MICRO CONSULT

Funktionale Sicherheit: Systematische Fehler mit Struktur und Prozessen eindämmen

Ein ganzheitlicher Ansatz und das entsprechende Wissen um die Details sind essentiell, wenn es um das Erstellen von funktional sicheren Systemen geht. Die Integrität der Software lässt sich durch strukturierte und zielgerichtete Methoden und Techniken erreichen.

Funktionale Sicherheit ist ein Teil des allgemeinen Sicherheitsbegriffs. Der Wortbildung liegt schon inne, dass es sich hierbei um Systeme handeln wird, die „funktionieren" müssen, weil sie bei einem Ausfall zu unsicheren Systemen führen können.

Hier gilt wie so oft das Motto „Es muss immer erst etwas passieren, damit etwas passiert". Internationale Normen tragen heute dafür Sorge, dass wir zu nachvollziehbar sicheren Systemen gelangen. Auch wenn Normen keine Gesetze darstellen, so werden sie doch vor Gericht als Referenz angesehen, um zu beurteilen, ob ein System nach dem aktuellen „Stand der Wissenschaft und Technik" gebaut wurde.

Basisnorm definiert Lebenszyklus von Systemen

Was steht nun in entsprechenden Normen, und welche Norm hat für den Entwickler Gültigkeit? Es gilt, dass marktspezifische Normen Vorrang gegenüber generischen Normen haben. Im industriellen Umfeld stellt die IEC 61508 die Basisnorm dar, nach deren Vorbild eine Vielzahl von spezifischen Normen entwickelt wurde. Trotzdem steht diese Basisnorm auch für sich alleine und sollte zur Anwendung gelangen, wenn es keine spezifische Norm gibt.

