VOLKSWAGEN

AKTIENGESELLSCHAFT

Group Basic Software Requirements

Basic requirements that the Volkswagen Group demands on vehicle-based and vehicle-related software and its development processes.

Development, General Project-Independent Performance Specification: LAH.893.909

First issue	06.09.2002
Date of revision	02.05.2024
Version	4.3

Contents

1 Preamble

1.1 Purpose

[I: KGAS_4110]

KGAS defines the framework specified by Volkswagen AG to ensure that the development of software for the deployment in the vehicle and vehicle environment is in accordance with the state of the art.

[I: KGAS_4111]

The Automotive SPICE process model serves as the fundamental basis for state-of-the-art software development (see KGAS_3887).

[I: KGAS_4112]

KGAS specifies and completes the Automotive SPICE process model by means of process requirements (see KGAS_3887).

[I: KGAS_4113]

KGAS addresses all project members of a software development project.

[I: KGAS_4114]

KGAS thus creates the basis for achieving the software quality in the product through the achievement of stable processes (see KGAS_3043).

[I: KGAS_3090]

The methods, basis of the evaluation and consequences of the quality assurance activities of the Volkswagen Group at the Contractors are described in the "Formel Q Capability Software" (KGAS_2834).

[I: KGAS_3633]

The terms "Contracting authority" and "Contractor" used in this document are equivalent to the terms "Customer" and "Supplier" used in the Formel Q Capability Software (KGAS_2834).

1.2 Document Owners

[I: KGAS_2085]

Volkswagen AG
Group Quality
Quality Software
Letterbox 81250
38436 Wolfsburg
Germany

Contact: software.qualitaet.vwag.r.wob@volkswagen.de

Audi AG
Quality Control Software
85045 Ingolstadt
Germany

Contact: software-quality.gq@audi.de

Porsche AG
Quality Software
Porscheplatz 1
70435 Stuttgart
Germany

Contact: software.quality@porsche.de

CARIAD SE
Berliner Ring 2,
Letterbox 1080/2
38440 Wolfsburg
Germany

Contact: quality@cariad.technology

1.3 Cybersecurity Incident Management

[R: KGAS_3890]

To communicate security incidents use the incident team of the brand or region which is responsible for the development of the affected product in the Volkswagen group. In case of unclear responsibility contact the incident team of Volkswagen Passenger Cars (KGAS_3876).

[I: KGAS_3876]

Volkswagen Passenger Cars and Volkswagen Commercial Vehicles (light commercial vehicles)

Contact: CSI-wob@volkswagen.de

[I: KGAS_3891]

Porsche AG

Contact: csi@porsche.de

[I: KGAS_3892]

MAN SE

Contact: carsecurity@man.eu

[I: KGAS_3926]

Audi AG

Contact: vulnerability@audi.de

[I: KGAS_3958]

Contact for products in development: The contact person is the respective project manager of the respective technical development.

[I: KGAS_3993]

Škoda Auto a.s.

Contact: teamcsi@skoda-auto.cz

[I: KGAS_4017]

SEAT S.A.

Contact: csi@seat.es

[I : KGAS_4117]

Bentley

Contact: csi@bentley.co.uk

[I : KGAS_4118]

Lamborghini

Contact: csi@lamborghini.com

[I : KGAS_4119]

Cariad SE

Contact for Cybersecurity Vulnerabilities: vuln@cariad.technology

Contakt for Cybersecurity Incidents: sirt@cariad.technology

[I: KGAS_3959]

Region China

Contact: csi-cn@volkswagen.com.cn

2 Scope

[A: KGAS_4120]

The KGAS applies to development processes and the associated development-accompanying processes for software and software-determined systems (e.g. ECU with software) that contribute to the realization of a function in the vehicle.

[R: KGAS_3028]

The requirements in this document are valid for the whole Volkswagen Group and its Contractors.

[A: KGAS_4121]

KGAS must be implemented in the development of software and software-defined systems (e.g. ECUs with software) from the beginning.

3 Rights of the Contracting authority and Obligations of the Contractor

[R: KGAS_4058]

Adherence to requirements must be demonstrable.

[R: KGAS_4059]

Adherance to framework conditions must be ensured.

[I: KGAS_4060]

Information [I] is used for additional understanding or as a hint of a possible implementation of the requirement.

[A: KGAS_3885]

All development artifacts must be either in English or German language.

[A: KGAS_1806]

On request from the Contracting authority, the Contractor must provide evidence of the compliance with the KGAS.

[A: KGAS_27]

The Contractor must obligate all its Sub-contractors to fulfil the KGAS and must ensure its execution.

[A: KGAS_2933]

If the Contractor or its Sub-contractors cannot fulfil the KGAS completely, the Contractor must seek written approval of the deviations from the Quality Assurance of the Contracting authority before the start of the project. The agreed and approved changes are to be sent to Group Quality (contact please refer KGAS_2085).

[R: KGAS_51]

The Contractor must allow the Contracting authority to verify the fulfillment of the KGAS with source code analysis, tool-based analysis and other appropriate methods.

[R: KGAS_4000]

The Contracting authority may have its analyses (KGAS_51) resulting from the KGAS performed in fully or in partly carried out by third parties.

[R: KGAS_2949]

The Contractor must support the analysis (KGAS_51) by providing the source code, corresponding to ECU configurations in the premises and presence of the Contractor.

[R: KGAS_3546]

If a requirement of the KGAS is inconsistent with a requirement from another applicable document, the Contractor must initiate a specific agreement between the Contracting authority and the Contractor.

4 Terminology

[I: KGAS_1984]

This chapter defines, how relevant technical terms in the KGAS are to be interpreted.

4.1 Feature

[I: KGAS_3665]

A feature is a scope of functionalities which is defined by the Contracting authority and which is represented by a subset of the requirements.

4.2 System

[I: KGAS_2877]

The system is the whole part to be delivered by the Contractor.

[I: KGAS_2879]

The system consists of system elements.

4.3 System Element

[I: KGAS_3604]

The definition of system element is in accordance with ASPICE PAM 4.0 .

4.4 Software

[I: KGAS_2876]

Software is the whole software enclosed in the deliverable.

[I: KGAS_3523]

Software consists of one or more software elements.

