Eine industrielle Fallstudie zur teilautomatischen Erzeugung von Komponentenfehlerbäumen aus Simulink-Modellen

Suryo Buono, Viktor Ramich, Bernhard Kaiser, Justyna Zander

Berner&Mattner Systemtechnik GmbH
Gutenbergstr. 15
10587 -Berlin
{suryo.buono|bernhard.kaiser|justyna.zander}@berner-mattner.com
viktor.ramich@googlemail.com

Kurzfassung: Seit mehreren Jahren wird an einer stärkeren Integration von modellbasierter Systementwicklung und Sicherheitsanalyse geforscht. In diesem Paper werden ein Ansatz und ein Werkzeug zur besseren und frühzeitigen Verzahnung zwischen der Systementwicklung und der Sicherheitsanalyse präsentiert. Im Vordergrund des Ansatzes steht die Verknüpfung der in der Industrie etablierten Werkzeuge für die Systemmodellierung (Matlab Simulink) und die Sicherheitsanalyse (Enterprise Architect). Ziel war es, die hierarchische Struktur von Komponenten-Fehlerbäumen (engl. *Component Fault Trees/ CFTs*) aus existierenden Simulink-Modellen zu generieren. Ein Transformationsalgorithmus, welcher die hierarchische Struktur und Signalflussinformationen von Simulink-Modellen analysiert und anschließend CFT-Rahmen mit möglichen Fehlermodi in Abhängigkeit von den in Simulink vordefinierten Signaltypen bildet, wurde im Rahmen einer Masterarbeit entwickelt. Anhand eines Fallbeispiels an einem vereinfachten elektrischen Antriebs für E-Fahrzeuge werden in diesem Paper die Vorgehensweise des Ansatzes sowie Erfolge und angetroffene Probleme erläutert.

1 Einführung

Die Komplexität und der Funktionsumfang heutiger Entwicklungen sind weiterhin zunehmend. Automotive Systeme unterliegen kurzen Entwicklungszyklen, einer große Variantenvielfalt und können meist nur in paralleler Zusammenarbeit mit mehreren Industriepartnern termingetreu entwickelt werden. Es bedarf daher Techniken, die die arbeitsteilige Entwicklung, die Wiederverwendung und eine enge Verzahnung von Entwicklungs- und Safety-Prozessen unterstützen [Pk12]. Ein systematisches Verfahren zur Analyse sicherheitskritischer Systeme und eine enge Integration von den System-Modellen und Safety-Modellen werden in dieser Arbeit präsentiert.

Die Wiederverwendung und die arbeitsteilige Entwicklung kann durch die Modularisierung von Systemen begünstigt werden. Modellbasierte Entwicklungswerkzeuge, wie z.B. das etablierte Werkzeug Matlab Simulink, werden heute im Bereich der Automobilin-

dustrie eingesetzt. Zur Analyse der sicherheitskritischen Systeme kann der Ansatz der Komponenten-Fehlerbäume (engl. *Component Fault Trees/ CFT*) [KLM03] verwendet werden. Dieser Ansatz strukturiert den Fehlerbaum modular, entsprechend der komponentenbasiert aufgebauten Systemarchitektur. Im Vergleich dazu sind klassische Sicherheitsanalysemodelle wie FMEA oder FTA in ihrer Ausdrucksfähigkeit und Modellierung eingeschränkt und können nicht ohne Weiteres das Potential von modular aufgebauten Systemarchitekturen nutzen [DT09].

Um die Sicherheitsanalyse enger mit dem Entwicklungsprozess zu verzahnen und somit eine effizientere Analyse zu erreichen, wurde der Ansatz verfolgt, die im Entwicklungsprozess erstellten Simulink-Modelle auch für die Sicherheitsanalyse zu nutzen. Ein Transformationsalgorithmus wurde in der Masterarbeit von Ramich [Rv14] entwickelt. Die Idee basiert darauf, mit Hilfe eines Algorithmus das in Simulink modellierte System automatisch in CFT-Rahmen mit möglichen Fehlermodi zu transformieren. Anschließende Analysen (qualitative sowie quantitative) können manuell durch den Benutzer durchgeführt werden. Als Frontend für die grafische Oberfläche wird das etablierte Werkzeug „Enterprise Architect (EA)“ eingesetzt. Die Erweiterung des Plugins in EA für das Importieren von Simulink-Modellen ist in Kooperation mit dem Fraunhofer IESE im Rahmen des Forschungsprojektes SPES-XT entstanden.

