Modellbasierte Sicherheitsanalysen im BMBF-Förderprojekt e performance

Rasmus Adler[1], Sören Kemmann[1], Markus Schurius[2], Dr. Christian Allmann[3]

[1]Fraunhofer IESE, Abteilung ESQ, Fraunhofer-Platz 1, 67663 Kaiserslautern
[2]Audi Electronics Venture GmbH, I/AEV-23, Sachsstr. 18, 85080 Gaimersheim
[3]AUDI AG, I/PG-EE, 85045 Ingolstadt

Abstract: Der Artikel behandelt aktuelle Herausforderungen, die in dem Spannungsfeld von Funktionaler Sicherheit und der Integration von software-intensiven Fahrfunktionen bei verteilter Entwicklung entstehen. Er stellt Safety Engineering Methoden aus der Wissenschaft vor und beschreibt, wie diese Methoden erfolgreich in dem BMBF-Förderprojekt *e performance* angewendet wurden. In diesem Projekt entwickelt Audi mit mehreren Partnern aus Industrie und Forschung ein Elektrofahrzeug. Der Artikel schildert die zugrundeliegende Motivation, beschreibt das allgemeine Vorgehen und stellt erzielte Ergebnisse am Beispiel des elektrifizierten Antriebsstrangs vor.

1 Motivation

In der Vergangenheit wurde einer software-basierten Fahrzeugfunktionalität – wie beispielsweise der Längsdynamikregelung – jeweils eine dedizierte Rechnerplattform mit eigener Sensorik und Aktorik zugeordnet. Da diese Lösung aufgrund von Bauraum- und Energielimitierung die Grenzen der Skalierbarkeit erreicht hat, teilen sich bereits heute mehrere Software-Systeme die selben Ressourcen und können nur noch bedingt unabhängig voneinander entwickelt werden.

Um eine verteilte Entwicklung zwischen einem Automobilhersteller und seinen Lieferanten zu ermöglichen, müssen die Fahrzeugfunktionalitäten in interagierende Funktionen partitioniert werden. Diese Partitionierung wird durch die Funktionsarchitektur des Gesamtsystems beschrieben, die wiederum mit Sprachen wie SysML [We08] oder EAST-ADL [DST05] in Modellen dargestellt werden kann. Sie ermöglicht es, die komplexen Zusammenhänge von vernetzten Funktionen besser zu verstehen, dient als Kommunikationsmittel zwischen den beteiligten Parteien in der verteilten Entwicklung und schafft durch Informationsstrukturierung die Grundlage für die Gewährleistung von Funktionaler Sicherheit. Funktionale Sicherheit gemäß ISO 26262 [1] ist eine Systemeigenschaft, die garantiert, dass die Risiken sowohl durch Fehler bei der Umsetzung der spezifizierten Systemfunktionalitäten als auch durch zufällige HW-Fehler akzeptabel sind.

Safety Engineering und die Entwicklung der Funktionsarchitektur stehen in einer engen Beziehung zueinander: Die Funktionsarchitektur ist die Basis für die Gewährleistung der Funktionalen Sicherheit. Jede Änderung an der Architektur hat potentiell Einfluss auf die Funktionale Sicherheit. Andererseits haben Entwicklungsartefakte, die im Sicherheitslebenszyklus entstehen direkt Auswirkungen auf die Architektur.

Fehlererkennungs- und Fehlerbehandlungsmechanismen führen beispielsweise zu neuen Funktionen, die ebenfalls in der Architektur abgebildet und analysiert werden. Aufgrund der steigenden Komplexität von Funktionsarchitekturen wird es daher immer aufwändiger, die Konsistenz zwischen der Funktionsarchitektur und den Safety-Artefakten in Form von Sicherheitsanalysen und -konzepten zu garantieren. Herkömmliche Safety Engineering Methoden aus der Praxis stoßen an ihre Grenzen. Betrachtet man beispielsweise eine traditionelle Fehlerbaumanalyse (engl. Fault Tree Analysis, FTA), dann bezieht sich der resultierende Fehlerbaum auf einen bestimmten Stand der Funktionsarchitektur. Änderungen an der Funktionsarchitektur ziehen oftmals eine entsprechende Modifikation im Fehlerbaum nach sich. Dies ist sehr aufwendig da es im Allgemeinen nicht klar ist welche Teile des Fehlerbaums wie angepasst werden müssen. Die dazu benötigte Änderungseinflussanalyse, wird zudem durch eine unvollständige Werkzeugintegration zwischen den Disziplinen System bzw. Software Engineering und konventionellem Safety Engineering erschwert.
Die gleiche Problematik betrifft auch das funktionales Sicherheitskonzept, solange es wie bisher weitestgehend textbasiert erstellt wird und grafische Notationen wie beispielsweise Fehlerbäume oder Modellteile anderer Entwicklungswerkzeuge lediglich als Abbildungen zum besseren Verständnis inkludiert. Das funktionale Sicherheitskonzept legt Sicherheitsanforderungen für die Funktionen in der Funktionsarchitektur fest und erklärt warum die Sicherheitsanforderungen ausreichen um die Sicherheitsziele zu erreichen. Wenn die Funktionsarchitektur geändert werden muss, ist somit die Anpassung des Sicherheitskonzepts ebenfalls sehr zeitaufwendig.
Um diese Konsistenzprobleme zu lösen muss eine verbesserte Verfolgbarkeit zwischen der funktionalen Architektur, den Fehlerbäumen und dem funktionalen Sicherheitskonzept hergestellt werden. Dazu wird ein Ansatz benötigt bei dem alle sicherheitsrelevanten Informationen die im Projektverlauf entstehen in einem gemeinsamen Modell integriert werden können. Bisher sind diese noch häufig in separaten Entwicklungsartefakten abgebildet, die manuell verlinkt werden müssen. Ausgangspunkt ist dabei die modellbasierte Entwicklung, beziehungsweise die systematische Erfassung von Funktionen und deren Abhängigkeiten anhand logischer Schnittstellen in einem Modell. Mit der bruchfreien Integration von Sicherheitsanalysen in die modellbasierte Entwicklung wird eine explizite und durchgängige Sicherheitsargumentation ermöglicht. Desweiteren vereinfacht diese modellbasierte, iterative Vorgehensweise eine kohärente Entwicklung von teilautomatisiert erstellten Sicherheitskonzepten und modularen Architekturen. Zudem wird durch die Ausdrucksmächtigkeit von Modellierungssprachen, z.B. in Form von unterschiedlichen Diagrammtypen, die Sicherheitsfunktion exakter spezifiziert und kann in definierbaren Sichten auf das Modell leichter verstanden und weiterentwickelt werden. Damit steigt auch die Wiederverwendbarkeit von Entwicklungsartefakten.
Da dieses Verfahren auf der Analyse von Funktionslogik basiert ist es für die konventionellen, d.h. im Fahrzeug verteilten Funktionen in gleichem Maße geeignet wie für hochintegrierte Funktionen auf einem einzigen Steuergerät. Speziell für diesen zweiten Fall bildet es vielmehr die Grundlage um Verteilungsentscheidungen (HW/SW-Deployment) in Bezug auf das Ausführungsverhalten so zu spezifizieren dass es weder zu Verletzungen von Echtzeitbedingungen noch von Sicherheitseigenschaften führt. Um z.B. die Einhaltung von Ende-zu-Ende Fehlertoleranzzeiten einer Funktion zu untersuchen kann ein Timing-Modell wie in [Sc12] beschrieben in das hier vorgestellte,