[I: KGAS_2878]

Typical software parts are application, driver, hardware abstractions, operating system, and implemented algorithms.

[I: KGAS_2880]

Software also includes platform elements, third party software and programmable integrated circuits.

4.5 Software Element

[I: KGAS_3095]

The definition of software element is in accordance with ASPICE PAM 4.0 .

4.6 Software Component

[I: KGAS_3651]

The definition of software component is in accordance with ASPICE PAM 4.0 .

4.7 Software Unit

[I: KGAS_2998]

The definition of software unit is in accordance with ASPICE PAM 4.0 .

4.8 Model Element

[I: KGAS_3527]

A model element is the logical representation of one or more basic objects within a tool for model based source code generation.

[I: KGAS_3528]

A basic object is an atomic object in a tool for model based source code generation, which cannot be divided into sub-objects.

4.9 Other definitions

[I: KGAS_3820]

Free and open source software (FOSS) is any software that is distributed under the terms of use and license terms for free and open source software and is subject to sharing or disclosure of the source code of the software under such substantial obligations.

[I: KGAS_3953]

Supplied Software is the software supplied by the Contractor (including contractor software, third-party software).

[I: KGAS_4066]

The contents of the deliverable are defined within the scope of an order.

[I: KGAS_3954]

Software of third party is a third party software that is not contractor software.

[I: KGAS_4067]

Whitebox testing is Testing based on an analysis of the internal structure of the component or system.

[I: KGAS_4068]

Blackbox test is a test technique based on an analysis of the specification of a component or system according to ISO/IEC/IEEE 29119 (KGAS_3479).

[I: KGAS_3956]

Dead code is a code that is included in the delivery and that cannot be executed by the program flow (including error handling) and not included in the specification.

[I: KGAS_3962]

Dead code is fixed, when the code is not executed because of these
- is no longer required,
- is not invoked,
- cannot be invoked.

[I: KGAS_3964]

Dead code is not fixed, when
- a requirement is planned, implemented, but not used by the Contracting authority,
- the code is not invoked by parameterization (e.g. target data container).

5 System and Software Development

[I: KGAS_3124]

This chapter contains requirements for the organization, development processes, work products and infrastructure of the Contractor.

5.1 Overall Process Requirements

[A: KGAS_2074]

The software-defined system or the software included in the deliverable must be developed with processes, which achieve at least a capability level **"level 2"** in an Automotive SPICE® Assessment according to Formula Q Capability Software.

[A: KGAS_4122]

Each release delivered to the Contracting authority must be completely developed, implemented and verified in accordance with KGAS with respect to the requirements agreed with the Contracting authority for that release.

[A: KGAS_4123]

The Contractor must also prove for software that has already been developed that the software development processes with which the software was developed are state-of-the-art.

[A: KGAS_4145]

The proof of KGAS_4123 must be done by using the ASPICE process REU.2 or comparable.

[I: KGAS_4146]

Under KGAS_4123 legacy, platform and bycatch (see KGAS_4147) are also considered.

[I: KGAS_4124]

Software that has already been developed includes, for example, software parts that have already been developed or purchased before nomination.

[A: KGAS_3896]

It must be ensured that no unused code (e.g. inaccessible or dead code) exists.

[A: KGAS_3679]

The Contractor must be able to implement in the software, all deemed necessary error corrections from the Contracting authority, up to 15 years after the end of production (EoP). The Contractor must guarantee to provision that the requirements from KGAS for the delivery of software and that all necessary prerequisites for processing and delivery of the software are met.

[A: KGAS_2035]

All work products specified by the process must be consistent with each other in terms of their content at the time of the release to the Contracting authority.

[A: KGAS_3552]

In order to refine specification elements (e.g. requirements, architectural elements) from one level of abstraction or hierarchy level to the level of abstraction below, the Contractor must define criteria and ensure that they are adhered to.

[A: KGAS_4125]

If the criteria from KGAS_3552 are not met in individual cases, the deviation must be demonstrably justified.

[I: KGAS_4126]

A common criterion of KGAS_3552 is a ratio of the level of abstraction to the underlying level of abstraction of 1 to 10.

[R: KGAS_3968]

In order to ensure that the use of individual control units in the field, provide data protection conformation , the data protection requirements must be taken into account from the start of development. The "Guideline for the data protection requirements for the (further) development of control units with memory function" is to be adhered (KGAS_3966).

5.2 Project Management

[A: KGAS_4127]

Project management must ensure adherence to deadlines and features for each release.

[I: KGAS_3595]

Effort estimates for all work packages have been made and are comprehensible.

[I: KGAS_3146]

For all work packages, existing dependencies to other work packages are evident.

[A: KGAS_3167]

For changes and problem solving appropriate lump sum expenses must be planned.

[I: KGAS_3154]

A feature release plan must be created that includes a division of the features into the Contracting authority milestones.

[A: KGAS_3594]

If a feature is implemented over multiple releases, the feature must be further refined in the feature release plan, so that an exact and testable scope per release can be implemented.

[A: KGAS_3157]

The features included in the feature release planning must be mapped to the requirements of the system and software requirements specification.

[I: KGAS_3177]

The schedule includes all activities resulting from problem-solving management and change management entries.

[I: KGAS_3171]

The critical path of the schedule must be systematically identifiable.

[I: KGAS_3178]

Unambiguous definitions for the degree of fulfillment of work packages and activities exist and are applied.

[I: KGAS_3191]

Project risks are evidently identified, evaluated and addressed with counteracting measures.

[A: KGAS_3727]

The status, progress and open points of all activities must be transparent to Contracting authority and Contractor at all times.

5.2.1 Project metrics

[A: KGAS_3612]

For project management, the Contractor must collect metrics beginning from the start of the project.

[A: KGAS_3915]

The Contractor must make the collected metrics available to the Contracting authority for each release and upon request, but at least every 4 weeks.

[A: KGAS_4107]

The minimum set of metrics is defined in KGAS_4093 unless otherwise specified by the Contracting authority.

[A: KGAS_4129]

The exchange format is defined by the Contracting authority.

[A: KGAS_3624]

If deviations are found that can be identified through the use of metrics (KGAS_3612), improvement measures must be defined with target dates.

