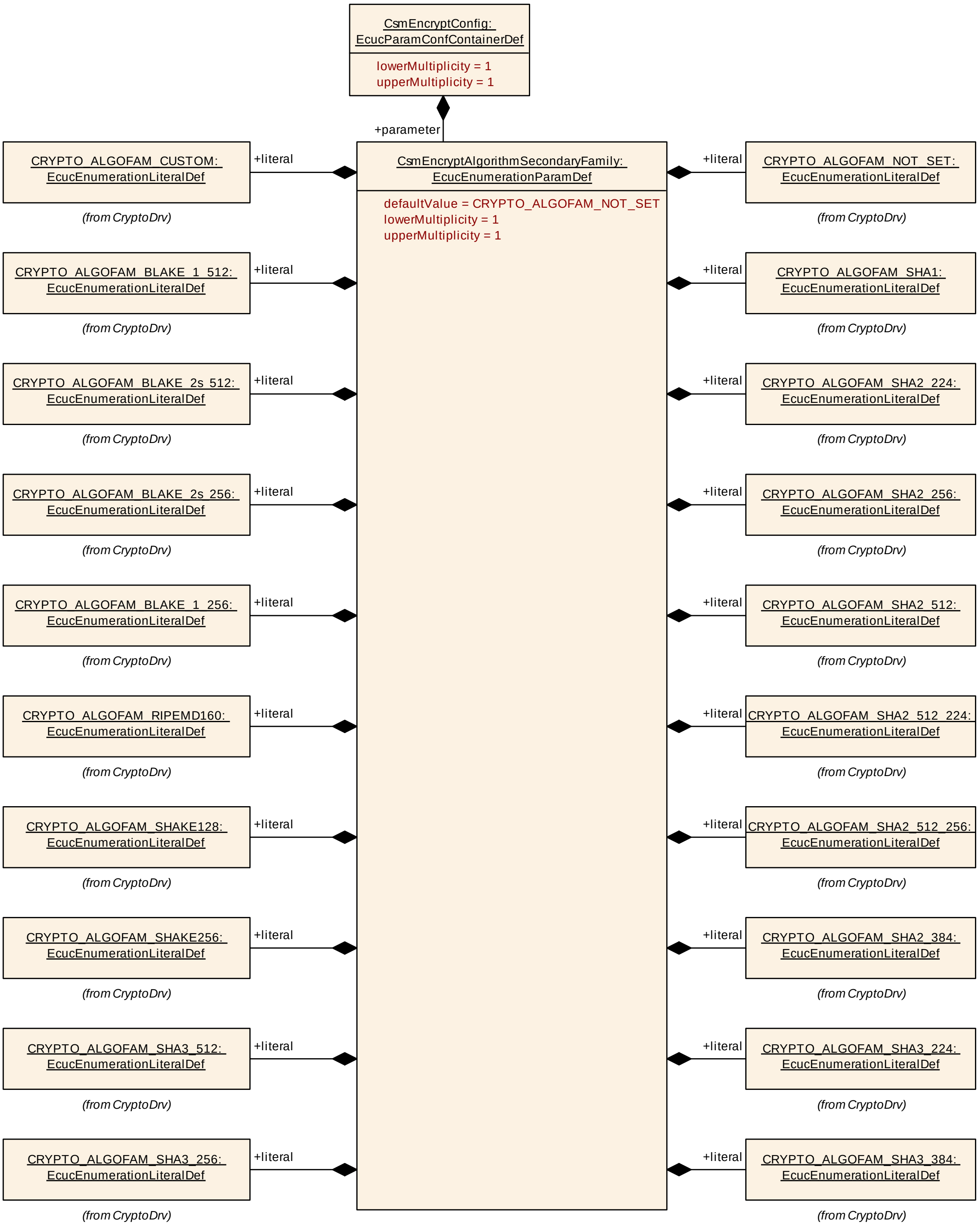

CsmEncryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmEncryptAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_1_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_1_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_RIPEMD160: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE128: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA1: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_384: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_384: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-13 CsmEncryptAlgorithmSecondaryFamily Layout

10.2.19 CsmEncrypt

SWS Item	[ECUC_Csm_00024]
Container Name	CsmEncrypt
Parent Container	CsmPrimitives

Description	Configurations of Encryption primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmEncrypt-Config	1	Container for configuration of a CSM encryption interface. The container name serves as a symbolic name for the identifier of an encryption interface.

10.2.20 CsmEncryptConfig

SWS Item	[ECUC_Csm_00057]
Container Name	CsmEncryptConfig
Parent Container	CsmEncrypt
Description	Container for configuration of a CSM encryption interface. The container name serves as a symbolic name for the identifier of an encryption interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00182]	
Parameter Name	CsmEncryptAlgorithmFamily	
Parent Container	CsmEncryptConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--
	CRYPTO_ALGOFAM_CHACHA	--
	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_EEA3	--
	CRYPTO_ALGOFAM_RSA	--

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00191]		
Parameter Name	CsmEncryptAlgorithmKeyLength		
Parent Container	CsmEncryptConfig		
Description	Size of the encryption key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00060]
Parameter Name	CsmEncryptAlgorithmMode
Parent Container	CsmEncryptConfig
Description	Determines the algorithm mode used for the crypto service
Multiplicity	1

Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_12ROUNDS	--	
	CRYPTO_ALGOMODE_20ROUNDS	--	
	CRYPTO_ALGOMODE_8ROUNDS	--	
	CRYPTO_ALGOMODE_AESKEYWRAP	--	
	CRYPTO_ALGOMODE_CBC	--	
	CRYPTO_ALGOMODE_CFB	--	
	CRYPTO_ALGOMODE_CTR	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_ECB	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_OFB	--	
	CRYPTO_ALGOMODE_RSAES_OAEP	--	
	CRYPTO_ALGOMODE_RSAES_PKCS1_v1_5	--	
	CRYPTO_ALGOMODE_XTS	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00144]	
Parameter Name	CsmEncryptAlgorithmSecondaryFamily	
Parent Container	CsmEncryptConfig	
Description	Determines the algorithm family used for the crypto service	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_BLAKE_1_256	--

	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00146]
Parameter Name	CsmEncryptDataMaxLength

Parent Container	CsmEncryptConfig		
Description	Size of the input plaintext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT."		

SWS Item	[ECUC_Csm_00147]		
Parameter Name	CsmEncryptResultMaxLength		
Parent Container	CsmEncryptConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00291]		
Parameter Name	CsmEncryptAlgorithmFamilyCustomRef		
Parent Container	CsmEncryptConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmEncryptAlgorithm Family is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00292]		
Parameter Name	CsmEncryptAlgorithmModeCustomRef		
Parent Container	CsmEncryptConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmEncryptAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00293]		
Parameter Name	CsmEncryptAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmEncryptConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmEncryptSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

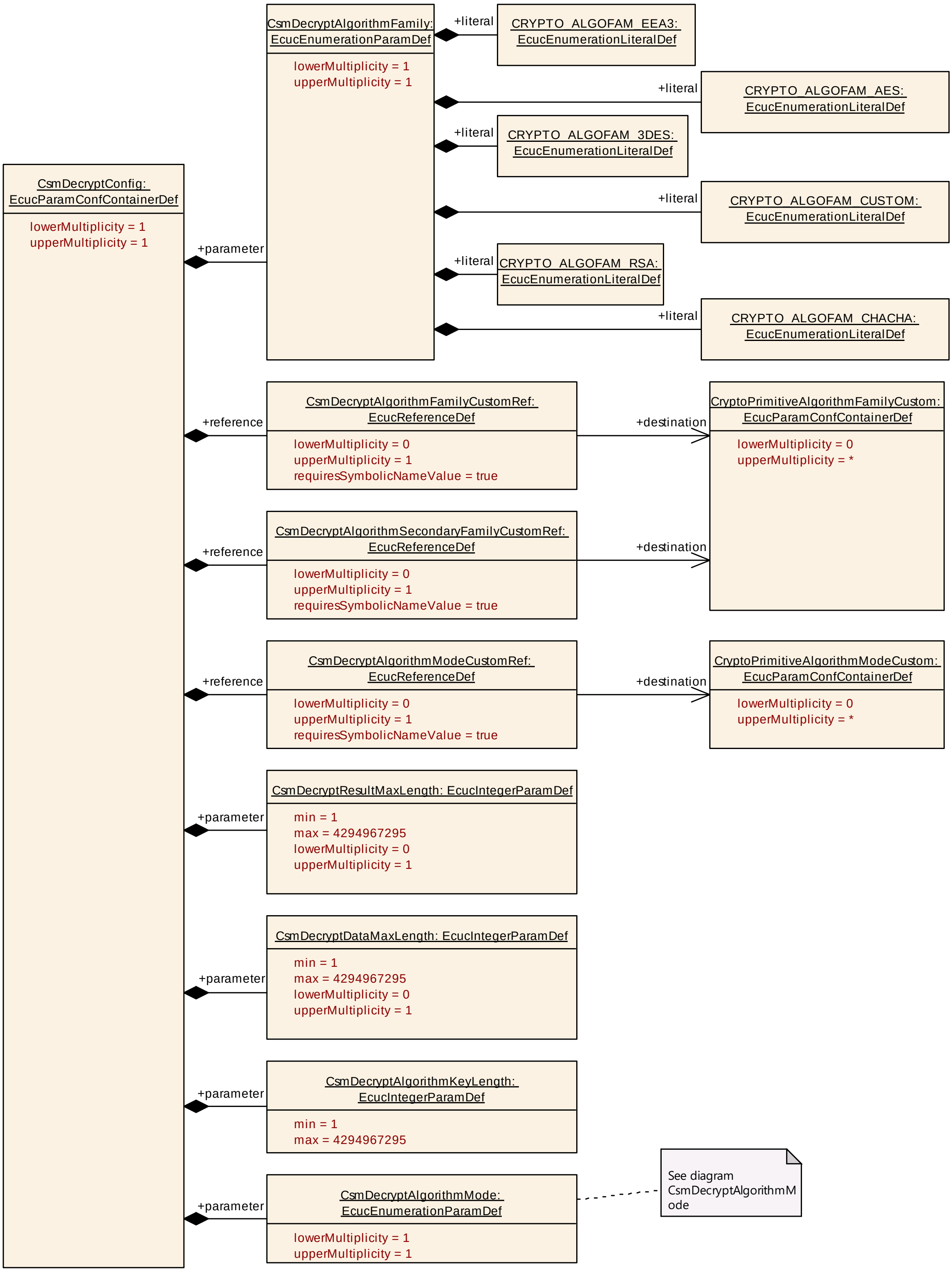

CsmDecryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_EEA3: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_RSA: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
CsmDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
+reference
CsmDecryptAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmDecryptAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmDecryptAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmDecryptResultMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmDecryptDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmDecryptAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
+parameter
CsmDecryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
See diagram CsmDecryptAlgorithmMode

Figure 10-14 CsmDecrypt Layout

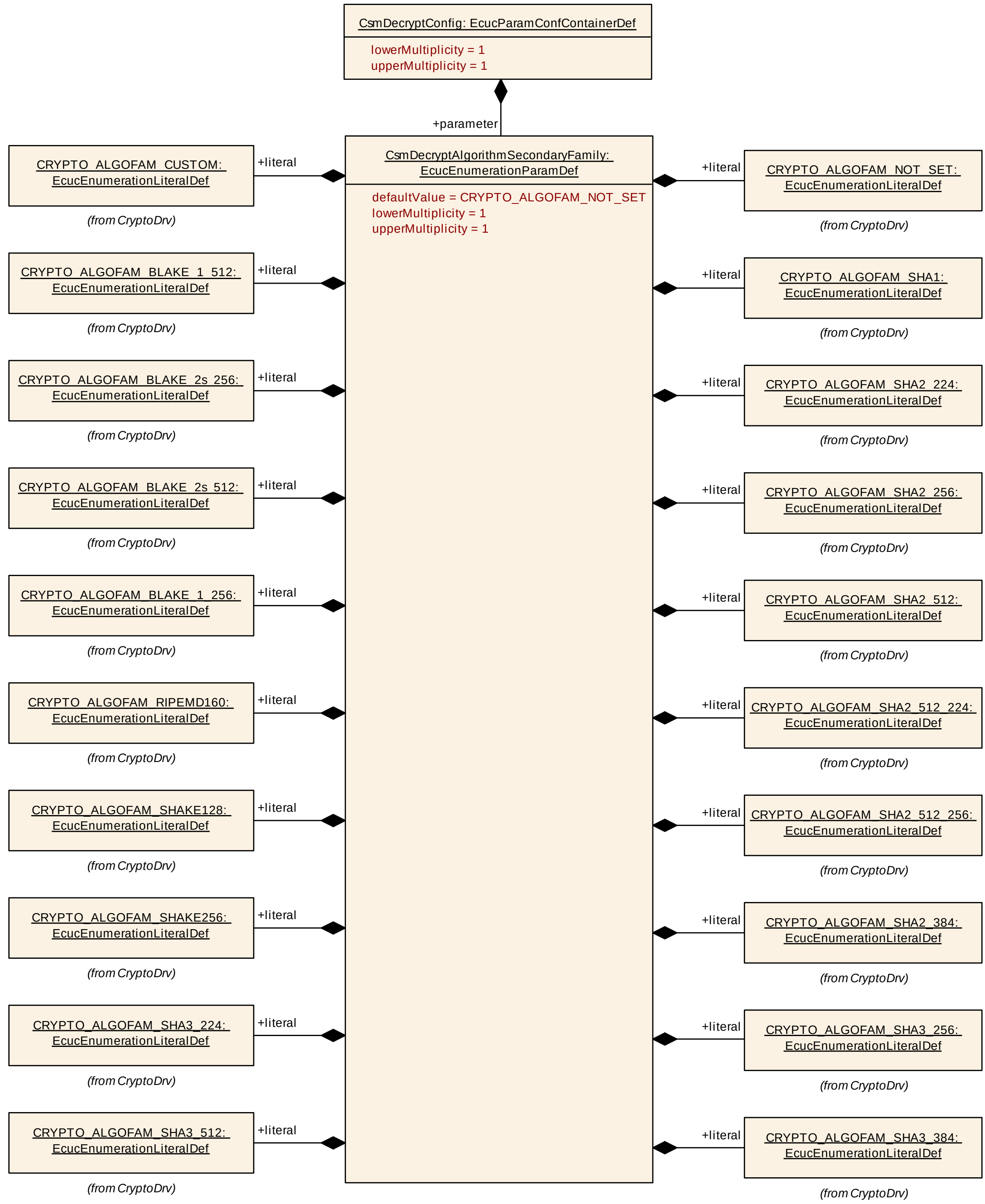

CsmDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmDecryptAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_BLAKE_1_512: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_BLAKE_2s_256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_BLAKE_2s_512: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_BLAKE_1_256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_RIPEMD160: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHAKE128: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHAKE256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA3_224: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA3_512: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA1: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_224: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_512: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_512_224: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_512_256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA2_384: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA3_256: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_SHA3_384: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-15 CsmDecryptAlgorithmSecondaryFamily Layout

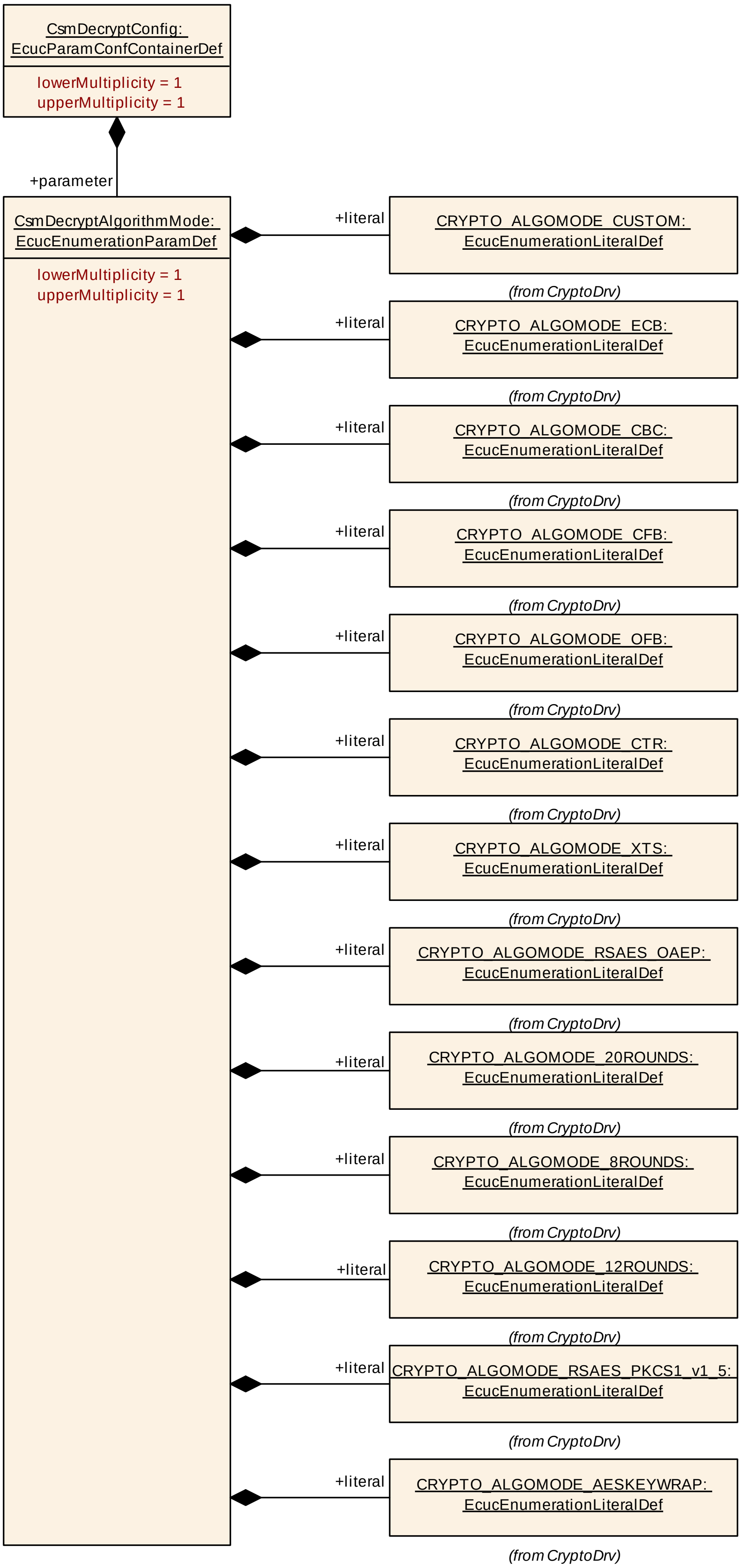

CsmDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmDecryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_ECB: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_CBC: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_CFB: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_OFB: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_CTR: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_XTS: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_RSAES_OAEP: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_20ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_8ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_12ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_RSAES_PKCS1_v1_5: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_AESKEYWRAP: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-16 CsmDecryptAlgorithmMode Layout

10.2.21 CsmDecrypt

SWS Item	[ECUC_Csm_00025]
Container Name	CsmDecrypt
Parent Container	CsmPrimitives
Description	Configurations of Decryption primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmDecrypt-Config	1	Container for configuration of a CSM decryption interface. The container name serves as a symbolic name for the identifier of an decryption interface.