modellbasierte Sicherheitsanalyse-Framework integriert werden. Ziel ist dabei die iterative Entwicklung von Lösungen, die unterschiedliche Aspekte berücksichtigen und eine kontinuierliche Optimierung der Funktions- und Steuergerätearchitektur erleichtern.

Der aktuelle Stand der Technik im Bereich sicherheitsgerichteter Entwicklung besitzt die folgenden Verbesserungspotentiale:

- Vermeidung von Brüchen in der Werkzeugkette (Sicherheitsanalyse vs. Modellierung) zur Verbesserung der Durchgängigkeit von sicherheitsrelevanten Informationen im Sicherheitslebenszyklus bzw. entlang des V-Modells
- Überwindung von organisatorischen Grenzen mittels Modularisierung und Integration von sicherheitsrelevanten Informationen in einem gemeinsamen Modell, sowohl hinsichtlich unterschiedlicher Rollen und Disziplinen innerhalb einer Firma als auch über die Schnittstelle zwischen OEM und Lieferanten hinweg
- Unterstützung einer effizienten und inkrementellen Bewertung des Sicherheitsnachweises

Die Akzeptanz der Anwender für eine intensivere, in diesem Falle phasen- und disziplinübergreifende Entwicklung auf einem gemeinsamen Informationsmodell ist vorhanden.

Dieser Artikel zeigt, wie innovative Safety Engineering Methoden genutzt werden können, um mit der steigenden Komplexität bei der Integration von Fahrzeugfunktionalitäten umzugehen. Er beschreibt zunächst einen modellbasierten Lösungsansatz, um die Komplexität software-intensiver, vernetzter Fahrfunktionen zu beherrschen. Anschließend wird das BMBF-Förderprojekt *e performance* vorgestellt und erklärt, wie darin Sicherheitskonzepte modellbasiert entwickelt wurden.

2 Lösungsansatz

Der Wunsch nach mehr Kundennutzen ist die Ursache für die steigende Anzahl und die Vernetzung von Fahrzeugfunktionen – eine Charakteristik, die speziell auf Fahrerassistenzsysteme zutrifft. Ein Mittel, um diese Komplexität zu beherrschen ist die Verwendung von modellbasierten Entwicklungsmethoden. Komplexitätsreduktion durch Abstraktion hilft systematische Fehler zu vermeiden und begünstigt somit die Funktionale Sicherheit. Aufgrund einer verbesserten Formalisierung der zu entwickelnden Funktionen wird modellbasierte Entwicklung von modernen Sicherheitsnormen wie der ISO 26262 empfohlen.
Um mit der steigenden Komplexität der verteilten Entwicklung von integrierten Fahrzeugfunktionen umzugehen, kann ein Funktionsmodell erstellt werden, das beschreibt, wie die Fahrzeugfunktionalitäten durch eine Komposition von Teilfunktionen realisiert werden. Modelle des Safety Engineering ermöglichen es hingegen mit den komplexen Zusammenhängen von sicherheitsrelevanten Aspekten umzugehen. Ein wesentlicher Punkt, der beim Safety Engineering berücksichtigt werden muss, ist die Ursache-Wirkungsbeziehung von Fehlern. In diesem Kontext muss insbesondere