5.3 Documentation of Deliverable

[A: KGAS_4115]

The documentation is provided in Release Notes KGAS_4116, unless otherwise agreed between the Contracting authority and the Contractor.

[I: KGAS_3214]

The release level of the deliverable (e.g. development status without road use, development status with road use or series release) must be documented.

[I: KGAS_3215]

The implemented changes in the deliverable must be documented, including a description of any bug fixes.

[I: KGAS_3938]

The release notes and feature overviews of all scopes (e.g. modules) of the Sub-contractors must be documented.

[I: KGAS_3216]

The tests to be performed out for the deliverable and their test results must be documented.

[I: KGAS_3219]

Each hardware version compatible with the software version of the deliverable must be documented.

[I: KGAS_3888]

The build environment, build configuration, definitions, compiler options and optimizations, including change history, must be documented.

5.4 System and Software Requirements Specification

[A: KGAS_4130]

All requirements must be traceable to their source.

[A: KGAS_3794]

All requirements that are not evidently related to the requirements of the Contracting authority must be reported by the Contractor.

[A: KGAS_3406]

All assumptions must be specified as requirements and agreed with the Contracting authority.

[A: KGAS_3548]

Own requirements of the Contractor (e.g. requirements for production, requirements from platform parts, etc.) must be documented in the system and software requirement specifications.

[I: KGAS_3266]

All requirements must be verifiably created and analysed considering at least the following aspects:

- Feasibility
- Verifiability
- Self-consistency
- Understandability
- Unambiguousness
- Atomicity

[I: KGAS_3535]

All requirements are assigned to a release or feature.

[A: KGAS_3257]

All requirements must be categorized at least in terms of safety relevance, legal relevance and cybersecurity relevance.

[A: KGAS_3263]

For each functional requirement, all technically possible scenarios must be specified (e.g. target behavior, error case, alternative path, limit cases and worst-case scenarios).

[I: KGAS_3262]

Requirements should not be combined from a higher level of requirements to a lower level of requirements if this results in a loss of information.

[I: KGAS_3264]

Each non-functional requirement must be demonstrably taken into account in requirements and work products derived from it.

5.5 System and Software Architecture Specification

[A: KGAS_3278]

All system and software elements must include a textual description containing at least its goal and purpose.

[A: KGAS_3275]

Syntax and semantics must be defined for the description of the system and software elements within the system and software architecture specifications.

[I: KGAS_3279]

Shared resources (e.g. global variables) shall be regarded as interfaces and must therefore be described in full.

[A: KGAS_3282]

For all software elements the resource consumptions requirements must be specified. These requirements must at least include maximum CPU time consumption, maximum volatile memory consumption, maximum non-volatile memory consumption.

5.6 Software Detailed Design

[A: KGAS_3285]

The software detailed design must include a textual description with goal, purpose and internal structure for each component and each contained unit to ensure traceability, quality, transparency and maintainability of the derived and implemented code.

[A: KGAS_3288]

Syntax and semantics must be defined to describe the detailed software specification.

[A: KGAS_3289]

All units and unit elements which are implemented must be described in the software detailed design.

[I: KGAS_4061]

In the detailed software specification, the solution approach (KGAS_4062) for the externally perceptible behavior of a unit must be described.

[I: KGAS_4062]

The solution approach defines algorithms, calculations, interfaces, function calls and macros and the behavior in the event of an error, as applicable in each case.

[I: KGAS_4063]

All necessary information to implement a solution approach (KGAS_4062) shall be described or referenced.

[I: KGAS_3298]

Shared resources (e.g. libraries, parameters, global and component-global variables) shall be regarded as interfaces and must therefore be described in full.

[A: KGAS_3455]

The software detailed design must also be created for every graphical or model-based program.

[A: KGAS_3682]

For each interface a validation check against the interface description must be specified.

[A: KGAS_3683]

In the case of negative validity tests of interfaces, a defined system and software behavior must be specified.

5.7 Software Construction

5.7.1 Programming Languages

[A: KGAS_2050]

The programming language of the software product must be an international standardized (e.g. ISO/IEC) high-level programming language.

[I: KGAS_2837]

The usage of different programming or script languages in the software product is only permitted after justification, verification of suitability and approval by the Contracting authority.

5.7.2 Manual Code Construction

[R: KGAS_3948]

This chapter only applies to software (deliverable) which uses methods of manually encoded programming.

[A: KGAS_3910]

The Contractor must apply coding guidelines demonstrably appropriate to the project for the entire source code. The relevant guidelines from KGAS_3908 also apply.

[A: KGAS_3321]

Naming conventions must be used in the source code (e.g. function's names, macros, variables, type definitions).

[A: KGAS_3878]

All deviations from the applied coding guidelines must be justified and documented.

[I: KGAS_3328]

Each unit is to be commented with at least a short description of the unit, input and output parameters as well as the return value.

[I: KGAS_3325]

The meaning and logical flow of all decision points in the source code (e.g. if-else, for, switch, while) is to be commented.

[I: KGAS_3326]

The source code is to be commented for all computations that include several variables or parameters with regard to the meaning or logic.

[A: KGAS_3324]

Each unit must be demonstrably verified by source code review.

[I: KGAS_3562]

The objectives of source code reviews (KGAS_3324) are at least: to check whether the source code complies with the software specification, to check non-functional requirements, to check compliance with coding guidelines that cannot be automatically tested.

5.7.2.1 Source Code Metrics

[I: KGAS_4099]

The source code metrics are used to measure the quality of the source code.
The metrics are indicators according to the ISO 25010 quality criteria (KGAS_3043).

[A: KGAS_4100]

In the case of manual source code development, the Contractor shall apply appropriate state-of-the-art source code metrics with defined thresholds.

[A: KGAS_3570]

The selection and appropriateness of the source code metrics (KGAS_4100) must be justified (e.g., in an appropriate strategy document).

[A: KGAS_4065]

Deviations from the source code metrics must be documented with comprehensible justifications.

[A: KGAS_4101]

Threshold violations must be justified in a comprehensible manner at the same level at which they are identified.

[A: KGAS_4102]

Threshold violations must be evaluated in terms of risks and impacts.

[A: KGAS_3571]

Based on risk assessments, appropriate measures must be taken to ensure software quality.

