VOLKSWAGEN

AKTIENGESELLSCHAFT

Konzern Grundanforderungen Software

Grundanforderungen, die der Volkswagen-Konzern an im Fahrzeug verbaute und fahrzeugbezogene Software/softwarebestimmt Systeme und deren Entwicklungsprozesse stellt.

Technische Entwicklung, Querschnittslastenheft: LAH.893.909

Erstausgabe	06.09.2002
Änderungsstand	21.05.2025
Lastenheftversion	4.6

Inhaltsverzeichnis

1 Vorwort ... 4
1.1 Zielsetzung ... 4
1.2 Besitzer dieses Dokumentes ... 4
1.3 Cybersecurity Incident Management ... 5
2 Geltungsbereich ... 7
3 Rechte des Auftraggebers und Pflichten des Auftragnehmers ... 8
4 Terminologie ... 9
4.1 Feature ... 9
4.2 System ... 9
4.3 Systemelement ... 9
4.4 Software ... 9
4.5 Softwareelement ... 9
4.6 Software Komponente ... 9
4.7 Software Unit ... 10
4.8 Modellelement ... 10
4.9 Weitere Definitionen ... 10
5 System- und Softwareentwicklung ... 12
5.1 Prozessübergreifende Anforderungen ... 12
5.2 Projektmanagement ... 13
5.2.1 Projektmetriken ... 14
5.3 Dokumentation des Lieferumfangs ... 14
5.4 System- und Softwareanforderungsspezifikation ... 15
5.5 System- und Softwarearchitekturspezifikation ... 16
5.6 Softwarefeinspezifikation (Detailed Design) ... 16
5.7 Softwareerstellung ... 18
5.7.1 Programmiersprachen ... 18
5.7.2 Manuelle Quellcodeerstellung ... 18
5.7.2.1 Quellcodemetriken ... 19
5.7.3 Grafische und modellbasierte Programmierung ... 19
5.7.3.1 Metriken für Grafische Programmierung ... 20
5.7.4 Tool-Qualifizierung ... 20
5.8 Test ... 21
5.8.1 Testplanung ... 21
5.8.2 Testfallspezifikation ... 21
5.8.3 Testdurchführung allgemein ... 22
5.8.4 Software Unittest ... 22
5.8.5 Softwareintegrationstest ... 23
5.8.6 Softwareverifikation ... 23
5.8.7 Systemintegrationsverification ... 23
5.8.8 Systemverifikation ... 23
5.9 Qualitätssicherung und -management ... 24
5.9.1 Qualitätsmanagement ... 24
5.9.2 Qualitätssicherung ... 24
5.9.2.1 Review der Arbeitsprodukte ... 24
5.9.2.2 Prüfung der Entwicklungsprozesse ... 25
5.10 Konfigurationsmanagement ... 25
5.11 Problemlösungsmanagement ... 25
5.12 Software von Dritten ... 26
5.13 Free and Open Source Software ... 27
5.14 Cybersecurity-relevante Entwicklung ... 29

5.14.1 Allgemeine Cybersecurity-Anforderungen 29
5.14.2 Cybersecurity-Terminologie....29
5.14.3 Cybersecurity-Management 31
5.14.4 Cybersecurity-Risikoanalyse 32
5.14.5 Cybersecurity-Risikomanagement....32
5.14.6 Cybersecurity-Architektur und Cybersecurity-Design....32
5.14.7 Cybersecurity-Implementierung....33
5.14.8 Cybersecurity-Nachweis....34
5.15 Cybersecurity-Aktivitäten nach der Entwicklungsphase (Serien-/Feldbetreuung)....34
6 Referenzierte Unterlagen 37
6.1 Dokumente der Volkswagen AG....37
6.2 Dokumente des Verbands der Automobilindustrie (VDA) 38
6.3 Internationale Standards und Normen....38
7 Release Notes....39
8 Vertraulichkeitshinweis....40

1 Vorwort

1.1 Zielsetzung

[I: KGAS_4110]

Die KGAS definiert den von der Volkswagen AG vorgegebenen Rahmen, um die Entwicklung einer Software, für den Einsatz im Fahrzeug und im Fahrzeugumfeld, nach Stand der Technik sicher zu stellen.

[I: KGAS_4111]

Als Grundlage für die Softwareentwicklung nach Stand der Technik dient das Prozessmodell Automotive SPICE® (siehe KGAS_3887).

[I: KGAS_4112]

Die KGAS präzisiert und ergänzt anhand von Prozessanforderungen das Prozessmodell Automotive SPICE® (siehe KGAS_3887).

[I: KGAS_4113]

Die KGAS richtet sich an alle Projektbeteiligten eines Softwareentwicklungsprojektes.

[I: KGAS_4114]

Die KGAS schafft die Grundlage, Softwarequalität im Produkt durch stabile Prozesse zu erreichen (siehe KGAS_3043).

[I: KGAS_3090]

Die Methoden, Bewertungsgrundlagen und Konsequenzen der Qualitätssicherungsaktivitäten des Volkswagen Konzerns bei Lieferanten sind in der "Formel Q Fähigkeit Software" (KGAS_2834) beschrieben.

[I: KGAS_3633]

Die in diesem Dokument verwendeten Begriffe "Auftraggeber" und "Auftragnehmer" sind gleichbedeutend mit den in der Formel Q Fähigkeit Software (KGAS_2834) verwendeten Begriffen "Kunde" und "Lieferant".

1.2 Besitzer dieses Dokumentes

[I: KGAS_2085]

Volkswagen PKW
Qualitätssicherung
Software-Qualität
Brieffach 1413
38436 Wolfsburg
Deutschland

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

Audi AG
Qualitätssteuerung Software
85045 Ingolstadt
Deutschland

Kontakt: software-quality.gq@audi.de

Porsche AG
Qualität Software
Porscheplatz 1
70435 Stuttgart
Deutschland

Kontakt: software.quality@porsche.de

CARIAD SE
Berliner Ring 2,
Brieffach 1080/2
38440 Wolfsburg
Deutschland

Kontakt: quality@cariad.technology

1.3 Cybersecurity Incident Management

[A: KGAS_3890]

Bei Sicherheitsvorkommnissen ist das Incident Team der Marke oder Region zu benachrichtigen, die die Entwicklungsverantwortung für das betroffene Produkt im Volkswagen Konzern hat. Im Fall einer unklaren Verantwortung ist das Incident Team von Volkswagen PKW (KGAS_3876) zu benachrichtigen.

[I: KGAS_3876]

Volkswagen PKW und Volkswagen Nutzfahrzeuge (leichte Nutzfahrzeuge)

Kontakt: CSI-wob@volkswagen.de

[I: KGAS_3891]

Porsche AG

Kontakt: csi@porsche.de

[I: KGAS_3892]

MAN SE

Kontakt: carsecurity@man.eu

[I: KGAS_3926]

Audi AG

Kontakt: vulnerability@audi.de

[I: KGAS_3958]

Kontakt für die in der Entwicklung befindlichen Produkte: Ansprechpartner ist der jeweilige Projektverantwortliche aus der jeweiligen technischen Entwicklung.

[I: KGAS_3993]

Škoda Auto a.s.

Kontakt: teamcsi@skoda-auto.cz

[I: KGAS_4017]

SEAT S.A.

Kontakt: csi@seat.es

[I: KGAS_4117]

Bentley

Kontakt: csi@bentley.co.uk

[I: KGAS_4118]

Lamborghini

Kontakt: csi@lamborghini.com

[I: KGAS_4119]

Cariad SE

Kontakt für Cybersecurity ausnutzbare Schwachstellen (Vulnerabilities): vuln@cariad.technology

Kontakt für Cybersecurity-Vorfälle (Incidents): sirt@cariad.technology

[I: KGAS_3959]

Region China

Kontakt: csi-cn@volkswagen.com.cn

2 Geltungsbereich

[A: KGAS_4120]

Die KGAS gilt für Entwicklungsprozesse und die dazugehörigen entwicklungsbegleitenden Prozesse für Software und softwarebestimmte Systeme (z.B. Steuergeräte mit Software), die zur Realisierung einer Funktion im Fahrzeug beitragen.

[A: KGAS_3028]

Die Anforderungen in diesem Dokument gelten für den gesamten Volkswagen Konzern und seine Auftragnehmer.

[A: KGAS_4121]

Die KGAS ist ab Beginn der Entwicklung von Software und softwarebestimmten Systemen (z.B. Steuergeräte mit Software) umzusetzen.

3 Rechte des Auftraggebers und Pflichten des Auftragnehmers

[A: KGAS_4058]

Anforderungen [A] sind nachweislich einzuhalten.

[I: KGAS_4060]

Informationen [I] dienen dem zusätzlichen Verständnis oder als Hinweis zu einer möglichen Umsetzung der Anforderung.

[A: KGAS_4169]

Informationen sind zu lesen und zu bewerten, ob sich hieraus projektspezifische Anforderungen ergeben. Bei Unklarheiten und Fragen sind die Besitzer dieses Dokumentes anzusprechen.

[A: KGAS_3885]

Alle Entwicklungsartefakte und Dokumente müssen in englischer oder deutscher Sprache verfasst werden.

[A: KGAS_1806]

Der Auftragnehmer muss auf Anfrage des Auftraggebers Nachweise über die Einhaltung der KGAS erbringen.

[A: KGAS_27]

Der Auftragnehmer muss alle von ihm eingesetzten Unterauftragnehmer auf die Erfüllung der KGAS verpflichten und deren Umsetzung sicherstellen.

[A: KGAS_2933]

Können der Auftragnehmer oder von ihm eingesetzte Unterauftragnehmer die KGAS nicht vollständig erfüllen, muss der Auftragnehmer die Abweichungen schriftlich vor Projektstart durch die Qualitätssicherung des Auftraggebers genehmigen lassen. Die vereinbarten und genehmigten Änderungen sind an die Konzern Qualität (Kontakt siehe KGAS_2085) zu senden.

[A: KGAS_51]

Der Auftragnehmer muss dem Auftraggeber die Möglichkeit einräumen, die Einhaltung der KGAS durch Quellcodeanalysen, werkzeuggestützte Analysen und andere geeignete Verfahren zu überprüfen.

[I: KGAS_4149]

Geeignete Verfahren zur Überprüfung der KGAS sind Quellcodeanalysen, werkzeuggestützte Analysen und Andere.

[A: KGAS_4000]

Der Auftragnehmer muss dem Auftraggeber die Möglichkeit einräumen, die Analysen (KGAS_51) auch durch Dritte vollständig oder teilweise wahrnehmen zu lassen.

[A: KGAS_2949]

Der Auftragnehmer muss die Analyse (KGAS_51) durch die Bereitstellung des Quellcodes in der entsprechenden Steuergerätekonfiguration in Räumen und in Anwesenheit des Auftragnehmers unterstützen.

[A: KGAS_3546]

Steht eine Anforderung der KGAS in Widerspruch zu einer Anforderung aus einer mitgeltenden Unterlage, so muss der Auftragnehmer eine spezifische Vereinbarung zwischen dem Auftragnehmer und dem Auftraggeber initiieren.

4 Terminologie

[I: KGAS_1984]

In diesem Kapitel wird definiert, wie relevante Fachbegriffe im Rahmen der KGAS zu interpretieren sind.

4.1 Feature

[I: KGAS_3665]

Ein Feature ist ein durch einen Stakeholder definierter Funktionsumfang, der durch eine Teilmenge der Anforderungen abgebildet wird.

4.2 System

[I: KGAS_2877]

Das System ist der gesamte vom Auftragnehmer zu erbringende Lieferumfang.

[I: KGAS_2879]

Das System besteht aus Systemelementen.

4.3 Systemelement

[I: KGAS_3604]

Die Definition Systemelement erfolgt gemäß ASPICE PAM 4.0 .

4.4 Software

[I: KGAS_2876]

Software ist die gesamte im Lieferumfang enthaltene Software.

[I: KGAS_3523]

Software besteht aus einem oder mehreren Softwareelementen.

[I: KGAS_2878]

Typische Bestandteile von Software sind Applikation, Treiber, Hardware-Abstraktionen, Betriebssystem, implementierte Algorithmen.

[I: KGAS_2880]

Zur Software zählen auch Plattformelemente, Software von Dritten und programmierte Schaltungen.

4.5 Softwareelement

[I: KGAS_3095]

Die Definition Softwareelement erfolgt gemäß ASPICE PAM 4.0 .

4.6 Software Komponente

[I: KGAS_3651]

Die Definition Software Komponente erfolgt gemäß ASPICE PAM 4.0 .

4.7 Software Unit

[I: KGAS_2998]

Die Definition Software Unit erfolgt gemäß ASPICE PAM 4.0 .

