Metriken-getriebene Prozessentwicklung im Rahmen zukünftiger vernetzter Mobilität

Ein Framework zur Risikobewertung von Fahrzeug-Funktionen

Christopher Kugler[1], German Baca Espinoza[2], Ralf Maquet[2], Jiju Vadakkepattath[2], Dirk Macke[2], Johannes Richenhagen[2], Stefan Kowalewski[1]

1 Lehrstuhl Informatik 11 – Embedded Software, RWTH Aachen University
Ahornstraße 55, D-52074 Aachen
ckugler@embedded.rwth-aachen.de, kowalewski@embedded.rwth-aachen.de
2 FEV Europe GmbH, Neuenhofstraße 181, D-52078 Aachen
richenhagen@fev.com, baca@fev.com, maquet@fev.com, macke@fev.com, vadakkepatta@fev.com

Mobilität verändert sich disruptiv: Neue Themenkomplexe wie Fahrzeugvernetzung und hochintelligente Assistenzsysteme stellen hohe Qualitätsanforderungen an SW-Produkte bei steigender Komplexität. Folglich müssen bestehende Entwicklungsprozesse kontinuierlich verbessert werden, um dem Zeit- und Kostenrahmen in Projekten gerecht zu werden.

In dieser Arbeit werden qualitative, metriken-basierte Meilensteine definiert, die einen reibungslosen Übergang zwischen Entwicklungsphasen sicherstellen sollen. Ein Rahmenwerk zur Risikobewertung von Fahrzeug-Funktionen wird vorgeschlagen, welches im Sinne des risikobasierten Testens zur Steuerung des Testumfangs genutzt werden kann. Es werden Risikofaktoren identifiziert, die eine Anwendung im Kontext vernetzter Mobilität erlauben. Zuvor definierte Metriken aus frühen Entwicklungsphasen fließen in die Bewertung ein und gewährleisten ein Mindestmaß an Objektivität. Das Rahmenwerk wird anhand einer Fallstudie evaluiert und der operative Mehrwert aufgezeigt.

Keywords: Vernetzte Mobilität, Risikobasiertes Testen, Risikobewertung, Key Performance Indikator (KPI), Vehicle-to-Everything (V2X).

1 Einführung

Hohe Komplexität und Qualitätsanforderungen, sowie vergleichsweise geringe Kosten- und Zeitbudgets sind Herausforderungen bei der Entwicklung von automobilen Steuerungssystemen. Insbesondere eine steigende Anzahl von Anforderungen für Fahrerassistenzsysteme (Advanced Driver Assistance Systems - ADAS) und für automatisiertes Fahren (Automated Driving -AD) erfordern technologische Innovationen und einen agilen Entwicklungsprozess. Vor allem im interdisziplinären Bereich der Software Entwicklung bringen vernetzte Steuergeräte neue Themengebiete wie Cyber-Security und Robustheit gegenüber externen Störfaktoren ins Spiel.

Eine besondere Rolle kommt automobilen Dienstleistern zu, die auf Grund von stetig wechselnden Projektrahmenbedingungen Methoden entwickeln müssen, um auf sich kurzfristig ändernde Anforderungen reagieren zu können. So wird regelmäßig die Nutzung externer Prozessabläufe und Tools vorausgesetzt. Entsprechend schwierig wird die Definition und Verankerung geeigneter Prozesse beim Dienstleister, die

dabei helfen können, den Entwicklungshub möglichst effizient zu gestalten.

Eine bewährte Praktik, um prozessseitige Flexibilität zu ermöglichen und wettbewerbsfähige Produkte zu entwickeln, ist der Einsatz von Qualitätsschranken (eng.: Quality Gates) im Entwicklungsprozess. Die grundlegende Idee ist mit dem Prinzip einer Fabrik vergleichbar (Abb. 1), wo ein folgender Verarbeitungsschritt erst startet, nachdem die definierte Qualität des Werkstücks sichergestellt worden ist.

