Vergleich einer graphenbasierten Methode mit der Klassifikationsbaummethode für die Testfallermittlung anhand einer automotiven Fallstudie

Friedrich Beidinger[1], Axel Hollmann[2], Markus Kleinselbeck[1], Dr. Wolf Ritschel[1]

[1] Hella KGaA Hueck & Co.,
Rixbecker Straße 75, 59552 Lippstadt
{Markus.Kleinselbeck, Wolf.Ritschel}@hella.com, fbeidinger@googlemail.com

[2] Fachgebiet Angewandte Datentechnik, Universität Paderborn,
Warburger Str. 100, 33098 Paderborn
hollmann@adt.upb.de

Abstract: Mechanische Steuerkomponenten, wie die in modernen Automobilen, werden mehr und mehr durch elektronische verdrängt. Diese sind meistens mit einem Steuergerät gekoppelt. Um die Funktionsfähigkeit dieser Steuergeräte zu gewährleisten, ist ein systematisches Testen notwendig. Die Automatisierung der Tests soll nicht nur die Qualität steigern, sondern auch die Effizienz erhöhen, um die Kosten zu reduzieren. Während die eigentliche Durchführung der Tests mittlerweile einen hohen Automatisierungsgrad erreicht hat, werden die Testfälle oft manuell erstellt. In diesem Beitrag wird der Einsatz der Ereignis-Sequenz-Graphen-basierten Methode untersucht, die aus einer formalen Beschreibung eines Steuergerätes die Erzeugung von Testfällen ermöglicht. Diese Methode wird dann mit der Klassifikationsbaummethode anhand einer Fallstudie verglichen.

1 Einführung und Motivation

Die Komplexität der elektronischen Systeme im Automobil wächst bei gleichzeitiger Verringerung der Entwicklungszeit. Die Integration der Steuergeräte zu einem funktionierenden Gesamtsystem wird immer schwieriger, da die Abhängigkeiten der Geräte voneinander immer höher werden. In diesem Beitrag werden die Erfahrungen bei der Anwendung einer graphenbasierten Methode (*Ereignis-Sequenz-Graphen*) zur Ermittlung von Testschritten auf ihre Anwendbarkeit im Testprozess für automotive Anwendungen bei der Firma Hella betrachtet. Zudem erfolgt ein Vergleich der Leistungsfähigkeit mit der gegenwärtigen Methode des manuellen Testens mit der Klassifikationsbaummethode. Dieses wird anhand eines Fallbeispiels eines Fahrerassistenzsystems zur automatischen Abstandsregelung durchgeführt.

In dem Fallbeispiel kommt ein Testsprozess zur Anwendung wie in Tabelle 1 dargestellt. Das betrachtete Steuergerät hängt direkt von der Korrektheit der Signale von vier weiteren Steuergeräten ab.

Etwa 2000 Anforderungen werden hierbei zu 400 Testzielen zusammengefasst, die in 4000 Testschritten umgesetzt werden. Die in der Tabelle angegebenen Aufwände sind empirisch für das Projekt ermittelt worden. Die Vermutung liegt daher nahe, dass eine Optimierung der Anzahl der Testschritte im Testentwurf direkt Zeit bei der Umsetzung der folgenden Prozessschritte einspart.

Prozess	Durchführung	Aufwand
Testplanung	manuell	gering 5%
Testspezifikation	manuell / geringe Toolunterstützung	mittel 20%
Testentwurf	manuell	mittel 20%
Testimplementierung	manuell	hoch 30%
Testdurchführung	automatisch	gering 5%
Testauswertung	manuell / geringe Toolunterstützung	mittel 20%

Tabelle 1: Testprozess und Aufwände

Der Beitrag ist wie folgt strukturiert: Im folgenden Kapitel werden Methoden zum Testentwurf vorgestellt. In Kapitel 3 wird das Testobjekt und seine Umgebung vorgestellt, sowie die Modell- und Testfallerstellung beschrieben. In Kapitel 4 werden die Ergebnisse des ESG-Ansatzes mit manuellem Test verglichen. Ein Ausblick schließt den Beitrag ab.

