Software Safety Concept – so könnte es klappen
Welche Analysen sind sinnvoll? Eine Fallstudie

Dr. Thomas Liedtke, Kugler Maag CIE GmbH
Christian Bayer, Elektrobit Automotive GmbH

In unserem Paper zeigen wir die bei der Definition und der Erstellung eines Software Safety Concepts gemachten Erfahrungen auf. Safety Analysen in der Softwareentwicklung sind (bis auf Ausnahmen) primär auf Software Architekturebene vorgesehen. Verbleibende Restrisiken können durch Anwendung empfohlener Maßnahmen der ISO 26262 [1] Band 6: Product development at software level im weiteren Entwicklungsverlauf als ausreichend klein eingestuft werden.

Unser Software Safety Concept fokussiert sich auf die Software Requirements specification-Ebene, sowie die Architectural design-Ebene. Wir stellen die im realen Kundenprojekt durchgeführten vier verschiedenen Safety Analysen vor. Dabei beschreiben wir den Zweck, gemachte Erfahrungen und typische Befunde.

1. Motivation | Projekt und die Aufgabenstellung

Für eine Fahrerassistenzfunktion stand ein OEM vor der Herausforderung einen vorhandenen SW-Algorithmus zu optimieren und zu industrialisieren. Der Algorithmus diente zwei Kunden-erlebbaren Funktionen im sicherheitskritischen Kontext. Durch Dekomposition wurde auf Systemebene eine ASIL-B (D) Einstufung vorgenommen. Ein technisches Sicherheitskonzept (technical safety conept/ TSK) fokussierte auf die zu erstellende Software Komponente (SW-C) und diente als Basis der Softwareentwicklung. Eine Platform (HW und SW) mit entsprechendem Safety Manual bot die notwendige Ausführungsumgebung an.

Die Firma Elektrobit Automotive GmbH (kurz: EB) wurde damit beauftragt für die Software Komponente darzulegen, welche Sicherheitsmaßnahmen notwendig sind, um den Anforderungen aus dem TSK gerecht zu werden. Desweiteren wurde EB in die Softwareentwicklung auf Requirements-, Architektur- sowie die entsprechenden Testebenen eingebunden. Zur externen Beratung wurde die Safety-Expertise von Kugler Maag CIE hinzugezogen.

Aus OEM Sicht sollte ein Dokument mit dem Titel „Software Safety Concept“ darlegen welche Sicherheitsmechanismen warum implementiert wurden und wie diese die Sicherheit gewährleisten. Das Dokument sollte zum Finden und Dokumentieren der notwendigen Sicherheitsmechanismen dienen. Das externe Assessorenteam wollte das Dokument um die Sicherheitsanalysen und die Sicherheitsargumentation für das Projekt ausgebaut wissen, um letztlich die Freigabe der Software zu verargumentieren.

Entsprechend blickt das Projekt mit dem Software Safety Concept auf das zentrale Dokument wenn es um Freigaben oder um die Bewertung neuer Change Requests für das Projekt geht.

2. Software Safety Concept

Nachdem ein seperates Software Safety Concept kein in der ISO 26262 [1] spezifiziertes Workproduct darstellt, musste zunächst der Umfang und Inhalt definiert werden.

Der Zweck des Software Safety Concepts ist zum einen die Abbildung der Safety Requirements auf die Architektur zu evaluieren. Zum anderen maßgebliche Evidenzen durch die Auswahl und die Ergebnisse durchgeführter Safety-Analysen für die Safety-Argumentation zu liefern.

Entsprechend stellten die SW Safety Requirements (betrafen im Wesentlichen die Behandlung von sicherheitsrelevanten und nichtsicherheitsrelevanten Sensoren und Signale), die abgeleitete SW Architektur und die Traceability Matrix die Analysebasis dar, welche iterativ untersucht und angepasst wurde.