4.8 Modellelement

[I: KGAS_3527]

Ein Modellelement ist die logische Repräsentation eines oder mehrerer Basisobjekte in einem Tool zur modellbasierten Codegenerierung.

[I: KGAS_3528]

Ein Basisobjekt ist ein atomares Objekt innerhalb eines Tools zur modellbasierten Codegenerierung, das nicht weiter in Unterobjekte teilbar ist.

4.9 Weitere Definitionen

[I: KGAS_3820]

Free and Open Source Software (FOSS) im Sinne dieses Querschnittslastenhefts bezeichnet jede Software, Teile von Software oder einzelne Files, die vom Rechteinhaber gegenüber jedermann im Quellcode unter Lizenzverträgen bereitgestellt wird, die grundsätzlich unter der Voraussetzung der Erfüllung der Lizenzverpflichtungen die freie Verwendung der Software, insb. auch zum Zwecke der Bearbeitung und Weitergabe (sowohl in ursprünglicher als auch in bearbeiteter Form) gestattet. Software, die unter die Public Domain fällt, gilt ebenfalls als Free and Open Source Software in diesem Sinne.

[I: KGAS_3953]

Gelieferte Software, ist die vom Auftragnehmer gelieferte Software (u.a. Auftragnehmer Software, Software von Dritten).

[I: KGAS_4066]

Die Inhalte des Lieferumfangs sind im Rahmen einer Beauftragung festgelegt.

[I: KGAS_3954]

Software von Dritten ist Fremdsoftware, welche nicht Auftragnehmer-Software ist.

[I: KGAS_4067]

Whitebox-Test ist ein Testverfahren, das auf der inneren Struktur einer Komponente oder eines Systems basiert.

[I: KGAS_4068]

Blackbox-Test ist ein Testverfahren, das auf einer Analyse der Spezifikation einer Komponente oder eines Systems basiert, gemäß ISO/IEC/IEEE 29119 (KGAS_3479) .

[I: KGAS_3956]

Toter Code ist in der Auslieferung enthaltener Code, der durch den von der Spezifikation vorgegebenen Programmablauf (inkl. Fehlerhandling) nicht ausgeführt werden kann.

[I: KGAS_3962]

Toter Code ist gegeben, wenn der Code nicht ausgeführt wird, weil dieser

- nicht mehr benötigt wird,
- nicht aufgerufen wird,
- nicht aufgerufen werden kann.

[I: KGAS_3964]

Toter Code ist nicht gegeben, wenn

- eine Anforderung geplant, umgesetzt, aber durch den Auftraggeber nicht eingesetzt wird,
- der Code durch Parametrierung (z.B. Zieldatencontainer) nicht aufgerufen wird.

[I: KGAS_4150]

Eine App-Entwicklung im Sinne dieses Dokumentes ist gegeben, wenn die folgenden Voraussetzungen sämtlich vorliegen:

- Es handelt sich um ein reines Software-Entwicklungsprojekt.
- Die Software hat keine Relevanz bzgl. funktionaler Sicherheit (Safety).
- Die Software hat keine Relevanz bzgl. Cybersecurity (siehe KGAS_3687).
- Die Software ist im Feld ohne Werkstattaufenthalt updatefähig, z.B. durch Over-the-Air Updates über einen App-Store.
- Für die Software kommen Maschinelles Lernen, Neuronale Netze oder vergleichbare datenbasierte Komponenten nicht zum Einsatz.

5 System- und Softwareentwicklung

[I: KGAS_3124]

Dieses Kapitel beinhaltet Anforderungen an die Organisation, die Entwicklungsprozesse, die Arbeitsprodukte und die Infrastruktur des Auftragnehmers.

5.1 Prozessübergreifende Anforderungen

[A: KGAS_2074]

Das gesamte im Lieferumfang enthaltene softwarebestimmte System oder die Software muss mit Prozessen entwickelt sein, die mindestens einen Reifegrad **„Level 2“** in einem Automotive SPICE® Assessment gemäß Formel-Q Fähigkeit Software erreichen.

[A: KGAS_4151]

Handelt es sich im Lieferumfang ausschließlich um eine App-Entwicklung (KGAS_4150), so muss die Software mit Prozessen entwickelt sein, die mindestens einen Reifegrad „Level 1“ in einem "SPICE for APPs" Assessment erreichen.

[A: KGAS_4152]

Handelt es sich im Lieferumfang ausschließlich um eine App-Entwicklung (KGAS_4150), so entfallen die folgenden Anforderungen und Informationen: KGAS_2074, KGAS_4123, KGAS_3257, KGAS_3117, KGAS_4164, KGAS_3556, KGAS_3334, KGAS_3335, KGAS_3657, KGAS_54, KGAS_3378, KGAS_3502, KGAS_3636, KGAS_3638, KGAS_3619, KGAS_3506, KGAS_3477, KGAS_3438, KGAS_3883, KGAS_3442, KGAS_3437, KGAS_3443.

[A: KGAS_4122]

Jedes an den Auftraggeber geliefertes Release muss in Bezug auf die mit dem Kunden für dieses Release vereinbarten Anforderungen vollständig gemäß KGAS entwickelt, implementiert und verifiziert sein (siehe auch Kapitel 5.3 Dokumentation des Lieferumfangs).

[A: KGAS_4123]

Der Auftragnehmer muss auch für bereits entwickelte Software nachweisen, dass die Softwareentwicklungsprozesse, mit denen die Software entwickelt wurde, dem aktuellen Stand der Technik entsprechen.

[A: KGAS_4145]

Der Nachweis von KGAS_4123 muss über den ASPICE Prozess REU.2 oder vergleichbar erfolgen.

[I: KGAS_4146]

Unter KGAS_4123 wird auch Legacy, Plattform und Beifang (siehe KGAS_4147) betrachtet.

[I: KGAS_4124]

Bereits entwickelte Software beinhaltet z.B. Softwareanteile, die bereits vor Nominierung entwickelt oder eingekauft wurden.

[A: KGAS_3896]

Es muss sichergestellt sein, dass kein unbenutzter Code (z.B. unzugänglicher oder toter Code) vorhanden ist.

[A: KGAS_4170]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (siehe KGAS_4150) so darf abweichend zu KGAS_3896 unbenutzter Code vorhanden sein. Ist dies der Fall, so muss der Auftragnehmer dem Auftraggeber sämtliche Vorkommen von unbenutztem Code (z.B. ungenutzte Library-Funktionen) benennen.

[A: KGAS_3679]

Der Auftragnehmer muss bis zu 15 Jahre nach End of Production (EoP) in der Lage sein, alle vom Auftraggeber für erforderlich erachteten Fehlerkorrekturen in der Software umzusetzen. Der Auftragnehmer muss sicherstellen, dass die gelieferte Software vorgehalten wird und alle notwendigen Voraussetzungen zur Bearbeitung und Lieferung der Software unter Beachtung der Anforderungen der KGAS gegeben sind.

[A: KGAS_2035]

Alle vom Prozess vorgeschriebenen Arbeitsprodukte müssen zum Zeitpunkt eines Releases an den Auftraggeber inhaltlich konsistent zueinander sein.

[A: KGAS_3552]

Zur Verfeinerung von Spezifikationselementen (z.B. Anforderungen, Architekturelemente) von einer Abstraktionsebene bzw. Hierarchieebene auf die darunter liegende Abstraktionsebene muss der Auftragnehmer Kriterien definieren und deren Einhaltung sicherstellen.

[A: KGAS_4125]

Wenn die Kriterien aus KGAS_3552 in Einzelfällen nicht eingehalten werden, muss die Abweichung nachweislich begründet werden.

[I: KGAS_4126]

Ein gängiges Kriterium der KGAS_3552 ist ein Verhältnis einer Abstraktionsebene auf die darunter liegende Abstraktionsebene von 1 zu 10.

[A: KGAS_3968]

Um einen datenschutzkonformen Einsatz der einzelnen Steuergeräte im Feld gewährleisten zu können, sind die datenschutzrechtlichen Anforderungen bereits ab dem Entwicklungsbeginn zu berücksichtigen. Die "Richtlinie für die datenschutzrechtlichen Anforderungen bei der (Weiter-)Entwicklung von Steuergeräten mit Speicherfunktion" ist einzuhalten (KGAS_3966).

5.2 Projektmanagement

[A: KGAS_4127]

Das Projektmanagement muss Termin- und Feature-Treue für jedes Release sicherstellen.

[I: KGAS_3595]

Aufwandsschätzungen für alle Arbeitspakete sind durchgeführt und nachvollziehbar.

[I: KGAS_3146]

Für alle Arbeitspakete sind vorhandene Abhängigkeiten zu anderen Arbeitspaketen ersichtlich.

[A: KGAS_3167]

Für Änderungs- und Problemlösungsumfänge müssen angemessene Pauschalaufwände eingeplant sein.

[I: KGAS_3154]

Es ist eine Feature-Release-Planung zu erstellen, die eine Aufteilung der Features auf die Meilensteine des Auftraggebers beinhaltet.

[A: KGAS_3594]

Wenn ein Feature über mehrere Releases umgesetzt wird, so muss dieses Feature in der Feature-Release-Planung weiter verfeinert werden, so dass pro Release ein exakt zu prüfender Umfang umgesetzt werden kann.

[A: KGAS_3157]

Die in der Feature-Release-Planung enthaltenen Features müssen den Anforderungen aus der System- und Softwareanforderungsspezifikation zugeordnet werden.

[I: KGAS_3177]

Der Terminplan beinhaltet alle Aktivitäten, die sich aus Einträgen des Problemlösungsmanagements und des Änderungsmanagements ergeben.

[I: KGAS_3171]

Der kritische Pfad des Terminplans hat systematisch identifizierbar zu sein.

[I: KGAS_3178]

Eindeutige Definitionen für Erfüllungsgrade von Arbeitspaketen und Aktivitäten existieren und werden angewendet.

[I: KGAS_3191]

Projektrisiken sind nachweislich identifiziert, bewertet und mit entgegenwirkenden Maßnahmen versehen.

[A: KGAS_3727]

Der Status, Fortschritt und offene Punkte aller Aktivitäten müssen zu jeder Zeit für Auftraggeber und Auftragnehmer transparent sein.

5.2.1 Projektmetriken

[A: KGAS_3612]

Zur Projektsteuerung muss der Auftragnehmer von Projektbeginn an Metriken erheben.

[A: KGAS_3915]

Der Auftragnehmer muss zu jedem Release und auf Anfrage die erhobenen Metriken dem Auftraggeber zur Verfügung stellen, mindestens aber alle 4 Wochen.

[A: KGAS_4107]

Der Mindestsatz der Projektmetriken ist definiert in KGAS_4093 sofern vom Auftraggeber nicht anders vorgegeben.

[A: KGAS_4153]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150) und sind abweichend zu KGAS_4093 die definierten Projektmetriken nicht sinnvoll erhebbar, muss der Auftragnehmer den Mindestsatz an Projektmetriken mit dem Auftraggeber zu Projektbeginn abstimmen.

[A: KGAS_4129]

Das Austauschformat wird durch den Auftraggeber festgelegt.

[A: KGAS_3624]

Bei festgestellten Abweichungen, die durch den Einsatz der Metriken (KGAS_3612) erkennbar werden, müssen Verbesserungsmaßnahmen mit Zielterminen festgelegt werden.

5.3 Dokumentation des Lieferumfangs

[A: KGAS_4115]

Die Dokumentation erfolgt in den ReleaseNotes KGAS_4116, sofern zwischen Auftraggeber und Auftragnehmer nicht anders vereinbart.

[I: KGAS_3214]

Das Freigabelevel des Lieferumfangs (z.B. Entwicklungsstand ohne Straßennutzung, Entwicklungsstand mit Straßennutzung oder Serienfreigabe) ist zu dokumentieren.

[I: KGAS_3215]

Die umgesetzten Änderungen des Lieferumfangs sind zu dokumentieren, inklusive Auflistung durchgeführter Fehlerbehebungen.

[I: KGAS_3938]

Die Releasenotes und Feature-Übersichten aller Umfänge (z.B. Module) der Unterauftragnehmer sind zu dokumentieren.

[I: KGAS_3216]

Die für den Lieferumfang auszuführenden Tests und deren Testergebnisse sind zu dokumentieren.

[A: KGAS_3219]

Jede mit der Softwareversion des Lieferumfangs kompatible Hardwareversion ist zu dokumentieren.

[A: KGAS_4154]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so muss abweichend zu KGAS_3219 jede mit der Softwareversion des Lieferumfangs kompatible Version des Betriebssystems dokumentiert sein.