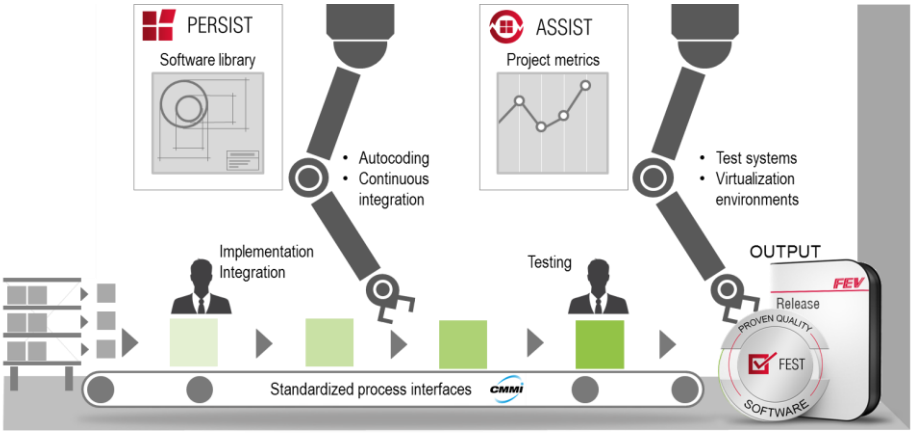

Abb. 1: Standardisierte Prozessschnittstellen für eine effizientere Entwicklung.

Zur Definition geeigneter Qualitätsschranken im Kontext automobiler Softwareentwicklung bieten sich eine Vielzahl von Metriken an, deren Aussagekraft allerdings von unternehmensspezifischen Strukturen und Zielen abhängt.

In den folgenden Kapiteln werden zunächst beispielhaft Qualitätsschranken für frühe Entwicklungsphasen – der Anforderungsdefinition und der Architekturentwicklung – eingeführt und deren Nutzen erörtert. Im Anschluss wird ein Rahmenwerk zur Risikobewertung von Fahrzeug-Funktionen vorgeschlagen, das auf diesen Metriken aufbaut und einen wesentlichen Beitrag zur Praxis der risikobasierten Qualitätssicherung leisten kann.

2 Metriken-getriebene Softwareprozesse

Metriken-getriebene Prozesse sind ein effektives Mittel, um typische Herausforderungen der Softwareentwicklung beherrschbar zu machen. Folgende Anforderungen an einen Prozess, die eine hohe Wertigkeit bedingen, sollen herausgestellt werden:

i. Agile Methoden in der Prozessausübung mit wohldefinierten Schnittstellen zu Kundenprozessen
ii. Mess- und bewertbare Prozessergebnisse
iii. Risikobewusste Steuerung der Prozesse (Robustheit gegen äußere Einflüsse, Flexibilität im Prozess, um (sich ändernde) Schwerpunkte zu setzen)
iv. Geeignete Komplexitätsreduktion der vorliegenden Aufgabe (beispielsweise durch Unterteilung in mehrere Schritte)

Im Kontext von Qualitätsschranken liegt hier vor allem Punkt ii) im Fokus, wobei Kapitel 3 mit dem risiko-basierten Rahmenwerk im Wesentlichen Punkt iii) adressiert.

Zur Gewährleistung einer hohen Softwarequalität ist es sinnvoll, bereits Artefakte aus frühen Entwicklungsphasen nach strengen Qualitätskriterien zu bewerten um so potentiellen zukünftigen Problemen vorzubeugen. Dies betrifft vor allem funktionale und nicht-funktionale Anforderungen, die als Grundlage der Software-Entwicklung dienen. Ein weiterer zentraler Aspekt ist die strukturelle Auslegung des Systems im Rahmen einer Software-Architektur und die damit einhergehende Zerlegung in Software-Komponenten, welche maßgeblichen Einfluss auf notwendige Schnittstellen und umzusetzende Funktionslogik nimmt.

2.1 Qualitätsschranken für Anforderungen

Qualitätsschranken können über aussagekräftige Metriken, sogenannte Key Performance Indikatoren (KPIs), realisiert werden. Im Bereich der Spezifikation und Entwicklung von Anforderungen sind in der Literatur vielfältige KPIs identifiziert und erörtert worden (vgl. [SM07], [He10]). Angelehnt an diese werden in [No17] drei wesentliche Anforderungsaspekte betrachtet, die eine qualitative Aussage über entstehende Anforderungsdokumente ermöglichen und als Basis für weitere Ausführungen in diesem Beitrag dienen.