Ursprünglich war geplant die Software in zwei Sub-Komponenten mit unterschiedlicher ASIL-Einstufung zu entwickeln. Jedoch konnte die Vermengung von Signalen mit unterschiedlicher ASIL-Allokation nicht verargumentiert und eine Freedom from Interference nicht hinreichend dargestellt werden. Es folgte die Forderung, dass sämtliche Softwareteile nach gleichem ASIL zu entwickeln sind.

Safety-Analysen in der Software werden aufgrund ihrer Wirksamkeit meist in der Architekturebene durchgeführt. Für die weitere SW-Entwicklung sollten die ASIL-orientierten Vorgaben der ISO 26262 [1] die notwendige Sicherheit geben, um verbleibende systematische Fehler auf ein Mindestmaß zu reduzieren.

3. Safety-Analysen

Zur Untersuchung folgender Fragestellungen wurden Safety-Analysen angesetzt:

- Sind die Sicherheitsanforderungen intern und extern konsistent, vollständig und verständlich? → Systematische TSK-Analyse

- Welches sind die kritischen Safety Mechanismen? Werden sie mit entsprechenden Maßnahmen hinreichend sicher entwickelt? → Requirements FMEA (Failure Mode and Effect Analysis [2])

- Ist die Architektur ausreichend abgesichert (im Sinne von vollständig logisch beschrieben), vollständig und konsistent? → Architektur HAZOP (Hazard and Operational Analysis [3])

- Analyse und Identifikation welche Basis-Fehler zu Top-Level-Fehlversagen führen können → Fehlerbaum Analyse (FTA)

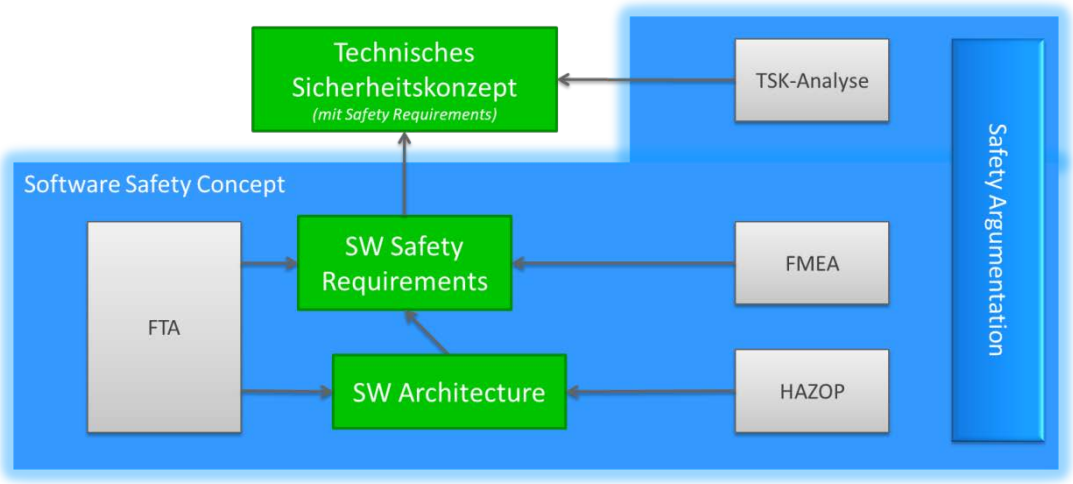

Abbildung 1: Bestandteile des Software Safety Concepts

3.1 Systematische TSK-Analyse

Das technische Sicherheitskonzept des OEM bestand aus ca. 1000 Datenbankeinträgen und beinhaltete Definitionen, Systemerklärungen, Relevante Ein- und Ausgangssignale, sowie die geforderten ca. 40 funktionalen Sicherheitsanforderungen. Das klingt zunächst überschaubar und trotzdem glänzten die Anforderungen durch

- verschachtelte Bedingungen
- teils mehrere Fußnoten mit weiteren Bedingungen, Ausnahmen, Referenzen auf Signaltabellen und Definitionen innerhalb des Dokuments

Im Projekt hatte man sich entschieden eine systematische Anforderungsanalyse gemeinsam mit dem OEM durchzuführen. Ziel war es ein gleiches Verständnis zu schaffen und Kundenanforderungen bei Missverständnissen oder aufgedeckten Unzulänglichkeiten zu ändern.