10.2.22 CsmDecryptConfig

SWS Item	[ECUC_Csm_00064]
Container Name	CsmDecryptConfig
Parent Container	CsmDecrypt
Description	Container for configuration of a CSM decryption interface. The container name serves as a symbolic name for the identifier of an decryption interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00066]	
Parameter Name	CsmDecryptAlgorithmFamily	
Parent Container	CsmDecryptConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--

	CRYPTO_ALGOFAM_AES	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_EEA3	--	
	CRYPTO_ALGOFAM_RSA	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00067]		
Parameter Name	CsmDecryptAlgorithmKeyLength		
Parent Container	CsmDecryptConfig		
Description	Size of the encryption key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00068]		
Parameter Name	CsmDecryptAlgorithmMode		
Parent Container	CsmDecryptConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_12ROUNDS	--	
	CRYPTO_ALGOMODE_20ROUNDS	--	
	CRYPTO_ALGOMODE_8ROUNDS	--	
	CRYPTO_ALGOMODE_AESKEYWRAP	--	
	CRYPTO_ALGOMODE_CBC	--	
	CRYPTO_ALGOMODE_CFB	--	
	CRYPTO_ALGOMODE_CTR	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_ECB	--	
	CRYPTO_ALGOMODE_OFB	--	
	CRYPTO_ALGOMODE_RSAES_OAEP	--	
	CRYPTO_ALGOMODE_RSAES_PKCS1_v1_5	--	
	CRYPTO_ALGOMODE_XTS	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00149]
Parameter Name	CsmDecryptAlgorithmSecondaryFamily
Parent Container	CsmDecryptConfig

Description	Determines the secondary algorithm family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_BLAKE_1_256	--	
	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00154]		
Parameter Name	CsmDecryptDataMaxLength		
Parent Container	CsmDecryptConfig		
Description	Size of the input ciphertext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00155]	
Parameter Name	CsmDecryptResultMaxLength	
Parent Container	CsmDecryptConfig	
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 4294967295	
Default value	--	

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00294]		
Parameter Name	CsmDecryptAlgorithmFamilyCustomRef		
Parent Container	CsmDecryptConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmDecryptAlgorithm Family is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00295]
Parameter Name	CsmDecryptAlgorithmModeCustomRef
Parent Container	CsmDecryptConfig
Description	Reference to a customer specific algorithm mode custom container
Multiplicity	0..1
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmDecryptAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00296]		
Parameter Name	CsmDecryptAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmDecryptConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmDecryptSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

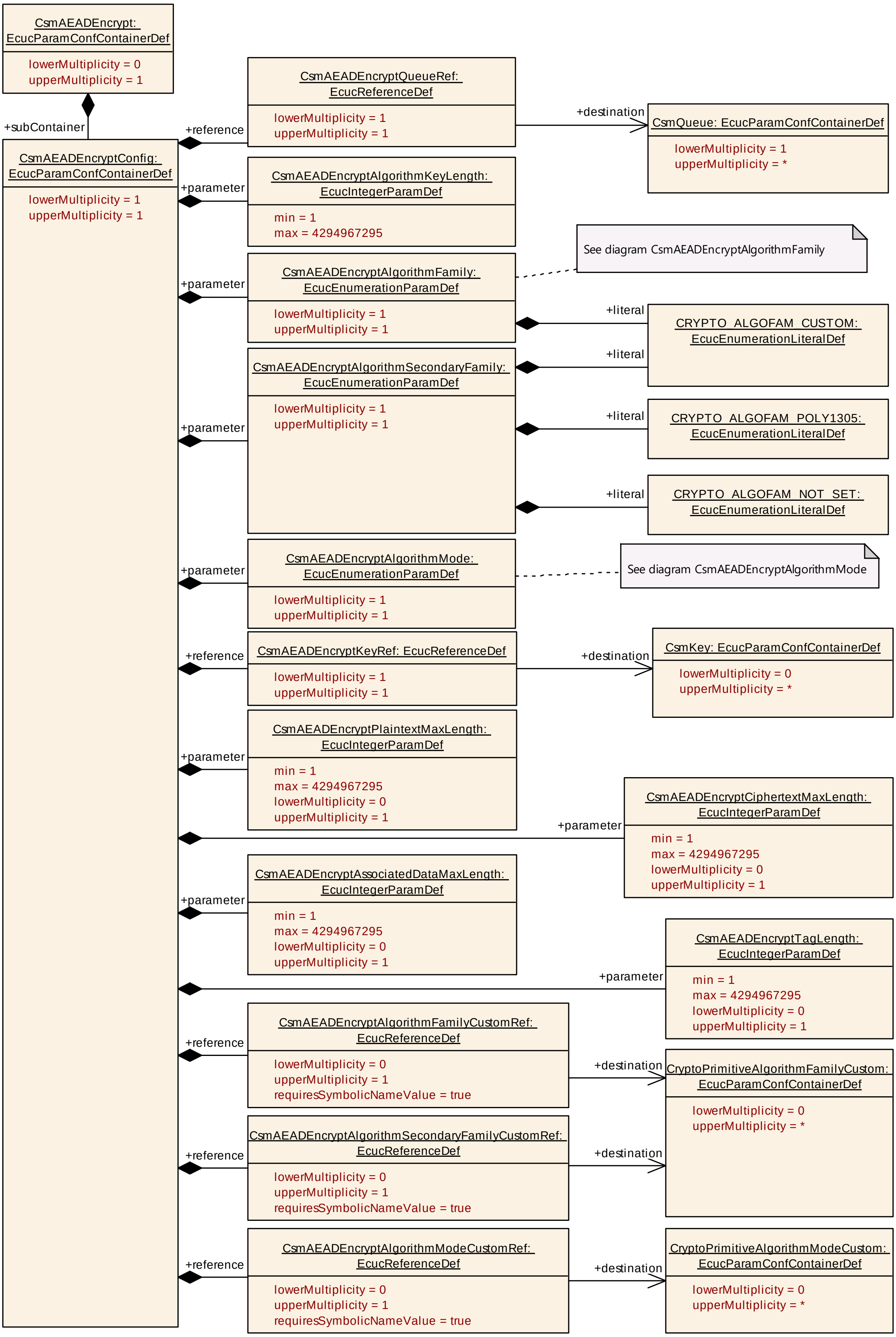

CsmAEADEncrypt: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmAEADEncryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+reference
CsmAEADEncryptQueueRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmQueue: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+parameter
CsmAEADEncryptAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
See diagram CsmAEADEncryptAlgorithmFamily
+parameter
CsmAEADEncryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CsmAEADEncryptAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
+literal
CRYPTO_ALGOFAM_POLY1305: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+parameter
CsmAEADEncryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
See diagram CsmAEADEncryptAlgorithmMode
+reference
CsmAEADEncryptKeyRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmKey: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmAEADEncryptPlaintextMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmAEADEncryptCiphertextMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmAEADEncryptAssociatedDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmAEADEncryptTagLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+reference
CsmAEADEncryptAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmAEADEncryptAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmAEADEncryptAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *

Figure 10-17 CsmAEADEncrypt Layout

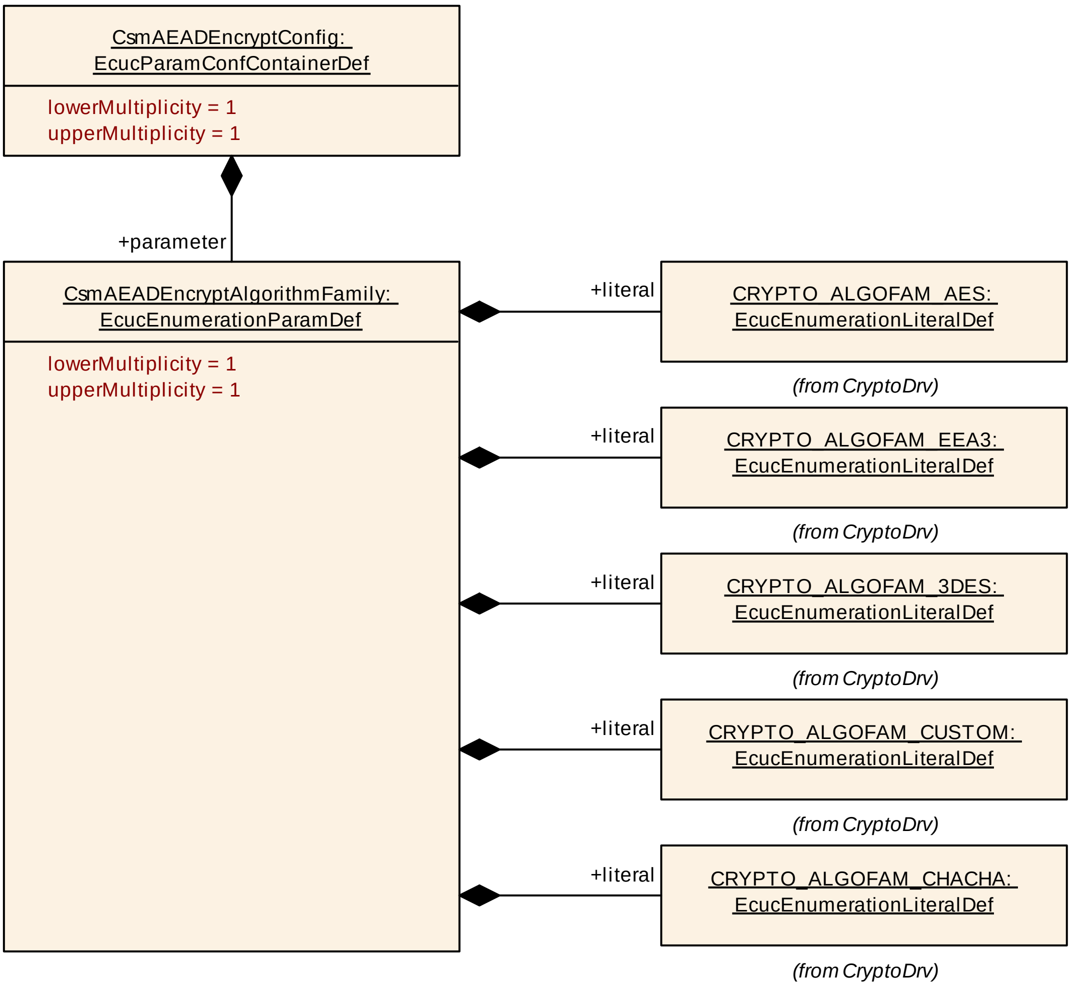

CsmAEADEncryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmAEADEncryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_EEA3: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-18 CsmAEADEncryptAlgorithmFamily Layout

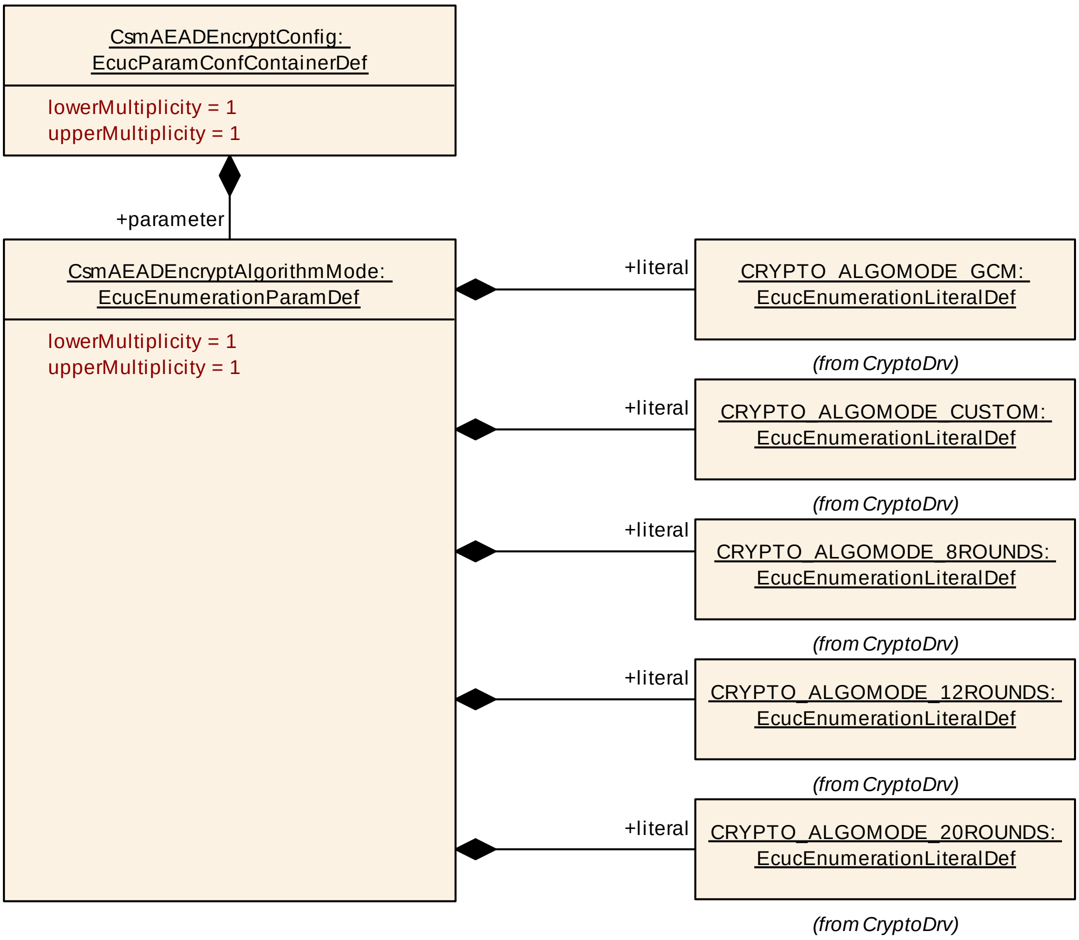

CsmAEADEncryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmAEADEncryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_GCM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_8ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_12ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_20ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-19 CsmAEADEncryptAlgorithmMode Layout

10.2.23 CsmAEADEncrypt

SWS Item	[ECUC_Csm_00026]
Container Name	CsmAEADEncrypt
Parent Container	CsmPrimitives
Description	Configuration of AEAD encryption primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmAEAD-EncryptConfig	1	Container for configuration of a CSM encryption interface. The container name serves as a symbolic name for the identifier of an encryption interface.

10.2.24 CsmAEADEncryptConfig

SWS Item	[ECUC_Csm_00072]
Container Name	CsmAEADEncryptConfig
Parent Container	CsmAEADEncrypt
Description	Container for configuration of a CSM encryption interface. The container name serves as a symbolic name for the identifier of an encryption interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00074]		
Parameter Name	CsmAEADEncryptAlgorithmFamily		
Parent Container	CsmAEADEncryptConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_3DES	--	
	CRYPTO_ALGOFAM_AES	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_EEA3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00075]
Parameter Name	CsmAEADEncryptAlgorithmKeyLength

Parent Container	CsmAEADEncryptConfig		
Description	Size of the AEAD encryption key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00076]		
Parameter Name	CsmAEADEncryptAlgorithmMode		
Parent Container	CsmAEADEncryptConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_12ROUNDS	--	
	CRYPTO_ALGOMODE_20ROUNDS	--	
	CRYPTO_ALGOMODE_8ROUNDS	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_GCM	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants

	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00278]		
Parameter Name	CsmAEADEncryptAlgorithmSecondaryFamily		
Parent Container	CsmAEADEncryptConfig		
Description	Defines the secondary family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_POLY1305	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00159]	
Parameter Name	CsmAEADEncryptAssociatedDataMaxLength	
Parent Container	CsmAEADEncryptConfig	
Description	Size of the input associated data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 4294967295	
Default value	--	
Post-Build Variant Value	false	

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00160]		
Parameter Name	CsmAEADEncryptCiphertextMaxLength		
Parent Container	CsmAEADEncryptConfig		
Description	Size of the output ciphertext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00158]
Parameter Name	CsmAEADEncryptPlaintextMaxLength
Parent Container	CsmAEADEncryptConfig

Description	Size of the input plaintext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00161]		
Parameter Name	CsmAEADEncryptTagLength		
Parent Container	CsmAEADEncryptConfig		
Description	Size of the output tag length buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00297]		
Parameter Name	CsmAEADEncryptAlgorithmFamilyCustomRef		
Parent Container	CsmAEADEncryptConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADEncrypt AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00298]		
Parameter Name	CsmAEADEncryptAlgorithmModeCustomRef		
Parent Container	CsmAEADEncryptConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADEncryptAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.