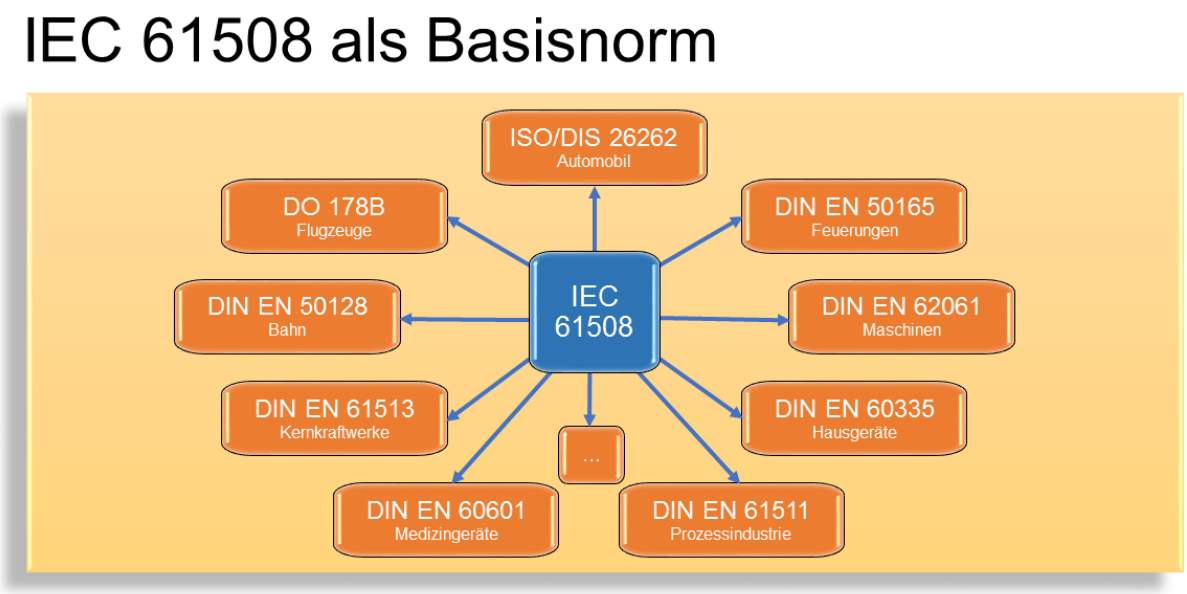

Bild 2: IEC 61508 als Basisnorm

Im Zentrum der Basisnorm stehen ein definierter Lebenszyklus von Systemen, unterteilt in mehrere Phasen, die jeweils mit entsprechenden Anforderungen versehen sind, sowie definierte Artefakte, die bei Einhaltung der Norm erstellt werden müssen. Dabei umfasst der Lebenszyklus die „Geburt“ des Systems durch ein entsprechendes Konzept bis hin zum „Tod“, sprich der Außerbetriebnahme bzw. Stilllegung.

Die Norm erkennt an, dass es keine 100% Sicherheit gibt

Ein weiteres zentrales Instrument ist das Bewerten der Sicherheit über Risiken und das eventuelle Senken des Risikos auf ein „sozial verträgliches Niveau“. Dazu werden zunächst alle möglichen Gefahren, die von dem System ausgehen können, analysiert und anschließend das Risiko bewertet. Dieses ergibt sich als Produkt aus Wahrscheinlichkeit des Eintretens der Gefahr und der Schwere der resultierenden Verletzungen. Je stärker ein Risiko gesenkt werden muss (auf das sozial verträgliche Niveau), umso höher sind die Anforderungen, die die Norm an sicherheitsrelevante Systeme stellt.

Die Anforderungen werden dabei in sogenannte Integritätslevel eingeordnet. Die IEC 61508 benennt vier Level, SIL 1 – 4, wobei letzterer die höchsten Anforderungen nach sich zieht. Die Norm erkennt an, dass es keine 100% Sicherheit gibt. Doch gibt sie über die Integritätslevel an, welche Maßnahmen zu treffen sind, damit die sicherheitsrelevanten Funktionen (jene, die dafür sorgen, dass das System sicher ist) auch ordnungsgemäß funktionieren.