[I: KGAS_3888]

Die Buildumgebung, Buildkonfiguration, Definitionen, Compileroptionen und -optimierungen inkl. Änderungshistorie sind zu dokumentieren.

5.4 System- und Softwareanforderungsspezifikation

[A: KGAS_4130]

Alle Anforderungen müssen zu Ihrer Quelle zurück verfolgbar sein.

[A: KGAS_3794]

Alle Anforderungen, die keinen nachweislichen Bezug zu den Anforderungen des Auftraggebers haben, müssen durch den Auftragnehmer angezeigt werden.

[A: KGAS_3406]

Alle getroffenen Annahmen müssen als Anforderungen spezifiziert und mit dem Auftraggeber abgestimmt werden.

[A: KGAS_3548]

Eigene Anforderungen des Auftragnehmers (z. B. Anforderungen zur Fertigung, Anforderungen aus Plattformanteilen, usw.) müssen in den System- und Softwareanforderungsspezifikationen dokumentiert sein.

[I: KGAS_3266]

Alle Anforderungen sind nachweislich unter Berücksichtigung mindestens folgender Aspekte zu erstellen und analysieren:

- Machbarkeit
- Verifizierbarkeit
- Widerspruchsfreiheit
- Verständlichkeit
- Eindeutigkeit
- Atomarität

[I: KGAS_3535]

Alle Anforderungen sind einem Release bzw. Feature zuzuordnen.

[A: KGAS_3257]

Alle Anforderungen müssen mindestens bezüglich Safety-, Gesetzes- und Cybersecurityrelevanz kategorisiert werden.

[A: KGAS_3263]

Für jede funktionale Anforderung müssen alle technisch möglichen Szenarien spezifiziert werden (z.B. Sollverhalten, Fehlerfall, Alternativpfad, Grenzfälle und Worst-Case Szenarien).

[I: KGAS_3262]

Anforderungen sind von einer höheren Anforderungsebene zu einer niedrigeren Anforderungsebene nicht zusammenzufassen, wenn dadurch Informationsverlust entsteht.

[I: KGAS_3264]

Jede nichtfunktionale Anforderung ist in daraus abgeleiteten Anforderungen und Arbeitsprodukten nachweislich zu berücksichtigen.

5.5 System- und Softwarearchitekturspezifikation

[A: KGAS_3278]

Jedes System- und Softwareelement muss eine textuelle Beschreibung mit mindestens Ziel und Zweck enthalten.

[A: KGAS_3275]

Für die Beschreibung der System- und Softwareelemente innerhalb der System- bzw. Softwarearchitekturspezifikationen müssen Syntax und Semantik definiert sein.

[A: KGAS_4155]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so müssen abweichend zu KGAS_3275 für die Beschreibung der Softwareelemente innerhalb der Softwarearchitekturspezifikationen Syntax und Semantik definiert oder intuitiv erfassbar sein.

[I: KGAS_3279]

Gemeinsam genutzte Ressourcen (z. B. globale Variablen) sind als Schnittstellen anzusehen und entsprechend vollständig zu beschreiben.

5.6 Softwarefeinspezifikation (Detailed Design)

[A: KGAS_4156]

Jede Softwareimplementierung muss aus einem dokumentierten Design abgeleitet sein.

[I: KGAS_3285]

Die Softwarefeinspezifikation soll für jede Komponente sowie jede darin enthaltene Unit eine nachvollziehbare Beschreibung mit Ziel, Zweck und internem Aufbau enthalten, um Nachvollziehbarkeit, Qualität, Transparenz und Wartbarkeit des daraus abgeleiteten und implementierten Codes zu gewährleisten.

[A: KGAS_3288]

Für die Beschreibung der Softwarefeinspezifikation müssen Syntax und Semantik definiert sein.

[A: KGAS_4157]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so müssen abweichend zu KGAS_3288 für die Beschreibung der Softwarefeinspezifikation Syntax und Semantik definiert oder intuitiv erfassbar sein.

[A: KGAS_3289]

Alle zu implementierenden Units und Unitelemente müssen in der Softwarefeinspezifikation beschrieben werden.

[A: KGAS_4158]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so entfällt KGAS_3289 für bereits entwickelte Softwareanteile (KGAS_4124).

[I: KGAS_4061]

In der Softwarefeinspezifikation ist der Lösungsansatz (KGAS_4062) für das nach außen wahrnehmbare Verhalten einer Unit zu beschreiben.

[I: KGAS_4159]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so ist es abweichend zu KGAS_4061 für bereits entwickelte Softwareanteile (KGAS_4124) ausreichend, den Lösungsansatz (KGAS_4062) für das nach außen wahrnehmbare Verhalten einer Unit lediglich für die Änderungen des ursprünglichen Verhaltens (vor Nominierung) zu beschreiben.

[I: KGAS_4062]

Der Lösungsansatz definiert Algorithmen, Berechnungen, Schnittstellen, Funktionsaufrufe und Makros und das Verhalten im Fehlerfall, soweit jeweils anwendbar.

[I: KGAS_4063]

Alle notwendigen Informationen zur Umsetzung eines Lösungsansatzes (KGAS_4062) sind zu beschreiben oder zu referenzieren.

[I: KGAS_3298]

Gemeinsam genutzte Ressourcen (z. B. Libraries, Parameter, globale und komponentenglobale Variablen) sind als Schnittstellen anzusehen und entsprechend vollständig zu beschreiben.

[I: KGAS_4160]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so ist es abweichend zu KGAS_3298 ausreichend, nur die gegenüber der bereits entwickelten Software (KGAS_4124) neu hinzugefügten gemeinsam genutzten Ressourcen vollständig zu beschreiben.

[A: KGAS_3455]

Die Softwarefeinspezifikation muss auch im Falle einer grafischen bzw. modellbasierten Programmierung erstellt werden.

[A: KGAS_3682]

Für alle Schnittstellen muss eine Gültigkeitsprüfung gegenüber der Schnittstellenbeschreibung spezifiziert sein.

[A: KGAS_3683]

Bei negativen Gültigkeitsprüfungen von Schnittstellen muss ein definiertes System- bzw. Softwareverhalten spezifiziert sein.

5.7 Softwareerstellung

5.7.1 Programmiersprachen

[A: KGAS_2050]

Als Programmiersprache des Softwareprodukts muss eine international standardisierte (z. B. ISO/IEC) Hochsprache verwendet werden.

[I: KGAS_2837]

Die Verwendung anderer Programmier- oder Scriptsprachen im Softwareprodukt ist nur nach Begründung, Eignungsnachweis und Genehmigung durch den Auftraggeber zulässig.

5.7.2 Manuelle Quellcodeerstellung

[A: KGAS_3948]

Dieses Kapitel gilt nur für Software (Lieferumfang), bei denen Methoden der handcodierten Programmierung zum Einsatz kommen.

[A: KGAS_3910]

Der Auftragnehmer muss nachweislich für die gesamte Quellcodeerstellung Codierrichtlinien nach Stand der Technik anwenden.

[I: KGAS_4161]

Codierrichtlinien nach Stand der Technik sind in KGAS_3908 aufgeführt.

[A: KGAS_3321]

Innerhalb des Quellcodes müssen definierte Namensregeln verwendet werden (z.B. für Funktionsnamen, Makros, Variablen, Typdefinition).

[A: KGAS_3878]

Alle Abweichungen von den angewandten Codierrichtlinien müssen begründet und dokumentiert werden.

[I: KGAS_3328]

Jede Unit ist mit mindestens einer Kurzbeschreibung der Unit, der Inputparameter und der Rückgabewerte zu kommentieren.

[I: KGAS_3325]

Der Quellcode ist an allen Entscheidungspunkten bezüglich der Bedeutung bzw. Logik zu kommentieren (z. B. bei if-else, for, switch, while).

[I: KGAS_3326]

Der Quellcode ist bei allen Berechnungen mit mehreren Variablen oder Parametern bezüglich der Bedeutung bzw. Logik zu kommentieren.

[A: KGAS_3324]

Jede Unit muss nachweislich durch ein Quellcodereview geprüft werden.

[A: KGAS_4162]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so ist es ausreichend KGAS_3328, KGAS_3325, KGAS_3326 und KGAS_3324 lediglich auf die Änderungen gegenüber der bereits entwickelten Software (KGAS_4124) zu beachten bzw. anzuwenden.

[I: KGAS_3562]

Ziele von Quellcodereviews (KGAS_3324) sind mindestens: Prüfung, ob der Quellcode der Softwarefeinspezifikation entspricht, Prüfung nicht funktionaler Anforderungen, Prüfung Einhaltung nicht automatisch prüfbarer Codierrichtlinien.

5.7.2.1 Quellcodemetriken

[I: KGAS_4099]

Die Quellcodemetriken dienen zur Messbarkeit der Qualität des Quellcodes.
Die Metriken sind Indikatoren gemäß der Qualitätskriterien der ISO 25010 (KGAS_3043).

[A: KGAS_4100]

Bei manueller Quellcodeerstellung muss der Auftragnehmer geeignete Quellcodemetriken nach Stand der Technik mit definierten Grenzwerten anwenden.

[A: KGAS_3570]

Die Auswahl und Eignung der Quellcodemetriken (KGAS_4100) muss begründet sein (z.B. in einem entsprechenden Strategiedokument).

[A: KGAS_4065]

Grenzwertverletzungen müssen nachvollziehbar dokumentiert werden.

[A: KGAS_4101]

Grenzwertverletzungen müssen auf der Ebene nachvollziehbar begründet werden, auf der diese erfasst werden.

[A: KGAS_4102]

Grenzwertverletzungen müssen bezüglich Risiken und Auswirkungen bewertet werden.

[A: KGAS_3571]

Basierend auf Risikobetrachtungen müssen angemessene Maßnahmen getroffen werden, um die Softwarequalität sicherzustellen.

[I: KGAS_4103]

Für die Programmiersprache "C" beschreibt KGAS_4104 geeignete Quellcodemetriken. Für andere Programmiersprachen sind diese sinngemäß zu übertragen.

5.7.3 Grafische und modellbasierte Programmierung

[A: KGAS_3947]

Dieses Kapitel gilt nur für Software (Lieferumfang), bei denen Methoden der grafischen Programmierung und/oder modellbasierten Programmierung zum Einsatz kommen.

[A: KGAS_3862]

Der Auftragnehmer muss nachweislich für die gesamte Modellierung Modellierungsrichtlinien nach Stand der Technik anwenden.

[I: KGAS_4163]

Modellierungsrichtlinien nach Stand der Technik sind in KGAS_3908 aufgeführt.

[A: KGAS_3886]

Alle Abweichungen von der/den angewandten Modellierungsrichtlinie(n) (KGAS_3862) müssen begründet und dokumentiert werden.

[I: KGAS_3889]

Die Hierarchieebene im Modell, aus der Code generiert wird, wird als Implementierung angesehen. Diese Implementierungsebene besteht i.d.R. aus Basisobjekten, die nicht weiter verfeinert werden können, und stellt das letzte menschlich erstellte Artefakt in der Softwareerstellungskette dar.

[A: KGAS_3313]

Jedes Modellelement muss nachweislich durch ein Review geprüft werden.

[A: KGAS_4131]

Das Review (KGAS_3313) eines Modellelementes muss auch die Einhaltung nicht automatisch prüfbarer Modellierungsrichtlinien enthalten.

[I: KGAS_3314]

Für jedes Modellelement ist eine Beschreibung zu erstellen, die mindestens Ziel und Zweck enthält.

[I: KGAS_3456]

Im Modell sind alle Entscheidungspunkte bezüglich der Bedeutung bzw. Logik zu kommentieren.

5.7.3.1 Metriken für Grafische Programmierung

[I: KGAS_4105]

Die Modellmetriken dienen zur Messbarkeit der Qualität der grafischen Programmierung.
Die Metriken sind Indikatoren gemäß der Qualitätskriterien der ISO 25010 (siehe KGAS_3043).

[A: KGAS_3865]

Bei der grafischen Programmierung muss der Auftragnehmer geeignete Modellmetriken nach Stand der Technik mit definierten Grenzwerten anwenden.

[A: KGAS_3866]

Die Auswahl und Eignung der Modellmetriken (KGAS_3865) muss begründet sein (z.B. in einem entsprechenden Strategiedokument).

[A: KGAS_4064]

Grenzwertverletzungen müssen nachvollziehbar dokumentiert werden.

[A: KGAS_3902]

Grenzwertverletzungen müssen auf der Ebene nachvollziehbar begründet werden, auf der diese erfasst werden.

[A: KGAS_4106]

Grenzwertverletzungen müssen bezüglich Risiken und Auswirkungen bewertet werden.