[I: KGAS_4103]

For the programming language "C" , KGAS_4104 describes suitable source code metrics. For other programming languages, these shall be transferred accordingly.

5.7.3 Graphical Programming and model-based Development

[R: KGAS_3947]

This chapter only applies to software (deliverable) that uses methods of graphic programming and/or model-based programming.

[A: KGAS_3862]

The Contractor must evidently apply appropriate modelling guidelines for the project for the entire modelling process. The relevant guidelines from KGAS_3908 also apply.

[A: KGAS_3886]

All deviations from the applied modeling guideline(s) (KGAS_3862) must be justified and documented.

[I: KGAS_3889]

The hierarchy level in the model which is used for code generation is to be considered as the implementation. This implementation usually consists of basic objects which cannot be further divided and it is the last human-created artifact in the software development chain.

[A: KGAS_3313]

Each model element must be verified by review.

[I: KGAS_4131]

The objectives of reviews (KGAS_3313) are at least: to check whether the model corresponds to the software detailed specification, to check non-functional requirements, to check compliance with modeling guidelines that cannot be automatically tested.

[I: KGAS_3314]

For each model element a description must be created that contains at least the objective and purpose.

[I: KGAS_3456]

In the model all decision points regarding meaning or logic are to be commented.

5.7.3.1 Metrics for Graphical Programming

[A: KGAS_4105]

The model metrics are used to measure the quality of the graphical programming.
The metrics are indicators according to the quality criteria of ISO 25010 (see KGAS_3043).

[A: KGAS_3865]

The Contractor must apply appropriate model metrics with defined boundaries for graphical programming.

[A: KGAS_3866]

Selection and qualification of the model metrics (KGAS_3865) must be justified (e.g. within a respective strategy document).

[A: KGAS_4064]

Deviations from the model metrics must be documented with comprehensible justification.

[A: KGAS_3902]

Threshold violations must be justified in a comprehensible manner at the same level at which they are identified.

[A: KGAS_4106]

Threshold violations must be evaluated in terms of risks and impacts.

[A: KGAS_3867]

Based on risk assessments, appropriate measures must be taken to ensure software quality.

5.7.4 Machine Learning (ML)

[R: KGAS_4009]

For software (deliverable) that uses machine learning, neural networks or comparable data-based components, KGAS_2074 applies.

5.7.5 Qualification of Tools

[I: KGAS_3117]

Every software-based tool in the software development tool chain must be qualified based on the normative requirements of ISO 26262: 2018 (KGAS_3895) and the requirements of ISO/SAE 21434 (KGAS_4094).

[A: KGAS_3481]

Manufacturer information (e.g. manuals, guidelines, erratas) of each software-based tool must be verifiably taken into account in the project.

5.8 Test

5.8.1 Test Planning

[A: KGAS_3556]

A test plan including a test strategy according to ISO/IEC/IEEE 29119 (KGAS_3479) must be created.

[I: KGAS_3334]

The test plan must include project specific test goals.

[I: KGAS_3335]

The test plan shall include a description of how a complete test coverage of all specifications is achieved (e.g. customer requirement specification, interface specification, software requirement specification, software architecture specification, software detailed design).

[A: KGAS_3364]

Black-box tests must be specified before white-box tests.

[I: KGAS_3657]

The test plan can contain a joint test strategy of the Contracting authority and Contractor.

5.8.2 Test Case Specification

[A: KGAS_3500]

Test case specification must fulfil the requirements of ISO/IEC/IEEE 29119 (KGAS_3479).

[A: KGAS_54]

Each test case specification must be created by anyone who neither implemented nor specified the object to be tested.

[A: KGAS_3359]

Possible boundary values must be tested for each requirement, interface, parameter and decision point.

[I: KGAS_3366]

If more than 10 test cases are necessary for verification of a requirement, the quality of the requirement must be assessed (e.g. check if it is atomic). If the requirement cannot be improved, an appropriate structuring of the test cases must be used.

5.8.3 Test Execution in general

[A: KGAS_3370]

Each test result must be allocated to an explicit configuration state of the test object (version of software, hardware, mechanics).

[A: KGAS_3372]

The used test environment must be documented (e.g. which type of test environment, test bench, software and hardware version).

[A: KGAS_3685]

If the deliverable contains failed test cases, the Contractor must analyze the associated risks and communicate them to the Contracting authority.

5.8.4 Software Unit Test

[A: KGAS_3376]

The software unit tests must verify a 100% coverage of the software detailed design.

[A: KGAS_3377]

The software unit tests must at least provide 100% branch coverage of the source code (C1).

[A: KGAS_3378]

Deviations of the 100% branch coverage required in KGAS_3377 must be justified.

[I: KGAS_3584]

Source code that cannot be covered by black-box-tests (also see KGAS_3378) can be verified by white-box tests.

[I: KGAS_3554]

The test case specification considers at least the following types of software errors: divide by zero, range violations, value range violations, infinite loops, type errors, initialization errors, unauthorized access, unreachable source code.

[A: KGAS_4148]

All software units must be statically verified.

5.8.5 Software Integration Test

[A: KGAS_3502]

The interfaces of all software elements and components must be tested regarding static structure, contents and timing behavior.

[I: KGAS_3383]

Software integration tests shall cover all requirements which are derived from the software architecture specification. Among others, these requirements include data interfaces (incl. structures and timing), function calls, global variables access, execution orders, resource consumptions, performance, task scheduling, process and interrupt server routines (ISR).

[A: KGAS_3636]

For all tasks, processes and interrupt service routines (ISR), the maximum and average net runtimes (see KGAS_3638) must be determined and documented on the target hardware for each release.

[I: KGAS_3638]

The net runtimes are the runtimes minus the runtime changes caused by the measurements.

[A: KGAS_3637]

For each release, the maximum and average resource consumptions (KGAS_3282) of all software elements on the target hardware must be determined, documented and verified against the resource consumption objectives (KGAS_3282).

5.8.6 Software Verification

[A: KGAS_3503]

The software verification must verify 100% coverage of the software requirements.

5.8.7 System Integration Verification

[A: KGAS_3619]

The interfaces of all system elements must be tested regarding static structure, contents and timing behavior.