1. Anforderungsreife - *beispielsweise bewertet über:*
   a. Erfüllungsgrad der SOPHIST Anforderungsschablone [Ge08]
   b. Anforderungs-Komplexität
2. Anforderungsabdeckung – *im Wesentlichen bewertet über:*
   a. Abdeckungsgrad der Kundenanforderungen (alternativ: Konvergenzrate zwischen Lasten- und Pflichtenheft)
3. Anforderungstesttiefe – *hier nicht im Fokus, da Ursprung in späteren Entwicklungsphasen*

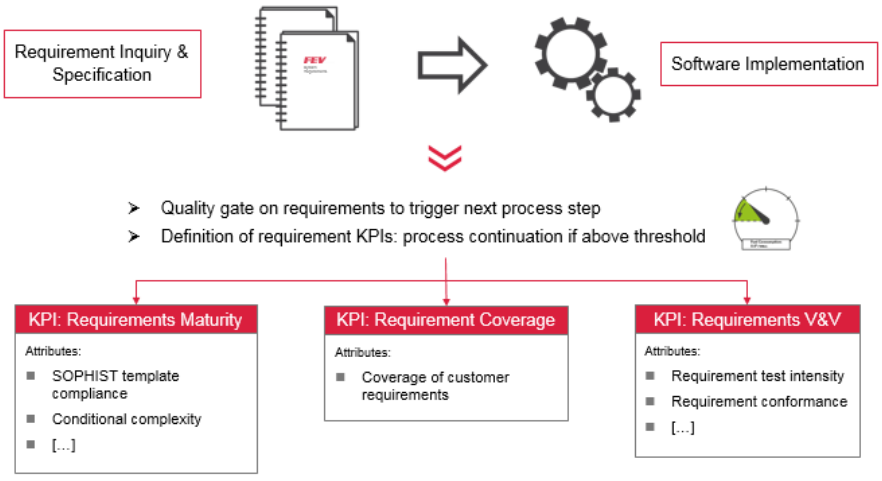

Abb. 2: Qualitätsschranken für Anforderungen.

Detaillierte Definitionen und weitere Ausführungen zu den genannten Metriken sind

in [No17] zu finden. Abb. 2 veranschaulicht dies grafisch. Die Metriken können regelmäßig für Anforderungsdokumente erhoben werden und dabei helfen, deren Qualität zu bewerten. Der Start weiterer Entwicklungsschritte, die auf diesen Anforderungen basieren ist nur dann sinnvoll, wenn in den definierten KPIs ein vorgegebener Schwellwert überschritten wird.

2.2 Qualitätsschranken für Architekturartefakte

Ein zentrales Element der frühen Software-Entwicklung ist die Software-Architektur und die damit verwandten Artefakte (Schnittstellendefinition, Software-Komponenten). Hier besteht die Herausforderung darin, die Komplexität der Architektur und auch der Komponenten möglichst gering zu halten, um Wartbarkeit, Testbarkeit und Analysierbarkeit zu gewährleisten. In [Ve17] werden diesbezüglich verschiedene Komplexitätsmetriken für Software-Komponenten erörtert und verglichen.

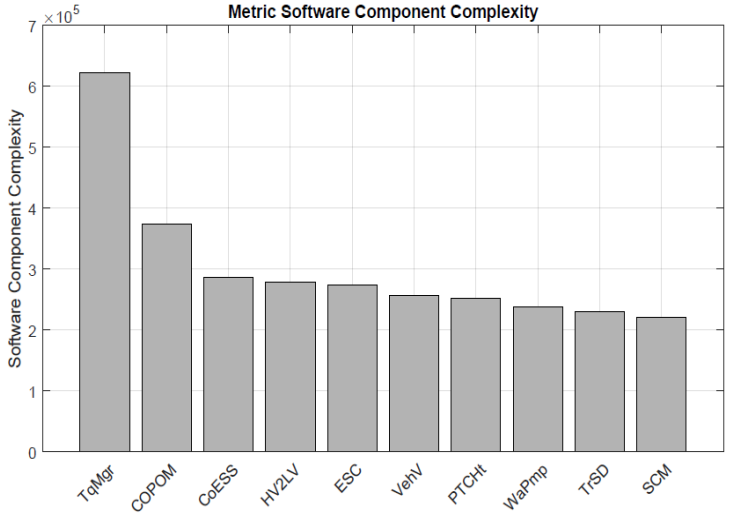

Abb. 3: Software-Komponenten Komplexität für verschiedene Komponenten einer Hybrid-Steuerung [Ve17].