2 Methoden zum Testentwurf

Weit verbreitet in der Praxis ist immer noch die *intuitive und erfahrungsbasierte Testfallermittlung*. Dabei werden vom Testingenieur die Eingabegrößen so variiert, dass die Ausgangsgrößen die zu erfüllenden Kriterien aufweisen. Der Erfolg dieser Vorgehensweise sowie die Anzahl und Effizienz der aufgestellten Testschritte und -fälle hängen stark von der Erfahrung des Testingenieurs ab. Sie sollte eher dazu benutzt werden, um die nach systematischen Methoden aufgestellten Testfälle sinnvoll zu ergänzen [SL05]. Zudem wird das Aufstellen von Negativ-Testfällen [BL07] meistens vernachlässigt, da die Anforderungen in der Regel nur positiv formuliert sind.

Ebenfalls weit verbreit ist die *Klassifikationsbaummethode* [GG93]. Sie ermöglicht über eine Klassifizierung aller Aspekte des Testobjekts die Komplexität zu reduzieren. Die Grundlage für diese Methode bildet die Systemspezifikation. Bei der Erstellung eines Klassifikationsbaums wird zunächst die Komplexität des in der Spezifikation verankerten Problems zerlegt. Durch die Analyse der Spezifikation werden einzelne für den Test relevante Aspekte (Klassifikationen) der Problemstellung herausgefiltert. Mehrere Klassifikationen können zu einer Komposition zusammengeführt werden. Der Wertebereich einzelner Klassifikationen wird betrachtet und in sinnvolle Klassen zerlegt.

Die Klassen werden so gewählt, dass alle Werte aus diesen als äquivalent betrachtet werden können (Äquivalenzklassen). Diese Äquivalenzklassen können nach der Betrachtung zusätzlicher Bedingungen weiter klassifiziert werden. Durch die wiederholte Bildung von Unterklassifikationen wird der Klassifikationsbaum erstellt.

Beschreibt ein Klassifikationsbaum ein dynamisches Verhalten eines Systems, so können die Blätter als Ereignisse aufgefasst werden. Diese sind gleichzeitig der Kopf einer Kombinationstabelle, wobei jede Zeile einen Testfall spezifiziert.

Ereignis-Sequenz-Graphen (ESG) [Be01] werden benutzt, um das Verhalten eines Systems und die Interaktion mit einem Benutzer bzw. der Umgebung zu modellieren. Ereignisse sind Eingaben des Benutzers und der Umgebung oder stellen Antworten des Systems dar. Ein Ereignis-Sequenz-Graph $ESG=(V,E)$ besteht aus einer endlichen Menge von Ereignissen V und einer endlichen Menge von Kanten $E \subseteq V \times V$. Eine Kante $(v,v') \in E$ gibt an, dass das Ereignis v' dann ausführbar ist, wenn das Ereignis v ausgeführt wurde. Die Pseudoknoten $[$ und $]$ markieren zulässige Start- bzw. Endereignisse. Der Graph beinhaltet somit alle gültigen Ereignissequenzen. Der inverse Graph $\overline{ESG} = (V, \overline{E})$ ist gegeben durch die Menge der Ereignisse V und die Kantenmenge $\overline{E} = (V \times V) \setminus E$. Er repräsentiert das ungültige Verhalten des Systems und wird zur systematischen Generierung von Negativtests benutzt. In Abbildung 1 ist beispielhaft der Graph $ESG=(V,E)$ mit $V=\{a,b,c\}$ und $E=\{(a,b),(a,c),(b,c),(c,b)\}$, der inverse Graph $\overline{ESG}$ mit $V=\{a,b,c\}$ und $\overline{E} = \{(a,a),(b,a),(b,b),(c,a),(c,c)\}$ und der komplettierte Graph gegeben.

