Integrationstest für die Prozesskette Softwarelogistik und Steuergeräteprogrammierung (Flash, Codierung, Freischaltung)

Beitrag im Tagungsband der Product Life live 2007 (6.-7.11.2007, Mainz),
Seiten 105 bis 112, VDE Verlag, 2007

Klaus Höllerer, BMW AG, Prozess-IT Vertrieb, München
Dr. Zoltán Ádám Mann, sd&m AG, CoC Automotive, München

1. Einleitung

Innerhalb des Gesamtprodukts Automobil spielen Elektronik und Software eine zunehmend entscheidende Rolle. Auf der einen Seite werden die meisten innovativen und wettbewerbsdifferenzierenden Funktionen durch Elektronik und Software verwirklicht, auf der anderen Seite stellt die immer steigende Komplexität von Elektronik und Software im Automobil eine große Herausforderung dar. Verschiedene Anwender bringen Daten in das Fahrzeug bzw. kommunizieren mit dem Fahrzeug.

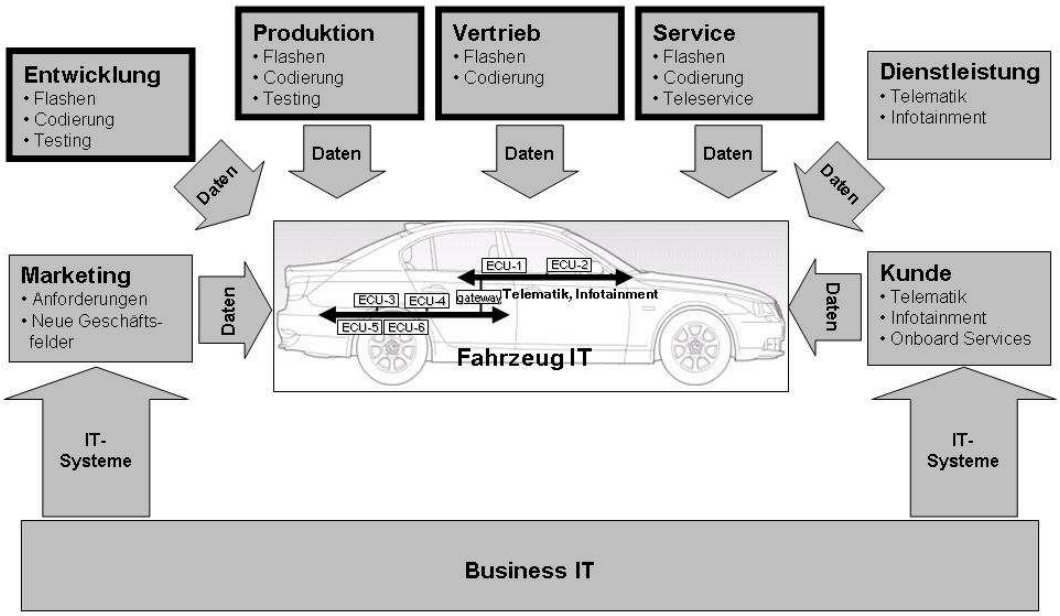

Abbildung 1: Einfluss externer Programme und Daten von verschiedenen Nutzern im Produktlebenszyklus

Die Einbettung von Software im Automobil ins Produktlebenszyklusmanagement ist dabei kritisch. Eine durchgängige und konsistente Verwaltung und Einbringung von Daten in das Fahrzeug ist bei allen beteiligten Stellen im Fahrzeuglebenszyklus notwendig, besonders aber beim Update von Steuergerätesoftware (dick umrandet in

Abbildung 1). Software ist allerdings in vielen Hinsichten anders als die materiellen Komponenten eines Fahrzeugs:

- Software muss logistisch anders gehandhabt werden, da Software in der Regel nur einmal verteilt wird, und dann beliebig oft verbaut werden kann.
- Neue Softwarefeatures und Bugfixes können im Monatstakt erstellt werden.
- Durch die hohe Vernetzung erfordern Elektronik und Software ein komplexes Konfigurationsmanagement, dass nichts mit der hierarchischen Konfiguration der Mechanik zu tun hat.
- Die Funktionsweise bereits verbauter Steuergeräte kann im Nachhinein geändert werden, in dem neue Software ins Steuergerät eingespielt wird.
- Durch die Anpassungs- und Erweiterungsmöglichkeit von Software ist mit steigendem Datenvolumen umzugehen.
- Ein konsistentes Softwareupdate muss über verschiedene Generationen von Steuergeräten während des Lebenszyklus des Fahrzeuges möglich sein, um die Reparaturfähigkeit sicherzustellen.

Um diesen Herausforderungen gerecht zu werden, führt die BMW Group eine Reihe von neuen IT-Systemen ein, die die effiziente Verwaltung, Verteilung und Einspielen von Software im Automobil ermöglichen (siehe Abbildung 2).