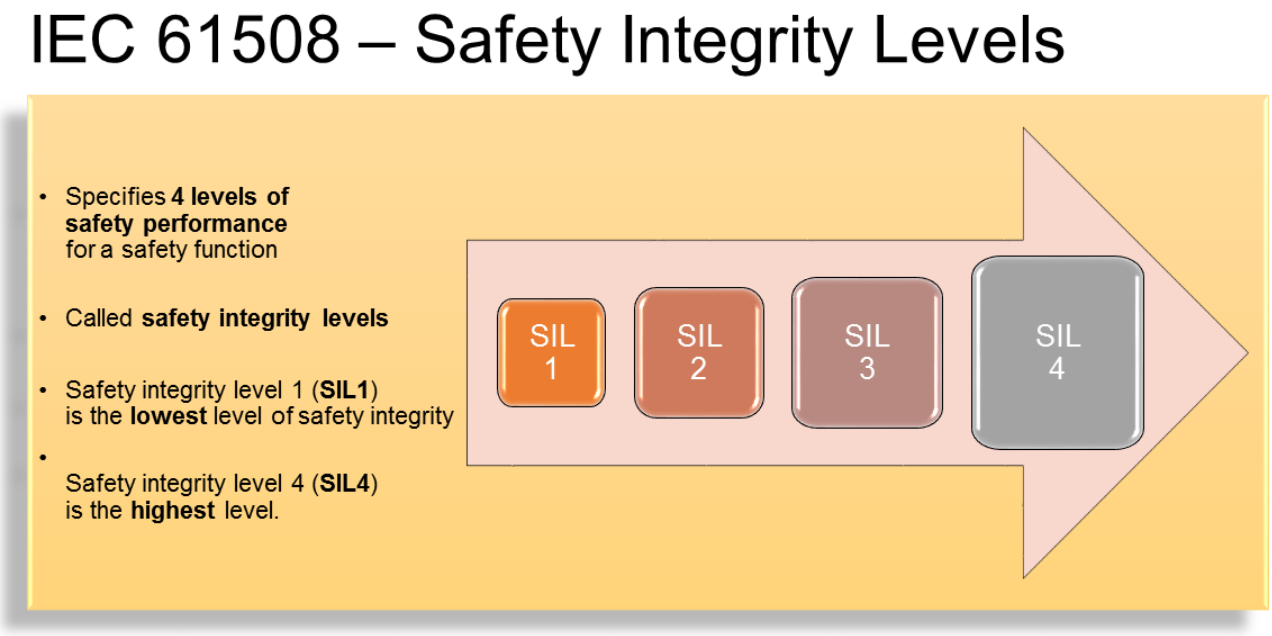

Bild 2: Sicherheits-Integritätslevel nach IEC 61508

Die Dokumentation – das Herzstück

Welche Vorgaben stellt die Norm? Grundsätzlich geht es um das Vermeiden von Fehlern. Dazu ist es notwendig, sich mit unterschiedlichen Fehlerklassen auseinanderzusetzen. Als oberstes Unterscheidungsmerkmal wird zwischen zufälligen (random) und systematischen Fehlern unterschieden. Die zufälligen Fehler (z.B. Ausfall eines Bauteils) werden vereinfacht auch gerne als Hardwarefehler bezeichnet. Diese Fehler kann man nicht wirklich vorhersehen oder verhindern, sondern nur deren Auftreten über geeignete Bauteilauswahl und Systemdesignmaßnahmen verringern.

Im Gegensatz dazu lassen sich systematische Fehler (solche, die menschlich verursacht sind) eindämmen, wenn strukturiert und nach Prozessen gearbeitet wird. Dazu gehören entsprechend hohe Anforderungen an die Dokumentation, das Management und die Verifikation in allen Phasen des Lebenszyklus. Häufig wird von Ingenieuren nur der damit verbundene große Aufwand gesehen und der eigentliche Grund und Nutzen darüber leider vergessen.

Selbst Anforderungen definieren

Aber nicht nur die Norm stellt Anforderungen, sie fordert auch, dass der Anwender welche definiert. Für jede sicherheitsrelevante Funktion sind demnach Anforderungen an die Funktion an sich sowie an die Integrität der Funktion zu dokumentieren und umzusetzen. Ganz gemäß den zwei Basisfehlerklassen lassen sich Anforderungen an die Hardwareintegrität und systematische Integrität unterscheiden, die allerdings wieder aus der Norm heraus definiert in Abhängigkeit des notwendigen SIL sind. Für den eigentlichen Systementwurf spielt neben den Anforderungen hinsichtlich der Behandlung von erkannten Fehlern und der Datenkommunikation eine Rolle.

Für die Hardwareintegrität kommen schließlich zwei Gruppen von Forderungen ins Spiel: Einerseits ist dies die Einhaltung von Vorgaben für zufällige Hardwarefehler, zusätzlich aber auch die Einhaltung von architekturellen Einschränkungen. Letzteres kann man über zwei sogenannte Pfade bzw. Routen erreichen (1H oder 2H). Die Route 2H bezieht sich dabei auf bereits betriebsbewährte Architekturen, für die aber zusätzlich Forderungen an Minimalwerte der sogenannten Hardware Fault Tolerance (HFT) gestellt werden. Route 1H bedeutet letztendlich, dass man gemäß der Norm entwickelt und den Nachweis für zwei Maße führt, nämlich für die HFT und SFF (Safe Failure Fraction), deren Werte je nach zu erfüllenden SIL eingehalten werden müssen.

Software-Integrität durch strukturierte und zielgerichtete Methoden erreichen

Ähnliches gilt für die systematische Integrität, deren Einhaltung sich über das Folgen von drei unterschiedlichen Routen (1S, 2S oder 3S) umsetzen lässt. Route 1S bedeutet wiederum das Entwickeln gemäß der Norm, Route 2S bezieht sich auf die Betriebsbewährtheit, und Route 3S sowie damit in Zusammenhang stehende Anforderungen gelten für den Einsatz von vorgefertigter Software, die allerdings nicht nach Norm entwickelt wurde.