Anhand eines vereinfachten elektrischen Antriebes wird die Vorgehensweise des Ansatzes in Ramich [Rv14] beispielhaft verifiziert und validiert. Angetroffene Probleme und Erfolge sowie zukünftige Verbesserungen werden in dieser Arbeit beschrieben.

2 Konzept der Transformationen

Das Grundkonzept der teilautomatischen Erzeugung von Komponenten-Fehlerbäumen aus Simulink-Modellen von Ramich [Rv14] beinhaltet die in Abbildung 1 dargestellten Schritte. In (1) findet ein Import der Simulink-Modelldatei (*.mdl) statt, um den darin enthaltenen Systemaufbau mit den Signal-/Datentypen der Ports zur Generierung der CFTs (2) zu nutzen. Anschließend werden die erzeugten CFT-Rahmen in (3) angezeigt sowie eine gegebenenfalls nötige Nachbearbeitung und das Einfügen von Fehlerbäumen ermöglicht.

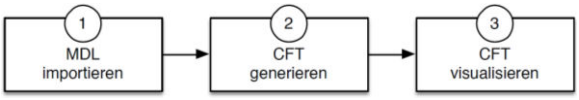

Abbildung 1: Grundkonzepts des Transformationsalgorithmus [Rv14]

2.1 Simulink-Modelldatei importieren

In Zusammenarbeit mit dem Fraunhofer IESE wurde das entwickelte Plugin für Enterprise-Architect um einen Menüeintrag zum Import von MDL-Dateien erweitert. Dies ermöglicht den Import einer Simulink-Modelldatei und mit Hilfe eines Parsers der Open-

Source-Bibliothek *Simulink Library für Java*[1] zu analysieren und relevante Modellinformationen zur Verfügung zu stellen. Dabei werden die Daten in das Komponenten-Daten-Modell (engl.: *Component-Data-Model (CDM)*) transformiert, welches in Abbildung 2 dargestellt ist. Darin wird für jedes Simulink-Subsystem mit den modellierten In- und Outports und angehängten Verbindungen ein entsprechendes Komponenten-Objekt (Component), Port-Objekt und Link-Objekt auf Basis des CDM erstellt. Über die child-/parent-Beziehung von Komponenten-Objekten wird die hierarchische Struktur des Simulink-Systems im CDM abgebildet.

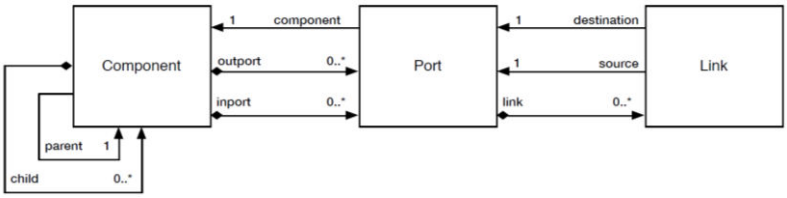

Abbildung 2: Component-Data-Model (CDM) [Rv14]

2.2 CFT generieren

Im nächsten Transformationsschritt werden die Modelldaten auf Basis des CDM in das Component-Data-Model with Failure-Modes (CDMFM) überführt. Dabei werden, abhängig vom Datentyp der Ports der zu übertragende Signale, festgelegte Standard-Fehlermodi vorgeschlagen und dem jeweiligen Port des CFT angehängt.

Abhängig vom Datentyp (*Double*, *Integer* oder *Boolean*) des jeweiligen Ports werden bei der CFT-Generierung jedem In- und Outport des Systems Fehlermodi auf Basis der Veröffentlichungen [Dd11], [FM93] und [Pd99] zugeordnet. Eine zusammenfassende Auflistung ist in Tabelle 1 aufgeführt. Es handelt sich dabei nicht um technische Fehlermodi wie z.B. Kurzschluss, Drift etc. sondern um funktionale Fehlermodi.