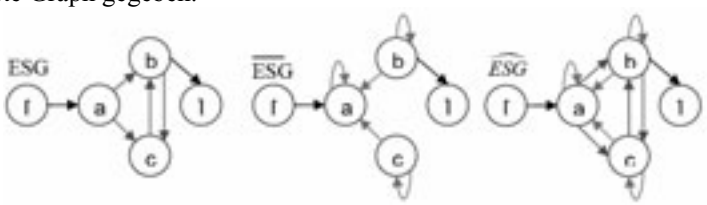

Abbildung 1: Ereignis-Sequenz-Graph, inverser und komplettierter Graph

3 Bewertungskriterien und Fallbeispiel

3.1 Anforderungen an die Testmethode

Die Anwendbarkeit der ESGs wurde bereits für das Testen von graphischen Oberflächen erfolgreich nachgewiesen [BHN07]. Das Testen von Steuergeräten besitzt andere Rahmenbedingungen. Die Anforderungen sind nur in textueller Form festgeschrieben. Die Steuergeräte sind physikalisch über CAN-Bus, Spannungsversorgung und mechanischer Halterung mit dem Fahrzeug verbunden.

Die Sensorik und die Einbindung des Steuergerätes in die Umwelt werden in dem Fallbeispiel nicht betrachtet, sondern nur der Teil, der für die Fahrzeugumgebung relevant ist. Da der Test der Fahrzeugintegration für fast alle Steuergeräte ähnlich ist, kann hier sehr gut die gegenwärtige Testmethode für den Vergleich mit der Anwendung von ESGs herangezogen werden.

3.2 Auswahl des Testobjekts

Das Steuergerät (ein Radarsensor auf 24GHz Technik) ist über den CAN-Bus mit fünf weiteren Steuergeräten und so indirekt mit mehr als 7 Sensoren gekoppelt und wird über die Standardspannungsversorgung mitversorgt. Von dem Gesamttest wurde ein Minimalfall (8 Zustände, abhängig von ca. 4 Eingangsgrößen) und ein Maximalfall (ca. 20 sog. Test-Targets, abhängig von über 20 Eingangsgrößen, zentraler Zustandsautomat) ausgewählt.

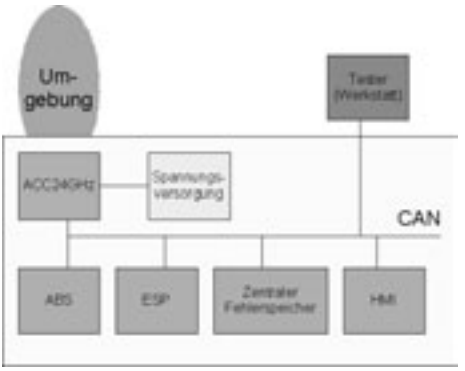

Abbildung 2: Abhängigkeit der Steuergeräte über das CAN-Netzwerk

Abbildung 2 zeigt die physikalischen Abhängigkeiten des Testobjekts (ACC24GHz:Adaptive Cruise Control, 24GHz Radar) zu den übrigen Steuergeräten. Der zentrale Zustandsautomat ist ein besonders komplexes und umfangreiches Testobjekt, da er in direkter Beziehung zu den Signalen aller aufgezeigten Steuergeräte steht. Aufgrund der sich kontinuierlich ändernden Anforderungen während der Entwicklung wurden zahlreiche Erweiterungen vorgenommen, die sich mit einem Zustandsautomaten nicht konsequent modellieren lassen („regelbasiertes Verhalten“). Abbildung 3 zeigt eine Zusammenfassung der für den Zustandsautomaten relevanten Signale. Grundsätzliche Randbedingung für die Durchführung der Tests war, dass nicht auf interne Signale oder Daten zugegriffen werden soll.