SWS Item	[ECUC_Csm_00299]		
Parameter Name	CsmAEADEncryptAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmAEADEncryptConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADEncryptSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00157]		
Parameter Name	CsmAEADEncryptKeyRef		
Parent Container	CsmAEADEncryptConfig		
Description	This parameter refers to the key used for that encryption primitive.		
Multiplicity	1		
Type	Reference to CsmKey		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00156]		
Parameter Name	CsmAEADEncryptQueueRef		
Parent Container	CsmAEADEncryptConfig		
Description	This parameter refers to the queue used for that encryption primitive.		
Multiplicity	1		
Type	Reference to CsmQueue		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

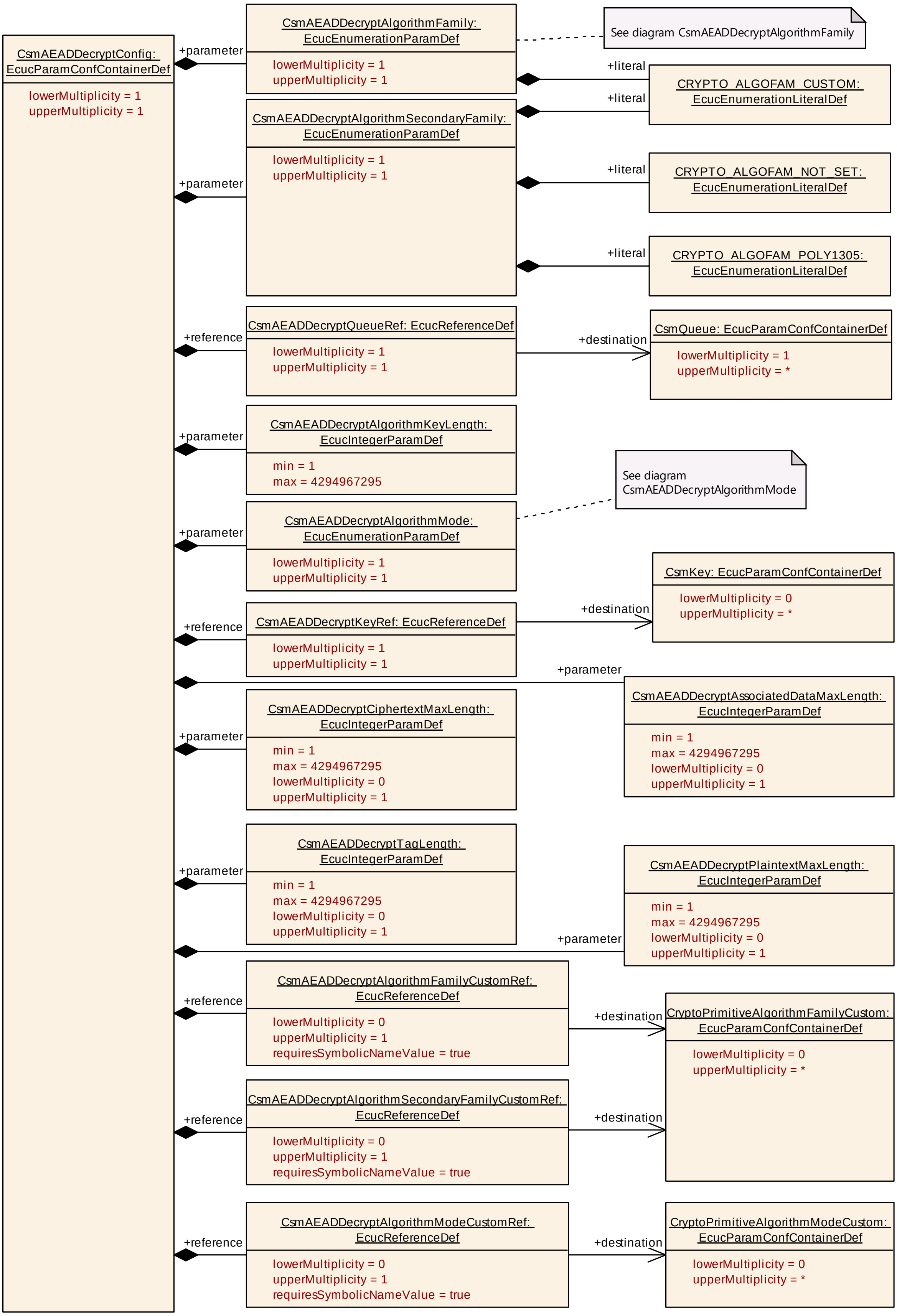

CsmAEADDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmAEADDecryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
See diagram CsmAEADDecryptAlgorithmFamily
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CsmAEADDecryptAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_POLY1305: EcucEnumerationLiteralDef
+reference
CsmAEADDecryptQueueRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CsmQueue: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+parameter
CsmAEADDecryptAlgorithmKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
See diagram CsmAEADDecryptAlgorithmMode
+parameter
CsmAEADDecryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
CsmKey: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmAEADDecryptKeyRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
+parameter
CsmAEADDecryptAssociatedDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmAEADDecryptCiphertextMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmAEADDecryptTagLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
CsmAEADDecryptPlaintextMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
+reference
CsmAEADDecryptAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmAEADDecryptAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmAEADDecryptAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *

Figure 10-20 CsmAEADDecrypt Layout

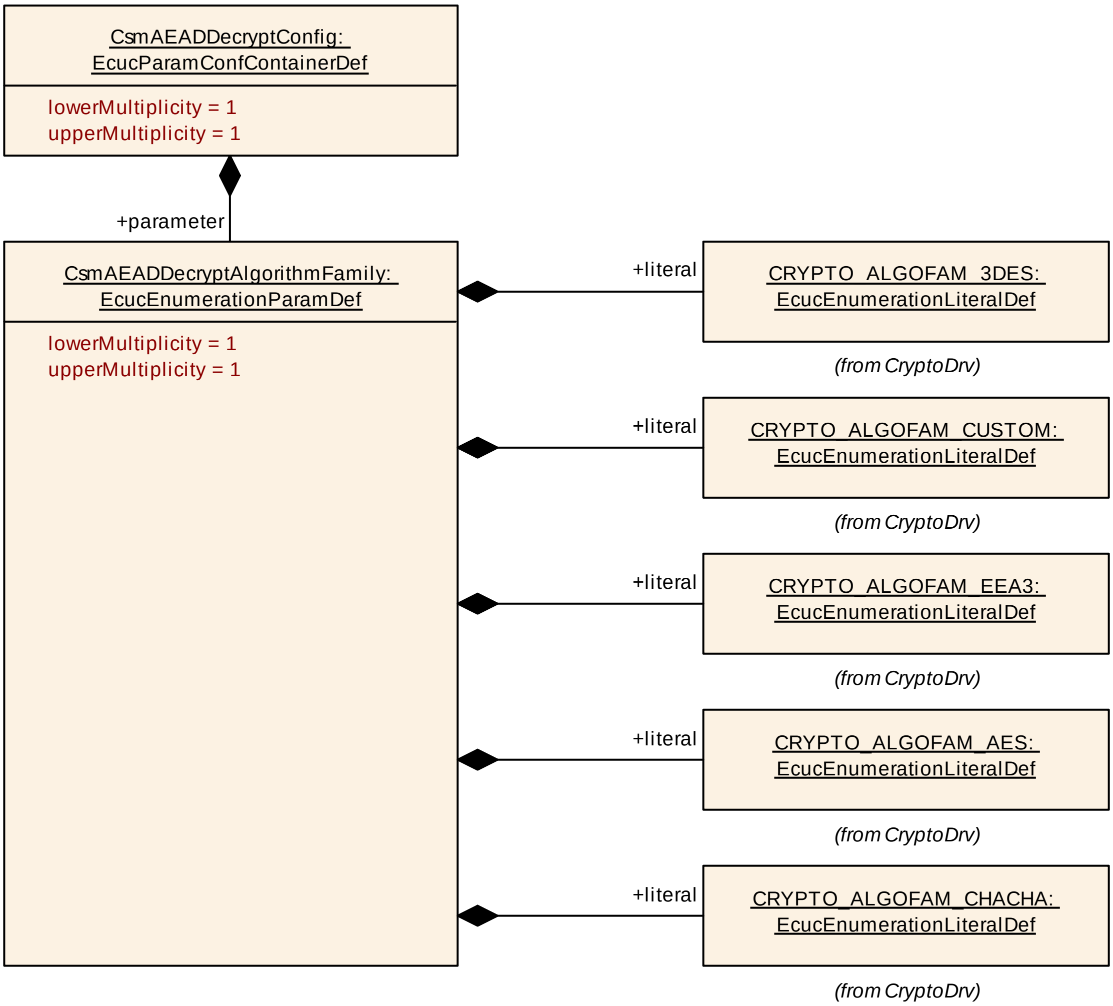

CsmAEADDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmAEADDecryptAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_EEA3: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-21 CsmAEADEncryptAlgorithmFamily Layout

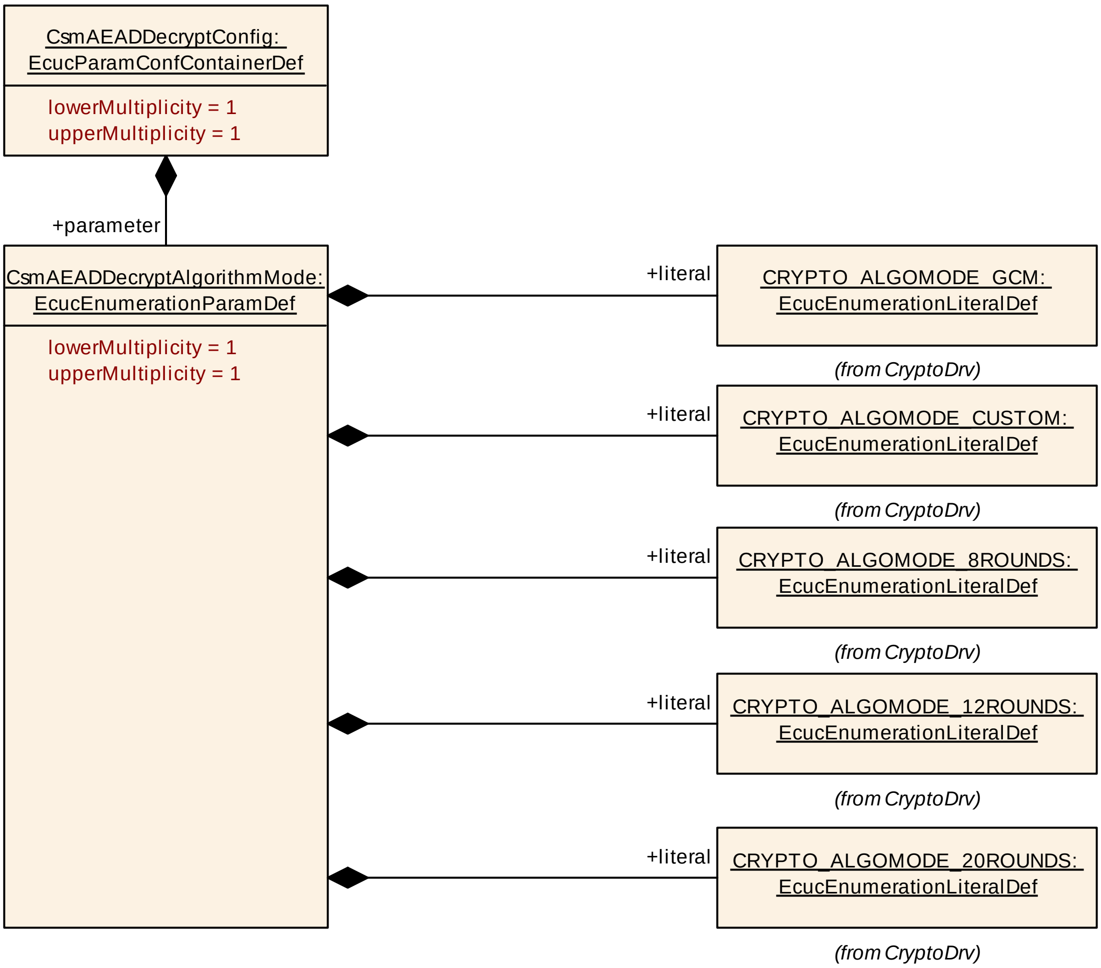

CsmAEADDecryptConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmAEADDecryptAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_GCM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_8ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_12ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_20ROUNDS: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-22 CsmAEADEncryptAlgorithmMode Layout

10.2.25 CsmAEADDecrypt

SWS Item	[ECUC_Csm_00027]
Container Name	CsmAEADDecrypt
Parent Container	CsmPrimitives
Description	Configuration of AEAD decryption primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmAEADDecryptConfig	1	Container for configuration of a CSM decryption interface. The container name serves as a symbolic name for the identifier of an decryption interface.

10.2.26 CsmAEADDecryptConfig

SWS Item	[ECUC_Csm_00080]
Container Name	CsmAEADDecryptConfig
Parent Container	CsmAEADDecrypt
Description	Container for configuration of a CSM decryption interface. The container name serves as a symbolic name for the identifier of an decryption interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00082]		
Parameter Name	CsmAEADDecryptAlgorithmFamily		
Parent Container	CsmAEADDecryptConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_3DES	--	
	CRYPTO_ALGOFAM_AES	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_EEA3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00083]		
Parameter Name	CsmAEADDecryptAlgorithmKeyLength		
Parent Container	CsmAEADDecryptConfig		
Description	Size of the AEAD decryption key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00084]		
Parameter Name	CsmAEADDecryptAlgorithmMode		
Parent Container	CsmAEADDecryptConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_12ROUNDS	--	
	CRYPTO_ALGOMODE_20ROUNDS	--	
	CRYPTO_ALGOMODE_8ROUNDS	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_GCM	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00277]		
Parameter Name	CsmAEADDecryptAlgorithmSecondaryFamily		
Parent Container	CsmAEADDecryptConfig		
Description	Defines the secondary family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_POLY1305	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00163]		
Parameter Name	CsmAEADDecryptAssociatedDataMaxLength		
Parent Container	CsmAEADDecryptConfig		
Description	Size of the input associated data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00162]		
Parameter Name	CsmAEADDecryptCiphertextMaxLength		
Parent Container	CsmAEADDecryptConfig		
Description	Size of the input ciphertext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT		

SWS Item	[ECUC_Csm_00165]
Parameter Name	CsmAEADDecryptPlaintextMaxLength
Parent Container	CsmAEADDecryptConfig

Description	Size of the output plaintext buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00164]		
Parameter Name	CsmAEADDecryptTagLength		
Parent Container	CsmAEADDecryptConfig		
Description	Size of the input tag buffer in BITS for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation."		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00300]		
Parameter Name	CsmAEADDecryptAlgorithmFamilyCustomRef		
Parent Container	CsmAEADDecryptConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADDecrypt AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00301]		
Parameter Name	CsmAEADDecryptAlgorithmModeCustomRef		
Parent Container	CsmAEADDecryptConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADDecrypt AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.

SWS Item	[ECUC_Csm_00302]		
Parameter Name	CsmAEADDecryptAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmAEADDecryptConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmAEADDecrypt SecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00086]		
Parameter Name	CsmAEADDecryptKeyRef		
Parent Container	CsmAEADDecryptConfig		
Description	This parameter refers to the key used for that decryption primitive.		
Multiplicity	1		
Type	Reference to CsmKey		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00081]		
Parameter Name	CsmAEADDecryptQueueRef		
Parent Container	CsmAEADDecryptConfig		
Description	This parameter refers to the queue used for that decryption primitive.		
Multiplicity	1		
Type	Reference to CsmQueue		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

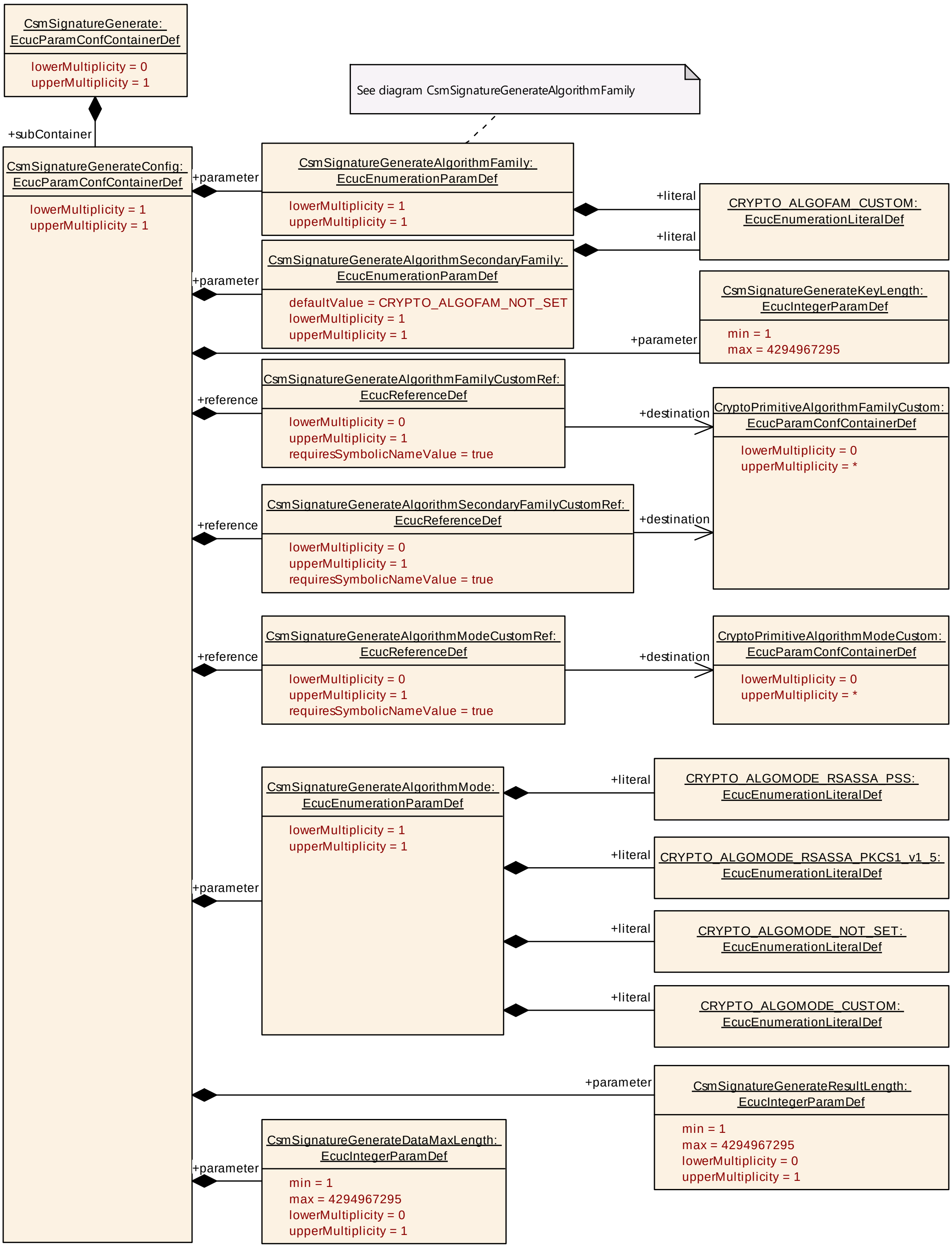

CsmSignatureGenerate: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
See diagram CsmSignatureGenerateAlgorithmFamily
+subContainer
CsmSignatureGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmSignatureGenerateAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
+parameter
CsmSignatureGenerateAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
CsmSignatureGenerateKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
+parameter
+reference
CsmSignatureGenerateAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmSignatureGenerateAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmSignatureGenerateAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
CsmSignatureGenerateAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_RSASSA_PSS: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_RSASSA_PKCS1_v1_5: EcucEnumerationLiteralDef
+parameter
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
+parameter
CsmSignatureGenerateResultLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
CsmSignatureGenerateDataMaxLength: EcucIntegerParamDef
+parameter
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-23 CsmSignatureGenerate Layout