Tabelle 1: Abhängig vom Datentyp automatisch vorgeschlagene Fehlermodi [KLM03]

	Double, Integer	Boolean
1	too high	true when false
2	too low	false when true
3	too early	too early
4	too late	too late
5	omission	omission
6	commission	commission

Um aus dem komponentenbasierten CDMFM die gewünschten CFTs zu generieren, ist eine finale Transformation nötig. Die CFTs sind auf Basis des Open Safety Model (OSM) [Rj13] zu erstellen, um eine Kompatibilität zu den vom Fraunhofer IESE entwickelten Algorithmen zu erzielen. OSM ist ein Meta-Model, welches die Handhabung von verschiedenen Sicherheitsanalysetechniken ermöglichen soll. Für weitere Informationen zum komplexen Transformationsalgorithmus sei auf die detaillierte Ausführung in [Rv14] verwiesen. Da der Algorithmus die funktionale Architektur des Simulink-

[1] http://www.cqse.eu/en/products/simulink-library-for-java/overview/

Modells exakt adaptiert, werden auch vorhandene Rückführkreise übernommen. Das Vorhandensein einer Schleife innerhalb einer Fehlerbaum-Analyse widerspricht der Definition von FTs bzw. CFTs. Diese muss zur Analyse unterbrochen werden. Innerhalb der Arbeit von Ramich [Rv14] wird keine automatische Erkennung von Schleifen durch den Algorithmus unterstützt. In [Vw02], [Yj97] und [Dd11] sind unterschiedliche Methoden und Vorgehen zur Erkennung von Schleifen erläutert, welche in den Algorithmus implementiert werden können.

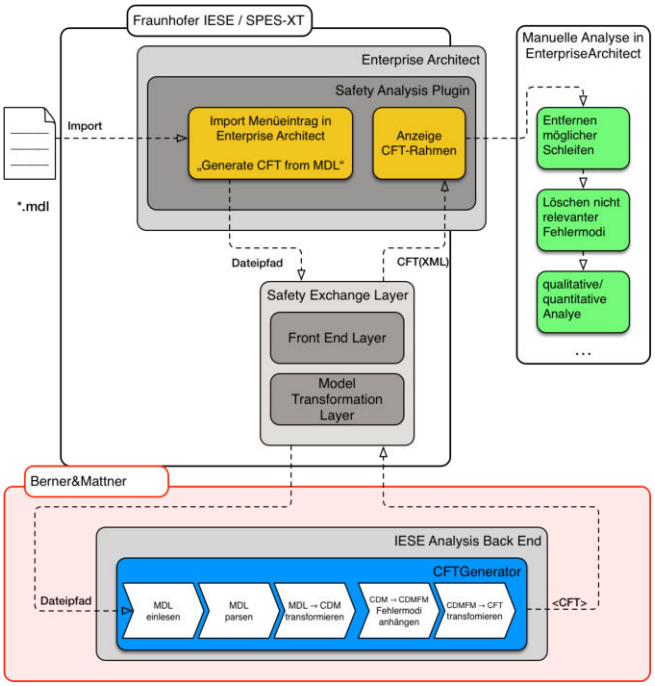

Abbildung 3: Zusammenfassung der Transformationsschritte zur Erzeugung von CFTs [Rv14]

2.3 CFT visualisieren und Einordnung der Arbeit

Um die erstellten CFTs in EA darzustellen, sind die dazu nötigen Informationen, die Daten des erzeugten CFT-Objekts, von der Java-Anwendung zu EA zu übertragen. Der Austausch findet mit Hilfe des von [Rj13] und [DT08] entwickelten Safety Exchange Layers (dt.: Sicherheitsaustauschschicht) statt. Die kooperative Integration wurde im Entwicklungsprojekt SPES-XT durchgeführt. Die Abbildung 3 stellt das Gesamtkonzept dar.