analysiert werden, wie sich Fehler von Funktionen auf Fahrzeugfunktionalitäten auswirken. Hierfür werden häufig Fehlerbäume verwendet, in denen für jede zuvor ermittelte Gefährdung (in absteigender Reihenfolge) eine Baumstruktur entwickelt wird – bis schließlich alle möglichen Ursachen, die zum Eintreten der Gefährdung führen können, identifiziert sind. Die Baumzweige wiederum sind über logische Operatoren miteinander verknüpft. Durch Auswerten der Minimalschnitte können Ereigniskombinationen ermittelt werden, die im Falle ihres gleichzeitigen Eintretens zur Gefährdung bzw. zu einer Verletzung des Sicherheitsziels führen können.
Fehlerbäume und andere Safety-Modelle beziehen sich häufig auf das Funktionsmodell. Dieses spezifiziert die zu entwickelnden Funktionen und deren Zusammenhänge. Modelle für die Analyse von Fehlern in Funktionen untersuchen zum Beispiel fehlerhafte Abweichungen von Funktionsspezifikationen und deren Auswirkungen. Andere Safety-Modelle beschreiben funktionale Sicherheitsanforderungen bzw. sicherheitsrelevante Teile der Funktionsspezifikation. Auch wenn sich Safety-Modelle auf die Information im Funktionsmodell beziehen, werden sie traditionell unabhängig vom Funktionsmodell und mit separaten Werkzeugen erstellt. Gemeinsamkeiten werden somit redundant und möglicherweise inkonsistent zueinander modelliert.
Um dieses Defizit im Hinblick auf Fehlerbaumanalysen zu beseitigen, wurde in [DT08] ein Ansatz entwickelt wie Fehlerbäume in ein Funktionsmodell integriert werden können. Darin besitzt jede Komponente in Funktionsmodell einen eigenen Fehlerbaum. Dies ist nicht nur im Rahmen der verteilten Entwicklung ein großer Vorteil, denn Konsistenzprobleme zwischen Fehlerbaum und Funktionsmodell treten auch bei einer nicht verteilten Entwicklung auf. Um die Konsistenzproblematik auch bei der Sicherheitskonzeptentwicklung in den Griff zu bekommen wurde ein ähnlicher Ansatz in [Do09] entwickelt. In dem Ansatz hat jede Komponente in dem Funktionsmodell ein eigenes Sicherheitskonzept. Um die beiden Integrationsansätze zu kombinieren und noch weitere Safety-Modelle integrieren zu können wurde in [Ad11] ein Ansatz entwickelt, der es ermöglicht verschiedene Safety-Sichten für ein Funktionsmodell zu erstellen. Gemäß des Prinzips *Separation of Concerns* können die Sichten separat modelliert und evaluiert werden.
Der Stand der Technik bietet viele verschiedene Arten von Safety-Modellen, die man als Safety- Sichten in ein Funktionsmodell integrieren könnte. Die Modelle kann man gemäß ihrem Zweck als analytisch oder als konstruktiv klassifizieren. Analytische Modelle dienen der Ursachenanalyse. Gemäß Fenelon et al. [Fe94], kann man analytische Ansätze anhand ihrer Suchrichtung nach Ursachen (von Gefährdungen) charakterisieren. Hierbei werden explorative, induktive, deduktive und beschreibende Ansätze unterschieden. Um den Aspekt Software und Funktion adäquater zu adressieren, haben wir dieses Schema etwas angepasst und erweitert. So unterscheiden wir grob, HAZOP (engl. Hazard and Operability) Analysen [Pu99][Ch93][RCC99], reine Vorwärtsanalyseansätze wie FMEA (engl. Failure Mode and Effects Analysis), reine Rückwärtsanalyseansätze wie FTA, FMEA-FTA kombinierte bidirektionale Analyseansätze [Pa04][LW99][PD99], Inspektions-ähnliche sowie formale Ansätze [De09]. Die Anzahl der analytischen Modelle zur Ursachenidentifikation ist deutlich größer als die Anzahl der konstruktiven Safety-Modelle zur Beschreibung der Maßnahmen gegen identifizierte Ursachen und Gefährdungen. Der populärste konstruktive Ansatz zur Darstellung des Sicherheitsnachweises (engl. Safety Case) ist die Goal Structuring Notation (GSN) [Wi96]. Die GSN eignet sich auch bedingt zur

Modellierung von Sicherheitskonzepten. Der Sicherheitsnachweis nach ISO 26262 ist das finale Arbeitsergebnis im Sicherheitslebenszyklus und beinhaltet die Gesamtheit der Entwicklungsartefakte, die zur Beurteilung und zum Nachweis der Funktionalen Sicherheit herangezogen werden. Die wichtigsten Kriterien für den Sicherheitsnachweis sind Vollständigkeit, Widerspruchsfreiheit und Durchgängigkeit, wobei speziell im Automotivebereich zusätzlich Vergleichbarkeit sowie Modularität zunehmend an Bedeutung gewinnen. Der Sicherheitsnachweis wird jeweils für eine Betrachtungseinheit erstellt, beispielsweise für eine sicherheitsrelevante Fahrzeugfunktion, die in einem System oder über mehrere Systeme verteilt realisiert ist. Er muss eine für den Sicherheitsverantwortlichen nachvollziehbare und durchgängige Sicherheitsargumentation dokumentieren. Diese Argumentation basiert auf Evidenzen in Form von Entwicklungsartefakten und zeigt auf, woher Sicherheitsanforderungen stammen, wie diese sowohl konstruktiv als auch prozessual korrekt implementiert wurden und dass deren Wirksamkeit auf Fahrzeugebene nachgewiesen wurde.
Der Hauptnutznießer eines klar strukturierten und durchgängigen Sicherheitsnachweises ist der Sicherheitsverantwortliche einer Fahrfunktion bzw. eines Systems. Entwicklungsingenieure können den Sicherheitsverantwortliche mit einer großen Menge sorgfältig erstellter Materialien wie FTAs oder FMEAs versehen, aber isoliert und ohne strukturierte Sicherheitsargumentation reicht dies nicht aus, um einen systematischen Nachweis der Sicherheit eines Systems zu ermöglichen. Der natürliche Zeitpunkt zur Erstellung des Sicherheitsnachweises liegt demnach nach der Beendigung der eigentlichen Produktenwicklung. Da aber speziell in der Automotiv Domäne die komplette Entwicklung meist iterativ und inkrementell erfolgt, muss auch der Sicherheitsnachweis aus Effizienzgründen inkrementell erstellt und am Ende der Produktentwicklung auf Vollständigkeit und Korrektheit überprüft werden können.
Für eine entwicklungsbegleitende Erstellung des Sicherheitsnachweises haben Graydon und Knight [GKS07] mit dem Assurance Based Development (ABD) ein Konzept vorgestellt, welches die frühzeitige und schrittweise Erstellung des Sicherheitsnachweises ermöglicht. Im Rahmen seiner Vorstellung der ABD nutzt Knight die GSN als Modellierungssprache für Sicherheitsnachweise. Egal wie der Sicherheitsnachweis dargestellt wird, er wird immer Sicherheitsanforderungen beinhalten. Diese sind bei hinreichend komplexen Systemen in der Regel nicht direkt aus Gefährdungen oder Sicherheitszielen auf Fahrzeugebene ableitbar sondern werden in Form von Sicherheitskonzepten unter Berücksichtigung von Funktionsarchitektur und anhand von Sicherheitsanalysen entwickelt.
Die einzige Modellierungssprache, die eigens für die Modellierung von Sicherheitskonzepten entwickelt wurde geht auf [Do09] zurück. Sie wurde am Fraunhofer IESE entwickelt, um den speziellen Herausforderungen der Erstellung eines Sicherheitskonzepts für komplexe, verteilt entwickelte Systeme zu begegnen. Diese Modellierungssprache für Sicherheitskonzepte beinhaltet insbesondere intuitive und hinreichend formale Konzepte zur Modularisierung, sowie die Unterstützung der Modellierung von ASIL[1]- Dekomposition. Durch ASIL-Dekomposition ist es möglich,

[1] ASIL (engl. Automotive Safety Integrity Level): Eine von vier Kategorien (A-D) zur Spezifizierung der notwendigen Anforderungen und Methoden gemäß ISO 26262 für die Betrachtungseinheit um ein akzeptables Restrisiko zu erreichen. ASIL D stellt dabei die höchsten Anforderungen dar.