Im Fokus stehen die Anzahl der externen und internen Schnittstellen einer Komponente und die Anzahl der in einer Komponente jeweils umgesetzten Funktionen. Diese Informationen werden schlussendlich zu einem KPI „Software Component Complexity“ aggregiert (vgl. Abb. 3). In der Entwicklung kann nun – basierend auf unternehmensspezifischen Parametern – ein Schwellwert für die maximal tolerierte Komplexität festgelegt werden. Architekturelemente, die diesen Schwellwert überschreiten, müssen hinsichtlich Modularität untersucht und weiter separiert werden.

3 Metriken-basiertes Rahmenwerk zur Risikobewertung von Fahrzeug-Funktionen im Kontext der Qualitätssicherung

Die Grundidee der risikobasierten Qualitätssicherung ist die Erkenntnis, dass in der Realität unter gegebenen Projektrahmenbedingungen eine 100%ige Absicherung (Verifikation & Validierung) des Produkts nicht möglich ist [PR12], [Si11]. Daher wird auf ein Priorisierungsschema zurückgegriffen, das sich im Wesentlichen auf bezugsspezifische Risikowerte stützt. Abb. 4 beschreibt dieses Vorgehen.

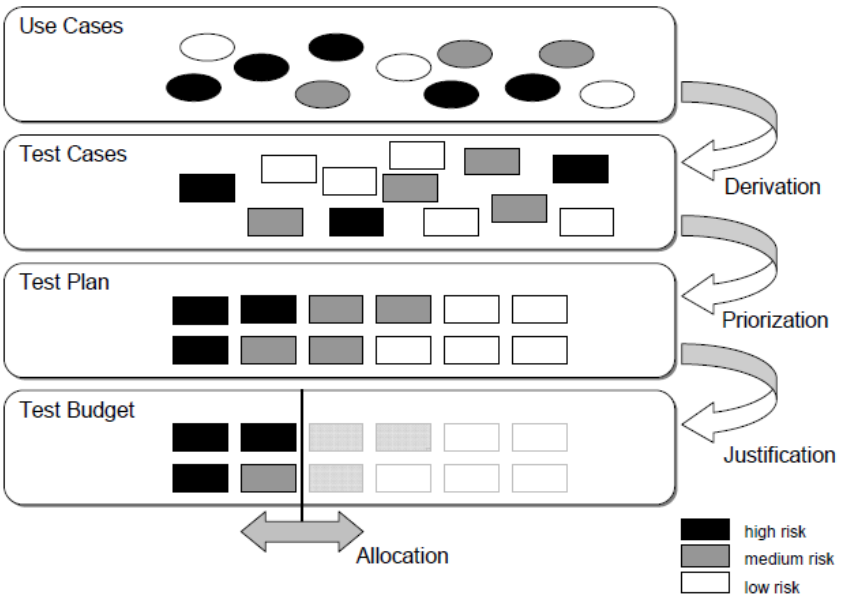

Abb. 4: Risikobasierte Test-Priorisierung basierend auf verfügbarem Test Budget [SM07].

Ansätze, den Prozess der Risikobeurteilung über den Einsatz von spezifischen Kenn- und Maßzahlen zu objektivieren, werden unter anderem in [No00], [Be99] und [SM07] untersucht. Benlarbi et. al. beschäftigen sich im Wesentlichen mit Code Metriken [Be99], wohingegen Nogueira et. al. ausschließlich Projektrisiken untersucht [No00], welche nicht im Fokus dieses Beitrags liegen. Die hier im Folgenden betrachteten Risiken sind produktbezogen und treffen primär eine Aussage über technische Eigenschaften der Betrachtungseinheit. Ein Ansatz, der dem Vorgehen in diesem Beitrag ähnelt, ist durch Stallbaum et. al. in [SM07] erörtert worden: Hier werden Anforderungsmetriken zur automatisieren Risikobewertung von Anforderungsdokumenten, beispielsweise Use-Case Definitionen, herangezogen. Im Vergleich gibt es drei wesentliche Unterschiede zu dem hier vorgestellten Rahmenwerk:

1. Unterschiedliche Betrachtungseinheit: Fokus auf systemweiten Funktionen anstatt auf einzelne Use-Cases.
2. Einbezug von Architekturmetriken als zusätzlicher Bewertungsfaktor.
3. Einbezug von qualitativen Bewertungsfaktoren (Expertenbeurteilung).