Wenn funktionale Sicherheit über Software auf programmierbaren Geräten sichergestellt werden soll, gelten weitere Maßnahmen, die je nach gefordertem SIL getroffen werden müssen. Da es allerdings keine zufälligen Fehler in Software gibt, zielen die Maßnahmen auf das Verhindern von systematischen Fehlern ab. Das Übel soll also an der Wurzel gepackt und die Integrität der Software durch strukturierte und zielgerichtete Methoden und Techniken erreicht werden.

Bild 3: Software-Integrität durch strukturierte und zielgerichtete Methoden

Software-Tools validieren

Die gute Nachricht an dieser Stelle geht an all jene, die bereits strukturiert und prozessorientiert Software entwickeln, sprich, allgemein anerkannte Software-Engineering-Praktiken anwenden. Der zusätzliche Aufwand hält sich in diesem Fall in Grenzen. Das Erstellen einer Software-Architektur sowie das Software-Design und Modul-Design zählen ebenfalls zu diesen Best Practices.

Neu mag allerdings die Notwendigkeit sein, sich auch die Software-Tools näher anzusehen, mithilfe derer Software realisiert wird. Je nach potentiellen Auswirkungen von Fehlern dieser Tools sind Validierungen durchzuführen, um sicherzustellen, dass über die Verwendung der Tools keine erhöhten Ausfallraten der Safety-relevanten Softwarefunktionen zu erwarten sind.

Verifikation und Validierung durch entsprechende Tests sind essentiell. Da Software häufig verändert wird, besteht die Notwendigkeit, solche Veränderungen über einen eigens dafür definierten Prozess in die Softwareentwicklung einfließen zu lassen. Dieser Prozess kann selbst definiert werden, aber er muss z.B. sicherstellen, dass nur freigegebene Änderungsanforderungen auch tatsächlich umgesetzt werden und dass über eine Impact-Analyse festgestellt wird, welche Phasen des Lebenszyklus von solchen Änderungen betroffen sind. Nur so lässt sich sicherstellen, dass das System als Ganzes weiterhin sicher entwickelt wird.

Training erlaubt effizientes Einarbeiten und schnelle Umsetzung

Funktionale Sicherheit ist also über weite Strecken das Anwenden von Best Practices und das Umsetzen der Anforderungen aus den Normen auf strukturierte Art und Weise. Sich in Normen einzulesen ist keine einfache Sache. Man verliert schnell den Überblick über das große Ganze, und auch die Ausdrucksweise der Normentexte lässt keine einfache Lesart zu.

Ein kompaktes Training zu den wichtigsten Punkten erlaubt eine effiziente Einarbeitung und schnelle Umsetzung.

Autor: Marcus Gößler

Nach seinem Studium der Elektrotechnik an der Technischen Universität Graz begann die berufliche Laufbahn von Marcus Gößler als Field Application Engineer für analoge und digitale Produkte im Bereich Luft-und Raumfahrt. Weitere Applikationsfelder umfassten Audio/Video, portable Systeme und Infotainment im Automobil. Er leitete Applikationsorganisationen in Zentral- und Osteuropa und zeichnete Verantwortung für große Halbleiterhersteller im Vertriebskanal und Marketing. Bei MicroConsult ist er heute als Trainer und Coach im Bereich Embedded Systems tätig, mit Schwerpunkten in sicherheitsrelevanten Anwendungen und Multicore-Bausteinen.

Weiterführende Informationen

MicroConsult Training & Coaching zum Thema Qualität, Safety & Security

MicroConsult Fachwissen zum Thema Qualität, Safety & Sicherheit

MicroConsult Training & Coaching zum Thema Test und Debug

MicroConsult Fachwissen zum Thema Test und Debug

MicroConsult Training & Coaching zum Thema Prozessmanagement