[A: KGAS_3867]

Basierend auf Risikobetrachtungen müssen Maßnahmen getroffen werden, um die Softwarequalität sicherzustellen.

5.7.4 Tool-Qualifizierung

[I: KGAS_3117]

Jedes softwarebasierte Tool in der Softwareerstellungs-Toolkette muss qualifiziert sein basierend auf den normativen Anforderungen der ISO 26262:2018 (KGAS_3895) sowie den Anforderungen der ISO/SAE 21434 (KGAS_4094).

[A: KGAS_4164]

Für nicht FUSI-relevante und nicht Security-relevante Lieferumfänge muss jedes Tool durch eine geeignete Verifikationsmethode (z. B. Review, Test, Validierungstool) qualifiziert werden, um sicherzustellen, dass generierte Arbeitsprodukte entsprechend der Generierungsregeln korrekt erstellt werden.

[A: KGAS_3481]

Herstellerinformationen (z. B. Handbücher, Richtlinien, Fehlerverzeichnis) jedes softwarebasierten Tools müssen im Projekt nachweislich berücksichtigt werden.

5.8 Test

5.8.1 Testplanung

[A: KGAS_3556]

Ein Testplan inklusive Teststrategie gemäß ISO/IEC/IEEE 29119 (KGAS_3479) muss erstellt werden.

[I: KGAS_3334]

Im Testplan sind projektspezifische Testziele zu dokumentieren.

[I: KGAS_3335]

Im Testplan ist zu beschreiben, wie die vollständige Testabdeckung der Spezifikationen (z. B. Pflichtenheft, Schnittstellenspezifikation, Softwareanforderungsspezifikation, Softwarearchitekturspezifikation, Softwarefeinspezifikation) erreicht wird.

[A: KGAS_3364]

Black-Box-Tests müssen vor White-Box-Tests spezifiziert werden.

[I: KGAS_3657]

Der Testplan kann eine gemeinsame Teststrategie des Auftraggebers und Auftragnehmers beinhalten.

5.8.2 Testfallspezifikation

[A: KGAS_3500]

Die Testfallspezifikation muss die Anforderungen der ISO/IEC/IEEE 29119 (KGAS_3479) erfüllen.

[A: KGAS_54]

Jede Testfallspezifikation muss von jemandem erstellt werden, der das Testobjekt weder umgesetzt noch spezifiziert hat.

[A: KGAS_3359]

Etwaige Grenzwerte müssen für jede(n) Anforderung, Schnittstelle, Parameter und Entscheidungspunkt getestet sein.

[I: KGAS_3366]

Falls mehr als 10 Testfälle für die Verifikation einer Anforderung notwendig sind, ist die Qualität der Anforderung zu überprüfen (z. B. daraufhin, ob diese atomar ist). Wenn die Anforderung nicht optimiert werden kann, ist eine geeignete Strukturierung der Testfälle zu verwenden.

5.8.3 Testdurchführung allgemein

[A: KGAS_3370]

Jedes Testergebnis muss einem eindeutigen Konfigurationsstand des Testobjekts (Version der Software und ggf. Hardware, Mechanik usw.) zugeordnet sein.

[A: KGAS_3372]

Die Testumgebung muss dokumentiert sein (z.B. welche Testumgebung, an welchem Prüfstand, Software- und Hardwareversion).

[A: KGAS_3685]

Wenn für einen Lieferumfang Testfälle fehlgeschlagen sind, muss der Auftragnehmer die damit verbundenen Risiken analysieren und diese dem Auftraggeber mitteilen.

5.8.4 Software Unittest

[A: KGAS_3376]

Die Software Unittests müssen eine 100%ige Abdeckung der Softwarefeinspezifikation nachweisen.

[A: KGAS_4165]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so ist abweichend zu KGAS_3376 eine ausreichende Abdeckung der Softwarefeinspezifikation anhand definierter Kriterien nachzuweisen. Hierfür können z.B. Ergebnisse der statischen Codeanalyse herangezogen werden.

[A: KGAS_3377]

Die Software Unittests müssen mindestens eine 100%ige Zweigabdeckung (C1 oder Branch Coverage) des Quellcodes nachweisen.

[A: KGAS_4166]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (KGAS_4150), so ist abweichend zu KGAS_3377 eine ausreichende Zweigabdeckung (C1 oder Branch Coverage) des Quellcodes anhand definierter Kriterien nachzuweisen. Hierfür können z.B. Ergebnisse der statischen Codeanalyse herangezogen werden.

[A: KGAS_3378]

Abweichungen von der in KGAS_3377 geforderten 100%igen Zweigabdeckung müssen begründet sein.

[I: KGAS_3584]

Durch Black-Box-Tests nicht abgedeckter Quellcode (siehe auch KGAS_3378) kann durch White-Box-Tests verifiziert werden.

[I: KGAS_3554]

Die Testfallspezifikation berücksichtigt mindestens die folgenden Softwarefehlerarten: Division durch Null, Bereichsüberschreitungen, Wertebereichsverletzungen, Endlosschleifen, Typ-Fehler, Initialisierungsfehler, unberechtigte Zugriffe, unerreichbarer Quellcode.

[A: KGAS_4148]

Alle Software Units müssen statisch verifiziert werden.

5.8.5 Softwareintegrationstest

[A: KGAS_3502]

Die Schnittstellen aller Softwareelemente und Komponenten müssen hinsichtlich statischer Struktur, Inhalt und Zeitverhalten getestet sein.

[I: KGAS_3383]

Softwareintegrationstests testen gegen alle Anforderungen, die durch die Softwarearchitekturspezifikation erstellt werden. Dazu zählen unter anderem Datenschnittstellen (inkl. Strukturen, zeitlicher Verlauf), Funktionsaufrufe, Zugriffe auf globale Variablen, Aufrufreihenfolgen, Ressourcenauslastung, Performance, Scheduling von Tasks, Prozesse und Interrupt Service Routines (ISR).

[A: KGAS_3636]

Für alle Tasks, Prozesse und Interrupt Service Routines (ISR) müssen die maximalen und durchschnittlichen Nettolaufzeiten (siehe KGAS_3638) auf der Zielhardware für jedes Release ermittelt und dokumentiert sein.

[I: KGAS_3638]

Die Nettolaufzeiten sind die Laufzeiten abzüglich der Laufzeitveränderungen, die durch die Messungen verursacht werden.

[A: KGAS_3637]

Die maximalen und durchschnittlichen Ressourcenverbräuche (KGAS_3282) aller Softwareelemente müssen auf der Zielhardware für jedes Release ermittelt, dokumentiert und gegen die Ressourcenanforderungen geprüft sein.

[A: KGAS_4171]

Handelt es sich im Lieferumfang ausschließlich um eine APP-Entwicklung (siehe KGAS_4150), so müssen abweichend zu KGAS_3637 die maximalen und durchschnittlichen Ressourcenverbräuche der APP für jedes unterstützte Betriebssystem und für jedes Release ermittelt, dokumentiert und gegen die Ressourcenanforderungen geprüft sein.

5.8.6 Softwareverifikation

[A: KGAS_3503]

Die Softwareverifikation muss eine 100%ige Abdeckung der Softwareanforderungen nachweisen.

5.8.7 Systemintegrationsverification

[A: KGAS_3619]

Die Schnittstellen jedes Systemelements müssen hinsichtlich statischer Struktur, Inhalt und Zeitverhalten getestet sein.

5.8.8 Systemverifikation

[A: KGAS_3506]

Die Systemverifikation muss eine 100%ige Abdeckung der Systemanforderungen nachweisen.

5.9 Qualitätssicherung und -management

[I: KGAS_4174]

Qualitätsmanagement beschreibt die systematische Planung und Steuerung von Abläufen mit Blick auf deren Qualität.

[I: KGAS_4175]

Qualitätssicherung ist die Sicherstellung von Qualitätsanforderungen an ein Produkt.

5.9.1 Qualitätsmanagement

[A: KGAS_53]

Das Qualitätsmanagement des Auftragnehmers muss von der Entwicklung des Produkts personell und organisatorisch unabhängig sein.

5.9.2 Qualitätssicherung

[A: KGAS_2904]

Die Ziele, Bewertungsmethoden, -aktivitäten und -kriterien der Qualitätssicherung des Auftragnehmers dürfen nicht durch die Projektleitung beeinflusst werden.

[A: KGAS_3129]

Die Ziele der Qualitätssicherung müssen messbar sein.

[A: KGAS_2911]

Die Qualitätssicherung des Auftragnehmers muss am Freigabeprozess der Software-Lieferung beteiligt sein, mindestens in Form einer Qualitätsaussage.

[A: KGAS_2913]

Mitarbeitende der Qualitätssicherung des Auftragnehmers müssen die fachlichen Qualifikationen besitzen, um die fachgerechte Durchführung (inhaltlich und formal) der Reviews bestätigen zu können.

5.9.2.1 Review der Arbeitsprodukte

[A: KGAS_2941]

Die Reviews müssen regelmäßig von der Qualitätssicherung begleitet werden, so dass eine fachgerechte Durchführung der Reviews bestätigt werden kann.

[I: KGAS_3508]

Die Prüfkriterien eines Reviews enthalten mindestens die folgenden Punkte:

- Formale Anforderungen
- Inhaltliche Anforderungen
- Konsistenz
- Plausibilität (sowohl innerhalb des Arbeitsproduktes als auch zu dem Arbeitsprodukt, aus dem es abgeleitet wurde)
- Eindeutigkeit
- Widerspruchsfreiheit
- Wartbarkeit
- Verständlichkeit

5.9.2.2 Prüfung der Entwicklungsprozesse

[A: KGAS_3477]

Die Einhaltung aller Prozesse muss regelmäßig, mindestens alle 3 Monate, durch die Qualitätssicherung des Auftragnehmers geprüft werden.

[A: KGAS_2922]

Der Auftragnehmer muss den Auftraggeber über alle für den Auftraggeber relevanten Projektrisiken informieren, die aus identifizierten Defiziten resultieren.

5.10 Konfigurationsmanagement

[A: KGAS_3389]

Zu jedem Projektmeilenstein, Qualitätsmeilenstein und Release müssen die Konfigurationselemente reproduzierbar und wiederherstellbar sein.

[A: KGAS_3759]

Für alle Softwareelemente muss die verwendete Version und ggf. Patch Level dokumentiert sein, z.B. in Form einer Software Bill of Materials.

5.11 Problemlösungsmanagement

[A: KGAS_3608]

Der Auftragnehmer muss dem Auftraggeber bei jedem Release an den Auftraggeber alle offenen, für den Auftraggeber relevanten Produktprobleme mitteilen.

[A: KGAS_3417]

Die Problembeschreibung muss den Prozessschritt enthalten, in dem das Produktproblem oder die Arbeitsproduktabweichung gefunden wurde (z. B. Softwarefeinspezifikation-Review, Quellcode-Review, Unittest, Softwaretest, Systemtest).

[I: KGAS_3418]

Bei allen Produktproblemen sind folgende Informationen ersichtlich und nachvollziehbar:

- Hardwareversion
- Softwareversion
- Ausgangssituation
- Fehlerschwere
- Durchgeführte Schritte
- Erwartete Ergebnisse
- Beobachtete Ergebnisse
- Bezüge zu verletzten Spezifikationen
- Informationen zur Reproduzierbarkeit des Problems
- Quelle des Problems

[I: KGAS_3421]

In allen Beschreibungen von Produktproblemen sind für die Reproduzierbarkeit notwendige Logdateien, Traces und Messergebnisse zu verlinken.

[I: KGAS_3609]

Die Quelle des Problems ist das erste, originäre fehlerhafte Arbeitsprodukt (z. B. Anforderung, Spezifikation, Quellcode, Testspezifikation).

[A: KGAS_4132]

Probleme im Produkt müssen systematisch bis zu ihrer Fehlerursache analysiert werden.

[A: KGAS_4133]

Wenn Probleme auf Prozessschwächen zurückzuführen sind, so müssen diese abgestellt werden.

[A: KGAS_4134]

Das Problemlösungsmanagement soll verhindern, dass bekannte Probleme erneut auftreten.

5.12 Software von Dritten

[A: KGAS_3940]

Dieses Kapitel gilt für Systeme und Software (Lieferumfang), bei denen Software von Dritten zum Einsatz kommt.

[I: KGAS_3941]

Free and Open Source Software (siehe Kapitel 5.14) ist eine Variante von Software von Dritten.

[A: KGAS_3438]

Der Auftragnehmer ist verpflichtet, jede verwendete Software von Dritten in Softwareelementen einzukapseln.