Das Ziel des hier beschriebenen Rahmenwerks ist die Herleitung von funktionsspezifischen Risikowerten basierend auf den in Kapitel 2 vorgestellten Metriken, sowie weiteren ad-hoc definierten Einflussfaktoren. Dazu wird in einem ersten Schritt definiert, hinsichtlich welcher Eigenschaften eine Funktion untersucht und bewertet werden soll. Im Anschluss werden Bewertungskriterien für diese Eigenschaften festgelegt und passende Wertebereiche definiert.

Exemplarisch wird in Kapitel 3.3 eine Evaluation des Rahmenwerks anhand von Funktionen aus dem Umfeld vernetzter Mobilität (Vehicle-to-Everything, V2X) skizziert.

3.1 Bewertungsschema für Funktionen

Zur Definition der für die Qualitätssicherung relevanten Eigenschaften einer Funktion kann das Software-Qualitätsmodell der ISO 25010 herangezogen werden (vgl. Abb. 5).

Abb. 5: ISO 25010 Software Qualitätsmodell [IS11].

Angelehnt an diese 8 Qualitätsattribute lässt sich der Risikowert einer Funktion über folgende Eigenschaften charakterisieren:

i. Funktionale Komplexität *(Functional Suitability)*
ii. Performanz-Relevanz *(Performance Efficiency)*
iii. Nutzbarkeits-Relevanz *(Usability)*
iv. Sicherheits-Relevanz *(Security, Reliability)*
v. Interoperabilitäts-Relevanz *(Compatibility, Portability)*

Für diese Eigenschaften gilt: Je höher der jeweilige Risikowert, desto höher ist das mit der Funktion assoziierte Risiko. Der methodische Ansatz ist also, Funktionen in diesen Eigenschaften unabhängig voneinander zu bewerten, um damit auf den Risikowert der Funktion als Ganzes Rückschlüsse ziehen zu können. Im Bereich der Qualitätssicherung können die individuellen Bewertungen dazu genutzt werden, die entsprechenden Eigenschaften über geeignete Testmethoden abzusichern und dabei gezielt Schwerpunkte zu setzen. So sind beispielsweise Ergonomie Tests nur dann notwendig, wenn auch eine Nutzbarkeits-Relevanz vorliegt.

3.2 Herleitung von spezifischen Risikofaktoren

Um die in 3.1 eingeführten Eigenschaften funktionsspezifisch bewerten zu können, müssen Bewertungskriterien und ein Aggregationsschema definiert werden. Für die Bewertungskriterien werden unter anderem die in Kapitel 2 eingeführten Metriken herangezogen. Tabelle 1 veranschaulicht die Komposition anhand der wesentlichen Bewertungskriterien.

Die Bewertungsskalen der einzelnen Kriterien sind nach Möglichkeit binär gewählt; in Einzelfällen werden kennzahlspezifische Skalen verwendet, die im Anschluss auf einen Wertebereich von 0 bis 1 normiert werden. Als Aggregationsschema wird das gewichtete arithmetische Mittel verwendet.

Betrachtete funktionale Eigenschaft	***Bewertungskriterien***	***Erläuterung***
Funktionale Komplexität	Anforderungs-Komplexität	Bewertung über Satzbau von textuellen

		Anforderungen (vgl. 2.1)
	Anforderungsaufkommen	Anzahl an funktionalen Anforderungen an eine Fkt.
	Durchschnitt. SW-Komponenten Komplexität	Einbezug aller SW-Komponenten, die an Umsetzung der Funktion beteiligt sind
	Maximale SW-Komponenten Komplexität	
Performanz-Relevanz	Existenz Echtzeitanforderungen	Binäre Bewertung (Ja/Nein)
	Zeitkritisches Funktionsverhalten	Binäre Bewertung (Ja/Nein)
Nutzbarkeits-Relevanz	Direkte Kundenschnittstelle	Binäre Bewertung (Ja/Nein)
Sicherheits-Relevanz	ASIL Rating	Entsprechend ISO 26262
	Funktion Teil der V2X Kommunikation	Binäre Bewertung (Ja/Nein)
	Einsatz von Verschlüsselung	Notwendigkeit Verschlüsselung auf Ebene Kommunikationsprotokoll; Binäre Bewertung (Ja/Nein)
Interoperabilitäts-Relevanz	Einsatz in verschiedenartigen Umgebungen	Mögliche Differenzierung: verschiedene Steuergeräte, verschiedene Fahrzeugarchitekturen, ...
	Varianz in Schnittstellen	Binäre Bewertung (Ja/Nein)

Tabelle 1: Exemplarische Bewertungskriterien für funktionale Eigenschaften.