Sicherheitsanforderungen redundant auf mehrere unabhängige Architekturelemente abzubilden und diesen einen reduzierten ASIL zuzuordnen.
Die Integration von Safety-Modellen in das Funktionsmodell wurde bereits an vielen akademischen und industriellen Beispielen erfolgreich evaluiert [Ad12]. In Bezug auf die Automotive Domäne, war die Komplexität der Beispiele allerdings nicht vergleichbar mit derer von heutigen Fahrerassistenzsystemen.

3 BMBF-Förderprojekt e performance

Das BMBF-Förderprojekt *e performance* [2] war prädestiniert, um die Praxistauglichkeit der Safety-Sichten zu evaluieren. Ziel dieses Projekts war die ganzheitliche Entwicklung eines Elektrofahrzeugs, wobei die entstehenden Komponenten in unterschiedlichen Fahrzeugkonzepten möglichst modular verwendbar sein mussten.
Im Förderprojekt hatte das Arbeitspaket „Funktionale Sicherheit“ die Aufgabe, ein ganzheitliches Sicherheitskonzept auf Gesamtfahrzeugebene für elektrisch betriebene Fahrzeuge zu erstellen. Im Fokus der Betrachtung standen dabei übergeordnete, vernetzte Funktionen, die sich aus einzelnen Subsystemen oder Komponenten heraus nicht effizient absichern ließen. Beispiele hierfür sind – wie in Abbildung 1 dargestellt – unter anderem die Betriebsstrategie, welche die Koordination zwischen Antrieb, Bremse, Fahrer und Fahrerassistenzsystemen darstellt, der Ladevorgang des Fahrzeugs sowie assoziierte Querschnittsfunktionen, wie das Energiemanagement.

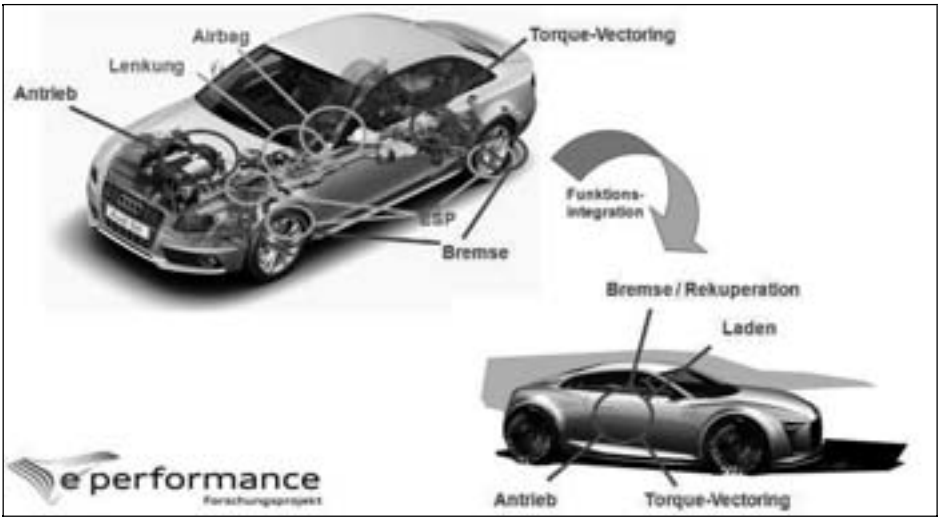

Abbildung 1: Hochintegration von Fahrfunktionen

Neue Funktionen, die Integration und Erweiterung bestehender Umfänge und prinzipielle Veränderungen im System und an den Systemgrenzen, ergeben insbesondere bei rein elektrisch betriebenen Fahrzeugen neue, für OEMs bisher weitgehend unbekannte Gefahrenquellen. Die ganzheitliche Analyse der Gefahren und Risiken auf Gesamtfahrzeugebene ist aufgrund der Komplexität und des Umfangs im Gegensatz zur Betrachtung eines einzelnen Subsystems jedoch ungleich anspruchsvoller. Dies gilt

insbesondere für die Entwicklung von vernetzten Funktionen mit stark verteilten Umfängen, wie es im vorliegenden Projekt der Fall ist. Um auch in diesem heterogenen Umfeld eine effiziente Sicherheitsanalyse zu gewährleisten, muss diese durch Modelle des Systems bzw. relevanten Teilsystemen und deren Fehlverhalten unterstützt werden.

4 Modellbasierte Sicherheitsanalysen und Sicherheitskonzepte

Das Fraunhofer IESE brachte seine Expertise im Bereich modellbasiertes Safety Engineering in das Förderprojekt ein. Es zeigte, wie Safety-Sichten auf das Funktionsmodell genutzt werden können, um mit den neuen Herausforderungen durch zunehmende Hochintegration von Fahrzeugfunktionen umzugehen.
Im Folgenden wird zunächst die allgemeine Vorgehensweise im Safety Engineering für Funktionale Sicherheit beschrieben. Anschließend wird erklärt welche Safety-Sichten in *e performance* evaluiert wurden.

4.1 Safety Engineering

Die sicherheitsgerichtete Entwicklung beginnt mit der Identifikation von Risiken durch Abweichungen vom Sollverhalten des Systems. Gemäß der ISO 26262 wurden dazu Gefährdungssituationen identifiziert und bezüglich ihres Risikos bewertet. In diesem Fall wurde für 5.600 Gefährdungssituationen bestimmt, wie hoch die notwendige Risikoreduktion sein muss, um Funktionale Sicherheit zu garantieren. Auf Basis dessen wurden anschließend die Sicherheitsziele abgeleitet. Ein solches bestand zum Beispiel darin abzusichern, dass das Differenzenmoment der Hinterräder den bestimmten Grenzwert nicht überschreitet, ab dem es zum Kontrollverlust über das Fahrzeug kommen kann. Die Sicherheitsmaßnahme für das genannte Sicherheitsziel erforderte unter anderem das Differenzenmoment der Hinterräder zu überwachen und diese in den momentenfreien Zustand zu versetzen, wenn das Differenzenmoment zu groß wird.

