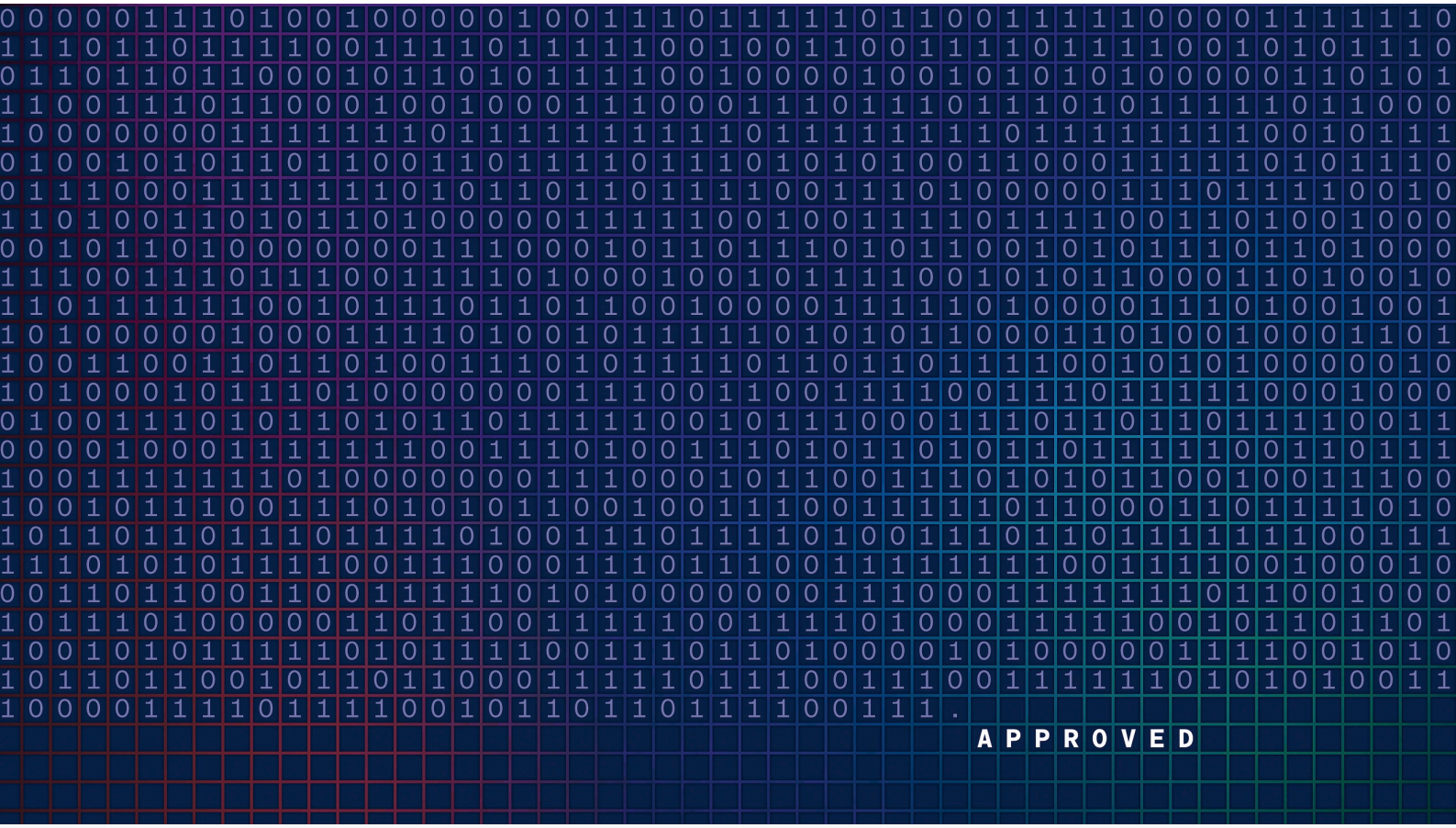

Sichere Software: Code auf dem Prüfstand

10/09/2024 Software spielt im modernen Leben, aber auch in Fahrzeugen eine Schlüsselrolle. Darum nutzen OEMs und Zulieferer bewährte Methoden und Werkzeuge, um Fehler in Programmen möglichst frühzeitig zu identifizieren. Forscher arbeiten bereits an neuen Ansätzen, unter anderem auf Basis von Künstlicher Intelligenz.

Moderne Fahrzeuge sind heute rollende Computer mit einem Netz von bis zu 100 elektronischen Steuergeräten, welche die Motor- und Batteriefunktionen steuern, die Klimaanlage überwachen oder das Infotainmentsystem kontrollieren. Hinzu kommen immer mehr intelligente Features wie Abstandsregeltempostaten und automatisierte Fahrfunktionen. All das ist nur mithilfe komplexer Software möglich.