3 Praktische Anwendung der teilautomatischen Erzeugung von CFT aus Simulink-Modellen

3.1 Fallbeispiel eines vereinfachten elektrischen Antriebs in Simulink

Als Fallbeispiel für die Verifizierung und Validierung der teilautomatischen Erzeugung von CFT-Modellen aus Simulink-Modellen wird ein vereinfachter DC-Motor mit Drehzahlregelung eingesetzt. Das System besteht aus Sensor, Regler, Leistungselektronik und Motor. Es ist als geschlossener Regelkreis zu betrachten. Das mit der CFT Methode zu analysierende Simulink-Modell des Systems ist in Abbildung 4 dargestellt. Die Führungsgröße wird durch die Solldrehzahl am Eingang *IN_w_target* vorgegeben. Die Regelgröße wird am Ausgang *OUT_w* ausgegeben. Das Subsystem *w_Sensor* ist durch einen Gain-Block implementiert. Die Referenzgröße kann damit skaliert werden. Im Subsystem *PI_Control* wird die Regelabweichung zwischen dem Sollwert (*w_target*) und dem Istwert (*w_act*) der Drehzahl bestimmt. Ein PI-Regler-Block aus der Simulink Library wurde für die Regelungsfunktion eingesetzt. Das Subsystem *Power_Electronics* wurde vereinfacht und besteht aus einem Gain-Block mit einem Verstärkungsfaktor. Das Modell des *DC_Motors* wurde aus Differentialgleichungen hinsichtlich der elektrischen und mechanischen Zusammenhänge des DC-Motors hergeleitet. Für die Herleitung und Umsetzung des Subsystems DC_Motor ist auf die Arbeit von Ramich [Rv14] verwiesen.

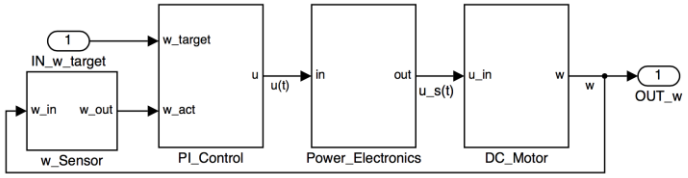

Abbildung 4: Systemaufbau des Simulink-Modells des Fallbeispiels

3.2 Erzeugung von Fehlerbäumen aus dem Simulink-Modell

Die Darstellung der automatisch generierten CFT-Rahmen mit vorgeschlagenen Fehlermodi erfolgt in dem etablierten Werkzeug „Enterprise Architect“. Wird die Simulink Modelldatei importiert, so werden die in Kapitel 2 erklärten Transformationen automatisch durchgeführt bis die CFT-Rahmen vorliegen.

Das Fallbeispiel entspricht einem typischen Modell eines geschlossenen Regelkreises, das einen Rückführkreis der Regelgröße und somit eine Schleife besitzt. Das Werkzeug übernimmt die Schleifen aus dem Simulink-Systemmodell in die CFT-Rahmen. Die automatisch generierten CFT-Rahmen sind in Abbildung 5 dargestellt. Da in Fehlerbäumen laut Definition keine Schleifen vorhanden seien dürfen, muss der Safety-Ingenieur die Schleifen manuell auflösen. Dieses Entfernen der Schleifen wird z.B. durch das Löschen der Feedback-Variable wie in (vgl. [Dd11]) beschrieben erreicht. Eine automatische Erkennung von Schleifen wird vom entwickelten Werkzeug derzeit nicht unterstützt. Im Fallbeispiel kann dies durch das Entfernen der Eingangsfehlermodi des

Drehzahlsensors w_Sensor erreicht werden. Dies entspricht dem Löschen der Feedback-Variable.