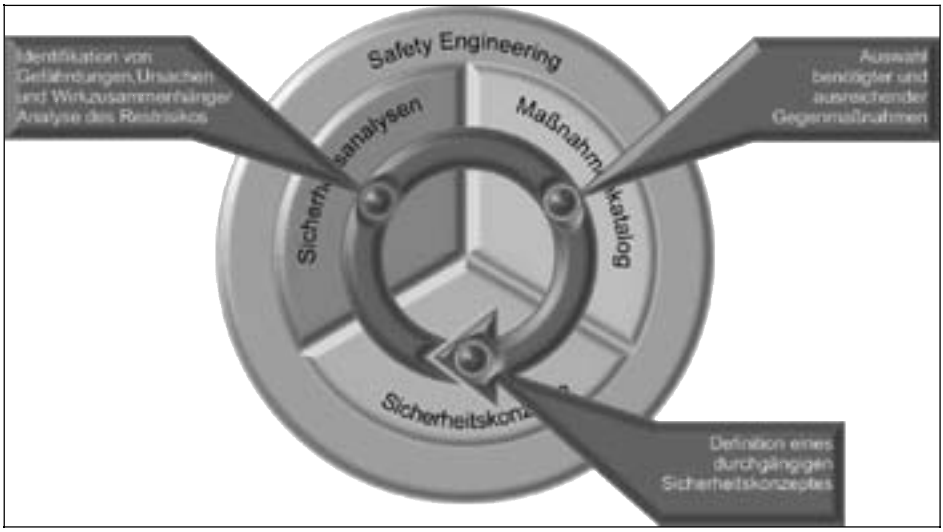

Abbildung 2: Iteratives Safety Engineering

Wie in Abbildung 2 veranschaulicht, ist die Maßnahmenidentifikation im Allgemeinen ein iterativer Prozess, bei dem analysiert wird, wie es dazu kommen kann, dass ein Sicherheitsziel verletzt ist. Sicherheitsanalysen werden dazu verwendet, um die Ursachen zu finden, warum ein oder mehrere Sicherheitsziele verletzt werden können. Basierend auf den identifizierten Ursachen werden im zweiten Schritt konstruktive Sicherheitsmaßnahmen systematisch ausgewählt.
Der iterative Prozess zwischen Sicherheitsanalysen und Maßnahmenauswahl endet sobald die Sicherheitsanalysen zeigen, dass alle Ursachen ausreichend durch Maßnahmen abgesichert sind. Um eine durchgängige, widerspruchsfreie und nachvollziehbare Darstellung der notwendigen Risikoreduktionen zu erhalten, wird in einem Sicherheitskonzept dokumentiert, welche Maßnahme warum ausgewählt wurde. Des Weiteren muss im Sicherheitskonzept auch dokumentiert sein, wie die Maßnahmen durch Sicherheitsfunktionen implementiert werden sowie welche Sicherheitsanforderungen an die normalem Systemfunktionen gestellt werden, um die Sicherheitsziele zu garantieren.

Um den Aufwand für das Safety Engineering zu minimieren wurden am Fraunhofer IESE Safety-Sichten für ein Funktionsmodell entwickelt. Abbildung 3 zeigt dieses Funktionsmodell[2], das im Rahmen des Projektes *e performance* erstellt wurde. Es beschreibt die funktionalen Zusammenhänge zwischen dem elektrifizierten Antriebsstrang und der Stromversorgung. Die Funktionen sind hierarchisch gruppiert. So sind beispielsweise die Funktionen für den Inverter, der den hinteren linken Elektromotor ansteuert, zu einer hierarchischen Funktion „AntriebsumrichterHL“ zusammengefasst. Für die Einregelung des Sollmoments ist eine Teilfunktion dieses Inverters verantwortlich.

[2] Hinweis: Die Details der abgebildeten Modelle sind für das weitere Verständnis unerheblich.

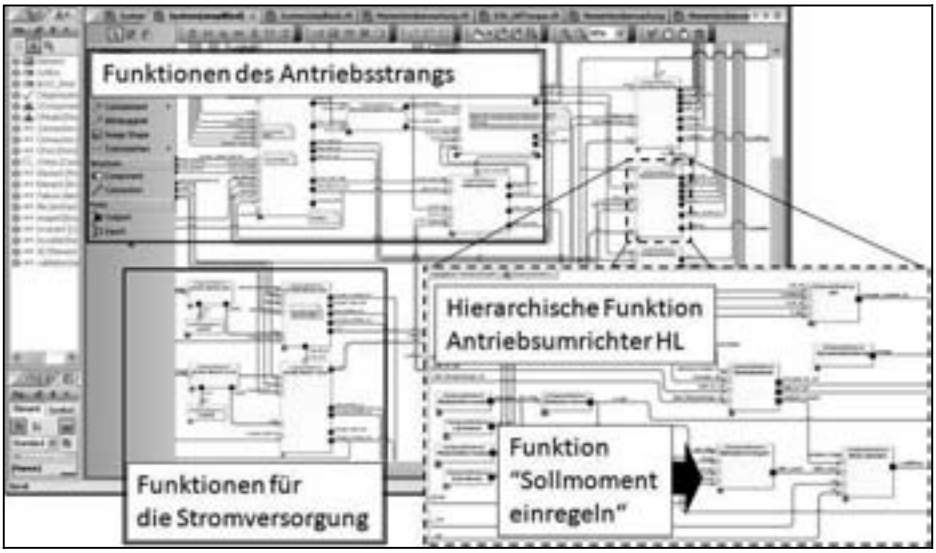

Abbildung 3: Funktionsmodell elektrifizierter Antriebsstrang im Projekt e performance

Im Folgenden wird beschrieben, mit welchen Sichten das Funktionsmodell erweitert wurde, um das Safety Engineering zu unterstützen.

4.2 Failure View

Eine wesentliche Sicht, um das Safety Engineering zu unterstützen, ist der *Failure View*. Dieser beschreibt die Ursache-Wirkungszusammenhänge von Fehlern und ermöglicht somit automatisierte Sicherheitsanalysen. Abbildung 4 zeigt den Failure View der Funktion „Sollmoment einregeln“. Er zeigt, wie Fehler von gesendeten Informationen bzw. Signalen mit internen Fehlern der Funktion und mit Fehlern von empfangenen Informationen bzw. Signalen zusammenhängen.
Die Fehler eines Ausgangsignals werden als „Fehlermodi“ bezeichnet und durch schwarze Dreiecke symbolisiert. Die Fehler eines Eingangssignals hingegen werden durch gelbe Dreiecke, interne Fehler wie bei der Fehlerbaumnotation durch Kreise dargestellt. Der Zusammenhang zwischen den Fehlern wird ebenfalls mit der üblichen Fehlerbaumnotation in Form von Booleschen Gattern modelliert. Um graphisch darzustellen, welcher Fehlermodus zu welchem Signal gehört, wird jeder Fehlermodus mit dem Port verbunden, über den das fehlerhafte Signal empfangen bzw. gesendet wird.