3.3 Evaluation anhand V2X Funktionen

Zur Evaluation des Bewertungsschemas für den Risikowert einer Funktion werden exemplarisch die notwendigen Rahmenbedingungen für zwei V2X Funktionen definiert und die Ergebnisse in den jeweiligen Kriterien miteinander hinsichtlich Plausibilität verglichen. Eine inhaltliche Beschreibung der betrachteten Funktionen findet sich im Folgenden:

Funktion A: Firmware over the Air (FOTA)
Um Software-Updates von Fahrzeug-Steuergeräten ohne Werkstattbesuch zur ermöglichen, können ausgewählte Steuergeräte außerhalb des Betriebs über ein Kommunikationsprotokoll nach geeigneter Authentifizierung durch den Kunden ferngesteuert aktualisiert werden.
Kernanforderungen an diese Funktion sind:

i. Robustheit gegenüber Unterbrechungen (z.B. Kommunikations- oder Stromausfall) und äußere Manipulationseinflüsse während des Flash-Vorgangs.
ii. Prüfung der zu aufzuspielenden Software auf Authentizität, bevor der Flash-Vorgang startet.
iii. Verifikation eines erfolgreichen Flash-Vorgangs über eine Prüfsumme (oder vergleichbar) und Feedback an den Kunden. Bei fehlgeschlagenem Flash-Vorgang, Möglichkeit eines Rollbacks zur alten Software.
iv. Kompatibilität des Flash-Vorgangs mit verschiedenartigen Steuergeräten. Voraussetzung ist die Umsetzung eines einheitlichen Flash-Protokolls.

Funktion B: Cooperative Collision Avoidance (CoCA)
Zur verbesserten Kollisionsvermeidung können Fahrzeuge untereinander laterale und longitudinale Kontrollparameter über einen Kommunikationsstandard, z.B. 5G Mobilfunk, austauschen und diese in Echtzeit auswerten. Es erfolgt eine Warnmeldung an den Fahrer und situationsabhängig ein Brems- oder Lenkeingriff, wenn Gefahrensituationen erkannt werden.
Kernanforderungen an diese Funktion sind:

i. Fahrzeug-zu-Fahrzeug (V2V) Kommunikation in Echtzeit; unidirektionale Latenz von weniger als 10ms [3G16].
ii. Robustheit gegen äußere Störeinflüsse inklusive Manipulation von Daten (umsetzbar beispielsweise über verschlüsselte Kommunikation).
iii. Eindeutige Identifizierung der Fahrzeuge gegeneinander.
iv. 99,999%ige Zuverlässigkeit bei sicherheitsrelevanten Fahrmanövern [3G16].

In Tabelle 2 findet sich die Bewertung beider Funktionen anhand der definierten Bewertungskriterien.

Betrachtete funktionale Eigenschaft	***Bewertungskriterien***	***Funktion A FOTA***	***Funktion B CoCA***	***Gew. Arithm. Mittel A***	***B***
Funktionale Komplexität	Anforderungs-Komplexität	0.5	0.6	0.58	0.75
	Anforderungsaufkommen	0.4	0.7		
	Durchschnitt. SW-Komponenten Komplexität	0.6	0.8		
	Maximale SW-Komponenten	0.8	0.9		

	Komplexität				
Performanz-Relevanz	Existenz Echtzeitanforderungen	0	1	0	1
	Zeitkritisches Funktionsverhalten	0	1		
Nutzbarkeits-Relevanz	Direkte Kundenschnittstelle	1	0	1	0
Sicherheits-Relevanz	ASIL Rating	B	C	0.83	0.92
	Funktion Teil der V2X Kommunikation	1	1		
	Einsatz von Verschlüsselung	1	1		
Interoperabilitäts-Relevanz	Einsatz in verschiedenartigen Umgebungen	1	1	1	1
	Varianz in Schnittstellen	1	1		

Tabelle 2: Exemplarische Auswertung für V2X Funktionen.