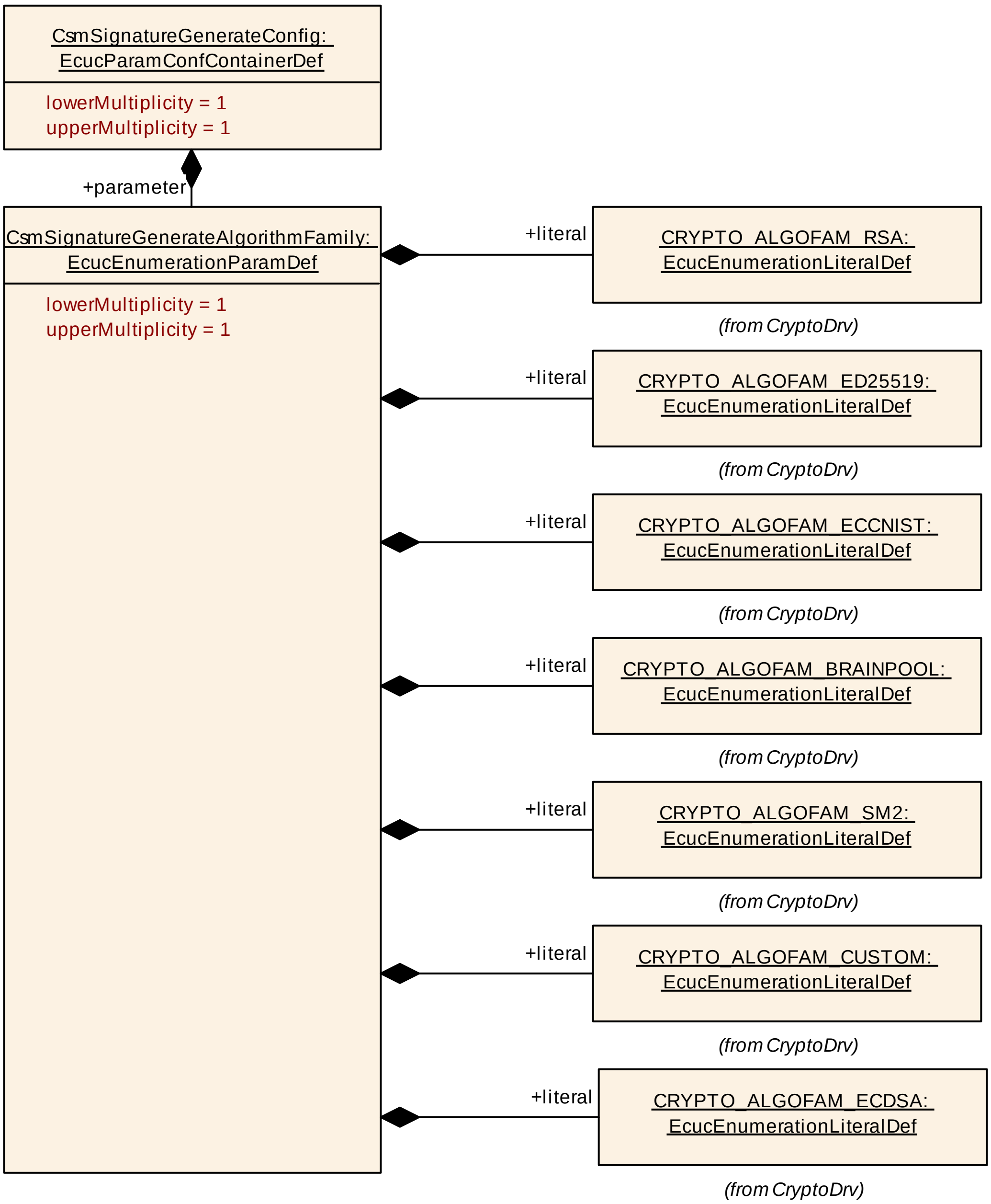

CsmSignatureGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmSignatureGenerateAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_RSA: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ED25519: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ECCNIST: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_BRAINPOOL: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_SM2: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ECDSA: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-24 CsmSignatureGenerateAlgorithmFamily Layout

10.2.27 CsmSignatureGenerate

SWS Item	[ECUC_Csm_00028]
Container Name	CsmSignatureGenerate
Parent Container	CsmPrimitives
Description	Configurations of SignatureGenerate primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmSignatureGenerateConfig	1	Container for configuration of a CSM signature generation interface. The container name serves as a symbolic name for the identifier of signature generation interface.

10.2.28 CsmSignatureGenerateConfig

SWS Item	[ECUC_Csm_00087]
Container Name	CsmSignatureGenerateConfig
Parent Container	CsmSignatureGenerate
Description	Container for configuration of a CSM signature generation interface. The container name serves as a symbolic name for the identifier of signature generation interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00089]		
Parameter Name	CsmSignatureGenerateAlgorithmFamily		
Parent Container	CsmSignatureGenerateConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_BRAINPOOL	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_ECCNIST	--	
	CRYPTO_ALGOFAM_ECDSA	--	
	CRYPTO_ALGOFAM_ED25519	--	
	CRYPTO_ALGOFAM_RSA	--	
	CRYPTO_ALGOFAM_SM2	--	
Post-Build Variant Value	false		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00091]		
Parameter Name	CsmSignatureGenerateAlgorithmMode		
Parent Container	CsmSignatureGenerateConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_RSASSA_PKCS1_v1_5	--	
	CRYPTO_ALGOMODE_RSASSA_PSS	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00183]
Parameter Name	CsmSignatureGenerateAlgorithmSecondaryFamily
Parent Container	CsmSignatureGenerateConfig
Description	Determines the algorithm mode used for the crypto service

Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_BLAKE_1_256	--	
	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00169]		
Parameter Name	CsmSignatureGenerateDataMaxLength		
Parent Container	CsmSignatureGenerateConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00090]		
Parameter Name	CsmSignatureGenerateKeyLength		
Parent Container	CsmSignatureGenerateConfig		
Description	Size of the signature generate key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00170]		
Parameter Name	CsmSignatureGenerateResultLength		
Parent Container	CsmSignatureGenerateConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00303]
Parameter Name	CsmSignatureGenerateAlgorithmFamilyCustomRef
Parent Container	CsmSignatureGenerateConfig
Description	Reference to a customer specific algorithm family custom container
Multiplicity	0..1
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmSignatureGenerateAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00304]		
Parameter Name	CsmSignatureGenerateAlgorithmModeCustomRef		
Parent Container	CsmSignatureGenerateConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmSignatureGenerateAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00305]		
Parameter Name	CsmSignatureGenerateAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmSignatureGenerateConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmSignatureGenerateSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

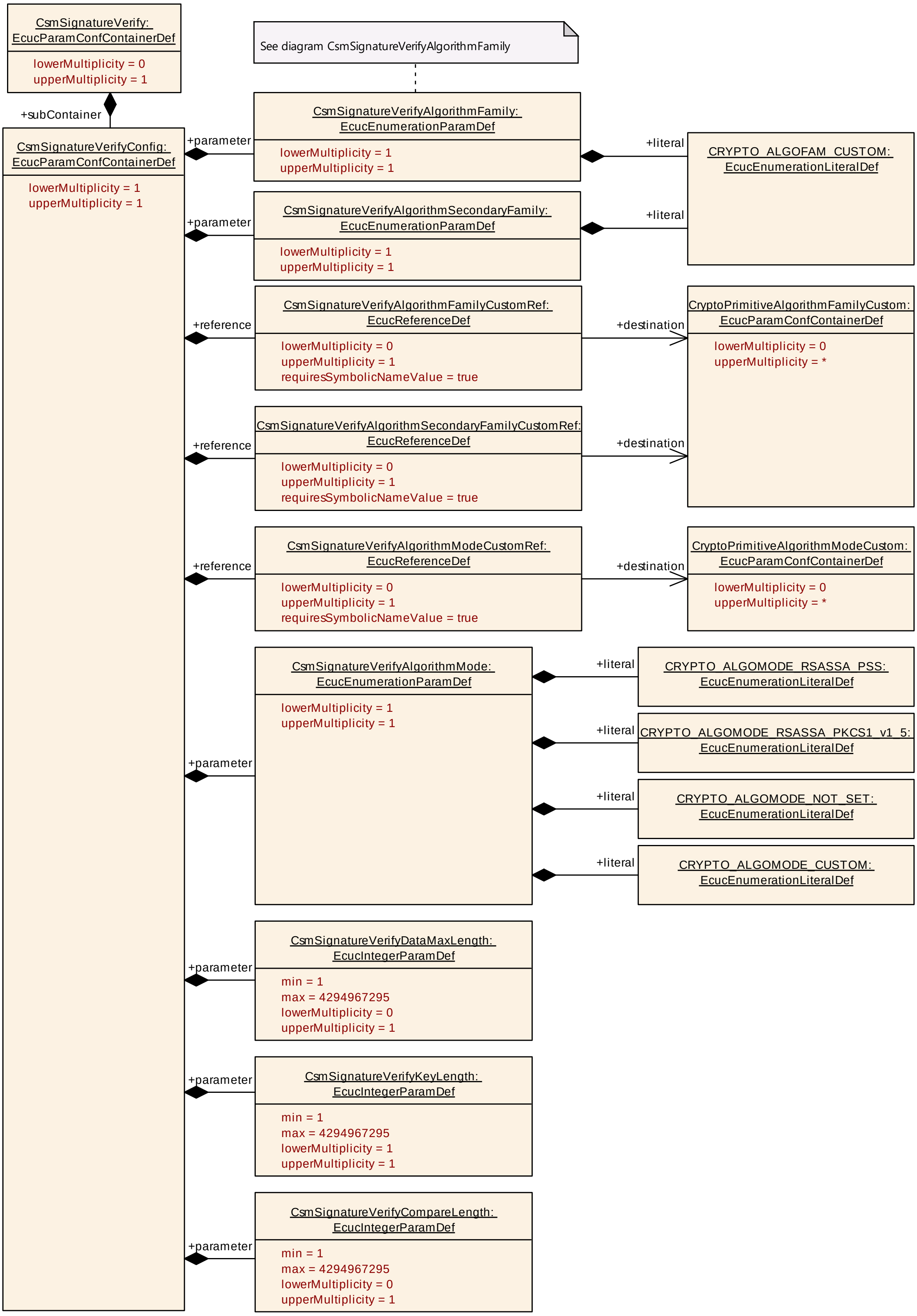

CsmSignatureVerify: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmSignatureVerifyConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
See diagram CsmSignatureVerifyAlgorithmFamily
+parameter
CsmSignatureVerifyAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+parameter
CsmSignatureVerifyAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
+reference
CsmSignatureVerifyAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmSignatureVerifyAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmSignatureVerifyAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+parameter
CsmSignatureVerifyAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_RSASSA_PSS: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_RSASSA_PKCS1_v1_5: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
+parameter
CsmSignatureVerifyDataMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
CsmSignatureVerifyKeyLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmSignatureVerifyCompareLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-25 CsmSignatureVerify Layout

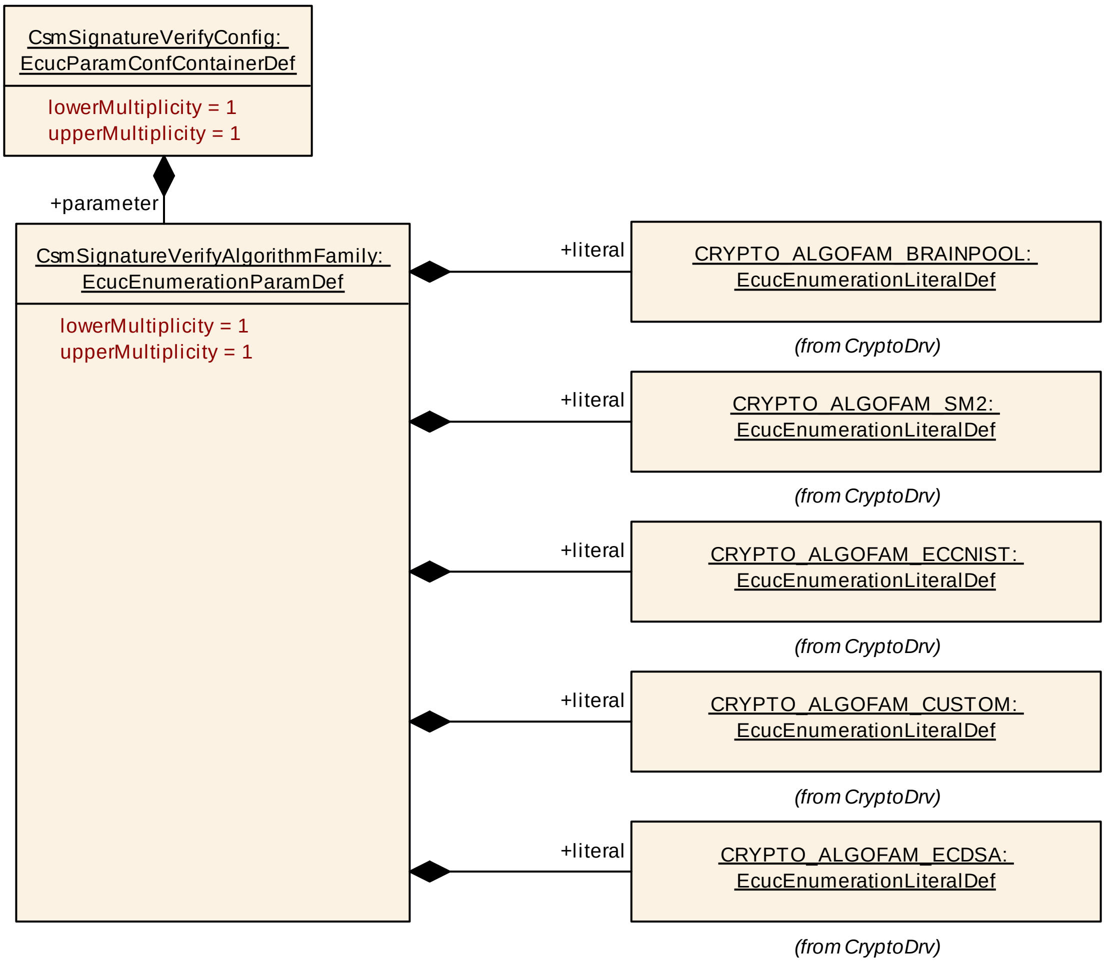


Figure 10-26 CsmSignatureVerifyAlgorithmFamily Layout

10.2.29 CsmSignatureVerify

SWS Item	[ECUC_Csm_00029]
Container Name	CsmSignatureVerify
Parent Container	CsmPrimitives
Description	Configurations of SignatureVerify primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmSignature-VerifyConfig	1	Container for configuration of a CSM signature verification interface. The container name serves as a symbolic name for the identifier of signature verification interface.

10.2.30 CsmSignatureVerifyConfig

SWS Item	[ECUC_Csm_00094]
Container Name	CsmSignatureVerifyConfig
Parent Container	CsmSignatureVerify
Description	Container for configuration of a CSM signature verification interface. The container name serves as a symbolic name for the identifier of signature verification interface.
Configuration Parameters	

SWS Item	[ECUC_Csm_00096]		
Parameter Name	CsmSignatureVerifyAlgorithmFamily		
Parent Container	CsmSignatureVerifyConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_BRAINPOOL	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_ECCNIST	--	
	CRYPTO_ALGOFAM_ECDSA	--	
	CRYPTO_ALGOFAM_ED25519	--	
	CRYPTO_ALGOFAM_RSA	--	
	CRYPTO_ALGOFAM_SM2	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00098]		
Parameter Name	CsmSignatureVerifyAlgorithmMode		
Parent Container	CsmSignatureVerifyConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_RSASSA_PKCS1_v1_5	--	
	CRYPTO_ALGOMODE_RSASSA_PSS	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00172]	
Parameter Name	CsmSignatureVerifyAlgorithmSecondaryFamily	
Parent Container	CsmSignatureVerifyConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_BLAKE_1_256	--
	CRYPTO_ALGOFAM_BLAKE_1_512	--
	CRYPTO_ALGOFAM_BLAKE_2s_256	--

	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00176]
Parameter Name	CsmSignatureVerifyCompareLength
Parent Container	CsmSignatureVerifyConfig
Description	Size of the input signature buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.

Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00175]		
Parameter Name	CsmSignatureVerifyDataMaxLength		
Parent Container	CsmSignatureVerifyConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.

SWS Item	[ECUC_Csm_00192]		
Parameter Name	CsmSignatureVerifyKeyLength		
Parent Container	CsmSignatureVerifyConfig		
Description	Size of the signature verify key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00306]		
Parameter Name	CsmSignatureVerifyAlgorithmFamilyCustomRef		
Parent Container	CsmSignatureVerifyConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local dependency: This reference shall only be present if CsmSignatureVerify AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.

SWS Item	[ECUC_Csm_00307]		
Parameter Name	CsmSignatureVerifyAlgorithmModeCustomRef		
Parent Container	CsmSignatureVerifyConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmSignatureVerify AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00308]		
Parameter Name	CsmSignatureVerifyAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmSignatureVerifyConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

	dependency: This reference shall only be present if CsmSignatureVerifySecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.

No Included Containers

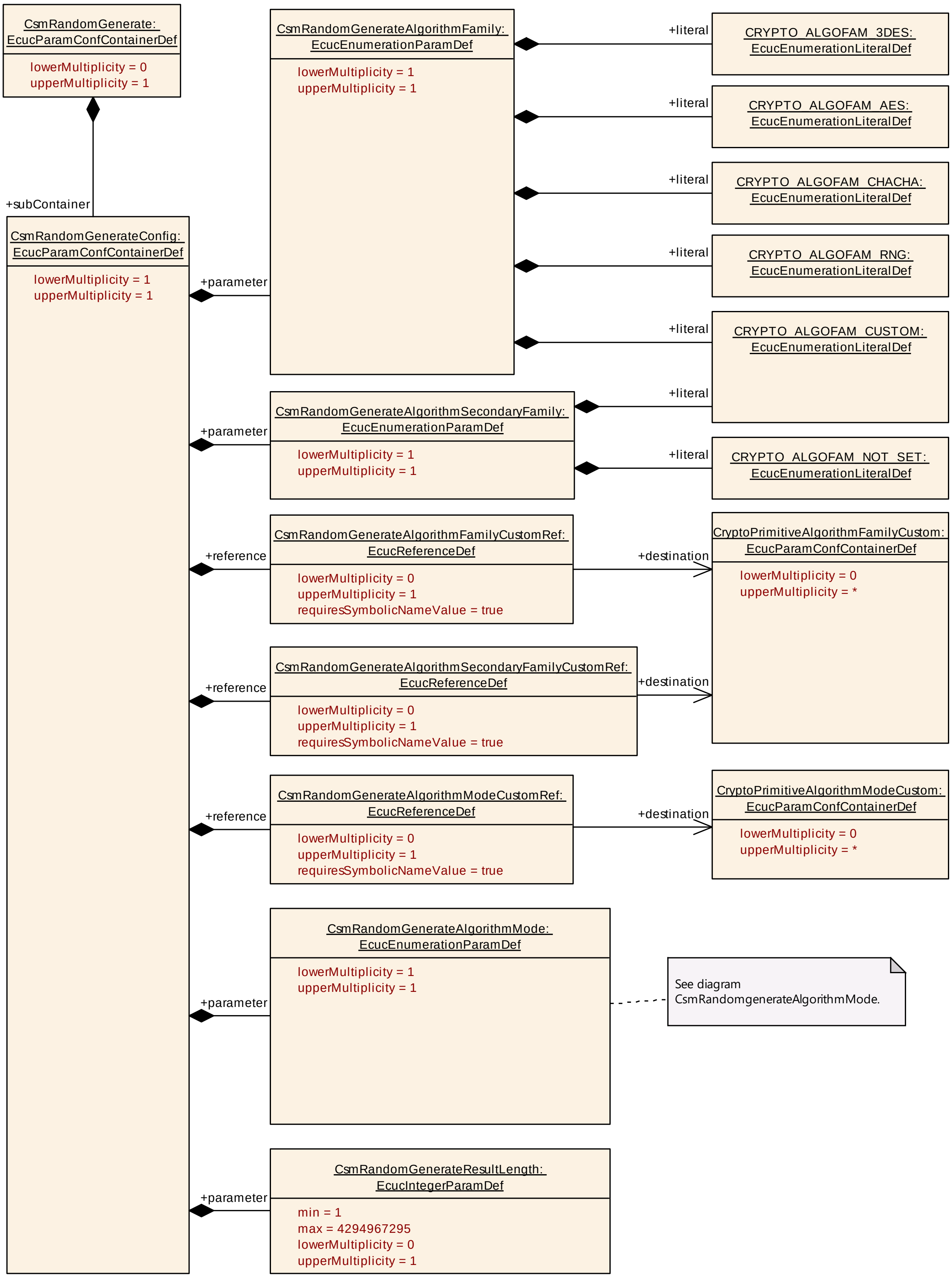

CsmRandomGenerate: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmRandomGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmRandomGenerateAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_RNG: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CsmRandomGenerateAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+reference
CsmRandomGenerateAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
CsmRandomGenerateAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
CsmRandomGenerateAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
CsmRandomGenerateAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
See diagram CsmRandomgenerateAlgorithmMode.
CsmRandomGenerateResultLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10-27 CsmRandomGenerate Layout