5.8.8 System Verification

[A: KGAS_3506]

The system verification must verify 100% coverage of the system requirements.

5.9 Quality Assurance and Management

5.9.1 Quality Management

[A: KGAS_53]

The process and product quality assurance of the Contractor must be personally and organizationally independent from the product development.

[A: KGAS_2904]

The goals, evaluation methods, activities and criteria of quality assurance of the Contractor must not be influenced by the project lead.

[A: KGAS_3129]

The quality assurance goals must be measurable.

[A: KGAS_2911]

The quality assurance of the Contractor must be involved in the release process of the software deliverables at least by providing a quality statement.

[A: KGAS_2913]

Employees of the contractor quality assurance department must have the technical qualifications to be able to confirm that the reviews have been carried out properly (content-related and formal).

5.9.2 Review of Work Products

[A: KGAS_2941]

The reviews must be regularly accompanied by quality assurance, so that a professional execution of the reviews can be confirmed.

[I: KGAS_3508]

The verification criteria of a review contain at least the following points:

- Formal requirements
- Content-related requirements
- Consistency
- Plausibility (regarding both within the work product and in relation to parent work products)
- Unambiguousness
- Self-consistency
- Maintainability
- Understandability

5.9.3 Verification of Development Processes

[A: KGAS_3477]

Compliance with all processes must be verified regularly by the contractor quality assurance, at least every two months.

[A: KGAS_2922]

The Contractor must inform the Contracting authority of all for Contracting authority relevant project risks arising from identified deficits.

5.10 Configuration Management

[A: KGAS_3389]

For each project milestone, quality milestone and release, all configuration elements must be reproducible and recoverable.

[A: KGAS_3759]

For all software elements, the used version and patch level (if any) must be documented, e.g. in form of a software bill of materials.

5.11 Problem Resolution Management

[A: KGAS_3608]

The Contractor must communicate all relevant open product problems to the Contracting authority with every release.

[A: KGAS_3417]

Problem descriptions must include the particular process step in which the product problem or work product deviation has been identified (e.g. software detailed design review, source code review, unit test, software test, system test).

[I: KGAS_3418]

For all product problems the following information is evident and traceable:

- Hardware version
- Software version
- Initial situation
- Failure severity
- Executed steps
- Expected results
- Observed results
- References to violated specifications
- A statement on the reproducibility of the problem
- Source of the problem

[I: KGAS_3421]

All product problem descriptions include links to available log files, traces and measurement results necessary for reproducibility.

[I: KGAS_3609]

The problem source is the first faulty work product (e.g. requirement, specification, source code, test specification).

[A: KGAS_4132]

Problems in the product must be systematically analyzed down to their cause of failure.

[A: KGAS_4133]

If problems are due to process weaknesses, they must be addressed and fixed.

[A: KGAS_4134]

Problem resolution management is designed to prevent known issues from recurring.

5.12 Third Party Software

[R: KGAS_3940]

This chapter applies for systems and software (deliverable) which uses third party software.

[I: KGAS_3941]

Free and Open Source Software (see chapter 5.14) is a version of Third Party Software.

[A: KGAS_3438]

The Contractor is obliged to encapsulate all third party software within software elements.

[A: KGAS_3883]

The encapsulation (KGAS_3438) must assure that only those functions and interfaces of the encapsulated software which are specified in the software requirements and software architecture can be used.

[A: KGAS_3442]

Systems developed by the Contractor must only use complete third party software elements. A partial use (e.g. copy & paste approaches) is not allowed.

[A: KGAS_3142]

Each third party software element must be marked in the software architecture specification.

[A: KGAS_3531]

For each third party software element, origin, author and right holders must be documented.

[A: KGAS_3437]

For all third party software elements, the original requirements on which those third party elements had been developed must be traceable to the software requirements.

[A: KGAS_3440]

The selection of third party software elements (incl. version and patch level) need to be justified and agreed with the Contracting authority.

[A: KGAS_3443]

The Contractor must ensure that all third party software elements are validated regarding that those components exclusively provide the specified functions and do not provide other, potentially undesired functions.

[R: KGAS_3446]

If third party software elements are used, the Contractor must ensure that the usage of all test methods and levels necessary for the development of the whole software is still possible.

[R: KGAS_3923]

The Contractor shall bear sole responsibility for ensuring that the use of the delivered software is permissible in accordance with the contract and the intended use.

5.13 Free and Open Source Software

[R: KGAS_3942]

This chapter applies to systems and software (deliverable) which uses Free and Open Source Software (see KGAS_3820).

[A: KGAS_3822]

The use of FOSS is only permitted if the Contractor has observed and successfully completed the respective FOSS process of the Contracting authority, fulfills all license requirements of the FOSS used and the specifications of this clause 5.14. This also applies if the relevant license conditions expressly permit this use in both original and edited or other form. In addition, if the Volkswagen, Volkswagen Commercial Vehicle or AUDI AG brands are the Contracting authority, FOSS may only be used if the prior consent of the Contracting authority has been obtained in written form.

[I: KGAS_3821]

A copyleft license is a form of use and license terms for open source software that contains conditions that may result in the software elements integrated or connected to the respective open source software also being distributed only under the respective terms of use and license of this copyleft license (effect of the so-called copyleft effect). The Contractor must ensure that the Delivered Software does not contain any license incompatibilities.

[A: KGAS_3833]

The Contractor shall not use FOSS in the deliverable in such a way that causes a copyleft effect for newly developed or pre-existing proprietary software under the agreement. Exceptions are adjustments within pre-existing FOSS components (e.g. bug fixes and adaptations to the specific hardware) and individual cases agreed with the Contracting authority.

[R: KGAS_3830]

The Contractor may only use FOSS in the delivered software which does not restrict the contractual and intended use of its services by the Contracting authority and Volkswagen Group companies.

[A: KGAS_4097]

At the time of delivery of the software, the Contractor grants the Contracting authority the sub-licensable and transferable right to modify proprietary software contained in the contractual products for his own use and to carry out reverse engineering for the purpose of troubleshooting (debugging) such processing, insofar as this proprietary software is compatible with the GNU Lesser General Public License v2.1 (LGPL-2.1) licensed software components.