[A: KGAS_3883]

Die Kapselung (KGAS_3438) ist so auszulegen, dass nur die durch Softwareanforderungen und Softwarearchitektur spezifizierten Funktionen und Schnittstellen der eingekapselten Software angesprochen werden können.

[A: KGAS_3442]

Vom Auftragnehmer entwickelte Systeme dürfen nur komplette Softwareelemente von Dritten verwenden. Eine teilweise Verwendung (z. B. über Copy & Paste Ansätze) ist nicht erlaubt.

[A: KGAS_3142]

Jedes verwendete Softwareelement von Dritten muss in der Softwarearchitekturspezifikation einzeln gekennzeichnet werden.

[A: KGAS_3531]

Für jedes verwendete Softwareelement von Dritten müssen Herkunft und Urheber bzw. Rechtsinhaber dokumentiert werden.

[A: KGAS_3437]

Für alle Softwareelemente von Dritten müssen die ursprünglichen Anforderungen, nach denen die Software von Dritten entwickelt wurde, den Softwareanforderungen zugeordnet werden.

[A: KGAS_3440]

Die Auswahl der verwendeten Softwareelemente von Dritten (inkl. Version und Patch Level) muss begründet und mit dem Auftraggeber abgestimmt werden.

[A: KGAS_3443]

Der Auftragnehmer muss sicherstellen, dass alle Softwareelemente von Dritten daraufhin geprüft werden, dass diese nur die spezifizierten Funktionen erbringen und keine anderen, möglicherweise unerwünschten Funktionen beinhalten.

[A: KGAS_3446]

Bei der Verwendung von Softwareelementen von Dritten muss der Einsatz aller für die Softwareentwicklung notwendigen Testmethoden und Teststufen für die Gesamtsoftware weiterhin möglich sein.

[A: KGAS_4167]

Handelt es sich im Lieferumfang ausschließlich um eine App-Entwicklung (KGAS_4150), so muss abweichend zu KGAS_3446 ein Test der Gesamtsoftware unter Anwendung geeigneter Testmethoden weiterhin möglich sein.

[A: KGAS_3923]

Der Auftragnehmer trägt die alleinige Verantwortung dafür, dass die Nutzung der Gelieferten Software vertrags- und bestimmungsgemäß zulässig ist und dokumentiert dieses gegenüber dem Auftraggeber.

5.13 Free and Open Source Software

[A: KGAS_3942]

Dieses Kapitel gilt für Systeme und Software (Lieferumfang), bei denen Free and Open Source Software zum Einsatz kommt (siehe KGAS_3820).

[A: KGAS_3822]

Die Verwendung von FOSS ist nur zulässig, wenn der Auftragnehmer den jeweiligen FOSS-Prozess des Auftraggebers beachtet und erfolgreich abgeschlossen hat, sämtliche Lizenzanforderungen der eingesetzten FOSS und die Vorgaben dieser Ziffer 5.14 erfüllt. Dies gilt auch dann, wenn die einschlägigen Lizenzbedingungen diese Verwendung sowohl in ursprünglicher als auch in bearbeiteter oder sonstiger Form ausdrücklich gestatten. Sofern die Marken Volkswagen, Volkswagen NFZ oder die AUDI AG Auftraggeber sind, darf FOSS zudem nur eingesetzt werden, wenn eine vorherige Zustimmung des Auftraggebers in Textform vorliegt.

[I: KGAS_3821]

Eine Copyleft-Lizenz ist eine Form von Nutzungs- und Lizenzbestimmungen für Open Source Software, die Bedingungen enthält, die dazu führen können, dass die mit der jeweiligen Open Source Software integrierten oder verbundenen Softwareelemente ebenfalls nur unter den jeweiligen Nutzungs- und Lizenzbestimmungen dieser Copyleft-Lizenz verbreitet werden dürfen (Auswirkung des sogenannten Copyleft Effekts). Der Auftragnehmer muss sicherstellen, dass die Gelieferte Software keine Lizenzinkompatibilitäten beinhaltet.

[A: KGAS_3833]

Der Auftragnehmer darf FOSS im Lieferumfang nicht in einer Art einsetzen, die einen Copyleft-Effekt für im Rahmen des Vertrages neu entwickelte oder vorbestehende proprietäre Software auslöst. Ausgenommen sind Anpassungen innerhalb von vorbestehenden FOSS-Komponenten (z.B. Fehlerbehebungen und Anpassungen an die konkrete Hardware) und mit dem Auftraggeber abgestimmte Einzelfälle.

[A: KGAS_3830]

Der Auftragnehmer darf in der Gelieferten Software nur solche FOSS einsetzen, die die vertrags- und bestimmungsgemäße Nutzung seiner Leistung durch den Auftraggeber und Unternehmen der Volkswagen Gruppe nicht beschränkt (siehe auch KGAS_3923).

[A: KGAS_4097]

Sofern und soweit proprietäre Software mit Softwarekomponenten unter einer GNU Lesser General Public License v2.1 (LGPL-2.1) verbunden ist, erteilt der Auftragnehmer dem Auftraggeber mit Lieferung der Software das von der LGPL-2.1 vorgesehene, unterlizenzierbare und übertragbare Recht, in den Vertragsprodukten enthaltene proprietäre Software für den eigenen Gebrauch zu modifizieren und Reverse Engineering zum Zwecke der Fehlerbehebung (Debugging) solcher Bearbeitung vorzunehmen.

[A: KGAS_4098]

Der Auftragnehmer stellt sicher, dass er dem Auftraggeber dieses Recht (KGAS_4097) auch in Bezug auf etwaige Softwarebestandteile Dritter einräumen kann.

[A: KGAS_3801]

Der Auftragnehmer muss dem Auftraggeber Informationen über alle in der Gelieferten Software verwendeten Free and Open Source Softwareelemente zur Verfügung stellen. Für jedes verwendete FOSS-Element müssen mindestens die folgenden Informationen enthalten sein:

- Komponenten-/Unitname
- Eindeutige Versionskennzeichnung
- Lizenzname mit eindeutiger Lizenzversionsnummer
- Vollständiger Lizenztext
- Download-Link des Lizenztexts sowie des Quellcodes inklusive letztem Zugriffsdatum
- Quellcode und Urheberrechtsvermerke
- Information, ob Quellcode und Urheberrechtsvermerke weiterzugeben bzw. zu veröffentlichen sind
- Etwaige Subelemente, die zur Verwendung des Softwareelements erforderlich sind, inklusive der vorgenannten Angaben zur Lizenzierung
- Information, ob die Lizenz eine obligatorische Bereitstellung der Lizenzinformationen an den Endkunden vorschreibt
- Schnittstelleninformationen zur Integration von Open Source Softwarekomponenten unter Ausschluss der Auslösung von Copyleft-Effekten
- Etwaige Dateien, die in der Softwarekomponente enthalten sind und unter abweichender Lizenz stehen, inklusive der vorgenannten Angaben zur Lizenzierung.

[A: KGAS_3834]

Der Auftragnehmer muss dem Auftraggeber die in KGAS_3801 geforderten Informationen mit jeder bereitgestellten Version der Software (Release, Update, Version etc.) sowie auf Anfrage des Auftraggebers zur Verfügung stellen, wobei jeweils sowohl eine vollständige Übersicht zur Verfügung gestellt werden muss als auch eine Delta-Übersicht, welche die Änderungen im Vergleich zum vorherigen Stand kenntlich macht.

[A: KGAS_3824]

Der Auftragnehmer muss die Gelieferte Software vor Auslieferung mit einer marktüblichen Analysesoftware auf enthaltene FOSS Elemente inklusive deren Abhängigkeiten und etwaiger Subelemente (u.a. Dateien) prüfen.

[A: KGAS_3828]

Auf Anfrage des Auftraggebers muss der Auftragnehmer dem Auftraggeber die Angaben, Materialien, Unterlagen und Ergebnisse der durchgeführten Analyse (KGAS_3824) zur Verfügung stellen.

[A: KGAS_3810]

Wenn der Auftragnehmer eine vom Auftraggeber patentierte oder zum Patent angemeldete technische Lösung umsetzt, dürfen keine Open Source Software Lösungen verwendet werden, deren Lizenzen die kostenpflichtige Lizenzierung des Patentes verhindern (siehe auch KGAS_3923).

[A: KGAS_4135]

Die Verwendung von FOSS durch den Auftragnehmer darf außerdem nur so erfolgen, dass kein Konflikt mit der digitalen Signatur oder dem authentisierten Fahrzeugprogrammierverfahren des

Auftraggebers besteht und dass Authentisierungsinformationen, kryptographische Schlüssel oder andere Informationen in Bezug auf die im Fahrzeug verwendete Software unberührt bleiben und insbesondere nicht an Dritte herausgegeben werden müssen und Dritten auch ansonsten keine Neuinstallation von (geändertem) Code im Fahrzeug ermöglicht werden muss.

[A: KGAS_4136]

Sofern der Auftraggeber vor Vertragsschluss eine Zertifizierung nach ISO/IEC 5230:2020(E) vom Auftragnehmer verlangt, übernimmt es der Auftragnehmer als wesentliche Vertragspflicht, die durch einen externen Zertifizierungsdienstleister erfolgte Zertifizierung entweder in geeigneter Form bei Vertragsschluss nachzuweisen oder diese durch einen solchen durchführen zu lassen und binnen sechs Monaten nach Vertragsschluss nachzuweisen.

5.14 Cybersecurity-relevante Entwicklung

5.14.1 Allgemeine Cybersecurity-Anforderungen

[A: KGAS_3687]

Dieses Kapitel gilt für Systeme und Software (Lieferumfang), die von der Marken-Security-Abteilung des Auftraggebers als Cybersecurity-relevant eingestuft wurden.

[A: KGAS_3738]

Der Auftragnehmer muss Cybersecurity-Risikoanalysen (Kap. 5.14.4) für den Lieferumfang auf System- und Softwareebene (auf Basis von Anforderungen und Architektur) durchführen und dokumentieren.

5.14.2 Cybersecurity-Terminologie

[I: KGAS_3703]

Bedrohungsanalyse und Risiko Assessment/Threat analysis and risk assessment - TARA
Bedrohungsanalyse und Risiko Assessment sind methodische Vorgehen, mit denen ermittelt werden kann, inwieweit das System/Element und seine Umgebung von einem Bedrohungsszenario betroffen sein können.

[I: KGAS_4138]

Eine Bedrohungsanalyse und Risiko Assessment ist auf dem eigenen Lieferumfang zu betrachten.

[I: KGAS_3704]

Wert/Asset
Werte sind für eine Institution im Sinne der Cybersecurity schützenswerte Güter.

[I: KGAS_3705]

Cybersecurity-Ziel/Cybersecurity goal
Cybersecurity-Anforderungen auf Konzeptebene, die mit einem oder mehreren Bedrohungsszenarien verbunden sind.

[I: KGAS_4139]

Schutzziel/Cybersecurity properties
Attribut, das schützenswert sein kann.

[I: KGAS_3706]

Bedrohung/Threat
Eine Bedrohung ist eine mögliche Ursache für die Kompromittierung von einem oder mehreren Schutzzielen, um ein Schadensszenario zu realisieren.

[I: KGAS_3868]

Backdoor
Eine Backdoor ist ein Zugang zu einer Software oder zu einem Hardwaresystem, die den spezifizierten Zugriff umgeht. Dabei kann der Zugang gewollt implementiert oder heimlich installiert sein.

[I: KGAS_3708]

Angriff/Attack
Eine Kette absichtlicher Handlungen zur Realisierung eines Bedrohungsszenarios.

[I: KGAS_3709]

Angriffsvektor/Attack vector
Ein Angriffsvektor ist ein potenzieller Weg, einen Angriff durchzuführen.

[I: KGAS_3710]

Risiko/Risk
Ein Risiko ist eine bezüglich möglicher Schäden durch Verletzung von Schutzzielen sowie bezüglich des für eine erfolgreiche Umsetzung nötigen Angriffsaufwands bewertete Bedrohung oder die Zusammenfassung mehrerer bewerteter Bedrohungen.

[I: KGAS_3969]

Cybersecurity-Information
Alle Informationen, die im Rahmen des Monitoring-Prozesses erfasst werden und deren Cybersecurity Relevanz (potenzielle Schwachstelle) noch nicht eingestuft ist.

[I: KGAS_3970]

Cybersecurity-Event
Cybersecurity-Information, welche als potenzielle Schwachstellen (ohne Risikobewertung) für das Unternehmen oder seine Produkte als relevant eingestuft wird und eine weitere Behandlung (CSI-Prozess, Information Assessment, etc.) bedingt.