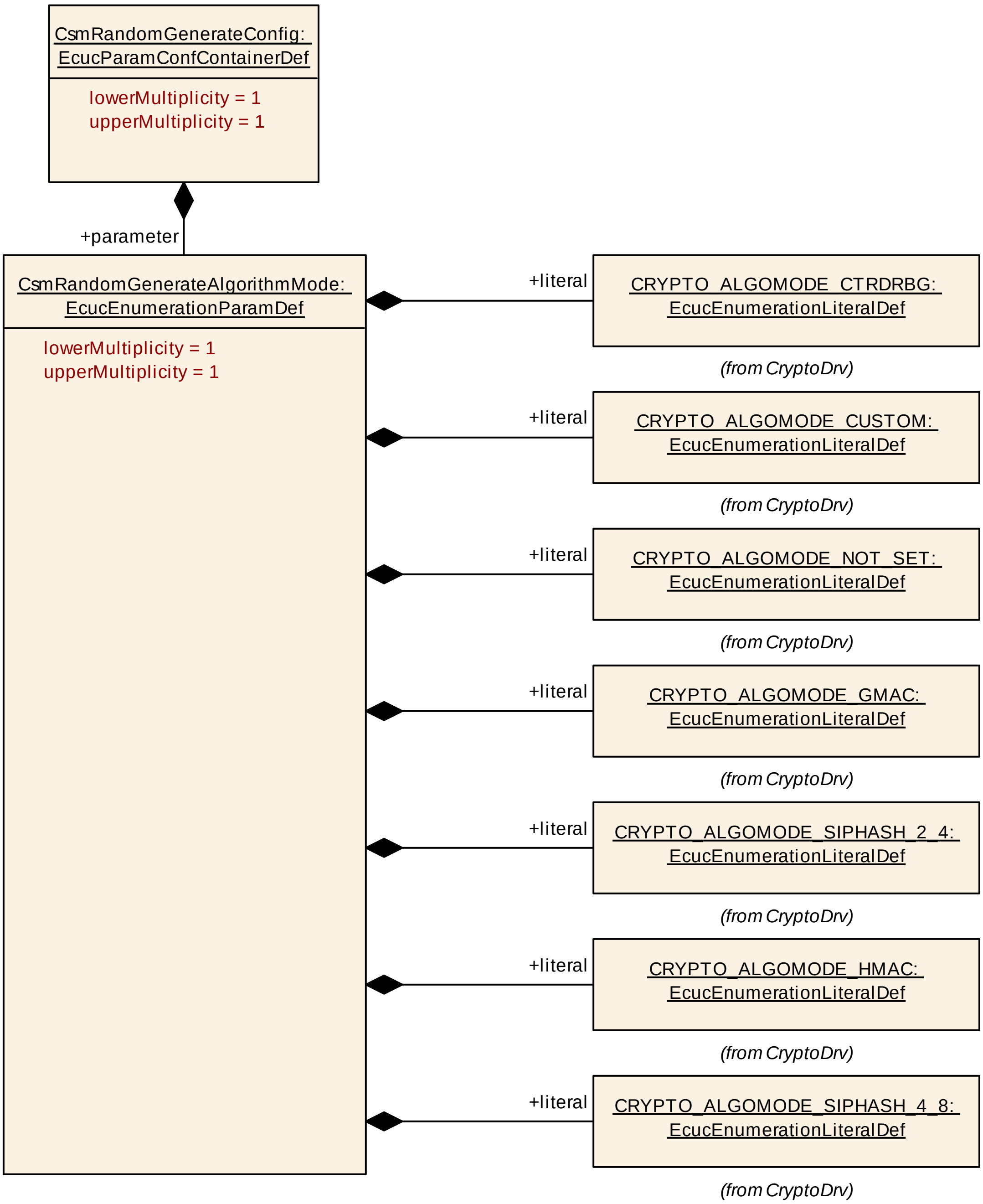

CsmRandomGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmRandomGenerateAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CTRDRBG: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_GMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_SIPHASH_2_4: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_HMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_SIPHASH_4_8: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-28 CsmRandomGenerateAlgorithmMode Layout

10.2.31 CsmRandomGenerate

SWS Item	[ECUC_Csm_00031]
Container Name	CsmRandomGenerate
Parent Container	CsmPrimitives
Description	Configurations of RandomGenerate primitives

Configuration Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmRandom-GenerateConfig	1	Container for configuration of a CSM random generator. The container name serves as a symbolic name for the identifier of a random generator configuration.

10.2.32 CsmRandomGenerateConfig

SWS Item	[ECUC_Csm_00103]
Container Name	CsmRandomGenerateConfig
Parent Container	CsmRandomGenerate
Description	Container for configuration of a CSM random generator. The container name serves as a symbolic name for the identifier of a random generator configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00105]	
Parameter Name	CsmRandomGenerateAlgorithmFamily	
Parent Container	CsmRandomGenerateConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--
	CRYPTO_ALGOFAM_BLAKE_1_256	--
	CRYPTO_ALGOFAM_BLAKE_1_512	--
	CRYPTO_ALGOFAM_BLAKE_2s_256	--
	CRYPTO_ALGOFAM_BLAKE_2s_512	--
	CRYPTO_ALGOFAM_CHACHA	--
	CRYPTO_ALGOFAM_CUSTOM	--

	CRYPTO_ALGOFAM_EEA3	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_RNG	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
	CRYPTO_ALGOFAM_SM3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00107]
Parameter Name	CsmRandomGenerateAlgorithmMode
Parent Container	CsmRandomGenerateConfig
Description	Determines the algorithm mode used for the crypto service
Multiplicity	1

Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CTRDRBG	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_GMAC	--	
	CRYPTO_ALGOMODE_HMAC	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_SIPHASH_2_4	--	
	CRYPTO_ALGOMODE_SIPHASH_4_8	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00178]		
Parameter Name	CsmRandomGenerateAlgorithmSecondaryFamily		
Parent Container	CsmRandomGenerateConfig		
Description	Determines the algorithm family used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00106]		
Parameter Name	CsmRandomGenerateResultLength		
Parent Container	CsmRandomGenerateConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter is only relevant if CsmJobInterfaceUsePort is configured as CRYPTO_USE_PORT.		

SWS Item	[ECUC_Csm_00309]		
Parameter Name	CsmRandomGenerateAlgorithmFamilyCustomRef		
Parent Container	CsmRandomGenerateConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmRandomGenerateAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00310]		
Parameter Name	CsmRandomGenerateAlgorithmModeCustomRef		
Parent Container	CsmRandomGenerateConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmRandomGenerateAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00311]		
Parameter Name	CsmRandomGenerateAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmRandomGenerateConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
	Pre-compile time	X	All Variants

Value Configuration Class	**Link time**	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmRandomGenerateSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

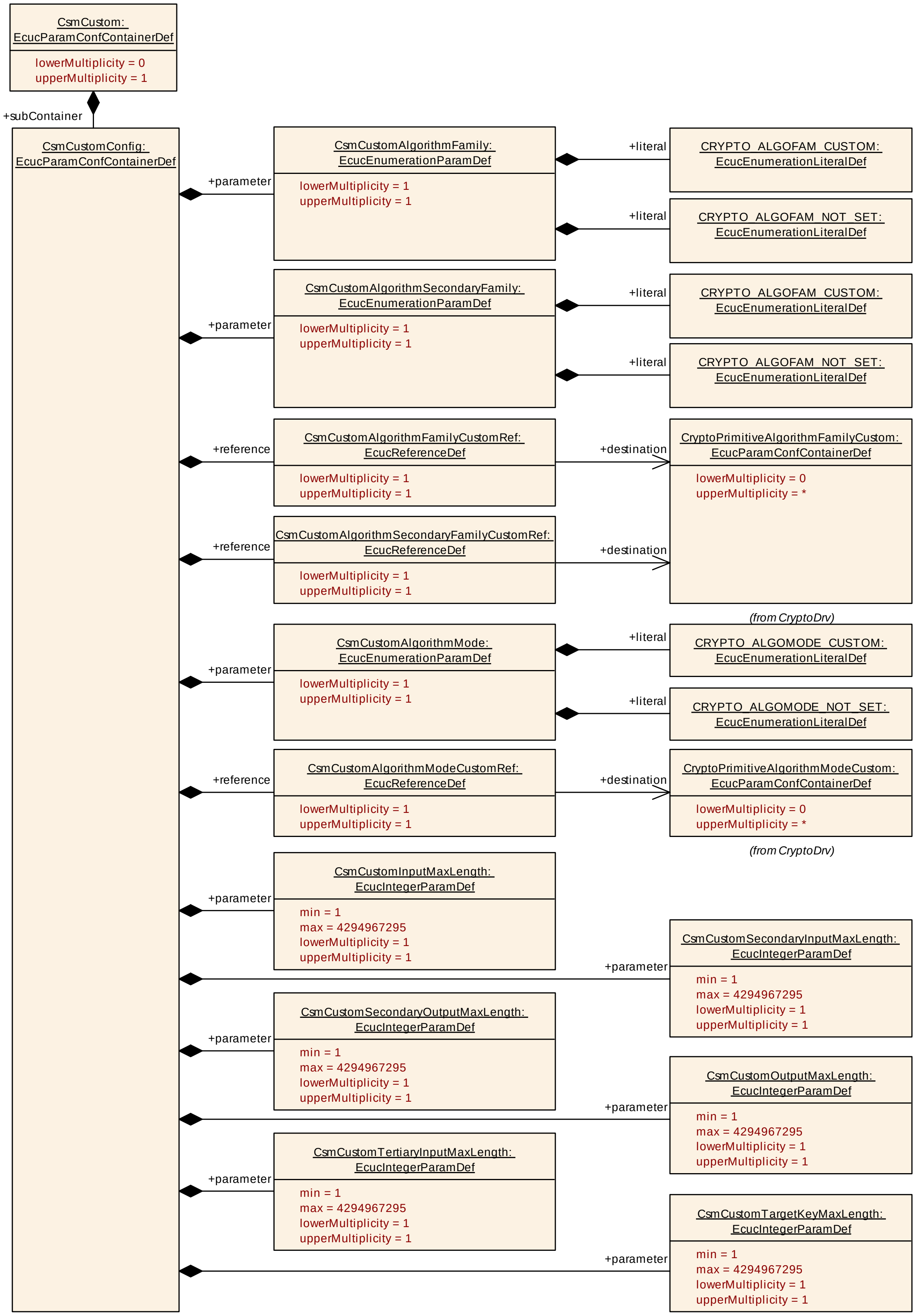

CsmCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmCustomConfig: EcucParamConfContainerDef
+parameter
CsmCustomAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+parameter
CsmCustomAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+reference
CsmCustomAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmCustomAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
(from CryptoDrv)
+parameter
CsmCustomAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
+reference
CsmCustomAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)
+parameter
CsmCustomInputMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmCustomSecondaryInputMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmCustomSecondaryOutputMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmCustomOutputMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmCustomTertiaryInputMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmCustomTargetKeyMaxLength: EcucIntegerParamDef
min = 1
max = 4294967295
lowerMultiplicity = 1
upperMultiplicity = 1

Figure 10-29 CsmCustom Layout

10.2.33 CsmCustom

SWS Item	[ECUC_Csm_00342]
Container Name	CsmCustom
Parent Container	CsmPrimitives
Description	Container for configuration of a CSM custom crypto primitive.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmCustom-Config	1	Container for custom primitive configuration configuration parameters

10.2.34 CsmCustomConfig

SWS Item	[ECUC_Csm_00343]
Container Name	CsmCustomConfig
Parent Container	CsmCustom
Description	Container for custom primitive configuration configuration parameters
Configuration Parameters	

SWS Item	[ECUC_Csm_00344]		
Parameter Name	CsmCustomAlgorithmFamily		
Parent Container	CsmCustomConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00346]		
Parameter Name	CsmCustomAlgorithmMode		
Parent Container	CsmCustomConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00348]		
Parameter Name	CsmCustomAlgorithmSecondaryFamily		
Parent Container	CsmCustomConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants

	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00350]		
Parameter Name	CsmCustomInputMaxLength		
Parent Container	CsmCustomConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00353]
Parameter Name	CsmCustomOutputMaxLength
Parent Container	CsmCustomConfig
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.
Multiplicity	1

Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00351]		
Parameter Name	CsmCustomSecondaryInputMaxLength		
Parent Container	CsmCustomConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00354]		
Parameter Name	CsmCustomSecondaryOutputMaxLength		
Parent Container	CsmCustomConfig		
Description	Size of the result data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00355]		
Parameter Name	CsmCustomTargetKeyMaxLength		
Parent Container	CsmCustomConfig		
Description	Size of the Key in bytes		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00352]		
Parameter Name	CsmCustomTertiaryInputMaxLength		
Parent Container	CsmCustomConfig		
Description	Size of the input data buffer for the synchronous RTE service interface if this primitive is referenced by a Csm job. It may also be possible that other BSW modules use the length parameter for buffer size calculation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 4294967295		
Default value	--		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00345]		
Parameter Name	CsmCustomAlgorithmFamilyCustomRef		
Parent Container	CsmCustomConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	1		
Type	Reference to CryptoPrimitiveAlgorithmFamilyCustom		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00347]		
Parameter Name	CsmCustomAlgorithmModeCustomRef		
Parent Container	CsmCustomConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	1		
Type	Reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00349]		
Parameter Name	CsmCustomAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmCustomConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	1		
Type	Reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants

	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

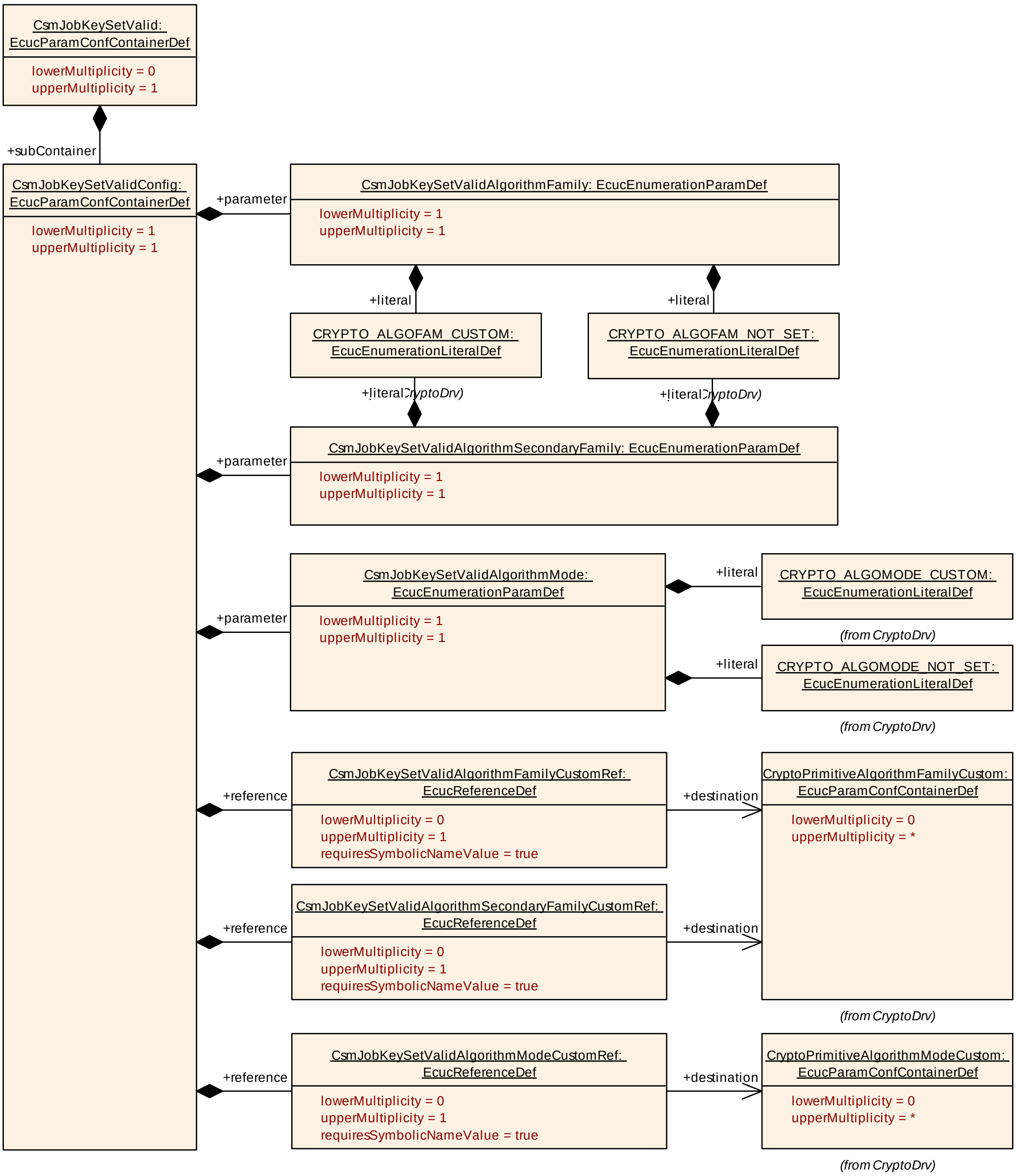

CsmJobKeySetValid: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeySetValidConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeySetValidAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+literal
(from CryptoDrv)
+literal
(from CryptoDrv)
+parameter
CsmJobKeySetValidAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeySetValidAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+reference
CsmJobKeySetValidAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmJobKeySetValidAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
(from CryptoDrv)
+reference
CsmJobKeySetValidAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)

Figure 10-30 CsmJobKeySetValid Layout

10.2.35 CsmJobKeySetValid

SWS Item	[ECUC_Csm_00196]
Container Name	CsmJobKeySetValid
Parent Container	CsmPrimitives
Description	Configurations of KeySetValid primitives

Configuration Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKeySet-ValidConfig	1	Container for configuration of a CSM key set valid operation. The container name serves as a symbolic name for the identifier of a key configuration.