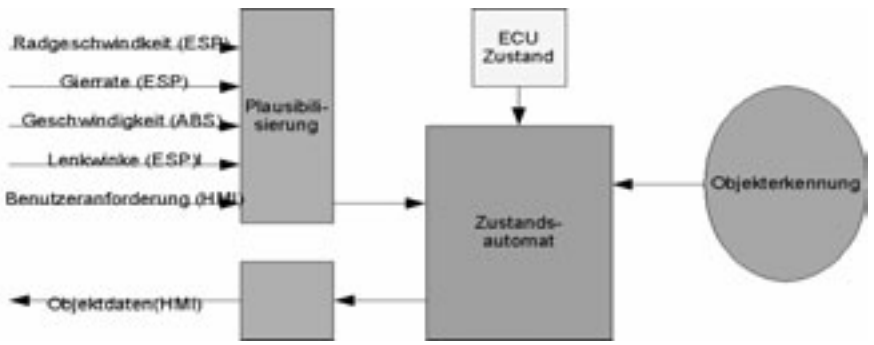

Abbildung 3: Ein- und Ausgangssignale

3.3 Testumgebung

Um die Durchführung der Integrationstests automatisieren zu können, wurde ein Integrationstestplatz eingerichtet. Mit Hilfe des Integrationstestplatzes werden unterschiedliche Situationen, die auf der Strasse auftreten können, simuliert und in den Test eingebunden. Mit Hilfe der SPS und programmierbaren Netzgeräte werden Kurzschlüsse, Fehlpotentiale sowie unterschiedliche Versorgungsspannungskurven am Bus sowie am zu testenden Steuergerät realisiert. Der gesamte Testablauf wird von einem Hostrechner mit der Softwarekomponente CANoe überwacht und durchgeführt. Außer der Überwachung und Steuerung des Testprozesses wird mit Hilfe von CANoe die imaginäre Vernetzung des Steuergerätes im Auto (Restbussimulation) ohne Vorhandensein von reellen Steuergeräten, die unmittelbar mit dem Prüfling über CAN-Bus gekoppelt sein müssen, realisiert. Am Ende eines Tests erstellt CANoe einen Testreport im *.xml oder *.html Format, in dem alle durchgeführten Testschritte von einzelnen Testfällen mit kurzen Beschreibungen aufgelistet sind. Der Testreport dient als Grundlage für die spätere Testauswertung.

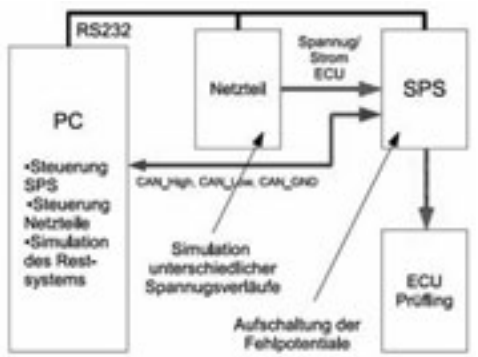

Abbildung 4: Schematischer Aufbau des Integrationstestplatzes

3.4 Anwendung der ESG-Methode und Testfallgenerierung

Die ESG-Methode wurde im Rahmen einer Diplomarbeit auf ihre Anwendbarkeit untersucht [Bei07]. Dazu wurden die Testfälle mit Hilfe der ESG-Methode aufgestellt und auf der beschriebenen Testumgebung durchgeführt. Dabei wurden die Original-Lastenhefte für die Erstellung der Testspezifikation mit herangezogen.

Als Grundlage für die Knoten des ESGs dienen die system- und testumgebungsspezifischen Ereignisse. Die systemspezifischen Ereignisse sind die Ereignisse, die auf der Kommunikationsebene (Netzwerkebene) geschehen. Damit sind die Signale gemeint, dessen Eingang bzw. Ausgang als Ereignis interpretiert wird. Die testumgebungsspezifischen Ereignisse beschreiben dagegen die äußeren Einflüsse auf das System. Zu solchen Ereignissen gehören die Aufschaltungen von Fehlpotentialen sowie Kurzschlüsse im Gesamtsystem. Jedem Ereignis wird ein Programmcode zugeordnet, der beim Auftreten des Ereignisses ausgeführt wird.