[I: KGAS_3971]

Cybersecurity-Schwachstelle/Cybersecurity Weakness
Defekt oder Eigenschaft welche zu einem unerwünschten Verhalten führt.

[I: KGAS_3707]

Cybersecurity ausnutzbare Schwachstelle/Cybersecurity Vulnerability
Die Schwachstelle eines Wertes, die durch eine oder mehrere Angriffe ausgenutzt werden kann.

[I: KGAS_3927]

Cybersecurity-Vorfall/Cybersecurity Incident
Situation im Feld, die durch die Ausnutzung von einer Cybersecurity-Schwachstelle eintreten konnte.

[I: KGAS_3811]

Reaktionsprozess/Incident response process
Ein Reaktionsprozess ist ein definierter Prozess, der das Ziel hat, Cybersecurity Informationen und Cybersecurity-Events zu bewerten und in Entwicklung und in Serie befindliche Produkte bei einer erkannten Schwachstelle schnellstmöglich anzupassen. Damit sollen die Risiken minimiert werden (evtl. mit funktionalen Einschränkungen) und die Schwachstellen unter Wiederherstellung der vollen Funktionalität beseitigt werden.

[I: KGAS_3711]

Cybersecurity-Anforderung/Cybersecurity Requirement
Cybersecurity-Anforderungen definieren Anforderungen an den Lieferumfang, die Eigenschaften zur Abwendung bzw. Reduktion von Bedrohungen spezifizieren.

[I: KGAS_3712]

Cybersecurity-Maßnahme/Cybersecurity Control
Eine Cybersecurity-Maßnahme beschreibt die (technische) Umsetzung von Cybersecurity-Anforderungen zur Reduzierung von Risiken und bildet eine logische Gruppierung für Cybersecurity-Anforderungen, die notwendig sind, um diese Cybersecurity-Maßnahme umzusetzen.

[I: KGAS_3713]

Cybersecurity-Konzept/Cybersecurity Concept
Das Cybersecurity-Konzept ist ein Arbeitsergebnis zur Dokumentation der Cybersecurity-relevanten Aspekte des Lieferumfangs, die Bedrohungen entgegenwirken. Es umfasst insbesondere Cybersecurity-Maßnahmen, berücksichtigte Einschränkungen, Architekturen sowie die getroffenen Annahmen und Randbedingungen.

[I: KGAS_3715]

Schützenswerte Daten/Data worthy of protection
Schützenswerte Daten sind Daten, die durch das Cybersecurity-Konzept bzw. durch Cybersecurity-Maßnahmen geschützt werden müssen.

[I: KGAS_3717]

Vertrauenswürdig/Trustworthy
Als vertrauenswürdig kann ein System, eine Datenquelle, usw. eingestuft werden, wenn ein Nachweis existiert, dass sich auf dieses bis zu einem bestimmten Ausmaß verlassen werden kann und keine Kompromittierung vorliegt.

[I: KGAS_3718]

Vertrauensgrenze/Trust boundary
Eine Vertrauensgrenze beschreibt den Übergang zwischen verschiedenen Ebenen des Vertrauens.

[I: KGAS_3720]

OWASP (Open Web Application Security Project)
OWASP ist eine Online Community, die unter anderem einen Standard zur Durchführung von Sicherheitsverifizierungen auf Applikationsebene zur Verfügung stellt.
Referenz: https://www.owasp.org/

[I: KGAS_3721]

CWE (Common Weakness Enumeration)
CWE ist eine Sammlung von Softwareschwachstellen, die durch eine Online Community zur Verfügung gestellt wird.
Referenz: https://cwe.mitre.org/

[I: KGAS_3904]

CVE (Common Vulnerabilities and Exposures)
CVE® ist eine Liste von Einträgen, die jeweils eine Identifikationsnummer, eine Beschreibung und mindestens eine öffentliche Referenz für bekannte Schwachstellen in Bezug auf Cybersecurity enthalten.
Referenz: https://cve.mitre.org/

5.14.3 Cybersecurity-Management

[A: KGAS_3851]

Nach einem erkannten unautorisierten Zugriff auf das Konfigurationsmanagementsystem muss der ursprüngliche Zustand der Konfigurationselemente wiederhergestellt werden und der Auftraggeber ist darüber zu informieren.

5.14.4 Cybersecurity-Risikoanalyse

[A: KGAS_4141]

Der Auftragnehmer muss Cybersecurity Risikoanalysen auf Systemebene und je nach Notwendigkeit auf Softwareebene und auf Softwareelementebene durchführen.

[A: KGAS_3740]

Im Rahmen der Cybersecurity-Risikoanalysen sind auch alle zu verarbeitenden Daten als Asset zu identifizieren.

[A: KGAS_3741]

Im Rahmen der Cybersecurity-Risikoanalysen sind alle Schnittstellen von und zu der beauftragten Software als Asset zu identifizieren.

[A: KGAS_3974]

Der Auftragnehmer muss aktuelle Bedrohungs- und Maßnahmenkataloge pflegen.

[A: KGAS_3743]

Für jede Bedrohung in einer Cybersecurity-Risikoanalyse ist das Risiko systematisch nach einem vom Auftragnehmer spezifizierten Bewertungsschema zu bewerten (z.B. Tabelle G.7 in ISO/SAE 21434).

[I: KGAS_4143]

Wenn Auswirkungen auf das Bauteil nicht bewertbar sind, können zur Evaluierung die Auswirkungen auf die Funktion herangezogen werden.

[A: KGAS_3744]

Der Auftragnehmer muss identifizierte und/oder vorgegebene Cybersecurity-Anforderungen in den Cybersecurity-Risikoanalysen berücksichtigen.

[A: KGAS_3750]

Cybersecurity-Maßnahmen müssen nachweislich zu Cybersecurity-Anforderungen führen.

5.14.5 Cybersecurity-Risikomanagement

[A: KGAS_3745]

Nach Änderungen auf System- und/oder Softwareebene müssen die Cybersecurity-Risikoanalysen sowie das Cybersecurity-Konzept entsprechend aktualisiert werden.

[A: KGAS_3980]

Identifizierte Schwachstellen müssen bis zur akzeptablen Minimierung des Risikos nachweislich gemanagt werden.

5.14.6 Cybersecurity-Architektur und Cybersecurity-Design

[A: KGAS_3755]

Alle Datenquellen müssen identifiziert und als vertrauenswürdig oder nicht vertrauenswürdig klassifiziert werden.

[I: KGAS_3855]

Datenquellen, die sich außerhalb der definierten Vertrauensgrenzen befinden, sind nicht vertrauenswürdig, und Datenquellen, die sich innerhalb der definierten Vertrauensgrenzen befinden, sind vertrauenswürdig. Der Lieferumfang muss nicht zwingend eine Vertrauensgrenze bilden. Der Lieferumfang kann auch mehrere Vertrauensgrenzen haben, z.B. bei mehreren µC.

[A: KGAS_3756]

Alle Daten, die aus nicht vertrauenswürdigen Quellen stammen, müssen vor der Verarbeitung validiert werden.

[A: KGAS_3758]

Fehlermeldungen, Logeinträge und Diagnoseeinträge dürfen keine sensitiven Daten enthalten, über die die Cybersecurity des Steuergerätes bzw. der Software gefährdet werden könnte.

[A: KGAS_3929]

Für die Analyse seines Lieferumfangs muss der Auftragnehmer geeignete Quellen für die Identifikation von Schwachstellen bestimmen und gegen diese prüfen.

[A: KGAS_3957]

Der Lieferumfang darf keine bekannten Schwachstellen enthalten. Abweichungen müssen in den Risikoanalysen berücksichtigt und begründet werden.

[A: KGAS_3930]

Der Auftragnehmer muss geeignete Quellen für die Identifikation von Schwachstellen festlegen und verwenden (siehe KGAS_4168).

[I: KGAS_4168]

Geeignete Quellen für die Identifikation von Schwachstellen sind CWE (KGAS_3721) , CVE (KGAS_3904), Meldungen des Auftraggebers und Vergleichbares.

[A: KGAS_3761]

Alle Architekturelemente, die keinen funktionellen Aspekt erfüllen, müssen kenntlich gemacht werden (z.B. Testschnittstellen). Diese potentiellen Einfallstore dürfen zur Seriensoftware nicht mehr zugänglich sein.

5.14.7 Cybersecurity-Implementierung

[A: KGAS_3762]

Neben der Anwendung geeigneter Codierrichtlinien oder Modellierungsrichtlinien (KGAS_3908) muss der Auftragnehmer Cybersecurity-Codierrichtlinien anwenden.

[A: KGAS_3772]

Der Auftragnehmer muss Codeanalysen durchführen, in denen die Einhaltung der KGAS_3762 geprüft wird.

[I: KGAS_3872]

Die Cybersecurity-Codeanalysen können sowohl manuell als auch toolgestützt durchgeführt werden.

[A: KGAS_4144]

Es muss sichergestellt werden, dass keine ungewollten Zugänge (Backdoors) implementiert sind.

[A: KGAS_3764]

Jede Abweichung zu den Anforderungen KGAS_3762, KGAS_4144, KGAS_3896 muss begründet und dokumentiert werden.

[A: KGAS_3765]

Umfasst der beauftragte Lieferumfang eine Web-Applikation, müssen die Richtlinien der OWASP (KGAS_3720) eingehalten werden.

5.14.8 Cybersecurity-Nachweis

[A: KGAS_3775]

Ein Cybersecurity-Nachweis muss vom Auftragnehmer zum Meilenstein "Function Complete" (100 % Software-Funktionalität ist implementiert) erbracht werden.

[A: KGAS_3931]

Der Cybersecurity-Nachweis muss spätestens zur 0-Serie, um den Nachweis des gegen unbefugte Zugriffe und Manipulation abgesicherten Flash Prozesses, ergänzt werden.

[A: KGAS_3818]

Der Cybersecurity-Nachweis muss bei Änderungen bis Serienlieferung stetig aktualisiert werden.

[A: KGAS_3776]

Der Cybersecurity-Nachweis muss die Ergebnisse der im Cybersecurity-Plan geplanten Cybersecurity-Aktivitäten beinhalten.

[A: KGAS_3817]

Der Cybersecurity-Nachweis muss eine Zusammenfassung der Ergebnisse der Cybersecurity-Risikoanalysen beinhalten.

[A: KGAS_3983]

Der Cybersecurity-Nachweis muss die Angemessenheit und die Effektivität der Cybersecurity-Maßnahmen beinhalten.

[I: KGAS_3873]

Die Risikoanalysen sowie die detaillierten Ergebnisse der Risikoanalysen können im Rahmen einer Technischen Revision beim Auftragnehmer eingesehen werden.

[A: KGAS_3777]

Der Cybersecurity-Nachweis muss aufzeigen, dass die Cybersecurity-Anforderungen umgesetzt und verifiziert wurden.

[A: KGAS_3819]

Der Cybersecurity-Nachweis muss aufzeigen, dass die Cybersecurity-Codierrichtlinien (Codierrichtlinien siehe KGAS_3762) eingehalten wurden.

[A: KGAS_3932]

Der Cybersecurity-Nachweis muss aufzeigen, dass der Reaktionsprozess zum Umgang mit identifizierten Schwachstellen (KGAS_3877) und die aktive Überwachung des Lieferumfangs (KGAS_3784) etabliert sind.

[A: KGAS_3984]

Der Cybersecurity-Nachweis muss durch eine vom Projekt unabhängige Instanz geprüft werden.

5.15 Cybersecurity-Aktivitäten nach der Entwicklungsphase (Serien-/Feldbetreuung)

[A: KGAS_3874]

Der Auftragnehmer hat eine für die Sicherheit seines Projektumfangs verantwortliche Person (Chief Information Security Officer, Chief Product Security Officer, o.ä.) zu benennen und eine funktionsfähige E-Mail-Adresse als Ansprechpartner für Nachrichten des Auftraggebers einzurichten. Der Auftragnehmer hat die Informationen in der Lieferantendatenbank (zugänglich über die Business-Plattform One.Group) zu speichern und bei Bedarf zu aktualisieren.

[A: KGAS_3985]

Der Auftragnehmer muss zum Austausch von Daten die von VW eingesetzten Mechanismen und Standards zur E-Mail-Verschlüsselung gem. IT-Sicherheitsrichtlinien unterstützen.
(siehe KGAS_4087, KGAS_4088 und KGAS_4089)

[A: KGAS_3877]

Der Auftragnehmer hat einen Ansprechpartner für einen bidirektionalen Antwortprozess zur Bearbeitung von Cybersicherheitsmeldungen zu benennen und Anfragen des Auftraggebers innerhalb einer angemessenen Frist zu beantworten.