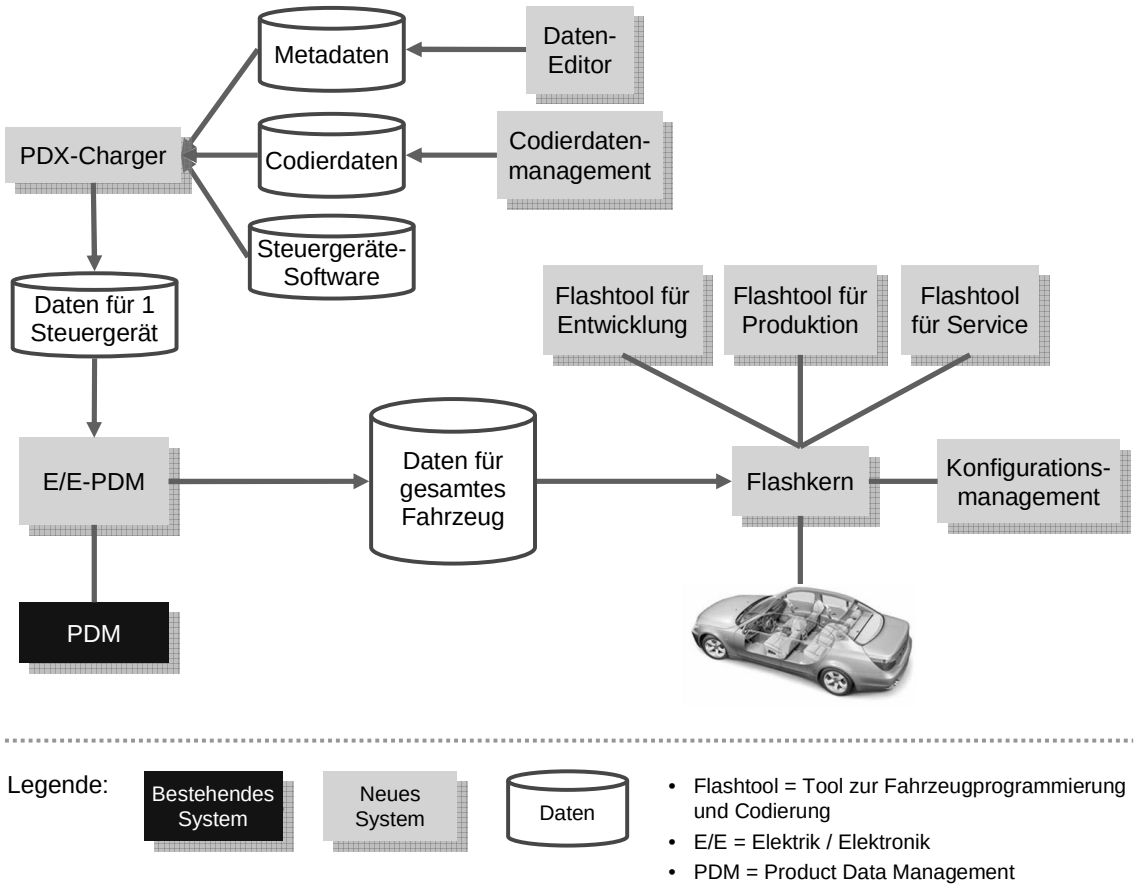

Abbildung 2: Übersicht zur Systemlandschaft

Das Herzstück der neuen Systemlandschaft ist ein System, das sämtliche Produktdaten für Elektrik/Elektronik und Software im Fahrzeug verwaltet (kurz E/E-PDM). Es findet ein Abgleich mit dem bestehenden PDM-System statt, das die teilebezogenen Daten verwaltet. Aus dem E/E-PDM kann die Software, inklusive nötiger Metadaten, für das Gesamtfahrzeug ausgeleitet werden.

Den Metadaten kommt eine entscheidende Rolle zu, da diese Informationen zur Softwareversion und Kompatibilitätsaussagen enthalten. Der korrekte Abgleich der Kompatibilitätsaussagen mit der Information im Fahrzeug ermöglicht erst eine sichere und konsistente Programmierung. Mit steigender Anzahl der Softwarevarianten über den Lebenszyklus des Fahrzeuges, kann die Menge der Metadaten durchaus höher sein, als die Datenmenge (Programme und Daten) der eigentlichen Steuergerätesoftware.

Aus diesem Grund ist die Entscheidung für die Verwendung des ASAM ODX Standards als Basis gefallen, da dieser den Nutzen als standardisiertes, selbst beschreibendes Datenformat zur Reduzierung fehlerhafter Bedatung durch einheitliches und maschinell prüfbares Format bietet. Dadurch ist eine integrierte Steuerung der Flash Programmierung, der Codierung, der Freischaltung und der Rettung von Individualdaten über eine einheitliche Transaktionsliste möglich.