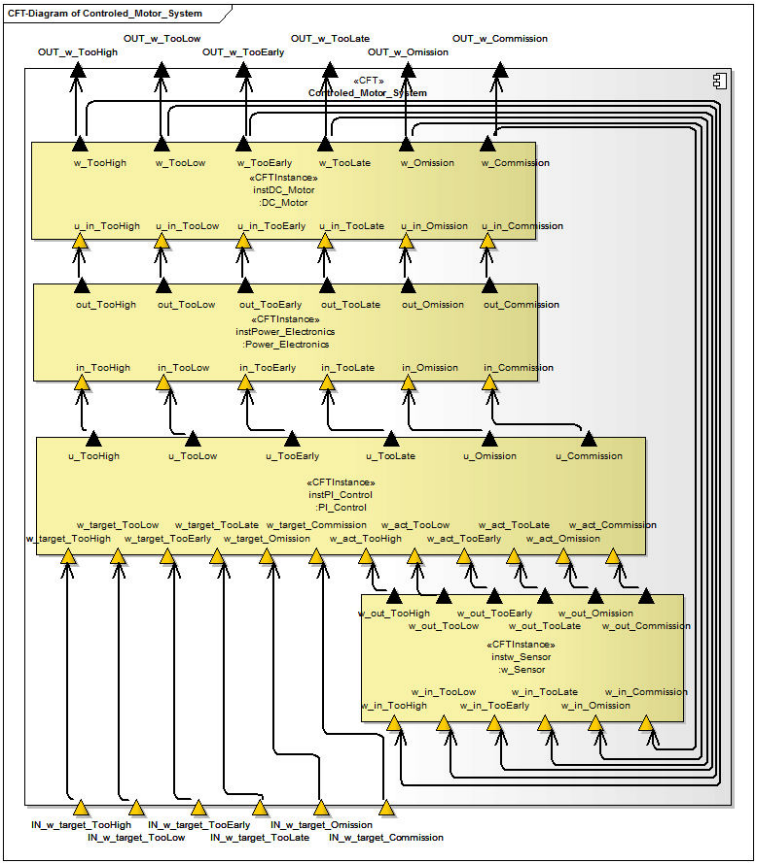

Abbildung 5: Durch das Werkzeug erzeugte CFT-Rahmen (Rückführkreis vorhanden)

3.3 Anschließende Analysen der generierten Komponenten-Fehlerbäume in EA

Nach der automatischen Generierung der CFT-Rahmen aus Simulink-Modellen sind relevante Fehlermodi aus den automatisch vorgeschlagenen Fehlermodi durch den Safety-Ingenieur auszuwählen. Die Auswahl ist abhängig vom betrachteten Kontext (z.B. aus einer Gefahren- und Risikoanalyse). Innerhalb der Sicherheitsanalyse sollten ausschließlich Fehler betrachtet werden, die nicht durch die Regelung kompensiert werden und zu einer Gefährdung (engl. *Hazard*) führen können. Somit ist z.B. der Fehlermodus „TooHigh“ als Abweichung vom Sollwert zu sehen, welcher eine Gefahr für das System darstellt.

Die Einordung und Auswahl relevanter Fehlermodi sollte parallel mit dem Vervollständigen der CFT-Rahmen durch Fehlerbäume erfolgen, da beides Systemverständnis vo-

raussetzt. So kann im DC-Motor-Beispiel ein zu großes Ausgangssignal (*u_TooHigh*) des Blockes *PI_Control* eintreten (siehe fett gedruckte Verbindungslinien in Abbildung 6), wenn der Drehzahlsollwert bereits fehlerhaft zu groß als Eingangsfehlermodus anliegt (*w_target_TooHigh*) oder der aktuelle Drehzahlwert zu klein vom Sensor gemessen wurde (*w_act_TooLow*). Des Weiteren kann z.B. ein Ausfall der Drehzahlmessung (*w_act_Omission*) oder auch ein interner Fehler im Mikroprozessor ebenso ein fehlerbedingt erhöhtes Ausgangssignal (*u_TooHigh*) verursachen. Diese anschließende Analyse, u.a. die Entfernung von Schleifen und die Auswahl von relevanten Fehlermodi, fordert die Analyse-fähigkeit des Safety-Ingenieurs.