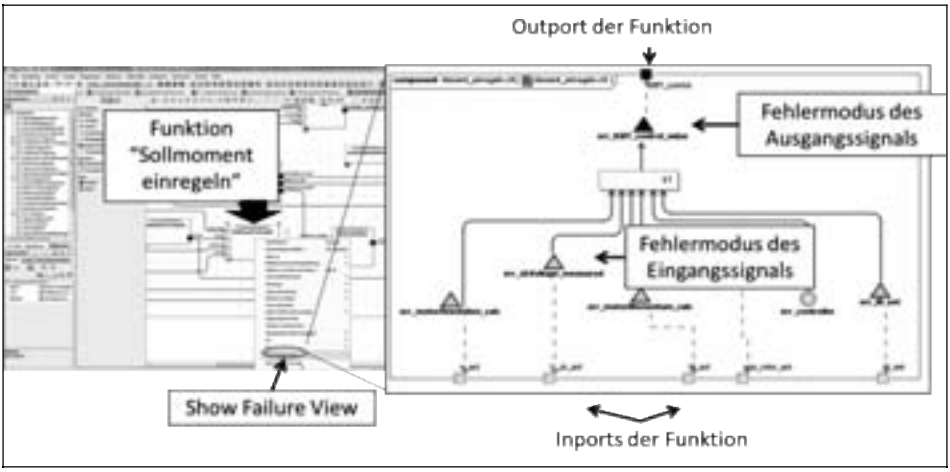

Abbildung 4: Failure View Sollmoment einregeln

Die obige Sicht zeigt jedoch nur, wie sich Fehler innerhalb einer Funktion fortpflanzen. Um zu beschreiben, wie sich Fehler von einer Funktion zur nächsten propagieren, wird für jede Verbindung im Funktionsmodell festgelegt, wie die Ausgangsfehlermodi der sendenden Funktion mit den Eingangsfehlermodi der empfangenen Funktion zusammenhängen. Abbildung 5 ist zu entnehmen, wie die funktionsübergreifende Fehlerfortpflanzung im Tool umgesetzt ist. Hierbei können verschiedene Failure Port Mappings für eine Verbindung festgelegt werden. Jedes dieser definiert, wie ein Ausgangsfehlermodus der sendenden Funktion einen Eingangsfehlermodus der empfangenen Funktion verursacht.

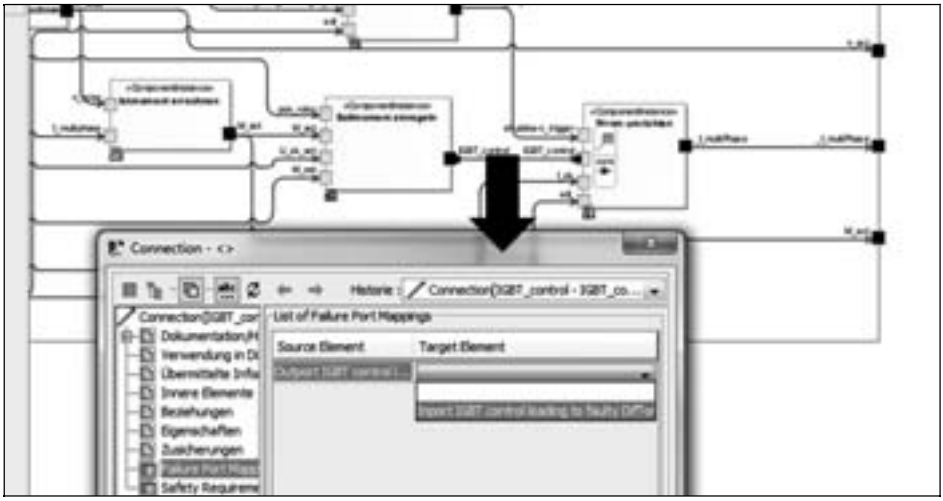

Abbildung 5: Fehlerfortpflanzung über Schnittstellen

Basierend auf den Failure Port Mappings werden die Failure Views der hierarchischen Funktionen generiert. Abbildung 6 verdeutlicht den generierten Failure View der

hierarchischen Funktion „AntriebsumrichterHL“. Er enthält alle Failure Views von den Unterfunktionen und beschreibt wie diese zusammenhängen. Im Beispiel enthält er den Failure View der Funktion „Sollmoment einregeln“ – als eine Unterfunktion der Funktion „AntriebsumrichterHL“. Der Failure View auf oberster Hierarchieebene beschreibt alle Fehlerzusammenhänge und somit alle notwendigen Informationen, um mit Hilfe von automatisierten Analysen herauszufinden, welche Fehlerkombinationen welche Sicherheitsziele verletzen.