[A: KGAS_3784]

Der Auftragnehmer muss einen Prozess zur aktiven und kontinuierlichen Überwachung des Cybersicherheitsstatus für den Projektumfang gemäß ISO21434 etablieren.

[A: KGAS_3785]

Wenn Cybersecurity-Informationen, Ereignisse, Cybersecurity-Schwachstellen, Cybersecurity ausnutzbare Schwachstellen und Cybersecurity-Vorfälle auftreten, muss der etablierte Reaktionsprozess (KGAS_3877) befolgt werden.

[A: KGAS_3933]

Sollte der Auftragnehmer eine Unterauftragnehmerkette benötigen, um das Produkt für den Auftraggeber zu liefern, muss der Auftragnehmer auch in seiner Unterauftragnehmerkette Reaktionsfähigkeit und Effektivität sicherstellen.

[A: KGAS_3934]

Erlangt der Auftragnehmer Kenntnis von Fällen der Kategorie Schwachstelle, ausnutzbare Schwachstelle oder Vorfall, die im Rahmen des Projekts enthalten sind, so hat er den Auftraggeber unverzüglich zu informieren. Bei Verdacht auf bereits in Verkehr gebrachte Produkte des Auftraggebers ist die Meldung an das Störungsteam des Auftraggebers unter der in KGAS angegebenen E-Mail-Adresse zu richten.

[A: KGAS_3986]

Erfolgt die erste Anzeige (KGAS_3934) durch den Auftraggeber, so hat der Auftragnehmer innerhalb einer üblichen Frist von zwei Werktagen eine Empfangsbestätigung zu übersenden, die eine Rückmeldung des Auftragnehmers enthält, ob die gelieferten Produkte vom Gegenstand der Mitteilung betroffen oder nicht betroffen sind. Bei verspäteter Information des Auftragnehmers ist einvernehmlich eine gemeinsame Statusbesprechung zu vereinbaren.

[A: KGAS_3988]

Die Empfangsbestätigung muss einen eindeutigen Verweis enthalten. Der Auftraggeber und der Auftragnehmer einigen sich auf eine eindeutige Referenz, die in der Kommunikation verwendet wird.

[A: KGAS_3939]

Jegliche Kommunikation des Auftragnehmers in Bezug auf Cybersecurity-Fälle erfolgt nach dem Need-to-know-Prinzip.

[A: KGAS_3936]

Plant der Auftragnehmer eine externe Kommunikation, die den Auftraggeber betrifft, so ist diese mit dem Incident Team des Auftraggebers abzustimmen. Dies gilt nicht für die Kommunikation aufgrund gesetzlicher Vorgaben. Im Falle einer Kommunikation aufgrund gesetzlicher Vorgaben ist der Auftraggeber über das jeweilige Cybersecurity Incident Management zu informieren.

[A: KGAS_3937]

Innerhalb von in der Regel 10 Arbeitstagen nach Bestätigung des Anliegens muss eine detaillierte technische Analyse, Risikobewertung, einschließlich Ursache, Auswirkungen und möglicher

Abhilfemaßnahmen an das zuständige Cybersecurity Incident Management des Volkswagen Konzerns (KGAS_3890) übermittelt werden.

[A: KGAS_3989]

Die Analyse muss in Übereinstimmung mit der ISO21434 erfolgen und mindestens die Kategorisierung und Beschreibung des Verdachtsfalls, eine Beschreibung des Angriffsweges, eine Beschreibung der Auswirkungen, eine Bewertung der Durchführbarkeit des Auftretens, der betroffenen Produkte, Scopes, Teile, Komponenten, Systeme und Projekte sowie die spezifische Softwareversion und das wahrscheinliche Risiko umfassen. Bei Bedarf kann der Umfang der Erstanalyse erweitert werden, wenn dies zwischen dem Auftragnehmer und dem Auftraggeber vereinbart wird.
Bei verspäteter Information des Auftragnehmers ist einvernehmlich eine gemeinsame Statusbesprechung zu vereinbaren.

[A: KGAS_3990]

Im Falle einer vom Auftragnehmer bereitgestellten Lösung ist diese auf Anfrage detailliert zu dokumentieren und muss folgendes enthalten:

- Unterschied zu vorher und nachher der Veränderung des Produktes (Bspw. bei Soft- oder Hardware)
- Beschreibung der Tests/Szenarien zur Wirksamkeitskontrolle
- die Testergebnisse

[A: KGAS_3991]

Auf Verlangen hat der Auftragnehmer auf eigene Kosten Hard- und/oder Softwaremuster zur Verfügung zu stellen, die es der Volkswagen AG ermöglichen, die bereitgestellte Lösung und die Schwachstelle zu verifizieren.

[A: KGAS_3992]

Identifizierte Cybersecurity ausnutzbare Schwachstellen müssen in aktuellen Entwicklungen berücksichtigt werden (Siehe KGAS_3746).

6 Referenzierte Unterlagen

6.1 Dokumente der Volkswagen AG

[I: KGAS_2834]

Formel Q Fähigkeit Software: Qualitätsfähigkeit Lieferanten Beurteilungsrichtlinie für Software-Entwicklungsprozesse [Volkswagen AG; Software-Qualitätssicherung]
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_3908]

Liste von Codier-/Modellierungsrichtlinien: Auflistung üblicher Codierrichtlinien und Modellierungsrichtlinien im Automotive Kontext [Volkswagen AG; Software-Qualitätssicherung]
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4093]

Smart Quality Analytics (SQA): Das ist der Mindestsatz von Projektmetriken. [Volkswagen AG; Software-Qualitätssicherung]
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4116]

ReleaseNotes: Dokumentation des Lieferumfangs und definierter Metriken.
[Volkswagen AG; Software-Qualitätssicherung]
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_3966]

Richtlinie für die datenschutzrechtlichen Anforderungen bei der (Weiter-)Entwicklung von Steuergeräten mit Speicherfunktion
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4003]

Allow-List FOSS-Lizenzen KGAS: Diese Liste enthält FOSS-Lizenzen, die vom Auftragnehmer in der Regel bedenkenlos eingesetzt werden können.
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4087]

Leitfaden - Sicherer Datenaustausch
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4088]

IS-Regelung Nr. 02.06. Handlungsleitlinie für Dritte
Erhältlich unter http://www.vwgroupsupply.com/
(für Beschäftigte der Volkswagen AG siehe KGAS_4089)

[I: KGAS_4089]

IS-Regelung Nr. 02.02 Handlungsleitlinie für Beschäftigte (Gültig für Volkwagen AG)
Erhältlich unter 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
Erhältlich unter http://www.vwgroupsupply.com/

[I: KGAS_4104]

VW SW Source Code Metrics: Das ist der Satz der Quellcodemetriken. [Volkswagen AG; Software-Qualitätssicherung]
Erhältlich unter http://www.vwgroupsupply.com/

6.2 Dokumente des Verbands der Automobilindustrie (VDA)

[I: KGAS_3887]

Automotive SPICE® Process Assessment / Reference Model (PAM/PRM) - RELEASE 4.0 oder höher.

6.3 Internationale Standards und Normen

[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]

Die Tabelle mit den Änderungen zur vorherigen Version finden sie unter http://www.vwgroup-supply.com/.

8 Vertraulichkeitshinweis

[A: KGAS_3488]

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	21.05.2025
Version	4.6

Content

1 Preamble....4
1.1 Purpose....4
1.2 Document Owners....4
1.3 Cybersecurity Incident Management....5
2 Scope....7
3 Rights of the Contracting authority and Obligations of the Contractor....8
4 Terminology....9
4.1 Feature....9
4.2 System....9
4.3 System Element....9
4.4 Software....9
4.5 Software Element....9
4.6 Software Component....9
4.7 Software Unit....10
4.8 Model Element....10
4.9 Other definitions....10
5 System and Software Development....12
5.1 Overall Process Requirements....12
5.2 Project Management....13
5.2.1 Project metrics....14
5.3 Documentation of Deliverable....14
5.4 System and Software Requirements Specification....15
5.5 System and Software Architecture Specification....16
5.6 Software Detailed Design....16
5.7 Software Construction....17
5.7.1 Programming Languages....17
5.7.2 Manual Code Construction....18
5.7.2.1 Source Code Metrics....18
5.7.3 Graphical Programming and model-based Development....19
5.7.3.1 Metrics for Graphical Programming....20
5.7.4 Qualification of Tools....20
5.8 Test....20
5.8.1 Test Planning....20
5.8.2 Test Case Specification....21
5.8.3 Test Execution in general....21
5.8.4 Software Unit Test....21
5.8.5 Software Integration Test....22
5.8.6 Software Verification....23
5.8.7 System Integration Verification....23
5.8.8 System Verification....23
5.9 Quality Assurance and Management....23
5.9.1 Quality Management....23
5.9.2 Quality Assurance....23
5.9.2.1 Review of Work Products....23
5.9.2.2 Verification of Development Processes....24
5.10 Configuration Management....24
5.11 Problem Resolution Management....24
5.12 Third Party Software....25
5.13 Free and Open Source Software....26
5.14 Cybersecurity Relevant Development....28

5.14.1 General Cybersecurity Requirements....28
5.14.2 Cybersecurity Terminology....28
5.14.3 Cybersecurity Management....30
5.14.4 Cybersecurity Risk Analysis....31
5.14.5 Cybersecurity Risk Management....31
5.14.6 Cybersecurity Architectural and Cybersecurity Design....31
5.14.7 Cybersecurity Implementation....32
5.14.8 Cybersecurity Case....32
5.15 Cybersecurity activities in post-development (Operations and Maintenance)....33
6 References....36
6.1 Documents of the Volkswagen AG....36
6.2 Documents of the German Association of Automotive Industry (VDA)....37
6.3 International Standards and Norms....37
7 Release Notes....38
8 Confidentiality Disclosure....39

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 PKW
Qualityassurance
Software Quality
Letterbox 1413
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

[A: 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.

[A: KGAS_3028]

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

[A: KGAS_4121]

KGAS shall 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

[A: KGAS_4058]

Adherence to requirements shall be demonstrable.

[I: KGAS_4060]

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

[A: KGAS_4169]

Information must be read and assessed to determine whether it gives rise to project-specific requirements. If there are any uncertainties or questions, please contact the owner of this document.

[A: KGAS_3885]

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

[A: KGAS_1806]

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

[A: KGAS_27]

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

[A: KGAS_2933]

If the Contractor or its Sub-contractors cannot fulfil the KGAS completely, the Contractor shall 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).

[A: KGAS_51]

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

[I: KGAS_4149]

Suitable methods for verifying KGAS are source code analyses, tool-based analyses and others.

[A: KGAS_4000]

The Contractor shall give the Contracting authority the opportunity to have the analyses (KGAS_51) carried out in whole or in part by third parties.

[A: KGAS_2949]

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

[A: KGAS_3546]

If a requirement of the KGAS is inconsistent with a requirement from another applicable document, the Contractor shall 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) within the meaning of the KGAS means any software, parts of software or individual files that are made available by the copyright holder to anyone in the source code under license agreements, which do allow in principle the free use of the software also for the purpose of adapting, modifying and distribution (both in original and modified form) provided that the license obligations are met. Software that falls under the public domain is also considered Free and Open Source Software as defined in this clause.

[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).

[I: KGAS_4150]

APP development within the meaning of this document is given if all of the following requirements are met:
• It is a pure software development project.
• The software has no relevance with regard to functional safety.
• The software has no relevance with regard to cybersecurity (see KGAS_3687).
• The software can be updated in the field without having to go to the workshop, e.g. through over-the-air updates via an app store.
• Machine learning, neural networks or similar data-based components are not used for the software.

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 shall be developed with processes, which achieve at least a capability level **"level 2"** in an Automotive SPICE® Assessment according to Formel Q Capability Software.

[A: KGAS_4151]

If the scope of delivery solely includes an APP development (KGAS_4150), the software shall be developed with processes, which achieve at least a capability level "level 1" in a "SPICE for APPs" assessment.

[A: KGAS_4152]

If the scope of delivery solely includes an APP development (KGAS_4150), the following requirements are not applicable: KGAS_2074, KGAS_4123, KGAS_3257, KGAS_3117, KGAS_4164, KGAS_3556, KGAS_3334, KGAS_3335, KGAS_3657, KGAS_54, KGAS_3378, KGAS_3502, KGAS_3636, KGAS_3638, KGAS_3619, KGAS_3506, KGAS_3477, KGAS_3438, KGAS_3883, KGAS_3442, KGAS_3437, KGAS_3443.

[A: KGAS_4122]

Each release delivered to the Contracting authority shall be completely developed, implemented and verified in accordance with KGAS with respect to the requirements agreed with the Contracting authority for that release (see also chapter 5.3 Documentation of Deliverable).