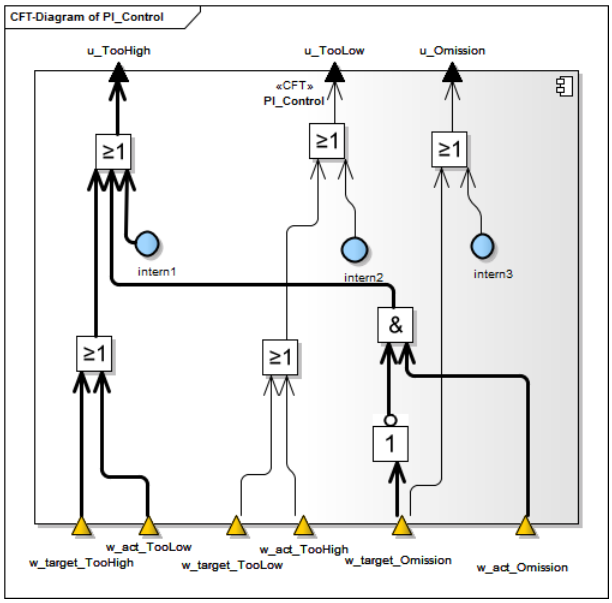

Abbildung 6: vervollständigter CFT mit Fehlerbäumen für die PI-Regelung

3.4 Änderung des Simulink-Modells und Wiederverwendung von CFTs

Bei einer Änderung des Simulink-Modells wurde die Wiederverwendung von bereits erstellten CFT-Modellen untersucht. Das System des Fallbeispiels wurde durch einen weiteren Sensor zur Drehzahlmessung erweitert (siehe Abbildung 7), um eine fehlerbedingt zu hohen Drehzahl des Motors durch Redundanz der Sensoren zu erkennen und zu verhindern. Die beiden Sensoren sollten dabei idealerweise mit unterschiedlichen Messprinzipien arbeiten. Existiert eine Abweichung zwischen den Messwerten der Sensoren (mit einer vorgegebenen Toleranz), so wird der Übergang in den sicheren Zustand eingeleitet, in diesem Fall die Abschaltung der Versorgungsspannung des Motors. Das Abschalten der Versorgungsspannung wird in der Praxis z.B. in der Leistungselektronik umgesetzt. Innerhalb des Fallbeispiels geschieht dies zur Vereinfachung innerhalb des Reglerblocks, um im Weiteren ausschließlich diesen Block betrachten zu müssen.