Um die Zusammenhänge zwischen einzelnen Ereignissen besser verstehen zu können, müssen einzelne Punkte der Systemspezifikation mit Hilfe von Kombinations- oder Zustandsübergangstabellen zusammengeführt und damit die Aufgabenstellung präzisiert werden. Anhand dieser Tabellen wird dann entschieden, ob einem Ereignis ein oder mehrere Knoten im ESG zugeordnet werden müssen. Die Kantenbildung erfolgt anhand der Systemspezifikation und/oder durch logische Zusammenhänge. Die Erkenntnisse, die bei der Präzisierung der Aufgabenstellung (Aufstellung der Kombinations-, Zustandsübergangstabellen usw.) gewonnen wurden, müssen bei der Kantenbildung berücksichtigt werden. Ausgehend von einem Knoten (Startknoten bevorzugt) im ESG werden zwischen ihm und allen seinen direkten Nachfolgern Kanten angelegt. Diese Prozedur wird solange wiederholt, bis alle Knoten mit ihren direkten Nachfolgern verbunden sind. Beim Anlegen der Kanten dürfen einzelne Knoten nicht isoliert, sondern müssen im Zusammenhang mit allen ihren Vorgängern betrachtet werden. Abbildung 5 zeigt ein Beispiel. Aus den Modellen wurden automatisch Testfälle generiert, so dass sämtliche Ereignispaare (Kanten) überdeckt werden. Diese Testsequenzen wurden mittels des *Chinese Postman Problems* [Ah91] minimiert. Zudem wurden Testfälle aufgestellt, um die ungültigen Ereignispaare des inversen Graphen abzudecken.

Abbildung 5: Beispiel eines Ereignis-Sequenz-Graphen zur Überprüfung der Zustandsübergänge

3.5 Manueller Testprozess mit Hilfe der Klassifikationsbaummethode

Zur selben Zeit hat ein Team von zwei Personen einen vollständigen Test nach der manuellen Methode, basierend auf Klassifikationsbäumen (aktionsorientierte Variante [HL02]), spezifiziert und implementiert. Ein Beispiel ist in Abbildung 6 gegeben. Bei der Erstellung der Testspezifikation zeigte sich, dass die Findung der Testziele relativ schnell und zügig möglich war. Aufwändig war dagegen die Verfeinerung der Testziele in einzelne Testschritte und die Pflege bei Änderungen. Die Zusammenhänge waren nicht mehr intuitiv verfolgbar.

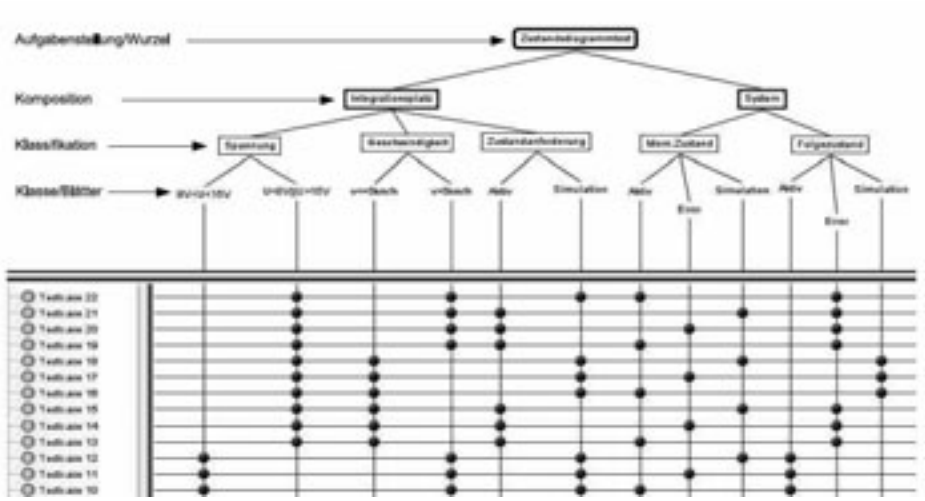

Abbildung 6: Beispiel Klassifikationsbaum

4 Bewertung der Anwendbarkeit der ESG-Methode und Ausblick

Die Anzahl der benötigten Stunden, die für die Erstellung der Testspezifikation gebraucht wurden, nahm mit der ESG-Methode ca. 26 % mehr Zeit in Anspruch (s. Tabelle 3). Die Anzahl der Testfälle konnte bei der ESG-Methode jedoch mehr als halbiert werden – die Anzahl der Testschritte sogar fast um den Faktor vier.