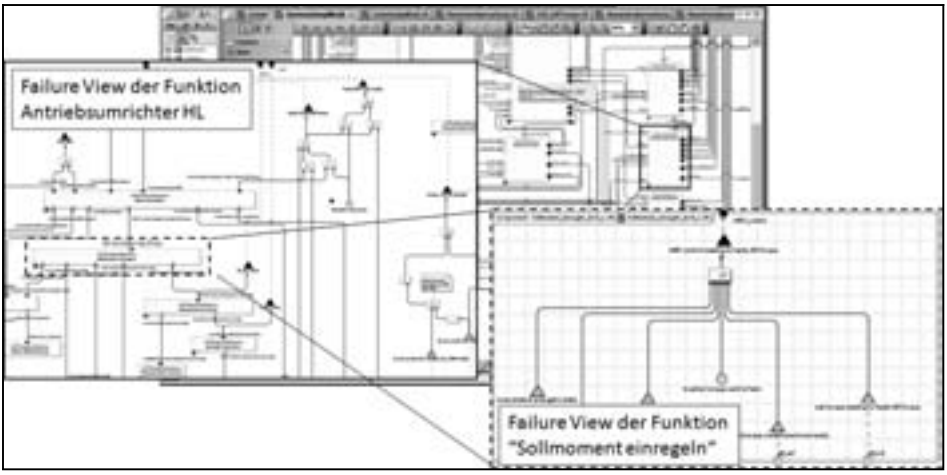

Abbildung 6: Hierarchischer Failure View

4.3 Safety Concept View

Eine weitere Sicht um das Safety Engineering zu unterstützen, ist der *Safety Concept View*. Dieser dokumentiert, was eine Funktion für ihre Ausgangssignale garantiert und wie diese Garantien mit Sicherheitsanforderungen der Funktion und den empfangenen Informationen zusammenhängen. Abbildung 7 zeigt den Safety Concept View der Funktion, die für die Erkennung eines zu hohen Differenzenmomentes verantwortlich ist. Die Funktion garantiert, dass sie ein Abschaltsignal sendet, wenn das hintere Differenzenmoment größer ist als ein definierter Grenzwert, der von den fahrdynamischen Eigenschaften des Fahrzeugs abhängt.
Um diese Garantie zu geben, fordert sie, dass die empfangenen Informationen über die Momente der hinteren Räder hinreichend genau sind. Des Weiteren implementiert sie die Sicherheitsanforderung, dass immer genau dann ein Abschaltsignal generiert wird, wenn die Differenz der empfangenen Radmomente den Grenzwert überschreitet.

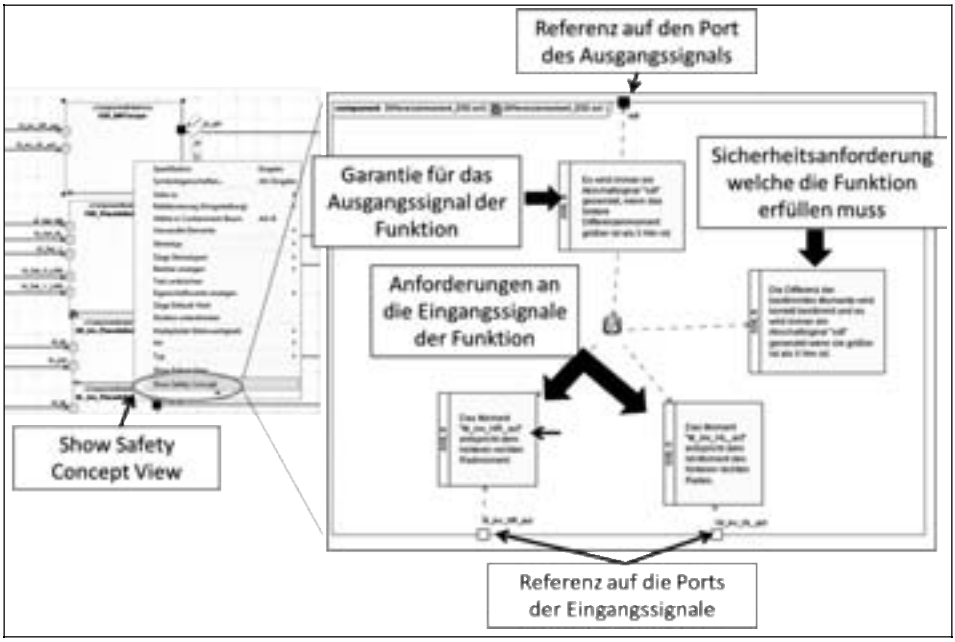

Abbildung 7: Safety Concept View

Der Safety Concept View lässt sich ebenso wie der Failure View hierarchisch im Modell in sogenannte Safety Concept Parts gliedern und unterstützt damit die iterativ-inkrementelle Entwicklung eines systematisch strukturierten und modularen Sicherheitskonzepts. Wir unterscheiden an dieser Stelle nicht zwischen einer Notation für funktionale oder technische Sicherheitskonzepte. Selbstverständlich ist es aber möglich in hierarchisch strukturierten Modellen funktionale Aspekte auf Ebene der logischen Funktionsarchitektur darzustellen, ebenso wie deren Verfeinerung durch Abbildung auf die technische Systemarchitektur, z.B. in UML Deployment-Diagrammen. Der zusicherungsbasierte Ansatz erleichtert die Wiederverwendung von Komponenten und die konsistente Einbindung von kontextfreien Sicherheitselementen (vgl. ISO 26262, Safety Element out of Context, SEooC). Solange an den Schnittstellen die Sicherheitsanforderungen der einen, durch ausreichende Sicherheitsgarantien der anderen Komponente erfüllt sind, besitzt auch die Komposition aus modularen Einzelkomponenten eine entsprechende Sicherheitsintegrität. Mit dieser Methode ist es nun möglich, aus verschiedenen kontextfreien Sicherheitselementen für bestimmte generische Sicherheitsfunktionen durch Komposition auf höherer Ebene eine ebenfalls kontextfreie und in diesem Sinne modulare Sicherheitsarchitektur, sozusagen eine Safety Architecture out of Context (SAooC), zu entwerfen.

5 Zusammenfassung und Ausblick

Allgemein ist der Failure View und seine Integration in einen modellbasierten Top-Down Entwicklungsprozess ein wesentlicher Schritt hin zu modularen Sicherheitsnachweisen, die es ermöglichen, Teile von Entwicklungsartefakten ohne

Mehraufwand wiederzuverwenden. Der Safety Concept View erlaubt es eine durchgängige Sicherheitsargumentation passend zu einer funktionalen Architektur zu modularisieren.
Im Projekt *e performance* wurden der Failure View und der Safety Concept View erfolgreich eingesetzt, um ein Sicherheitskonzept für den elektrifizierten Antriebsstrang zu entwickeln und modular zu beschreiben. Das Sicherheitskonzept wurde auf mehreren Ebenen verteilt implementiert. Die Antriebsumrichter erkennen und behandeln lokale Fehler eigensicher unter Verwendung von diversitären Prinzipien sowie deren Umsetzung auf dissimilarer Hardware. Im integrierten Antriebsstrang, bestehend aus mehreren Einzelantrieben, plausibilisiert sich die Antriebslogik gegenseitig. Ein separates Steuergerät überwacht die Längs- und Querdynamik auf oberster Systemebene und kann abhängig vom Fehlergrad den Antriebsstrang redundant momentenfrei schalten.
Im modellbasierten Entwicklungsprozess des elektrifizierten Antriebsstrangs erwies sich das Funktionsmodell als unumgänglich um die funktionalen Zusammenhänge zu verstehen. Eine informale Beschreibung basierend auf Textdokumenten (Item Definitions) wäre bei der stark verteilten Entwicklung nicht ausreichend gewesen, um mit der hohen Änderungsrate von neuen Funktionen umzugehen. Die beiden Sichten Failure View und Safety Concept View waren essentiell um eine Konsistenz zwischen dem Safety Engineering und der Entwicklung der Funktionsarchitektur zu gewährleisten. Insbesondere Änderungen der Funktionsarchitektur konnten effizient im Safety Engineering behandelt werden, indem nur der Failure View und der Safety Concept View von den betroffenen Funktionen angepasst wurden.
Das vom Fraunhofer IESE entwickelte Werkzeug für die Erstellung modellintegrierter Sicherheitsanalysen und Sicherheitskonzepte basiert auf XML und kann deshalb mit vertretbarem Aufwand in gängige Modellierungswerkzeuge integriert werden.