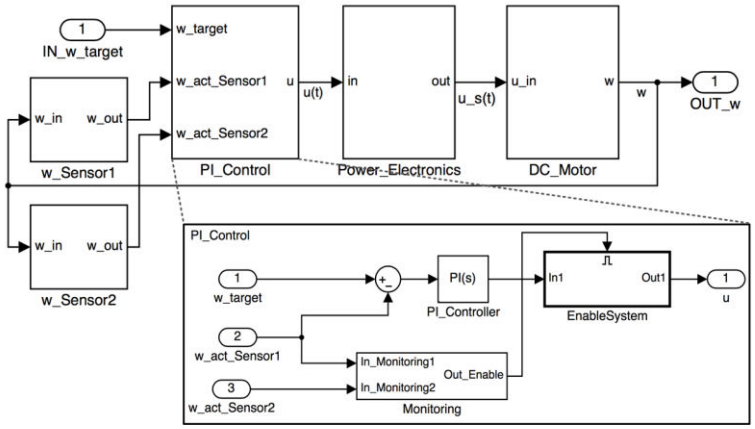

Abbildung 7: Erweitertes Simulink-Modell

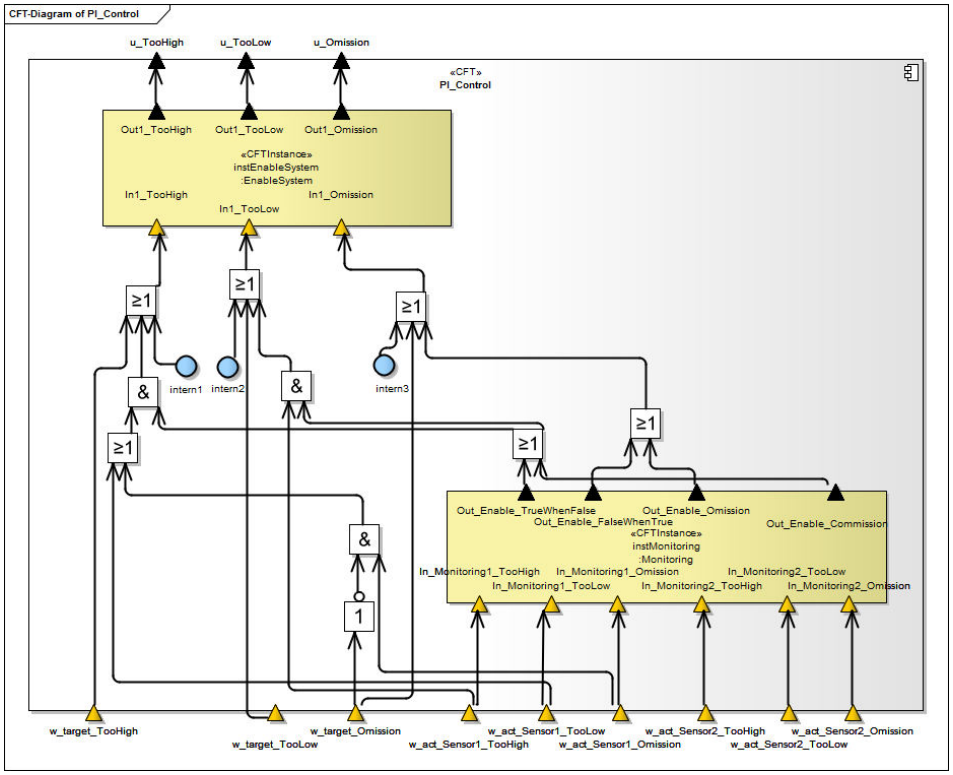

Abbildung 8: CFT des erweiterten Reglerblocks

Zur Analyse wird das erweiterte Simulink-Modell erneut in Enterprise Architect eingelesen, um die entsprechenden CFTs automatisch zu generieren. Nicht geänderte Systemkomponenten, wie z.B. der DC-Motor können durch den bereits zuvor erstellten CFT wiederverwendet bzw. der Fehlerbaum in die neue CFT-Instanz übernommen werden. Selbst für geänderte Komponenten, wie z.B. das Subsystem *PI_control* mit dem hinzu-

gefügten Eingangsignals *w_act_Sensor2*, können Teilbereiche des bereits zuvor analysierten CFT übernommen werden. Es ergibt sich z.B. der in Abbildung 8 dargestellte CFT.

4 Zusammenfassung und Ausblick

Ein Beitrag zur Verzahnung der klassischen Systementwicklung mit den Sicherheitsaktivitäten (Safety) wurde bei Berner&Mattner durch die entwickelte Vorgehensweise und dem vorgestellten Werkzeug geleistet. Dies ermöglicht die teilautomatische Generierung von Komponenten-Fehlerbäume (CFT), basierend auf der Systemarchitektur eines Simulink-Modells. Vorteilhaft dabei ist, dass das Simulink-Modell, welches innerhalb der Systementwicklung entwickelt wurde, mit Hilfe des entwickelten Werkzeuges auch für die Sicherheitsanalyse verwendet und damit die Sicherheitsanalyse frühzeitig in das Systemdesign einbezogen werden kann. Einsatzmöglichkeiten wurden an Fallbeispielen zwecks Verifizierung und Validierung des Werkzeuges demonstriert. Dazu wurde in der Fallstudie eine ‚Kriterium-Checkliste' zur Verifizierung des Algorithmus erstellt sowie eine eingeschränkte Evaluation des Arbeitsresultates hinsichtlich des Erreichens der definierten Ziele (effiziente Verzahnung der klassischen Systementwicklung mit den Sicherheitsaktivitäten, frühzeitige Einbeziehung der Sicherheitsanalyse in das Systemdesign, Modularisierung etc.) durchgeführt. Ausführliche Ergebnisse der Fallstudie sind in der Arbeit von Ramich [Rv14] präsentiert.