	Manuelle Methode	ESG
Stunden zur Aufstellung der Testspezifikation	120	152
Anzahl Testfälle	26	11
Anzahl Testschritte	240	62

Tabelle 3: Stundenvergleich

Mit Hilfe der Ereignissequenzgraphen wurden zudem alle Fehler gefunden, die mit dem manuellen Testprozess gefunden wurden. Zusätzlich zu diesen Fehlern wurde ein weiterer, erheblicher (da sicherheitskritisch) Fehler in der Implementierung des zentralen Zustandsautomaten gefunden. (Fehler Nr. 3). Tabelle 4 gibt einen Überblick über die mit der ESG-Methode gefundenen Fehler.

1	Befindet sich das System im Fehlerzustand, die Geschwindigkeit sinkt unter 5km/h und es erfolgt eine Simulationsaufforderung, bleibt das System im Fehlerzustand anstatt in den Simulationszustand zu wechseln.
2	Beim Aufschalten des Fehlers wechselt das System nicht rechtzeitig in den Fehlerzustand.
3	Das System befindet sich im Fehlerzustand und die Fehlerbedingung verschwindet. Dann wechselt das Gerät nicht rechtzeitig in den angeforderten Zustand Aktiv (Fehler gefunden mit Hilfe von negativen Testfällen).
4	Wenn sich das System im Fehlerzustand befindet und die Fehlerbedingung verschwindet, wechselt das Gerät nicht rechtzeitig in den angeforderten Zustand Inaktiv (Fehler gefunden mit Hilfe von negativen Testfällen).
5	Das System wechselt nicht rechtzeitig aus dem Zustand Init in Aktiv oder Inaktiv.

Tabelle 4: Fehlerliste

Abschließend lässt sich sagen, dass die Zeit für die Implementierung der Testfälle und die Kosten, die durch die Testdurchführung entstehen mit Hilfe von der ESG-Methode erheblich reduziert wurden. Mit dem Ereignis-Sequenz-Graphen ist eine Methode gegeben, die die Automatisierung der Testvorbereitung ermöglicht und zudem das Regressionstesten vereinfacht. Weitere Schritte zur Effizienzsteigerung sind zudem möglich, und zwar durch die Entwicklung eines Rahmenprogramms, das die generierten Testfälle aus den ESGs in ein von CANoe lesbares Format wie z.B. XML umsetzt.

Literaturverzeichnis

[Ah91] Aho, A. V. et.al.: An Optimization Technique for Protocol Conformance Test Generation Based on UIO Sequences and Rural Chinese Postman Tours. IEEE Transactions on Communications, Vol. 39(11), S. 1604-1615, 1991.

[Be01] Belli, F.: Finite-State Testing and Analysis of Graphical User Interfaces. Proceedings of 12th International Symposium on Software Reliability Engineering (ISSRE 01), S. 34-43, IEEE CS, 2001.

[BHN07] Belli, F.; Hollmann, A.; Nissanke, N.: Modeling, Analysis and Testing of Safety Issues - An Event-Based Approach and Case Study. Computer Safety, Reliability, and Security, LNCS 4680, Springer Berlin, 2007.

[BL07] Belli, F.; Linschulte, M.: 'Negativ'-Tests interaktiver Systeme und ihre Automatisierung. Lecture Notes in Informatics, Vol. 106, S. 35-44, Software Engineering 2007 - Beiträge zu den Workshops, Gesellschaft für Informatik, Bonn, 2007.

[Bei07] Beidinger, F.: Bewertung einer graphenbasierten Methode für den Testentwurf. Diplomarbeit Universität Paderborn, Fachgebiet Angewandte Datentechnik, 2007.

[BN02] Broekman, B.; Notenboom, E.: Testing Embedded Software. Addison-Wesley-Verlag, 2002.

[GG93] Grochtmann, M.; Grimm, K.: Classification Trees for Partition Testing. Software Testing, Verification and Reliability, 3(2), S. 63-82, John Wiley Verlag, 1993.

[HL02] Hanisch, J.; Letzel, J.: Automatisierung von Regressionstests eines Programms zur Halbleiter-Strukturanalyse. Diplomarbeit HU Berlin, 2002.

[SL05] Spillner, A.; Linz, T.: Basiswissen Softwaretest. 3. Auflage, Dpunkt Verlag, 2005.