Die Durchsprache mit dem OEM in zahlreichen Treffen wurde mitprotokolliert und die Ergebnisse dokumentiert. Bei notwendigen Änderungen an den Kundenanforderungen wurden diese solange nachverfolgt, bis die Änderung in Form neuer Kundenanforderungen bei EB vorlag.

Lessons Learned:

- Komplexität der Anforderungen wurde unterschätzt; die Ergebnisse einer ersten TSK Durchsicht mit Anforderungsableitung mussten verworfen werden
- Viele implizite Annahmen wurden explizit nachdokumentiert und Terminologie geklärt
- Widersprüche konnten aufgedeckt und geklärt werden

3.2 Requirements FMEA

Eine FMEA auf Requirementsebene hatte folgende Ziele:

- Als Teil der Risiko-Analyse sollte sie frühzeitig mögliche Inkonsistenzen (interne und externe) in den Safetymechanismen aufdecken

- Die Kritikalität der einzelnen Sicherheitsmechanismen sollte evaluiert und dokumentiert werden

- Die protokollierte Wissensbasis sollte erhöht werden, der Kommunikationsfluss mit dem Kunden und damit verbundenem erforderlichen Wissenstransfer sollte gefördert werden

Die Analyse-Sitzungen starteten jeweils auf Basis der negierten Anforderungen. In zwei Richtungen wurden diese evaluiert:

1. Fehlerursache: Welche Fehler können zu einer Verletzung von Safety-Anforderungen führen?

2. Fehlerauswirkung: Welches mögliche Fehlversagen kann eintreten? Dabei stellte sich heraus, dass die Kombinatorik unübersichtlich werden kann, weshalb eine einheitliche Risikobewertung erstellt wurde (s. Abbildung 2).

sensor signals an error	error status will be interpreted correctly	sensor data is used for algotithm	signalling of error on the output	risk impact
Yes	Yes	Yes	Yes	7
			No	9
		No	Yes	0
			No	8
	No	Yes	Yes	9
			No	9
		No	Yes	0
			No	8
No	Yes	Yes	Yes	3
			No	0
		No	Yes	4
			No	5
	No	Yes	Yes	3
			No	0
		No	Yes	3
			No	5

Abbildung 2: FMEA Risikobewertung

In mehreren Sitzungen wurden Fehlerkombinationen und deren Auswirkungen evaluiert. Typische Befunde bezogen sich auf bislang unberücksichtigte komplexe Fehlerkombinationen und Fehlerarten (lang oder kurz anliegend, instabile Situation, …). Anforderungen wurden aufgrund von Befunden entsprechend angepasst.

Lessons learned:

- Wichtiger Know-How-Transfer für den weiteren Projektverlauf
- Zeitaufwändige Klärungsdiskussionen ergaben mehrdeutige Interpretationen von Anforderungen, so dass diese nachgeschärft wurden.

3.3 Architektur HAZOP

Sämtliche Diagramme (u.a. Komponenten, Sequenzen) sollten systematisch hinterfragt werden. Hierzu wurde für jede Diagrammart ein passender Fragenkatalog mit HAZOP-Leitworten [3] erstellt.