[A: KGAS_4098]

The Contractor shall ensure to grant the Contracting authority the aforesaid right (KGAS_4097) also with regard to any third party software components.

[A: KGAS_3801]

The Contractor must provide the Contracting authority with information on all free and open source software elements used in the delivered software. For each software element used, the following information must be included:

- Component Name/Unit Name
- Unique version identifier
- License name with unique license version number
- Complete license text
- Download link of the license text and source code including the last access date
- Source code and copyright notices
- Information on whether source code and copyright notices are to be shared or disclosed
- Any sub-elements required for the use of the software element, including the aforementioned details on licensing

- Information as to whether the license prescribes a mandatory provision of the license information to the end user
- Interface information for the integration of open source software components with the exclusion of the triggering of copyleft effects
- Any files contained in the software component and under a different license, including the aforementioned licensing information.

[A: KGAS_3834]

The Contractor must provide the Contracting authority with the information required in KGAS_3801 with each version of the software (release, update, version, etc.) as well as at the request of the Contracting authority, whereby both a complete overview must be made available as well as a delta overview that marks the changes in comparison with the previous status.

[A: KGAS_3824]

Before delivery the Contractor must test the Delivered Software with commercially available analysis software for contained FOSS elements including their dependencies and any sub-elements (including files).

[A: KGAS_3828]

At the request of the Contracting authority, the Contractor must provide the Contracting authority with the details, materials, documents and results of the analysis carried out (KGAS_3824).

[R: KGAS_3810]

If the Contractor implements a technical solution patented or for which a patent has been applied for by the Contracting authority, no open source software solutions may be used, whose licenses impede the cost liable licensing of the patent.

[A: KGAS_4135]

Furthermore, the use of FOSS by the Contractor may only happen in such a way that there is no conflict with the digital signature or the authenticated vehicle programming procedure of the Contracting authority and that authentication information, cryptographic keys or other information relating to the software used in the vehicle remain unaffected and, in particular, do not have to be disclosed to third parties and that third parties do not otherwise have to reinstall (changed) code in the vehicle must be made possible.

[A: KGAS_4136]

If the Contracting authority requires certification according to ISO/IEC 5230:2020(E) from the Contractor before the conclusion of the contract, the Contractor assumes as an essential contractual obligation either to prove the certification carried out by an external certification service provider in a suitable form upon conclusion of the contract or to have it carried out by such a provider and to prove it within six months of the conclusion of the contract.

5.14 Cybersecurity Relevant Development

5.14.1 General Cybersecurity Requirements

[R: KGAS_3687]

This chapter applies to systems and software (deliverable) that have been classified as security relevant by the Contracting authority's Brand Security Department.

[A: KGAS_3738]

The Contractor must conduct and document cybersecurity risk analysis (Chapter 5.14.4) on system and software level (based on requirements and architecture) for the entire deliverable.

5.14.2 Cybersecurity Terminology

[I: KGAS_3703]

Threat analysis and risk assessment - TARA
Threat analysis and risk assessment are methodological procedures that can be used to determine the extent to which the system/element and its environment may be affected by a threat scenario.

[I: KGAS_4138]

A threat analysis and risk assessment must be taken into account on the own deliverable.

[I: KGAS_3704]

Asset
In the sense of cybersecurity, assets are entities worth protecting for an institution.

[I: KGAS_3705]

Cybersecurity goal
Concept-level cybersecurity requirements associated with one or more threat scenarios.

[I: KGAS_4139]

Cybersecurity properties
Attribute that may be worth protecting.

[I: KGAS_3706]

Threat
A threat is a possible cause of the compromise of one or more protection objectives in order to realize a damage scenario.

[I: KGAS_3868]

Backdoor
A Backdoor is an access to a software or hardware system that bypasses the specified access thereby it was implemented intentionally or secretly.

[I: KGAS_3708]

Attack
An attack is an unwanted or unauthorized act that realizes a threat.

[I: KGAS_3709]

Attack vector
An attack vector describes a possibility to perform an attack.

[I: KGAS_3710]

Risk
A risk is a set of threats that are evaluated with respect to the potential damages caused by the violation of security goals as well as the efforts needed for a successful attack or the aggregation of multiple scored threats.

[I: KGAS_3969]

Cybersecurity Information
All information that is recorded as part of the Monitoring Process and whose cybersecurity relevance (potential weakness) has not yet been classified.

[I: KGAS_3970]

Cybersecurity Event
Cybersecurity information, which is classified as potential weakness (without risk assessment) for the company or its products and requires further steps (CSI Process, Information Assessment, etc.).

[I: KGAS_3971]

Cybersecurity Weakness
Defect or property that leads to an undesirable behavior.

[I: KGAS_3707]

Cybersecurity Vulnerability
The weakness of an asset that can be exploited by one or more attacks.

[I: KGAS_3927]

Cybersecurity Incident
Situation in the field that could occur due to the exploitation of a Cybersecurity Weakness.

[I: KGAS_3811]

Incident response process
An incident response process is a defined process with the goal to adjust series products as soon as possible in case of detected vulnerabilities in order to minimize any risks (possibly accompanied by functional constraints) and to eliminate vulnerabilities with recovery of full functionality.

[I: KGAS_3711]

Cybersecurity Requirement
Cybersecurity requirements define requirements for the deliverable specifying properties to prevent or reduce threats.

[I: KGAS_3712]

Cybersecurity Control
A cybersecurity control describes the (technical) realization of cybersecurity requirements to reduce risks and logically group cybersecurity requirements that are needed to successfully implement this cybersecurity control.

[I: KGAS_3713]

Cybersecurity Concept
The cybersecurity concept is a work product to document cybersecurity relevant aspects of the deliverable, which mitigate threats. The cybersecurity concept comprises especially cybersecurity controls, considered constraints, architectures as well as made assumptions and conditions.

[I: KGAS_3715]

Data worthy of protection
Data worthy of protection is data which must be secured by means of the security concepts or cybersecurity measures.

[I: KGAS_3717]

Trustworthy
A system, data source, etc. is trustworthy if a proof exists on which one can rely to a certain extent and there is no compromising.

[I: KGAS_3718]

Trust boundary
A trust boundary describes the transition between different levels of confidence.