10.2.36 CsmJobKeySetValidConfig

SWS Item	[ECUC_Csm_00204]
Container Name	CsmJobKeySetValidConfig
Parent Container	CsmJobKeySetValid
Description	Container for configuration of a CSM key set valid operation. The container name serves as a symbolic name for the identifier of a key configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00328]		
Parameter Name	CsmJobKeySetValidAlgorithmFamily		
Parent Container	CsmJobKeySetValidConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00329]		
Parameter Name	CsmJobKeySetValidAlgorithmMode		
Parent Container	CsmJobKeySetValidConfig		
Description	Determines the algorithm mode used for the crypto service.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00340]		
Parameter Name	CsmJobKeySetValidAlgorithmSecondaryFamily		
Parent Container	CsmJobKeySetValidConfig		
Description	Determines the secondary algorithm family used for the crypto service.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00330]		
Parameter Name	CsmJobKeySetValidAlgorithmFamilyCustomRef		
Parent Container	CsmJobKeySetValidConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetValid AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00331]		
Parameter Name	CsmJobKeySetValidAlgorithmModeCustomRef		
Parent Container	CsmJobKeySetValidConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetValid AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00332]		
Parameter Name	CsmJobKeySetValidAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmJobKeySetValidConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetValid SecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

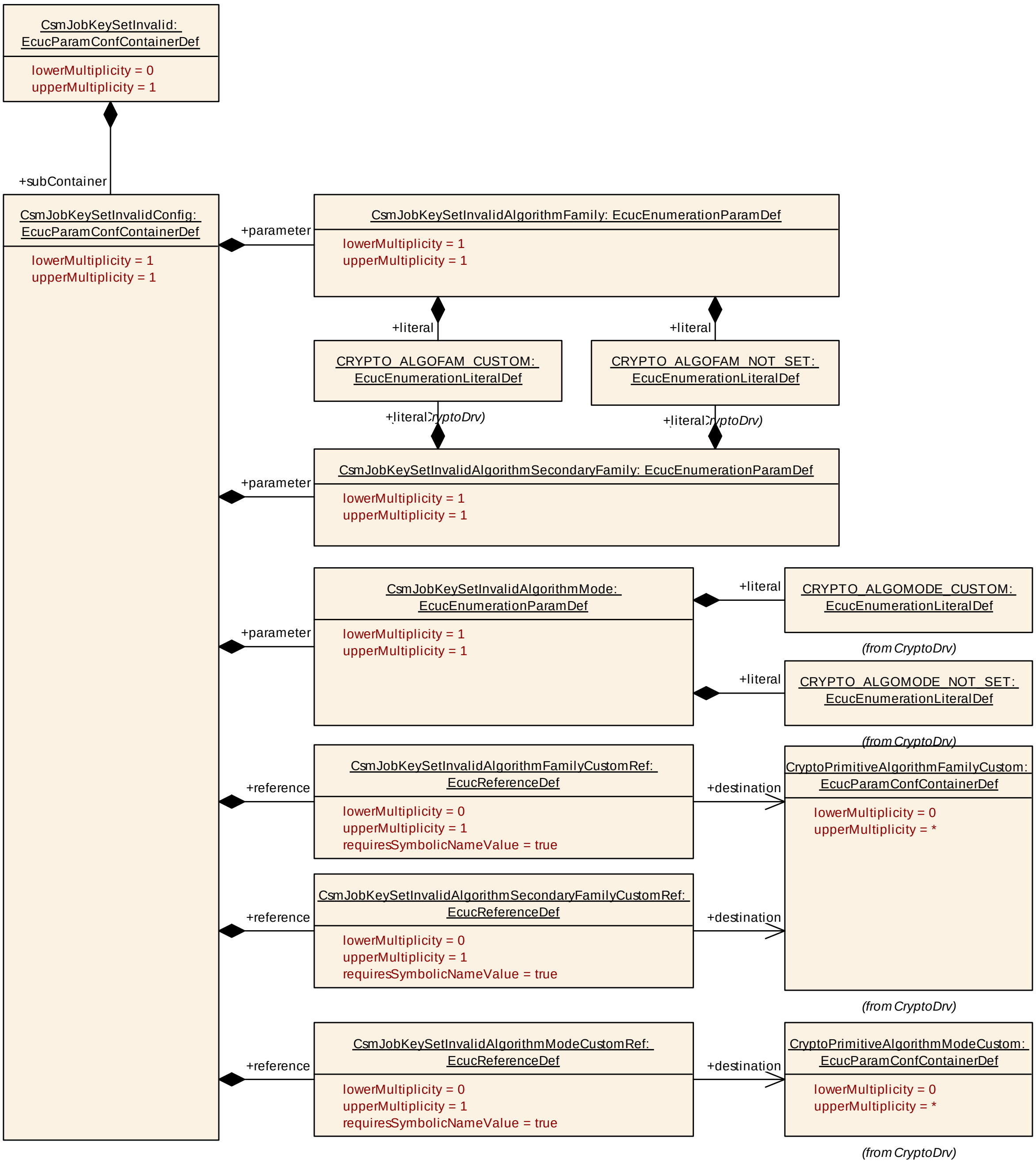

CsmJobKeySetInvalid: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeySetInvalidConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeySetInvalidAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
+literal (from CryptoDrv)
+literal (from CryptoDrv)
+parameter
CsmJobKeySetInvalidAlgorithmSecondaryFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeySetInvalidAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+reference
CsmJobKeySetInvalidAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmJobKeySetInvalidAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
(from CryptoDrv)
+reference
CsmJobKeySetInvalidAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)

Figure 10-31 CsmJobKeySetInvalid Layout

10.2.37 CsmJobKeySetInvalid

SWS Item	[ECUC_Csm_00333]
Container Name	CsmJobKeySetInvalid
Parent Container	CsmPrimitives
Description	Configurations of KeySetInvalid primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKeySetInvalidConfig	1	Container for configuration of a CSM key set invalid operation.

10.2.38 CsmJobKeySetInvalidConfig

SWS Item	[ECUC_Csm_00334]
Container Name	CsmJobKeySetInvalidConfig
Parent Container	CsmJobKeySetInvalid
Description	Container for configuration of a CSM key set invalid operation.
Configuration Parameters	

SWS Item	[ECUC_Csm_00335]		
Parameter Name	CsmJobKeySetInvalidAlgorithmFamily		
Parent Container	CsmJobKeySetInvalidConfig		
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00336]
Parameter Name	CsmJobKeySetInvalidAlgorithmMode

Parent Container	CsmJobKeySetInvalidConfig		
Description	Determines the algorithm mode used for the crypto service.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00341]		
Parameter Name	CsmJobKeySetInvalidAlgorithmSecondaryFamily		
Parent Container	CsmJobKeySetInvalidConfig		
Description	Determines the secondary algorithm family used for the crypto service.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00337]

Parameter Name	CsmJobKeySetInvalidAlgorithmFamilyCustomRef		
Parent Container	CsmJobKeySetInvalidConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetInvalidAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00339]		
Parameter Name	CsmJobKeySetInvalidAlgorithmModeCustomRef		
Parent Container	CsmJobKeySetInvalidConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetInvalidAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00338]
Parameter Name	CsmJobKeySetInvalidAlgorithmSecondaryFamilyCustomRef
Parent Container	CsmJobKeySetInvalidConfig

Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeySetInvalidAlgorithmSecondaryFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

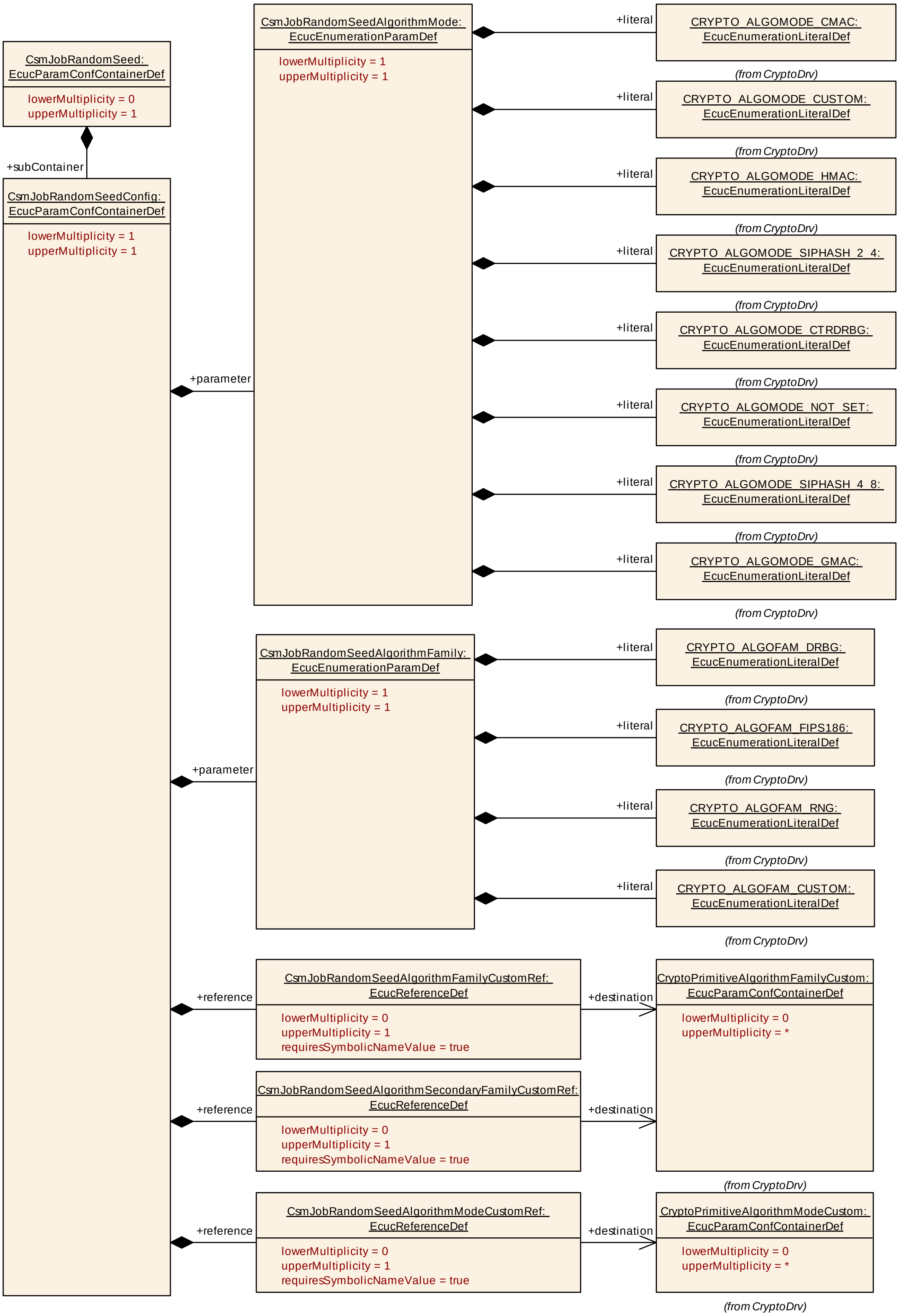

CsmJobRandomSeed: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobRandomSeedConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobRandomSeedAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_HMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_SIPHASH_2_4: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_CTRDRBG: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_SIPHASH_4_8: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOMODE_GMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+parameter
CsmJobRandomSeedAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_DRBG: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_FIPS186: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_RNG: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+reference
CsmJobRandomSeedAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)
+reference
CsmJobRandomSeedAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmJobRandomSeedAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)

Figure 10-32 CsmJobRandomSeed Layout

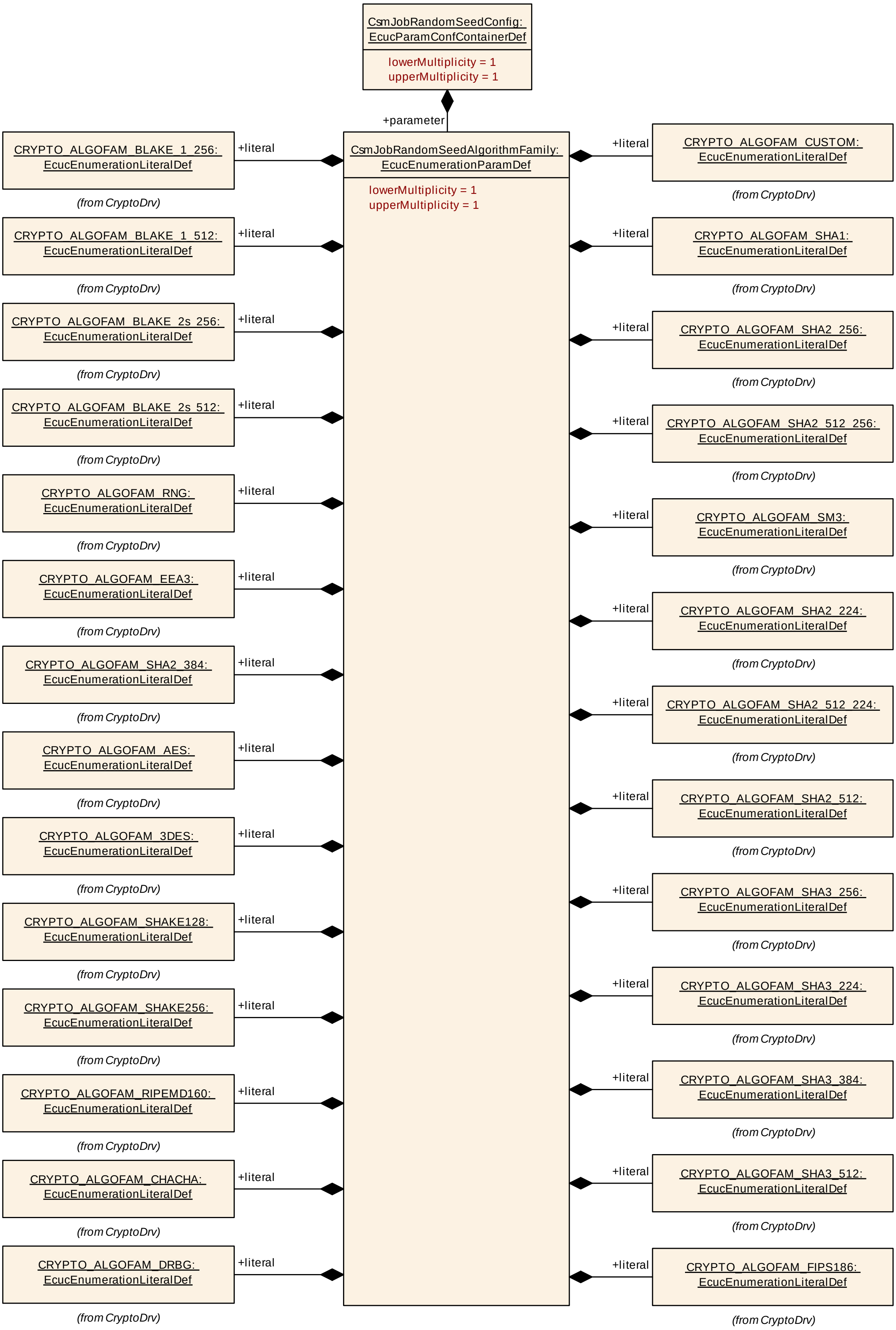

CsmJobRandomSeedConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobRandomSeedAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_BLAKE_1_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_1_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_RNG: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_EEA3: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_384: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_AES: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_3DES: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE128: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_RIPEMD160: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CHACHA: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_DRBG: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA1: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SM3: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_384: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_FIPS186: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-33 CsmJobRandomSeedAlgorithmFamily Layout

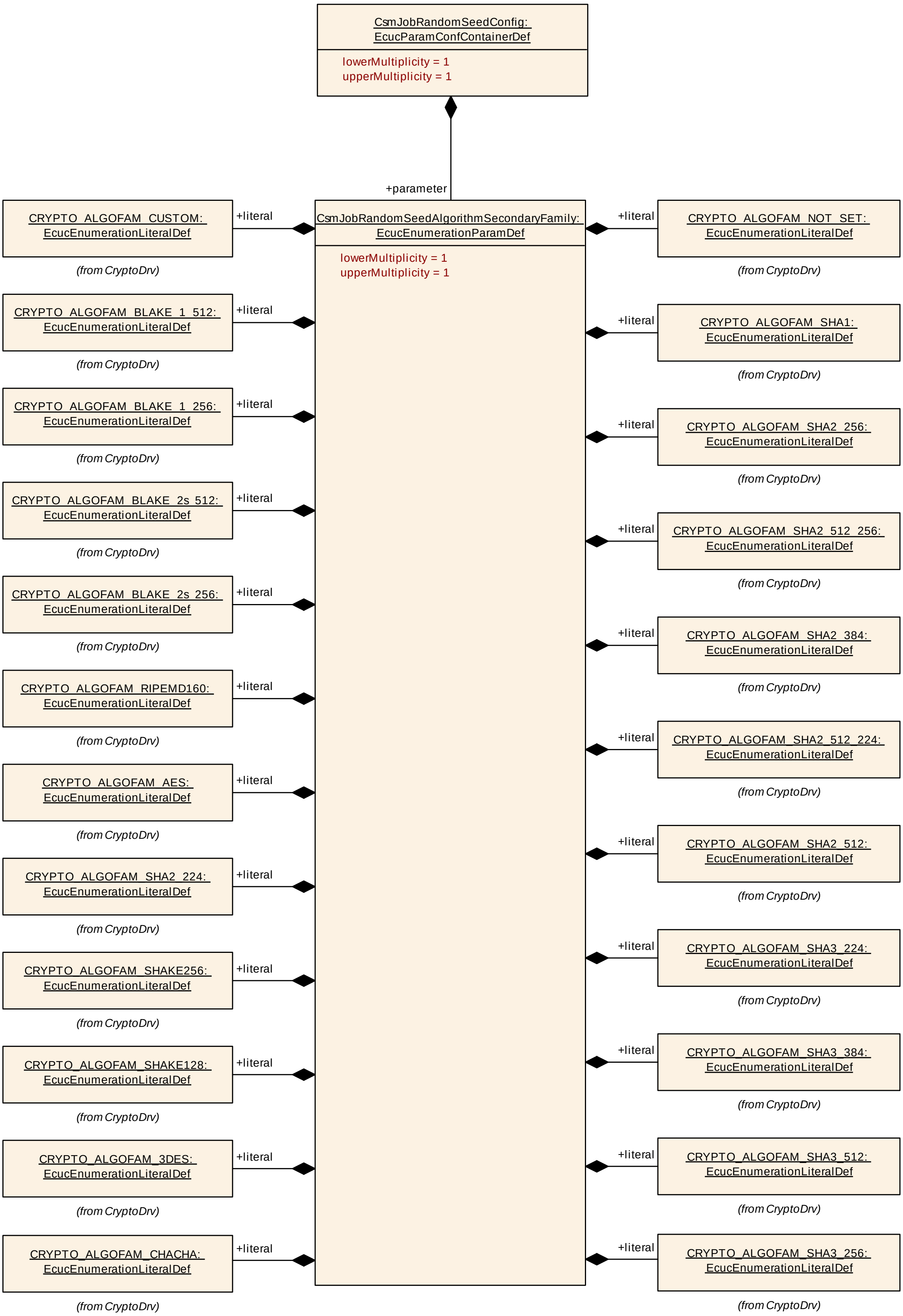


Figure 10-34 CsmJobRandomSeedAlgorithmSecondaryFamily Layout

10.2.39 CsmJobRandomSeed

SWS Item	[ECUC_Csm_00197]
Container Name	CsmJobRandomSeed
Parent Container	CsmPrimitives
Description	Configurations of RandomSeed primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJob-RandomSeed-Config	1	Container for configuration of a CSM Ramdom Seed operation. The container name serves as a symbolic name for the identifier of a random seed configuration.