Der Prozess lief iterativ in den folgenden Phasen ab:

1. Identifikation der Design Elemente für die die Analyse durchgeführt werden soll

2. Bestimmung der Abweichungen: Anwendung der HAZOP-Leitworte auf alle Elemente eines Diagramms

3. Analyse der Abweichungen bzgl. Ursache, Folge und möglicher Verletzung einer Anforderung

4. Definition neuer Anforderungen für fehlende Maßnahmen und Anwendung von Maßnahmen

5. Allokation und Realisierung neuer Anforderungen an die Sicherheitsarchitektur oder Komponente

6. Übertragung Benutzeranforderung an die HW-/ SW-Randbedingung

7. Aktualisierung des Designs → neue Iteration beginnend bei Schritt 1

Beispiele der Anwendung möglicher Leitworte für ein Komponentendiagramm sind:

- Vollständigkeit:
  - Komponente fehlt – Memory Partition fehlt – Schnittstelle fehlt – Konnektor fehlt - …
- Quantitativ:
  - enthält mehr: Memory Partition umfasst unabsichtlich weitere Komponenten – Assoziation falsche Anzahl - … |
  - enthält weniger: Memory Partition umfasst nicht alle Komponenten - …

Bei Sequenz- und Aktivitätsdiagrammen kommen weitere Leitworte wie z.B. für Timing und Abfolge dazu.

Lessons Learned:

- Typische Befunde bezogen sich mehrheitlich auf die Testbarkeit (z.B. wie kann die vorgegebene Sequenz geprüft werden?) sowie auf Vollständigkeit (wie kann das vollständige Durchlaufen einer Schleife geprüft werden).
- Die Diagramme wurden im Verlauf der Analyse schlüssiger und stimmiger und damit die Architektur wertvoller.

3.4 Fehlerbaumanalyse (FTA)

Um nicht nur Bottom up auf mögliche Fehlerquellen schließen zu können wurde eine Fehlerbaumanalyse basierend auf den Ebenen Software Safety Requirements und der Safety Architektur erstellt.

Hierbei wurden minimal cut sets aufgestellt und untersucht. Die Abhängigkeiten zwischen verschiedenen Safety Mechanismen wurden hergestellt.

Die Fehlerbäume wurden in mehreren gemeinsamen Runden von Safety-Experten und Architekten aufgestellt.

Lessons Learned:

- Obwohl die Top-level-Events sehr schnell gefunden waren gestaltete sich die Ableitung durch die Safety Mechanismen auf einzelne Single-Events schwierig.
- Die FTA-Analyse ergab eine gute Übersicht über mögliche Kombinationen von Fehlerursachen und der Wirkung von Safety Mechanismen.

4 Resümee

Die Erstellung eines Software Safety Concepts half den Umfang des Projektes zu begreifen. Es bildet nun das zentrale Dokument inklusive wesentlicher Evidenzen für die Safety Argumentation. Bei neuen Kundenanforderungen ermöglicht es dem Team effizient Einflußanalysen auf bestehende Safety Mechanismen durchzuführen.

Der Aufwand für die Safety Analysen war zunächst unterschätzt worden. Eine Übersicht über Dauer, Aufwand und Nutzen gibt folgende tabellarische Übersicht:

	TSK Analyse	FMEA	HAZOP	FTA
Dauer	3 Monate	3 Monate	12 Monate	2 Monate
Aufwand	6 Personenmonate	1,5 Personenmonate	8 Personenmonate	2 Personenmonate
Nutzen	7% des TSKs vom OEM überarbeitet (inkl. Klarstellungen, neuer und herausgenommener Anforderungen)	Kritikalität der Safety Mechanismen als Input für weitere präventive und analytische Maßnahmen in der Entwicklung	Hohe Qualität der Diagramme und der Architektur, Diverser Input für Teststrategie	Unabhängigkeit der Safety Mechanismen

Tabelle 1: Aufwandsübersicht Safety Analysen

Literatur

[1] ISO 26262 Road vehicles – Functional safety 1st edition (2011)
[2] VDA-Band 4: Produkt- und Prozess FMEA (2009)
[3] HAZOP Studies on Systems Containing Programmable Electronics Part 2 General Application Guidance. Ministery of Defence. Defence Standard 00-58, (2000)