Bei Berechnung des Eigenschaften-übergreifenden Risikowerts für die gesamte Funktion ergibt sich ein Wert von *(0.58+0+1+0.83+1)/5=0.68* für Funktion A, und entsprechend *(0.75+1+0+0.92+1)/5=0.73* für Funktion B.

Es ist jedoch zumindest im Kontext der Qualitätssicherung sinnvoller, die einzelnen Eigenschaften individuell zu betrachten. So ist aus den Bewertungen zum Beispiel ableitbar, dass Ergonomie Tests bei der Funktion *FOTA* sinnvoll sind - auf Grund der notwendigen Authentifizierung durch den Kunden und anschließendem Feedback bei abgeschlossenem Flashvorgang -, bei der Funktion *CoCA* allerdings im Umfang wesentlich eingeschränkt werden können. Des Weiteren ist aus den höheren Werten der *CoCA* Funktion bei Funktionaler Komplexität und Sicherheits-Relevanz ersichtlich, dass die Testintensität und damit der Testumfang für diese Funktion vergleichsweise höher angesetzt werden muss.

4 Zusammenfassung und Ausblick

Im Rahmen dieses Beitrags wurde zunächst der Bedarf für metriken-getriebene Prozesse erörtert und im Anschluss geeignete Qualitätsschranken für die Anforderungs- und Architekturentwicklung definiert. Anschließend wurde ein Rahmenwerk zur Risikobewertung von Fahrzeugfunktionen vorgeschlagen, das die zuvor definierten Metriken aufgreift und in die Risikobewertung einbezieht. Somit wird eine höhere Aussagekraft erreicht und der Prozess der Risikobewertung anteilig objektiviert. Auf Basis der Risikowerte kann im Anschluss der Umfang und der Schwerpunkt der Qualitätssicherungs-Maßnahmen sinnvoll abgeschätzt und gesteuert werden.

Die exemplarischen Ergebnisse sind vielversprechend und die hier skizzierte Methodik soll zukünftig weiter evaluiert werden. Dazu werden im Rahmen der Qualitätssicherung unternehmensspezifische Risikogrenzschwellen definiert, die unmittelbaren Einfluss auf Testumfang und –intensität haben.

Weiter ist geplant, Metriken aus späteren Software-Entwicklungsphasen, beispielsweise der Systemverifikation, in die Risikobewertung miteinzubeziehen und damit eine iterative Annäherung während der Projektlaufzeit zu ermöglichen. Denkbare Einflussfaktoren sind beispielsweise funktionsspezifische Fehlerraten und die Stabilität der Anforderungen – gemessen über eingehende Änderungsanfragen.

Bibliographie

[SM07] Stallbaum, H.; Metzger, A.: Employing Requirements Metrics for Automating Early Risk Assessment. Proceedings of MeReP07, 2007.

[He10] Heidenreich, M.: Metriken und Werkzeugunterstützung zur Überprüfung von Anforderungen. OBJEKTspektrum, Ausgabe RE, 2010

[No17] Nowack, J. et. al.: Continuous Process for the Validation of Transmission Controls, CTI Symposium Shanghai, *To appear*

[Ge08] Geltinger, G. et. al.: Das SOPHIST Regelwerk, DHBW Ravensburg, 2008

[Ve17] Venkitachalam, H. et. al.: Metric-based Evaluation of Powertrain Software Architecture, SAE Int. J. Passeng. Cars – Electron. Electr. System., S. 194-208, 2017

[PR12] Veenendaal, Erik van: The PRISMA Approach, Uitgeverij Tutein Nolthenius, 2012

[Be99] Benlarbi, S. et. al.: Issues in Validating Object-Oriented Metrics for Early Risk Prediction. Proceedings of 10th ISSRE, International Symposium on Software Reliability Engineering, Flordia, USA, 1999

[No00] Nogueira, J.C. et. al. : A Formal Risk Assessment Model for Software Evolution. Proceedings of the 22nd ICSE, International Conference on Software Engineering, 2000

[Si11] Siller, A. et. al.: Systematisch zu Testfällen – Anforderungsbasierte Strategie für Test und Absicherung von Elektrik/Elektronik. Automotive 2011

[IS11] ISO/IEC 25010:2011: System and software quality models, http://www.iso.org/iso [Oktober 2017]

[3G16] 3GPP TR 22.886 V15.0.0, Study on enhancement of 3GPP Support for 5G V2X Services, Release 15, 2016