10.2.40 CsmJobRandomSeedConfig

SWS Item	[ECUC_Csm_00261]
Container Name	CsmJobRandomSeedConfig
Parent Container	CsmJobRandomSeed
Description	Container for configuration of a CSM random seed operation. The container name serves as a symbolic name for the identifier of a random seed configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00206]	
Parameter Name	CsmJobRandomSeedAlgorithmFamily	
Parent Container	CsmJobRandomSeedConfig	
Description	Determines the algorithm family used for the crypto service.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--
	CRYPTO_ALGOFAM_BLAKE_1_256	--

	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_DRBG	--	
	CRYPTO_ALGOFAM_EEA3	--	
	CRYPTO_ALGOFAM_FIPS186	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_RNG	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
	CRYPTO_ALGOFAM_SM3	--	
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	[ECUC_Csm_00208]		
Parameter Name	CsmJobRandomSeedAlgorithmMode		
Parent Container	CsmJobRandomSeedConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CMAC	--	
	CRYPTO_ALGOMODE_CTRDRBG	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_GMAC	--	
	CRYPTO_ALGOMODE_HMAC	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_SIPHASH_2_4	--	
	CRYPTO_ALGOMODE_SIPHASH_4_8	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00210]	
Parameter Name	CsmJobRandomSeedAlgorithmSecondaryFamily	
Parent Container	CsmJobRandomSeedConfig	
Description	Determines the algorithm family used for the crypto service.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_3DES	--
	CRYPTO_ALGOFAM_AES	--
	CRYPTO_ALGOFAM_BLAKE_1_256	--
	CRYPTO_ALGOFAM_BLAKE_1_512	--

	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CHACHA	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00312]
Parameter Name	CsmJobRandomSeedAlgorithmFamilyCustomRef
Parent Container	CsmJobRandomSeedConfig
Description	Reference to a customer specific algorithm family custom container
Multiplicity	0..1
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobRandomSeed AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00313]		
Parameter Name	CsmJobRandomSeedAlgorithmModeCustomRef		
Parent Container	CsmJobRandomSeedConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmJobRandomSeed AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00314]		
Parameter Name	CsmJobRandomSeedAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmJobRandomSeedConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
	Pre-compile time	X	All Variants

Multiplicity Configuration Class	**Link time**	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmJobRandomSeedSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

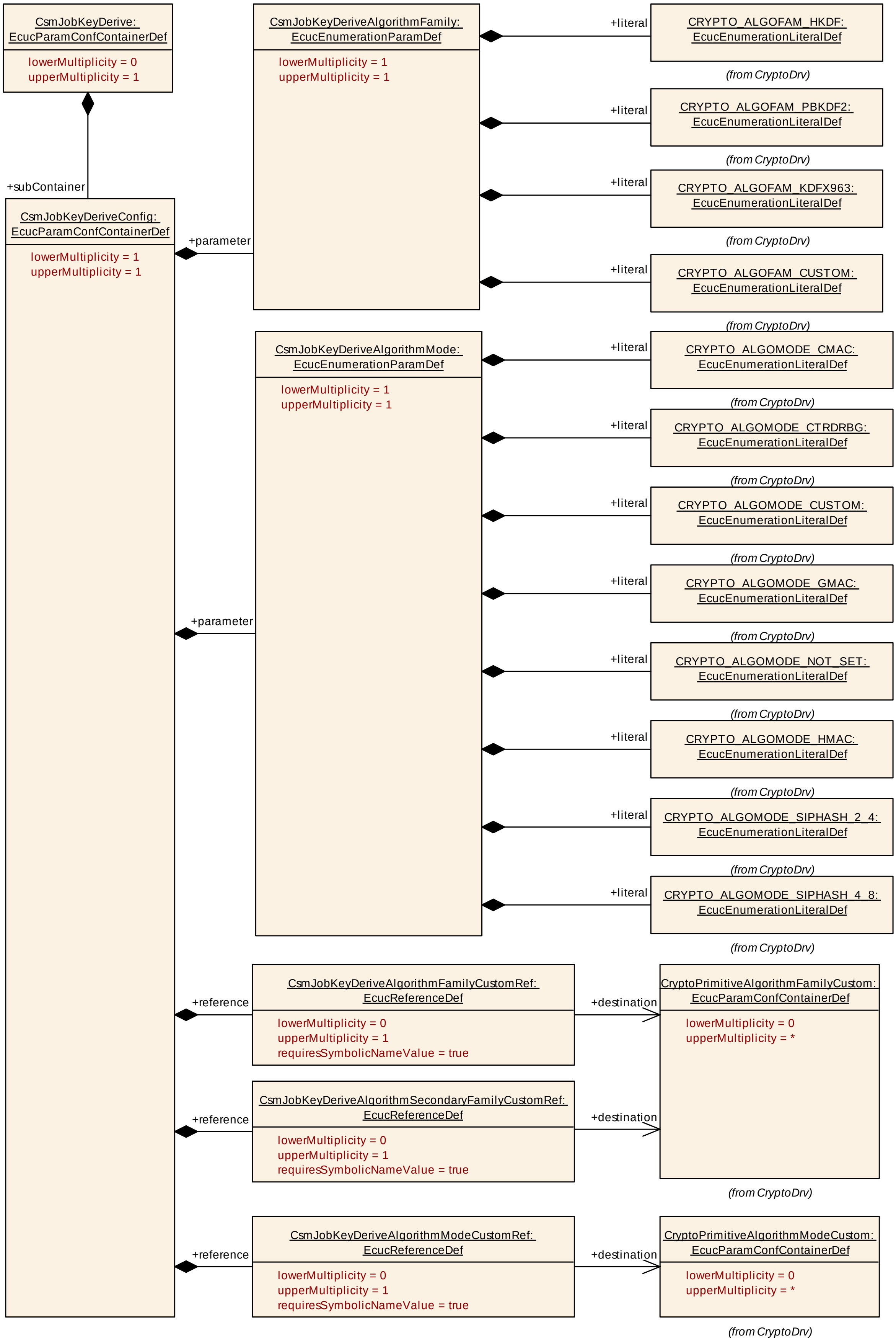

CsmJobKeyDerive: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeyDeriveConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeyDeriveAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_HKDF: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_PBKDF2: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_KDFX963: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+parameter
CsmJobKeyDeriveAlgorithmMode: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_CTRDRBG: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_GMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_HMAC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_SIPHASH_2_4: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_SIPHASH_4_8: EcucEnumerationLiteralDef
(from CryptoDrv)
+reference
CsmJobKeyDeriveAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)
+reference
CsmJobKeyDeriveAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
+reference
CsmJobKeyDeriveAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)

Figure 10-35 CsmJobKeyDerive Layout

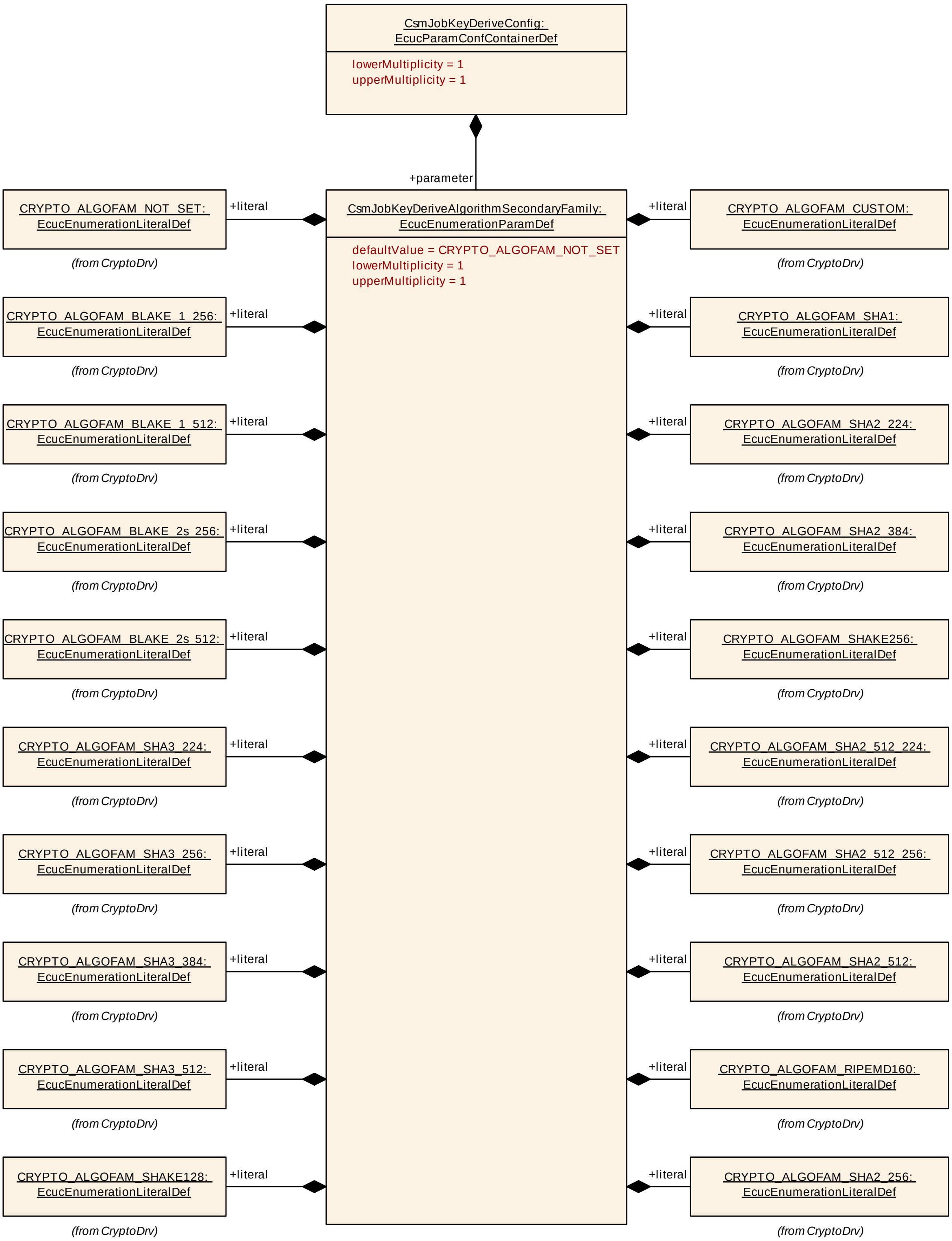

CsmJobKeyDeriveConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeyDeriveAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_1_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_1_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_BLAKE_2s_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_384: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA3_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE128: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA1: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_384: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHAKE256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_224: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512_256: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_512: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_RIPEMD160: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_SHA2_256: EcucEnumerationLiteralDef
(from CryptoDrv)

Figure 10-36 CsmJobKeyDeriveAlgorithmSecondaryFamily Layout

10.2.41 CsmJobKeyDerive

SWS Item	[ECUC_Csm_00198]
Container Name	CsmJobKeyDerive

Parent Container	CsmPrimitives
Description	Configurations of KeyDerive primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKey-DeriveConfig	1	Container for configuration of a CSM key derive operation. The container name serves as a symbolic name for the identifier of a key derive configuration.

10.2.42 CsmJobKeyDeriveConfig

SWS Item	[ECUC_Csm_00213]
Container Name	CsmJobKeyDeriveConfig
Parent Container	CsmJobKeyDerive
Description	Container for configuration of a CSM key derive operation. The container name serves as a symbolic name for the identifier of a key derive configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00215]	
Parameter Name	CsmJobKeyDeriveAlgorithmFamily	
Parent Container	CsmJobKeyDeriveConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_HKDF	--
	CRYPTO_ALGOFAM_KDFX963	--
	CRYPTO_ALGOFAM_PBKDF2	--
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00216]		
Parameter Name	CsmJobKeyDeriveAlgorithmMode		
Parent Container	CsmJobKeyDeriveConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CMAC	--	
	CRYPTO_ALGOMODE_CTRDRBG	--	
	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_GMAC	--	
	CRYPTO_ALGOMODE_HMAC	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
	CRYPTO_ALGOMODE_SIPHASH_2_4	--	
	CRYPTO_ALGOMODE_SIPHASH_4_8	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00218]
Parameter Name	CsmJobKeyDeriveAlgorithmSecondaryFamily
Parent Container	CsmJobKeyDeriveConfig
Description	Determines the algorithm family used for the crypto service.
Multiplicity	1
Type	EcucEnumerationParamDef

Range	CRYPTO_ALGOFAM_BLAKE_1_256	--	
	CRYPTO_ALGOFAM_BLAKE_1_512	--	
	CRYPTO_ALGOFAM_BLAKE_2s_256	--	
	CRYPTO_ALGOFAM_BLAKE_2s_512	--	
	CRYPTO_ALGOFAM_CUSTOM	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_RIPEMD160	--	
	CRYPTO_ALGOFAM_SHA1	--	
	CRYPTO_ALGOFAM_SHA2_224	--	
	CRYPTO_ALGOFAM_SHA2_256	--	
	CRYPTO_ALGOFAM_SHA2_384	--	
	CRYPTO_ALGOFAM_SHA2_512	--	
	CRYPTO_ALGOFAM_SHA2_512_224	--	
	CRYPTO_ALGOFAM_SHA2_512_256	--	
	CRYPTO_ALGOFAM_SHA3_224	--	
	CRYPTO_ALGOFAM_SHA3_256	--	
	CRYPTO_ALGOFAM_SHA3_384	--	
	CRYPTO_ALGOFAM_SHA3_512	--	
	CRYPTO_ALGOFAM_SHAKE128	--	
	CRYPTO_ALGOFAM_SHAKE256	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00315]
Parameter Name	CsmJobKeyDeriveAlgorithmFamilyCustomRef
Parent Container	CsmJobKeyDeriveConfig
Description	Reference to a customer specific algorithm family custom container

Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyDeriveAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00316]		
Parameter Name	CsmJobKeyDeriveAlgorithmModeCustomRef		
Parent Container	CsmJobKeyDeriveConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	dependency: This reference shall only be present if CsmJobKeyDeriveAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00317]
Parameter Name	CsmJobKeyDeriveAlgorithmSecondaryFamilyCustomRef
Parent Container	CsmJobKeyDeriveConfig
Description	Reference to a customer specific algorithm family container in the Crypto Driver
Multiplicity	0..1
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyDerive SecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

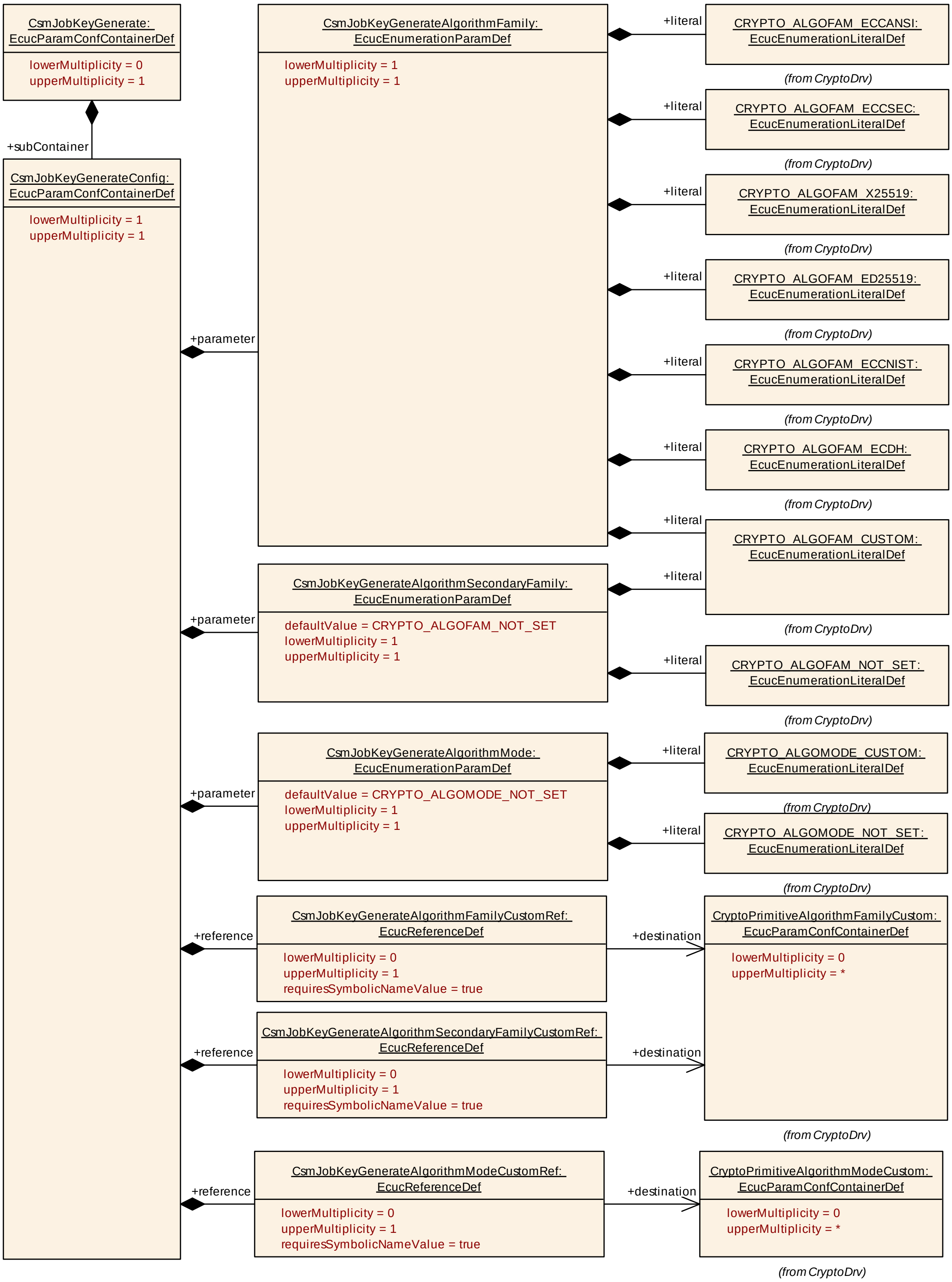

CsmJobKeyGenerate: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeyGenerateConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeyGenerateAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_ECCANSI: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ECCSEC: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_X25519: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ED25519: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ECCNIST: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_ECDH: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
+literal
(from CryptoDrv)
+parameter
CsmJobKeyGenerateAlgorithmSecondaryFamily: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOFAM_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+parameter
CsmJobKeyGenerateAlgorithmMode: EcucEnumerationParamDef
defaultValue = CRYPTO_ALGOMODE_NOT_SET
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
(from CryptoDrv)
+literal
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
(from CryptoDrv)
+reference
CsmJobKeyGenerateAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
+reference
CsmJobKeyGenerateAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
(from CryptoDrv)
+reference
CsmJobKeyGenerateAlgorithmModeCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from CryptoDrv)

Figure 10-37 CsmJobKeyGenerate Layout

10.2.43 CsmJobKeyGenerate

SWS Item	[ECUC_Csm_00199]

Container Name	CsmJobKeyGenerate
Parent Container	CsmPrimitives
Description	Configurations of KeyGenerate primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKey-GenerateConfig	1	Container for configuration of a CSM key generate operation. The container name serves as a symbolic name for the identifier of a key generate configuration.