Mit zunehmender Komplexität der Software wächst auch der Aufwand, den OEMs und Zulieferer betreiben, um Fehler zu vermeiden und so eine hohe Software-Qualität sicherzustellen. Sie orientieren sich dabei einerseits an den Standardprozessen der Branche – vor allem Automotive SPICE (ASPICE) und ISO 26262 – sowie eigenen Coding- und Qualitätsrichtlinien, die beispielsweise die Nutzung

fehleranfälliger Funktionen in bestimmten Programmiersprachen verbieten. Porsche Engineering erreicht mit seinen Entwicklungsprozessen zuverlässig und reproduzierbar ASPICE Level 2. Dies entspricht nicht nur dem aktuellen Stand der Technik und bildet die essenzielle Basis für die Freigabe im Fahrzeug, sondern stellt auch gegenüber den Kunden sicher, dass Fehler und Abweichungen frühzeitig erkannt und behoben werden können.

Bei der Software-Entwicklung folgt Porsche Engineering dem V-Modell: Links finden sich von oben nach unten die Schritte Systemanforderung, Systemarchitektur, Software-Anforderung und Software-Architektur. An der Basis des V ist der Software-Entwurf, dem auf der rechten Seite von unten nach oben die Schritte Unit-Tests, Integrationstests und Qualifikationstests sowie Abnahme und Nutzung folgen. „Zu jedem Schritt links gehört ein Testschritt rechts", erklärt Stefan Rathgeber, Leiter der Software-Entwicklung bei Porsche Engineering. „Bei Unit-Tests prüfen wir zum Beispiel die kleinste Einheit auf Funktionsebene, Komponententests folgen dann eine Stufe darüber." Für sämtliche Ebenen gibt es Testkataloge, die auch alle Varianten berücksichtigen.

Ein eigenes Team von Qualitätsmanagern überprüft bei Porsche Engineering, ob bei der Software-Entwicklung alle Prozessschritte eingehalten und dokumentiert werden. Sie führen ständig Reviews durch, um Probleme so früh wie möglich zu finden. Denn in späteren Entwicklungsphasen steigt der Aufwand für die Fehlersuche und -behebung stark an. Man kann beispielsweise nicht sämtliche möglichen Konstellationen testen, weshalb bestimmte unglückliche Kombinationen zu Problemen führen können.

Die zahlreichen Fahrzeugderivate, die vielen Ausstattungsvarianten sowie Software-Updates nach Auslieferung der Fahrzeuge stellen eine besondere Herausforderung dar. „Derivate sind im Moment ein großes Thema", weiß Thomas Machauer, Fachprojektleiter bei Porsche Engineering. „Man versucht, möglichst nur die Unterschiede zwischen den Derivaten zu testen und so den besten Kompromiss zwischen Aufwand und Qualität zu finden."

Mit diesem Thema beschäftigt sich Professorin Ina Schaefer vom Karlsruher Institut für Technologie (KIT). Sie konzentriert sich auf die intelligente Erzeugung und Priorisierung der Testfälle, vor allem für Software-Varianten. „Die Kombinatorik des Einzelsystems wird durch die Kombinatorik der Derivate noch potenziert", erklärt sie. „Wir stellen uns darum folgende Fragen: Was muss man testen, um den Variantenraum gut abzudecken? In welcher Reihenfolge soll getestet werden? Wenn es gelingt, nur die Unterschiede zu testen, kommen wir zu einem effizienteren Testprozess."

Intelligente Testauswahl

Eine aktuelle Promotion an Schaefers Lehrstuhl beschäftigt sich mit dem intelligenten Testen verschiedener Fahrzeugvarianten, wenn ein Update Over-the-Air ausgespielt wird. „Updates machen alles komplizierter, weil die Fahrzeuge im Feld ganz unterschiedliche Software-Stände aufweisen", so Schaefer. „Früher hat man die Software nur beim Start der Serienproduktion getestet. Künftig muss man für jedes Update nachtesten, was den Aufwand stark erhöht." Dadurch ergeben sich auch ganz

neue Fragen: Was muss man bei einem individuellen Fahrzeug testen, damit alles sicher ist? Welchen Einfluss hat das Update auf die Komponenten des Fahrzeugs? Um diese Fragen zu beantworten, hat Schaefers Team ein Prototypenwerkzeug entwickelt, das den Variantenraum untersucht und eine Liste von Konfigurationen liefert, die getestet werden sollen. Andere Werkzeuge schlagen die beste Testreihenfolge vor.