[A: KGAS_4123]

The Contractor shall also prove that the software development processes used to develop software that has already been developed correspond to the current state of the art.

[A: KGAS_4145]

The proof of KGAS_4123 shall 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 shall be ensured that no unused code (e.g. inaccessible or dead code) exists.

[A: KGAS_4170]

If the scope of delivery is exclusively an APP development (see KGAS_4150), unused code may be present in deviation from KGAS_3896. If this is the case, the Contractor must inform the Contracting authority of all occurrences of unused code (e.g. unused library functions).

[A: KGAS_3679]

The Contractor shall 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 shall

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 shall 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 shall 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 shall 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.

[A: KGAS_3968]

In order to ensure that the use of individual control units in the field, provide data protection conformation , the data protection requirements shall 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 shall 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 shall be planned.

[I: KGAS_3154]

A feature release plan should 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 shall 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 shall 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 should 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 shall be transparent to Contracting authority and Contractor at all times.

5.2.1 Project metrics

[A: KGAS_3612]

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

[A: KGAS_3915]

The Contractor shall 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_4153]

If the scope of delivery solely includes an APP development (KGAS_4150), and, in deviation from KGAS_4093, the defined project metrics cannot be collected in a meaningful way, the Contractor shall agree a minimum set of project metrics with the Contracting authority at the beginning of the project.

[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 shall 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) should be documented.

[I: KGAS_3215]

The implemented changes in the deliverable should 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 should be documented.

[I: KGAS_3216]

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

[A: KGAS_3219]

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

[A: KGAS_4154]

If the scope of delivery solely includes an APP development (KGAS_4150), each version of the operating system compatible with the software version of the scope of delivery must be documented in deviation from KGAS_3219.

[I: KGAS_3888]

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

5.4 System and Software Requirements Specification

[A: KGAS_4130]

All requirements shall be traceable to their source.

[A: KGAS_3794]

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

[A: KGAS_3406]

All assumptions shall 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.) shall be documented in the system and software requirement specifications.

[I: KGAS_3266]

All requirements should 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 shall 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 shall 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 should 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 shall include a textual description containing at least its goal and purpose.

[A: KGAS_3275]

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

[A: KGAS_4155]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3275, syntax and semantics for the description of the software elements within the software architecture specifications shall be defined or intuitively understandable.

[I: KGAS_3279]

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

5.6 Software Detailed Design

[A: KGAS_4156]

Every software implementation shall be derived from a documented design.

[I: KGAS_3285]

The detailed software specification should contain a comprehensible description with aim, purpose and internal structure for each component as well as each unit contained therein in order to ensure traceability, quality, transparency and maintainability of the code derived and implemented from it.

[A: KGAS_3288]

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

[A: KGAS_4157]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3288, the syntax and semantics to describe the detailed software specification shall be defined or intuitively understandable.

[A: KGAS_3289]

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

[A: KGAS_4158]

If the scope of delivery solely includes an APP development (KGAS_4150), KGAS_3289 is not applicable for software components that have already been developed (KGAS_4124).

[I: KGAS_4061]

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

[I: KGAS_4159]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_4061, for software components already developed (KGAS_4124) it is sufficient to describe the solution approach (KGAS_4062) for the externally perceptible behaviour of a unit only for the changes to the original behaviour (before nomination).

[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) should be described or referenced.

[I: KGAS_3298]

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

[I: KGAS_4160]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3298, it is sufficient to fully describe only the newly added shared resources compared to the already developed software (KGAS_4124}.

[A: KGAS_3455]

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

[A: KGAS_3682]

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

[A: KGAS_3683]

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

5.7 Software Construction

5.7.1 Programming Languages

[A: KGAS_2050]

The programming language of the software product shall 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

[A: KGAS_3948]

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

[A: KGAS_3910]

The Contractor shall demonstrably apply state-of-the-art coding guidelines for the entire source code creation.

[I: KGAS_4161]

State-of-the-art coding guidelines are listed in KGAS_3908.

[A: KGAS_3321]

Naming conventions shall 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 shall 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 shall be demonstrably verified by source code review.

[A: KGAS_4162]

If the scope of delivery solely includes an APP development (KGAS_4150), it is sufficient to apply KGAS_3328, KGAS_3325, KGAS_3326 and KGAS_3324 only to the changes compared to the already developed software (KGAS_4124).

[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) shall be justified (e.g., in an appropriate strategy document).

[A: KGAS_4065]

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

[A: KGAS_4101]

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

[A: KGAS_4102]

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

[A: KGAS_3571]

Based on risk assessments, appropriate measures shall 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 should be transferred accordingly.

5.7.3 Graphical Programming and model-based Development

[A: 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 shall demonstrably apply state-of-the-art modelling guidelines for the entire modelling.

[I: KGAS_4163]

State-of-the-art modelling guidelines are listed in KGAS_3908.

[A: KGAS_3886]

All deviations from the applied modeling guideline(s) (KGAS_3862) shall 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 shall be verified by review.

[A: KGAS_4131]

The review (KGAS_3313) of a model element shall also include compliance with modeling guidelines that cannot be automatically verified.

[I: KGAS_3314]

For each model element a description should 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

[I: 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 shall apply appropriate model metrics with defined boundaries for graphical programming.

[A: KGAS_3866]

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

[A: KGAS_4064]

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

[A: KGAS_3902]

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

[A: KGAS_4106]

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

[A: KGAS_3867]

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

5.7.4 Qualification of Tools

[I: KGAS_3117]

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

[A: KGAS_4164]

For non-FUSI-relevant and non-security-relevant deliverables, each tool shall be qualified by an appropriate verification method (e.g. review, test, validation tool) to ensure that the generated work products has been created correctly with regard to the generation rules.

[A: KGAS_3481]

Manufacturer information (e.g. manuals, guidelines, erratas) of each software-based tool shall 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) shall be created.

[I: KGAS_3334]

The test plan should include project specific test goals.

[I: KGAS_3335]

The test plan should 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 shall 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 shall fulfil the requirements of ISO/IEC/IEEE 29119 (KGAS_3479).

[A: KGAS_54]

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

[A: KGAS_3359]

Possible boundary values shall 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 should be assessed (e.g. check if it is atomic). If the requirement cannot be improved, an appropriate structuring of the test cases should be used.

5.8.3 Test Execution in general

[A: KGAS_3370]

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

[A: KGAS_3372]

The used test environment shall 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 shall analyze the associated risks and communicate them to the Contracting authority.

5.8.4 Software Unit Test

[A: KGAS_3376]

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

[A: KGAS_4165]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3376, sufficient coverage of the detailed software specification shall be demonstrated using defined criteria. For example, results of the static code analysis can be used for this purpose.

[A: KGAS_3377]

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

[A: KGAS_4166]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3377, sufficient branch coverage (C1) of the source code shall be demonstrated using defined criteria. For example, results of the static code analysis can be used for this purpose.

[A: KGAS_3378]

Deviations of the 100% branch coverage required in KGAS_3377 shall 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 shall be statically verified.

5.8.5 Software Integration Test

[A: KGAS_3502]

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

[I: KGAS_3383]

Software integration tests should 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) shall 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 shall be determined, documented and verified against the resource consumption objectives.

[A: KGAS_4171]

If the scope of delivery is exclusively an APP development (see KGAS_4150), the maximum and average resource consumption of the APP for every supported operating system and for every release must be determined, documented and checked against the resource requirements in deviation from KGAS_3637.

5.8.6 Software Verification

[A: KGAS_3503]

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

5.8.7 System Integration Verification

[A: KGAS_3619]

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

5.8.8 System Verification

[A: KGAS_3506]

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

5.9 Quality Assurance and Management

[I: KGAS_4174]

Quality management describes the systematic planning and control of processes with a view to their quality.

[I: KGAS_4175]

Quality assurance is the assurance of quality requirements for a product.

5.9.1 Quality Management

[A: KGAS_53]

The quality management of the Contractor shall be independent of the development of the product in terms of personnel and organization.

5.9.2 Quality Assurance

[A: KGAS_2904]

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

[A: KGAS_3129]

The quality assurance goals shall be measurable.

[A: KGAS_2911]

The quality assurance of the Contractor shall 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 shall have the technical qualifications to be able to confirm that the reviews have been carried out properly (content-related and formal).

5.9.2.1 Review of Work Products

[A: KGAS_2941]

The reviews shall 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

5.9.2.2 Verification of Development Processes

[A: KGAS_3477]

Compliance with all processes shall be verified regularly by the Contractor quality assurance, at least every three months.

[A: KGAS_2922]

The Contractor shall 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 shall be reproducible and recoverable.

[A: KGAS_3759]

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

5.11 Problem Resolution Management

[A: KGAS_3608]

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

[A: KGAS_3417]

Problem descriptions shall 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 shall be systematically analyzed down to their cause of failure.

[A: KGAS_4133]

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

[A: KGAS_4134]

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

5.12 Third Party Software

[A: 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) shall 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 shall 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 shall be marked in the software architecture specification.

[A: KGAS_3531]

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

[A: KGAS_3437]

For all third party software elements, the original requirements on which those third party elements had been developed shall 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 shall 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.

[A: KGAS_3446]

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

[A: KGAS_4167]

If the scope of delivery solely includes an APP development (KGAS_4150), then, in deviation from KGAS_3446, it shall still be possible to test the entire software using suitable test methods.

[A: KGAS_3923]

The Contractor is solely responsible for ensuring that the use of the delivered software is permitted in accordance with the contract and intended provisions and documents this to the Contracting authority.

5.13 Free and Open Source Software

[A: 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 should 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.

[A: 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 (see also KGAS_3923).

[A: KGAS_4097]

If and insofar as proprietary software is linked to software components licensed under GNU Lesser General Public License v2.1 (LGPL-2.1), the Contractor hereby grants the Contracting authority the transferrable and sub-licensable right to modify proprietary software contained in the contractual products for its own use and to carry out reverse engineering for the purpose of debugging such processing.

[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 shall 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 shall 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 shall 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 shall 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 shall 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 shall provide the Contracting authority with the details, materials, documents and results of the analysis carried out (KGAS_3824).

[A: 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 (see also KGAS_3923).

[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 shall 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

[A: 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 shall 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 should 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 shall be restored and the Contracting authority is to be informed.

5.14.4 Cybersecurity Risk Analysis

[A: KGAS_4141]

The Contractor shall 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 shall be identified as an asset.

[A: KGAS_3741]

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

[A: KGAS_3974]

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

[A: KGAS_3743]

For each threat identified in a cybersecurity risk analysis, the risk shall 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 shall consider identified and/or specified cybersecurity requirements in the risk analysis.

[A: KGAS_3750]

Cybersecurity controls shall 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 shall be updated accordingly.

[A: KGAS_3980]

Identified weaknesses shall 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 shall 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 shall be validated before being processed.

[A: KGAS_3758]

Error messages, log records and diagnostic records shall 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 shall identify suitable sources for identifying weak points and check them against them.

[A: KGAS_3957]

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

[A: KGAS_3930]

The Contractor shall define and use appropriate sources for the identification of vulnerabilities (see KGAS_4168).

[I: KGAS_4168]

Suitable sources for identifying vulnerabilities are CWE (KGAS_3721), CVE (KGAS_3904), reports of the Contractor and comparable.

[A: KGAS_3761]

All architectural elements that do not fulfill a functional aspect shall be identified (e.g. test interfaces). These potential gateways shall 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 shall apply cybersecurity coding guidelines.

[A: KGAS_3772]

The Contractor shall 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 shall be ensured that no unwanted access (Backdoors) are implemented.

[A: KGAS_3764]

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

[A: KGAS_3765]

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

5.14.8 Cybersecurity Case

[A: KGAS_3775]

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

[A: KGAS_3931]

The cybersecurity case shall 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 shall be continuously updated until delivery of the series product.

[A: KGAS_3776]

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

[A: KGAS_3817]

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

[A: KGAS_3983]

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

[A: KGAS_3819]

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

[A: KGAS_3932]

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

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

[A: KGAS_3874]

The Contractor shall 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 shall 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 shall support the mechanisms and standards used by VW for e-mail encryption in accordance with IT Security Guidelines.
(see KGAS_4087, KGAS_4088 and KGAS_4089)

[A: KGAS_3877]

The Contractor shall 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 shall 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) shall 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 shall 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 shall 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 shall 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, shall be communicated to the appropriate cybersecurity incident management of the Volkswagen Group (KGAS_3890).

[A: KGAS_3989]

The analysis shall be in accordance with 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 shall be provided on request and shall 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 shall 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 shall 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 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

[A: KGAS_3488]