10.2.44 CsmJobKeyGenerateConfig

SWS Item	[ECUC_Csm_00220]
Container Name	CsmJobKeyGenerateConfig
Parent Container	CsmJobKeyGenerate
Description	Container for configuration of a CSM key generate operation. The container name serves as a symbolic name for the identifier of a key generate configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00222]	
Parameter Name	CsmJobKeyGenerateAlgorithmFamily	
Parent Container	CsmJobKeyGenerateConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_ECCANSI	--
	CRYPTO_ALGOFAM_ECCNIST	--
	CRYPTO_ALGOFAM_ECCSEC	--

	CRYPTO_ALGOFAM_ECDH	--	
	CRYPTO_ALGOFAM_ED25519	--	
	CRYPTO_ALGOFAM_X25519	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00223]		
Parameter Name	CsmJobKeyGenerateAlgorithmMode		
Parent Container	CsmJobKeyGenerateConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Default value	CRYPTO_ALGOMODE_NOT_SET		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00225]	
Parameter Name	CsmJobKeyGenerateAlgorithmSecondaryFamily	
Parent Container	CsmJobKeyGenerateConfig	
Description	Determines the algorithm family used for the crypto service.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--

	CRYPTO_ALGOFAM_NOT_SET	--	
Default value	CRYPTO_ALGOFAM_NOT_SET		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00318]		
Parameter Name	CsmJobKeyGenerateAlgorithmFamilyCustomRef		
Parent Container	CsmJobKeyGenerateConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyGenerate AlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00319]		
Parameter Name	CsmJobKeyGenerateAlgorithmModeCustomRef		
Parent Container	CsmJobKeyGenerateConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyGenerate AlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00320]		
Parameter Name	CsmJobKeyGenerateAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmJobKeyGenerateConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyGenerate SecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

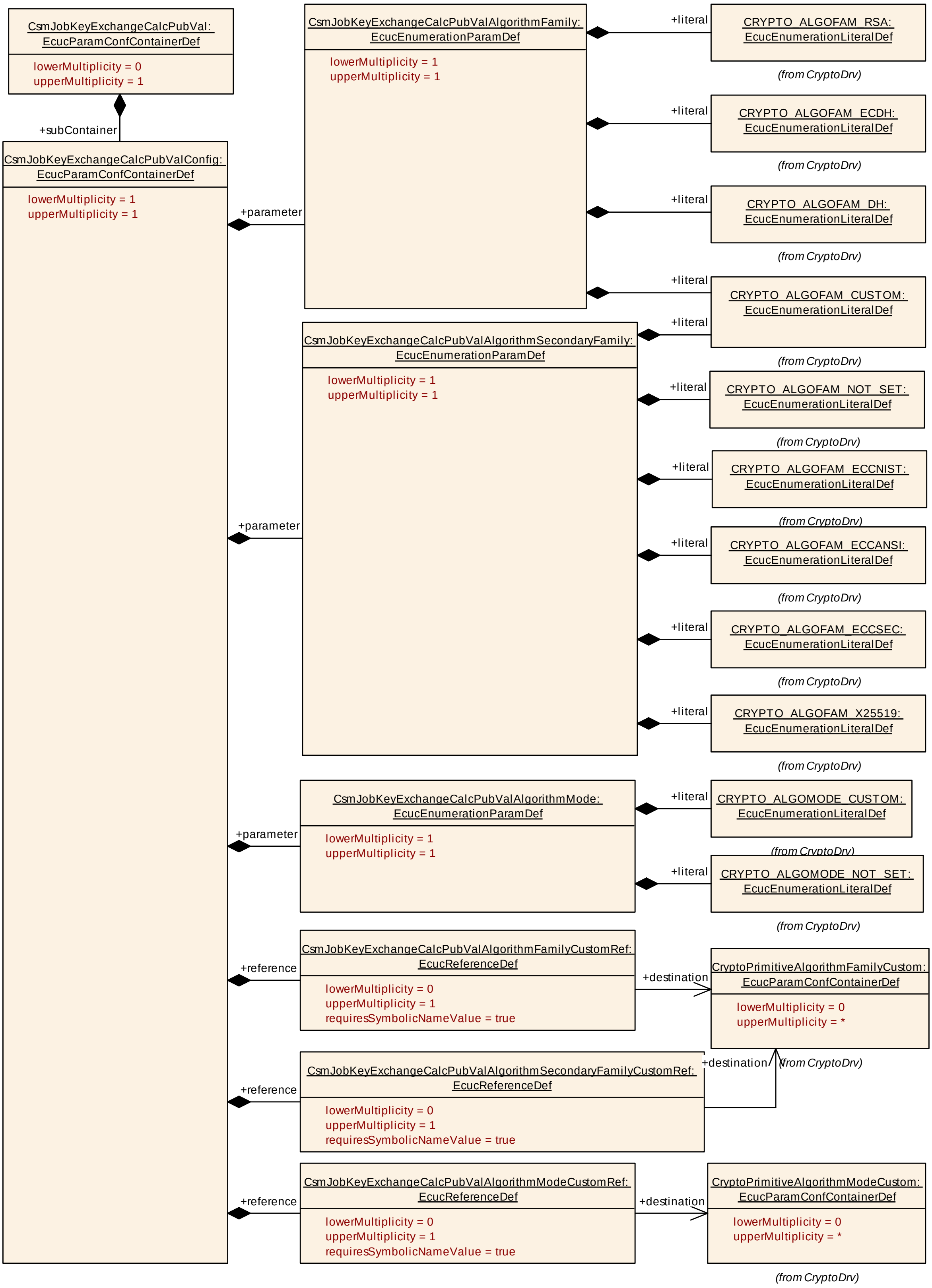

CsmJobKeyExchangeCalcPubVal: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeyExchangeCalcPubValConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeyExchangeCalcPubValAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_RSA: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_ECDH: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_DH: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CsmJobKeyExchangeCalcPubValAlgorithmSecondaryFamily: EcucEnumerationParamDef
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCNIST: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCANSI: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCSEC: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_X25519: EcucEnumerationLiteralDef
CsmJobKeyExchangeCalcPubValAlgorithmMode: EcucEnumerationParamDef
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
+reference
CsmJobKeyExchangeCalcPubValAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
CsmJobKeyExchangeCalcPubValAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
CsmJobKeyExchangeCalcPubValAlgorithmModeCustomRef: EcucReferenceDef
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef

Figure 10-38 CsmJobKeyExchangeCalcPubVal Layout

10.2.45 CsmJobKeyExchangeCalcPubVal

SWS Item	[ECUC_Csm_00200]
Container Name	CsmJobKeyExchangeCalcPubVal
Parent Container	CsmPrimitives
Description	Configurations of KeyExchangeCalcPubVal primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKeyExchangeCalcPubValConfig	1	Container for configuration of a CSM JobKeyExchangeCalcPubVal. The container name serves as a symbolic name for the identifier of a key configuration.

10.2.46 CsmJobKeyExchangeCalcPubValConfig

SWS Item	[ECUC_Csm_00226]
Container Name	CsmJobKeyExchangeCalcPubValConfig
Parent Container	CsmJobKeyExchangeCalcPubVal
Description	Container for configuration of a CSM JobKeyExchangeCalcPubVal. The container name serves as a symbolic name for the identifier of a key configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00227]	
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmFamily	
Parent Container	CsmJobKeyExchangeCalcPubValConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_DH	--
	CRYPTO_ALGOFAM_ECDH	--

	CRYPTO_ALGOFAM_RSA	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00229]		
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmMode		
Parent Container	CsmJobKeyExchangeCalcPubValConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00231]	
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmSecondaryFamily	
Parent Container	CsmJobKeyExchangeCalcPubValConfig	
Description	Determines the algorithm family used for the crypto service.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_ECCANSI	--
	CRYPTO_ALGOFAM_ECCNIST	--
	CRYPTO_ALGOFAM_ECCSEC	--

	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_X25519	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00321]		
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmFamilyCustomRef		
Parent Container	CsmJobKeyExchangeCalcPubValConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyExchangeCalcPubValAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00322]		
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmModeCustomRef		
Parent Container	CsmJobKeyExchangeCalcPubValConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This parameter shall only be present if CsmJobKeyExchangeCalcPubValAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00323]		
Parameter Name	CsmJobKeyExchangeCalcPubValAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmJobKeyExchangeCalcPubValConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyExchangeCalcPubValSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

No Included Containers

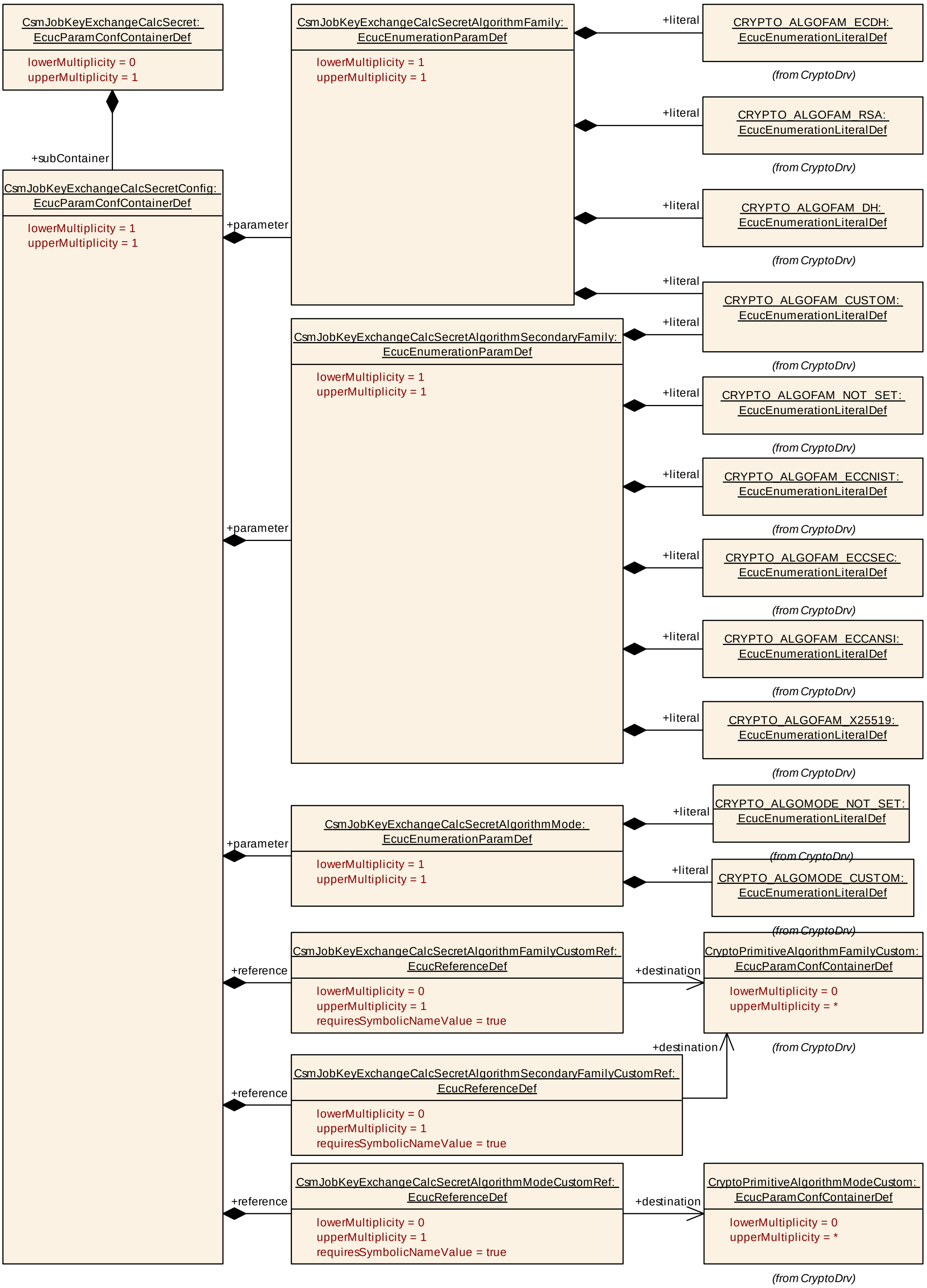

CsmJobKeyExchangeCalcSecret: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
CsmJobKeyExchangeCalcSecretConfig: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
CsmJobKeyExchangeCalcSecretAlgorithmFamily: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRYPTO_ALGOFAM_ECDH: EcucEnumerationLiteralDef
(from CryptoDrv)
CRYPTO_ALGOFAM_RSA: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_DH: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_CUSTOM: EcucEnumerationLiteralDef
CsmJobKeyExchangeCalcSecretAlgorithmSecondaryFamily: EcucEnumerationParamDef
CRYPTO_ALGOFAM_NOT_SET: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCNIST: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCSEC: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_ECCANSI: EcucEnumerationLiteralDef
CRYPTO_ALGOFAM_X25519: EcucEnumerationLiteralDef
CsmJobKeyExchangeCalcSecretAlgorithmMode: EcucEnumerationParamDef
CRYPTO_ALGOMODE_NOT_SET: EcucEnumerationLiteralDef
CRYPTO_ALGOMODE_CUSTOM: EcucEnumerationLiteralDef
+reference
CsmJobKeyExchangeCalcSecretAlgorithmFamilyCustomRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CryptoPrimitiveAlgorithmFamilyCustom: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
CsmJobKeyExchangeCalcSecretAlgorithmSecondaryFamilyCustomRef: EcucReferenceDef
CsmJobKeyExchangeCalcSecretAlgorithmModeCustomRef: EcucReferenceDef
CryptoPrimitiveAlgorithmModeCustom: EcucParamConfContainerDef

Figure 10-39 CsmJobKeyExchangeCalcSecret Layout

10.2.47 CsmJobKeyExchangeCalcSecret

SWS Item	[ECUC_Csm_00201]
Container Name	CsmJobKeyExchangeCalcSecret
Parent Container	CsmPrimitives
Description	Configurations of KeyExchangeCalcSecret primitives
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmJobKeyExchangeCalcSecretConfig	1	Container for configuration of a CSM JobKeyExchangeCalc Secret. The container name serves as a symbolic name for the identifier of a JobKeyExchangeCalcSecret configuration.

10.2.48 CsmJobKeyExchangeCalcSecretConfig

SWS Item	[ECUC_Csm_00234]
Container Name	CsmJobKeyExchangeCalcSecretConfig
Parent Container	CsmJobKeyExchangeCalcSecret
Description	Container for configuration of a CSM JobKeyExchangeCalcSecret. The container name serves as a symbolic name for the identifier of a JobKeyExchangeCalcSecret configuration.
Configuration Parameters	

SWS Item	[ECUC_Csm_00235]	
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmFamily	
Parent Container	CsmJobKeyExchangeCalcSecretConfig	
Description	Determines the algorithm family used for the crypto service. This parameter defines the most significant part of the algorithm.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_DH	--

	CRYPTO_ALGOFAM_ECDH	--	
	CRYPTO_ALGOFAM_RSA	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00237]		
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmMode		
Parent Container	CsmJobKeyExchangeCalcSecretConfig		
Description	Determines the algorithm mode used for the crypto service		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRYPTO_ALGOMODE_CUSTOM	--	
	CRYPTO_ALGOMODE_NOT_SET	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00239]	
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmSecondaryFamily	
Parent Container	CsmJobKeyExchangeCalcSecretConfig	
Description	Determines the algorithm family used for the crypto service.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRYPTO_ALGOFAM_CUSTOM	--
	CRYPTO_ALGOFAM_ECCANSI	--
	CRYPTO_ALGOFAM_ECCNIST	--

	CRYPTO_ALGOFAM_ECCSEC	--	
	CRYPTO_ALGOFAM_NOT_SET	--	
	CRYPTO_ALGOFAM_X25519	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	[ECUC_Csm_00324]		
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmFamilyCustomRef		
Parent Container	CsmJobKeyExchangeCalcSecretConfig		
Description	Reference to a customer specific algorithm family custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyExchange CalcSecretAlgorithmFamily is set to CRYPTO_ALGOFAM_CUSTOM.		

SWS Item	[ECUC_Csm_00325]		
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmModeCustomRef		
Parent Container	CsmJobKeyExchangeCalcSecretConfig		
Description	Reference to a customer specific algorithm mode custom container		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmModeCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	

	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyExchange CalcSecretAlgorithmMode is set to CRYPTO_ALGOMODE_CUSTOM.		

SWS Item	[ECUC_Csm_00326]		
Parameter Name	CsmJobKeyExchangeCalcSecretAlgorithmSecondaryFamilyCustomRef		
Parent Container	CsmJobKeyExchangeCalcSecretConfig		
Description	Reference to a customer specific algorithm family container in the Crypto Driver		
Multiplicity	0..1		
Type	Symbolic name reference to CryptoPrimitiveAlgorithmFamilyCustom		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local dependency: This reference shall only be present if CsmJobKeyExchange CalcSecretSecondaryAlgorithmFamily is set to CRYPTO_ALGOFAM_ CUSTOM.		

No Included Containers

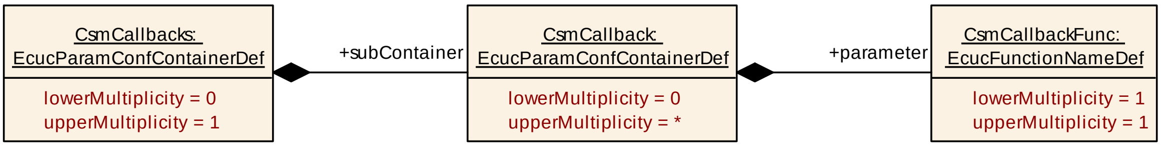


Figure 10-40 CsmCallbacks Layout

10.2.49 CsmCallbacks

SWS Item	[ECUC_Csm_00008]
Container Name	CsmCallbacks
Parent Container	Csm
Description	Container for callback function configurations
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
CsmCallback	0..*	Container for configuration of a callback function

10.2.50 CsmCallback

SWS Item	[ECUC_Csm_00109]		
Container Name	CsmCallback		
Parent Container	CsmCallbacks		
Description	Container for configuration of a callback function		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	[ECUC_Csm_00110]
Parameter Name	CsmCallbackFunc
Parent Container	CsmCallback
Description	Callback function to be called if an asynchronous operation has finished. The corresponding job has to be configured to be processed asynchronously.
Multiplicity	1
Type	EcucFunctionNameDef
Default value	--
Regular Expression	--

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.3 Published Information

For details refer to the chapter 10.3 “Published Information” in *SWS_BSWGeneral.*