[I: KGAS_3720]

OWASP (Open Web Application Security Project)
OWASP is an online community providing among other things a standard to conduct cybersecurity verifications on the application layer.
Reference: https://www.owasp.org/

[I: KGAS_3721]

CWE (Common Weakness Enumeration)
CWE is a software community project providing a catalog of software weaknesses and vulnerabilities.
Reference: https://cwe.mitre.org/

[I: KGAS_3904]

CVE (Common Vulnerabilities and Exposures)
CVE® is a list of entries - each containing an identification number, a description, and at least one public reference - for publicly known cybersecurity vulnerabilities. Reference: https://cve.mitre.org/

5.14.3 Cybersecurity Management

[A: KGAS_3851]

After a known unauthorized access to the configuration management system the original state of configuration items must be restored and the Contracting authority is to be informed.

5.14.4 Cybersecurity Risk Analysis

[A: KGAS_4141]

The Contractor must perform cybersecurity risk analysis at the system level and, as necessary, at the software level and at the software element level.

[A: KGAS_3740]

As part of the cybersecurity risk analyses all processed data must be identified as an asset.

[A: KGAS_3741]

As part of the cybersecurity risk analyses all interfaces to and from the commissioned software must be identified.

[A: KGAS_3974]

The Contractor must keep the threats and measures catalogs up to date.

[A: KGAS_3743]

For each threat identified in a cybersecurity risk analysis, the risk must be classified according to a set of grading criteria specified by the Contractor (e.g. Table G.7 in ISO/SAE 21434).

[I: KGAS_4143]

If effects on the component cannot be assessed, the effects on the function can be used for evaluation.

[A: KGAS_3744]

The Contractor must consider identified and/or specified cybersecurity requirements in the risk analysis.

[A: KGAS_3750]

Cybersecurity controls must demonstrably lead to cybersecurity requirements.

5.14.5 Cybersecurity Risk Management

[A: KGAS_3745]

In case of changes at system and/or software level, the cybersecurity risk analysis as well as the cybersecurity concept must be updated accordingly.

[A: KGAS_3980]

Identified weaknesses must be verifiably managed until the risk has been minimized to an acceptable level.

5.14.6 Cybersecurity Architectural and Cybersecurity Design

[A: KGAS_3755]

All data sources must be identified and classified either as trustworthy or non-trustworthy.

[I: KGAS_3855]

Data sources that are located outside the specified trust boundaries are not trustworthy and data sources that are located within the specified trust boundaries are trustworthy. The deliverable may not necessarily be a trust boundary. The deliverable can also have more than one trust boundaries, e.g. in case of multiple µC.

[A: KGAS_3756]

All data from non-trustworthy sources must be validated before being processed.

[A: KGAS_3758]

Error messages, log records and diagnostic records must not contain any sensitive data that could jeopardize the cybersecurity of the ECU or software.

[A: KGAS_3929]

To analyze its scope of delivery, the Contractor must identify suitable sources for identifying weak points and check them against them.

[A: KGAS_3957]

The deliverable must not contain any known vulnerabilities. Deviations must be considered and justified in the risk analyzes.

[A: KGAS_3930]

Sources for the identification of weaknesses must be beside others publications of CWE (KGAS_3721), CVE (KGAS_3904) or reports from the Contracting authority or similar.

[A: KGAS_3761]

All architectural elements that do not fulfill a functional aspect must be identified (e.g. test interfaces). These potential gateways must no longer be accessible to the series software.

5.14.7 Cybersecurity Implementation

[A: KGAS_3762]

In addition to applying appropriate coding guidelines or modeling guidelines (KGAS_3908), the Contractor must apply cybersecurity coding guidelines.

[A: KGAS_3772]

The Contractor must conduct code analysis, in which the compliance of the KGAS_3762 is checked.

[I: KGAS_3872]

The cybersecurity code analysis may be conducted manually or by a tool.

[A: KGAS_4144]

It must be ensured that no unwanted access (Backdoors) are implemented.

[A: KGAS_3764]

Every deviation from the requirements KGAS_3762, KGAS_4144, KGAS_3896 must be justified and documented.

[A: KGAS_3765]

In case the contracted deliverable contains web application(s) or similar, the OWASP (KGAS_3720) guidelines must be followed.

5.14.8 Cybersecurity Case

[A: KGAS_3775]

The Contractor must provide the cybersecurity case by the "Function Complete" milestone (100% software functionality is implemented).

[A: KGAS_3931]

The cybersecurity case must be supplemented not later then 0-series by the evidence that the flash process has been secured against unauthorized accesses and manipulations.

[A: KGAS_3818]

In case of any changes, the cybersecurity case must be continuously updated until delivery of the series product.

[A: KGAS_3776]

The cybersecurity case must include the results of the cybersecurity activities planned in the cybersecurity plan.

[A: KGAS_3817]

The cybersecurity case must include a summary of the results of all risk analysis.

[A: KGAS_3983]

The cybersecurity case must include the adequateness and effectiveness of the cybersecurity measures.

[I: KGAS_3873]

The risk analysis as well as the detailed results of the risk analysis can be viewed within a technical revision at the Contractor.

[A: KGAS_3777]

The cybersecurity case must show that all cybersecurity requirements were implemented and verified.

[A: KGAS_3819]

The cybersecurity case must show the compliance with the cybersecurity coding guidelines.

[A: KGAS_3932]

The cybersecurity case must show that the incident response process to handle identified weaknesses (KGAS_3877) and the active monitoring of the deliverable (KGAS_3784) is established.

[A: KGAS_3984]

The cybersecurity case must be verified by an authority independent of the project.

5.15 Cybersecurity activities in post-development (Operations and Maintenance)

[A: KGAS_3874]

The Contractor must appoint a person responsible for the security of its project scope (Chief Information Security Officer, Chief Product Security Officer, or similar) and set up a functional e-mail

address as the contact for messages from the Contracting authority. The Contractor must save the information in the supplier database (accessible via the One.Group business platform) and update it as required.

[A: KGAS_3985]

For the exchange of data, the Contractor must support the mechanisms and standards used by VW for e-mail encryption in accordance with IT Security Guidelines.
(see KGAS_4087 and KGAS_4088)

[A: KGAS_3877]

The Contractor must establish a contact for a bi-directional response process for handling cyber security notifications and respond to inquiries from the Contracting authority within a reasonable period of time.