Aber selbst der intelligenteste Test kann nicht alle mathematisch möglichen Kombinationen von Ein- und Ausgabewerten einer Software abdecken. Darum arbeiten andere Wissenschaftler an der Verifikation von Code. „Informatiker meinen damit einen Beweis", erklärt Professor Ralf Reussner vom KIT. „Sie wollen mathematisch zeigen, dass ein Programm genau das tut, was die Spezifikation vorgibt. In besonders sicherheitskritischen Bereichen der Luftfahrt wird dieser Ansatz bereits eingesetzt, und auch einzelne Automobilhersteller haben ihn im Rahmen von Forschungsprojekten bereits untersucht."

Was auf den ersten Blick vielversprechend klingt, stößt allerdings oft an theoretische Grenzen. So gibt es mathematische Probleme, deren Wahrheit oder Falschheit nicht automatisch festgestellt werden kann – zum Beispiel wahre Aussagen, für die kein Beweis existiert. Für Software bedeutet das: Gewisse Eigenschaften kann man nicht für alle Computerprogramme mit einem Algorithmus prüfen – etwa ob ein Programm anhält oder in einer Endlosschleife hängen bleibt. „Im Allgemeinen werden wir darum nie ein Verifikationswerkzeug bauen können, das einen beliebigen Programmcode nimmt und vollautomatisch den Nachweis seiner Fehlerfreiheit führt", sagt Reussner. Oft haben die Wissenschaftler aber Glück und finden dennoch automatisch einen Korrektheitsbeweis. Und wenn das nicht möglich ist, kommen interaktive Werkzeuge zum Einsatz, bei denen der Mensch bei Bedarf in den Prozess eingreift. „In der Praxis machen diese Tools aber verblüffend viel automatisch", so Reussner.

Spezifikation aus dem Lastenheft

Ein weiteres Problem der Software-Verifikation: Wenn die Spezifikation falsch ist, ist auch der Beweis wertlos. Um das zu vermeiden, haben Informatiker eigene Spezifikationssprachen entwickelt, die Programmiersprachen ähneln. „Mit ihnen lassen sich logische Zusammenhänge, aber auch zeitliche Aussagen beschreiben", sagt Reussners KIT-Kollege Professor Bernhard Beckert. „Damit ist aber ein hoher Aufwand verbunden: Die präzise Spezifikation einer Software dauert heute etwa genauso lange wie das Schreiben des Codes. Weniger aufwendig ist aber die einfache Beschreibung, dass die Software typische Fehler wie eine Division durch null nicht enthält." Am einfachsten wäre es, wenn sich die präzise Spezifikation automatisch aus dem Software-Pflichtenheft ableiten ließe, was nach Beckerts Meinung in Zukunft mithilfe von Künstlicher Intelligenz möglich sein könnte: „Large-Language- Modelle wie ChatGPT wären eventuell in der Lage, das zu leisten."

Künstliche Intelligenz könnte sich aber auch – zum Beispiel als Ergänzung zur formalen Verifikation – eigenständig auf die Suche nach Programmfehlern machen. Diesen Ansatz unter dem Namen „Neural Bug Detection" untersucht die Arbeitsgruppe von Professorin Heike Wehrheim an der Universität Oldenburg. „Man kann eine KI mithilfe von korrekten und fehlerhaften Programmen darauf trainieren, Fehler zu finden", so Wehrheim. „Das Thema ist noch sehr jung, boomt im Moment aber stark und wird

von vielen Forschern untersucht." Bisher eignet sich der Ansatz allerdings nur für kleine Programme oder einzelne simple Funktionen. „Bessere Software ist definitiv möglich, auch wenn sie nie hundertprozentig fehlerfrei sein wird", fasst KIT-Expertin Schaefer die aktuelle Situation zusammen. „Aber mit guten Prozessen und innovativen Verfahren können wir die Qualität ständig verbessern. Mir wird in der Forschung darum wohl nie die Arbeit ausgehen."

Info

Text erstmals erschienen im Porsche Engineering Magazin, Ausgabe 1/2024.

Text: Christian Buck

Copyright: Alle in diesem Artikel veröffentlichten Bilder, Videos und Audio-Dateien unterliegen dem Copyright. Eine Reproduktion oder Wiedergabe des Ganzen oder von Teilen ist ohne die schriftliche Genehmigung der Dr. Ing. h.c. F. Porsche AG nicht gestattet. Bitte kontaktieren Sie newsroom@porsche.com für weitere Informationen.

MEDIA ENQUIRIES

Sandro Kälin

Head of Communications Porsche Schweiz AG
+41 41 487 91 16
sandro.kaelin@porsche.ch

Link Collection

Link to this article
https://newsroom.porsche.com/de_CH/2024/innovation/porsche-engineering-software-fehler-erkennung-37264.html

Media Package
https://pmdb.porsche.de/newsroomzips/0947ac27-cc2e-4f57-8144-731e9a4375dc.zip