Systematische Wiederverwendung sowie synergetische Funktions- und Software-Baukästen sind wesentliche Bestandteile der Technologie-Roadmap von Audi. Die im Förderprojekt evaluierten Methoden und Werkzeuge für modellbasierte Sicherheitsanalysen und Sicherheitskonzepte sind vielversprechend und werden in das etablierte Vorgehen sukzessive integriert und erweitert. Aus diesem Grund schließt die Audi Electronics Venture GmbH Folgeaktivitäten an und beteiligt sich im BMBF-Förderprojekt SPES_XT [3] als Partner sowie auch auf europäischer Ebene im ITEA-2 Projekt SAFE [4] als OEM Advisor.
Mögliche Erweiterungen zum bisher Erreichten sind beispielsweise eine modellbasierte Kopplung von Item Definition mit datenbankgestützter Gefahren- und Risikoanalyse. Dabei ist es denkbar, die Aspekte der Gebrauchssicherheit in das gemeinsame Funktionsmodell einzubeziehen. Ein naheliegender Ansatz wäre die Verwendung von Anwendungsfall-, Aktivitäts- und Kontextdiagrammen, um damit die Interaktion des Fahrers mit der als funktionssicher angenommenen Fahrfunktion und der Systemumgebung abzubilden. Eine zusätzliche Anforderung ist sicherlich die Anbindung von Fahrsimulationen, um aus modellbasierten Szenarien abgeleitete Sicherheitskonzepte frühzeitig validieren zu können.

6 Literatur

[Ad11] Rasmus Adler, Dominik Domis, Kai Höfig, Sören Kemmann, Thomas Kuhn, Jean-Pascal Schwinn and Mario Trapp. Integration of Component Fault Trees into the UML. In: Juergen Dingel and Arnor Solberg, Models in Software Engineering. Lecture Notes in Computer Science. Pages 312-327. Springer Berlin / Heidelberg, 2011.

[Ad12] Rasmus Adler, Sören Kemmann, Peter Liggesmeyer, Pascal Schwinn. Model-based Development of a Safety Concept. To be published in: Proceedings of the PSAM11 & ESREL 2012 Conference.

[Ch93] Chudleigh M.: Hazard analysis using HAZOP: A case study. 12th International Conference on Computer Safety, Reliability and Security (SAFECOMP 93), Springer Verlag, pages 99 – 108, 1993.

[De09] Denger, C. SafeSpection – A Framework for Systematization and Customization of Software Hazard Identification by Applying Inspection Concepts. PhD-Thesis, Technical University of Kaiserslautern, 2009

[Do09] D. Domis, M. Forster, S. Kemmann, and M. Trapp. Safety concept trees. In Reliability and Maintainability Symposium, 2009. RAMS 2009. Annual, pages 212 {217, jan. 2009.

[DST05] Vincent Debruyne, Françoise Simonot-Lion and Yvon Trinquet. EAST-ADL: An Architecture Description Language. In: Architecture Description Languages, IFIP The International Federation for Information Processing. Chapter 12, Pages 181–195. Springer Boston, 2005.

[DT08] Dominik Domis, Mario Trapp: Integrating Safety Analyses and Component-Based Design. SAFECOMP 2008: 58-71

[Fe94] P Fenelon, et al, Towards Integrated Safety Analysis and Design, In ACM Applied Computing Review, 2(1):21-32, 1994, ACM Press

[GKS07] P.J. Graydon, J.C. Knight, E.A. Strunk., „Assurance Based Development of Critical Systems“, in Proc. of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pages 347 – 357, 2007.

[LW99] Lutz, R.R., Woodhouse R. M.: Bi-directional Analysis for Certification of Safety-Critical Software. In the proceedings of the International Software Assurance Certification Conference (ISACC 99), Springer Verlag, pages 1 – 9, 1999.

[Pa04] Papadopoulos Y. et al.: Automating the Failure Mode and Effects Analysis of Safety Critical Systems. In the proceedings of the 8th International Symposium on High Assurance Systems Engineering (HASE 2004), pages 310 – 311, 2004.

[PD99] Y. Papadopoulos, J. McDermid, “Hierarchically Performed Hazard Origin and Propagation Studies”, in Proceedings of the 18th International Conference on Computer Safety, Reliability and Security, LNCS 1608:139-152, 1999, Springer Verlag.

[Pu99] Pumfrey D.J.: The Principled Design of Computer System Safety Analysis. PhD thesis. Department of Computer Science, University of York, York, UK, 1999.

[RCC99] Redmill F., Chudleigh M., Catmur J.: System Safety: HAZOP and Software HAZOP. Chichester: John Wiley & Sons Ltd. 248, 1999

[Sc12] Karsten Schmidt, Markus Buhlmann, Christoph Ficek, Kai Richter. Entwurfsaspekte für hochintegrierte Steuergeräte mit unterschiedlichen ASIL –Stufen, ATZ elektronik, 01/2012, Seiten 34-40

[We08] Tim Weilkiens. Systems Engineering mit SysML/UML: Modellierung, Analyse, Design. Dpunkt-Verl., 2., aktualisierte u. erweiterte Edition, Oktober 2008.

[Wi96] S.P. Wilson, J.A. Mcdermid, C.H. Pygott, and D.J. Tombs, "Assessing Complex Computer Based Systems using the Goal Structuring Notation", in ICECCS '96: Proceedings of the 2nd IEEE International Conference on Engineering of Complex Computer Systems (ICECCS '96), Washington, DC, USA: IEEE Computer Society, 1996, p. 498.

[1] ISO 26262: Road vehicles - Functional safety.

[2] http://www.audi.de/eperf/brand/de.html

[3] http://spes2020.informatik.tu-muenchen.de/ (Vorgänger von SPES_XT)

[4] http://www.safe-project.eu/