[A: KGAS_3784]

The Contractor must establish a process for active and continuous monitoring of the cyber security status for the project scope accordance to ISO21434.

[A: KGAS_3785]

When cybersecurity informations, events, weaknesses, vulnerabilities and incidents occur, the established response process (KGAS_3877) must be followed.

[A: KGAS_3933]

Should the Contractor require a chain of Sub-contractors to deliver the Product for the Contracting authority, the Contractor shall ensure responsiveness and effectiveness in its chain of Sub-contractors as well.

[A: KGAS_3934]

If the Contractor becomes aware of cases belonging to the category Weakness, Vulnerability or Incident category contained within the project scopes, it shall notify the Contracting authority immediately. Where suspicion arises in the following cases Contracting authority products already put into circulation the notification shall be sent to the Contracting authority Incident Team at the email address which are defined in KGAS.

[A: KGAS_3986]

If the first notification (KGAS_3934) is made by the Contracting authority, the Contractor shall send an acknowledgment of receipt, containing a feedback from the Contractor if the supplied products are affected/not affected by the object of the notification within a usual deadline of two working days. In case of delayed information from Contractor, a joint status meeting has to be mutually agreed.

[A: KGAS_3988]

The acknowledgement of receipt must contain a unique reference. The Contracting authority and the Contractor agree on a clear reference that will be used in communication.

[A: KGAS_3939]

Any communication by the Contractor regarding cybersecurity cases shall be on a need-to-know basis.

[A: KGAS_3936]

If the Contractor plans on external communication that concerns to the Contracting authority, this must be coordinated with the Contracting authority Incident Team. This does not apply to communication due to legal requirements. In the event of communication due to legal requirements, the Contracting authority must be informed about the respective cybersecurity incident management.

[A: KGAS_3937]

Within usually 10 working days after the concern has been confirmed, a detailed technical analysis, risk assessment, including cause, effects and possible remedial measures, must be communicated to the appropriate cybersecurity incident management of the Volkswagen Group (KGAS_3890).

[A: KGAS_3989]

The analysis shall be in accordance with the metrics of ISO21434 and shall include at least the categorization and description of the suspected case, a description of the path of attack, a description of the effects, an assessment of the feasibility of occurrence, the affected products, scopes, parts, components, systems and projects together with the specific software version and the probable risk. If necessary, the scope of the initial analysis can be expanded if agreed between the Contractor and the Contracting authority.
In case of delayed information from Contractor, a joint status meeting has to be mutually agreed.

[A: KGAS_3990]

In case of a solution provided by the Contractor, detailed documentation of this must be provided on request and must contain:

- Difference between before and after the change in the product (e.g. with software or hardware)
- Description of the tests / scenarios for effectiveness control.
- The test results.

[A: KGAS_3991]

Upon request, the Contractor must provide hardware and/or software samples on his own costs that enable Volkswagen AG to verify the solution provided and the vulnerability.

[A: KGAS_3992]

Identified Cybersecurity Vulnerabilities must be taken into account in current developments (see KGAS_3746).

6 References

6.1 Documents of the Volkswagen AG

[I: KGAS_2834]

Formel Q Capability Software: contractor quality capability evaluation guidelines for software development processes [Volkswagen AG; Software-Quality Assurance] Available on http://www.vwgroupsupply.com/

[I: KGAS_3908]

List of Coding-/Modeling Guidelines: List of common coding guidelines and modeling guidelines in the automotive context [Volkswagen AG; Software Quality Assurance] Available at http://www.vwgroupsupply.com/

[I: KGAS_4093]

Smart Quality Analytics (SQA): This is the minimum set of project metrics. [Volkswagen AG; Software Quality Assurance]
Available at http://www.vwgroupsupply.com/

[I: KGAS_4116]

ReleaseNotes: Documentation of the deliverable and the defined metrics. [Volkswagen AG; Software Quality Assurance]
Available at http://www.vwgroupsupply.com/

[I: KGAS_3966]

Guideline for the data protection requirements for the (further) development of control units with memory function
Available at http://www.vwgroupsupply.com/

[I: KGAS_4003]

Allow List FOSS Licenses KGAS: This list contains FOSS licenses that can usually be used by the Contractor without hesitation.
Available at http://www.vwgroupsupply.com/

[I: KGAS_4087]

Guideline Secure Data Exchange
Available at http://www.vwgroupsupply.com/

[I: KGAS_4088]

IS-Regulation No. 02.06. Guideline for third parties
Available at http://www.vwgroupsupply.com/
(for employees of Volkswagen AG see KGAS_4089)

[I: KGAS_4089]

IS-Regulation No. 02.02 Guideline for employees (only valid for Volkswagen AG)
Available at https://volkswagen-net.de/wikis/display/ISRegelwerk/IS+Regelungen

[I: KGAS_4147]

LAH.893.909.D Besondere Merkmale in Software und/oder Umgang mit nicht beauftragten Softwareumfängen
Available at http://www.vwgroupsupply.com/

[I: KGAS_4104]

VW SW Source Code Metrics: This is the set of sourcecode metrics. [Volkswagen AG; Software Quality Assurance]
Available at http://www.vwgroupsupply.com/

6.2 Documents of the German Association of Automotive Industry (VDA)

[I: KGAS_3887]

Automotive SPICE® Process Assessment / Reference Model (PAM/PRM) - RELEASE 4.0 or higher

6.3 International Standards and Norms

[I: KGAS_3043]

ISO/IEC 25010:2023 Systems and software engineering -- Systems and software Quality Requirements and Evaluation ("SQuaRE") – Product quality model

[I: KGAS_3479]

ISO/IEC/IEEE 29119:2022 Software and systems engineering - Software testing

[I: KGAS_3895]

ISO 26262:2018 Road vehicles -- Functional safety

[I: KGAS_4094]

ISO/SAE 21434:2021 Road vehicles – Cybersecurity engineering

7 Release Notes

[I: KGAS_4055]

The table with the changes to previous version could be found under http://www.vwgroupsupply.com/.

8 Confidentiality Disclosure

[R: KGAS_3488]