Es wurde gezeigt, dass die Wiederverwendung von CFTs von zuvor analysierten Komponenten aus aktuellen oder vorherigen Projekten die Sicherheitsanalyse beschleunigt. Dies trägt nicht nur zur Effizienzsteigerung, sondern auch zur Verbesserung der Qualität des Ergebnisses bei und ermöglicht eine arbeitsteilige Analyse. Da bei einem architekturnahen CFT nur die Ports der bestehenden Systemarchitektur zur automatischen Generierung der Fehlermodi berücksichtigt werden, besteht nun bei der Sicherheitsanalysemodellierung die Gefahr, dass nicht alle sicherheitsrelevanten Fehlermodi (v.a. durch Umwelteinflüsse) berücksichtigt worden sind, weil ein in dieser Fehlermodi zugeordnetes Signal im Systemmodell nicht modelliert wurde. Die elektromagnetische Verträglichkeit sowie die Umwelttemperatur (zu hoch/ zu klein) werden beispielweise in der Systemarchitektur oft nicht modelliert. Dies führt dazu, dass bei der teilautomatischen Erstellung der CFT-Rahmen mögliche Fehlermodi zu diesen äußeren Umwelteinflüssen nicht mitberücksichtigt sind.

Neben dem automatischen Erkennen von Schleifen in CFT-Modellierung, könnte im nächsten Entwicklungsschritt die Berücksichtigung der Operationsblöcke der Simulink-Modelle, z.B. Additions- und Subtraktionsblöcke, erfolgen. Dies würde eine Übernahme der Signal-Kausalitäten ermöglichen und so die Qualität der Sicherheitsanalyse verbessern. Außerdem ist die Anbindung der teilautomatisch generierten Sicherheitsanalyse mit der contract-basierten Modellierung [Ss13] erstrebenswert. Eine Verletzung eines von Contracts, welche zu einer Gefährdung führen kann, könnte somit als relevante Fehlermodus in CFT interpretiert werden.

Literaturverzeichnis

[Dd11] Domis, D.: Integrating fault tree analysis and component-oriented model-based design of embedded systems. Verl. Dr. Hut, 2011.

[DT09] Domis, D.; Trapp, M.: Component-Based Abstraction in Fault Tree Analysis. In Computer Safety, Reliability, and Security. Springer Verlag, Berlin, 2009; S. 297-310.

[DT08] Domis, D.; Trapp, M.: Integrating Safety Analyses and Component-Based Design. In Proceedings of the 27th International Conference on Computer Safety, Reliability, and Security. Springer Verlag, Berlin, Heidelberg, 2008; S. 58-71.

[FM93] Fenelon, P.; McDermid, J.: An Integrated Toolset for Software Safety Analysis. In Journal of Systems and Software (3), July 1993; S. 279-290.

[KLM03] Kaiser, B.; Liggesmeyer, P; Mäckel, O.: A New Component Concept for Fault Trees. In SCS. In Proceedings of the 8th Australian workshop on Safety critical systems and software-Volume 33. Australian Computer Society, 2003; S. 37-46.

[Pk12] Pohl, K.; Hönniger, H.; Achatz, R; Broy, M.: Model-based Engineering of Embedded Systems: The SPES 2020 Methodology. Springer, 2012.

[Pd99] Pumfrey, D.: The Principled Design of Computer System Safety Analyses. University of York, 1999.

[Rv14] Ramich, V.: Teilautomatische Erstellung von Component-Fault-Trees aus Simulink-Modellen. Master thesis, Universität Kassel, 2014.

[Rj13] Reich, J.: Konzeption und Entwicklung eines Werkzeugs zur Spezifikation und Analyse von Safety-Modellen auf Basis von Enterprise Architect. Master thesis, Technische Universität Kaiserslautern, 2013.

[Ss13] Sonksi, S.: Contract-based modeling of component properties for safety-critical systems. Master thesis, Hochschule Darmstadt, 2013.

[Vw02] Vesely, W.; Stamatelatos, M.; Dugan, J.; Fragola, J.; Minarick, J.; Railsback, J.: Fault Tree Handbook with Aerospace Applications. NASA Office of Safety and Mission Assurance, 2002.

[Yj97] Yang, J.: Analytic Method to Break Logical Loops Automatically in PSA. In Reliability engineering & system safety. Elvesier, 1997; S. 101-106.