Diese Daten (Programm, Daten, Metadaten) werden dann in Entwicklung, Produktion und Service benutzt, um die Steuergeräte korrekt und effizient mit den Programmier- und Codiertools auf den neuesten Stand zu bringen.

Die Programmierung erfolgt mit einem Flashtool, das den jeweiligen Anforderungen der Nutzer (Entwicklung / Produktion / Service) angepasst ist. Um einen einheitlichen Prozessablauf während des Programmiervorgangs sicher zu stellen, verwenden alle

Flashtools denselben Flashkern, der die Kommunikation mit dem Fahrzeugbordnetz sicherstellt. Dieser basiert auf einem im ASAM ODX Standard entwickelten Produkt.

2. Aufbau der Integrationstests

Um die Prozesse der Programmierung und Softwarelogistik korrekt und vollständig abzubilden, ist eine nahtlose Integration all der oben erwähnten IT-Systeme erforderlich. Tests der einzelnen Projekte haben bereits Teile dieser Integration abgedeckt, diese zielten aber noch nicht auf eine Integration des gesamten Verbunds. Deshalb wurde ein weiteres Projekt aufgesetzt, das sich dediziert mit dem Integrationstest für den gesamten Verbund beschäftigte. Wir berichten hier von diesem Projekt.

Integrationstest bedeutet in diesem Kontext einen Test, der die Durchgängigkeit des Systemverbunds absichert. Es geht hierbei also nicht um die Integration eines Systems aus seinen Modulen, sondern um die Integration der einzelnen Systeme zu einem Systemverbund. Der Zweck des Integrationstests ist nicht die Absicherung aller möglichen Abläufe. Eine vollständige Testabdeckung sollte im Test der einzelnen Systeme angestrebt werden; der Integrationstest dient vielmehr dazu, das technische und fachliche Zusammenspiel der einzelnen Systeme in einigen typischen Szenarien abzusichern.

Der Integrationstest war durch zwei inhärente „Henne-Ei-Probleme“ geprägt:

1. Parallele Entwicklung von onboard und offboard Systemen: Einerseits sind für die Absicherung der neuen Steuergeräte die neuen Offboard-Systeme notwendig. Andererseits sind aber für den Test der neuen Offboard-Systeme die neuen Steuergeräte notwendig. Konkrete Ausprägungen dieses Problems:

    - Gegenseitige Abhängigkeit zwischen dem Programmiersystem und den Steuergeräten. D.h. um das Programmiersystem testen zu können, braucht man Steuergeräte, aber für die Absicherung der Steuergeräte braucht man schon das Programmiersystem.
    - Gegenseitige Abhängigkeit zwischen der Softwarelogistik und den von der Softwarelogistik verwalteten Daten. D.h. um die Daten erfassen zu können, braucht man die Softwarelogistik-Systeme, aber für den Test der Softwarelogistik-Systeme braucht man schon die Daten.

2. Parallele Entwicklung der Offboard-Systeme: Einerseits werden für den Integrationstest eingefrorene Stände benötigt. Andererseits können aber die Systeme erst nach Einarbeitung der Ergebnisse aus dem Integrationstest eingefroren werden.

Diese Probleme konnten nur aufgelöst werden, in dem der Integrationstest in mehreren Iterationen, unter Verwendung mehrerer Vorabversionen der einzelnen Systeme erfolgte. Eine weitere Folge war die erschwerte Fehlersuche, da man bei keinem der Systeme davon ausgehen konnte, dass es beim Integrationstest bereits fehlerfrei funktioniert.

Für den Integrationstest wurden verschiedene Testgegenstände benötigt: Steuergeräte-Hardware, Steuergeräte-Software sowie diverse Metadaten über die Steuergeräte (z.B. Kompatibilitätsangaben). Bei jedem der nötigen Testgegenstände stellte sich die Frage, welchen Vorabstand man für die Tests benutzen kann. Dabei war es einerseits wünschenswert, einen möglichst produktionsnahen Stand zu

nutzen. Andererseits musste aber auch beachtet werden, wann und mit welchem Aufwand ein entsprechender Vorabstand zu erstellen war.

Die einzelnen Projekte mussten nicht nur das entwickelte IT-System sondern auch dessen Zusammenspiel mit seinen Nachbarsystemen testen (siehe Abbildung 3). Das reicht aber noch nicht aus, um den ganzen Systemverbund übergreifend abzusichern. Denn die Wechselwirkung der Systeme ist so komplex, dass inhaltliche Abhängigkeiten auch zwischen Systemen existieren, die keine direkten Nachbarsysteme sind. Dies ist z.B. der Fall, wenn System A Daten erzeugt, die durch System B transportiert und durch System C genutzt werden. In diesem Fall gibt es eine inhaltliche Abhängigkeit zwischen System A und C, die aber weder von Projekt A noch von Projekt C getestet wird. Solche Probleme abzufangen war genau das Ziel des Integrationstests.

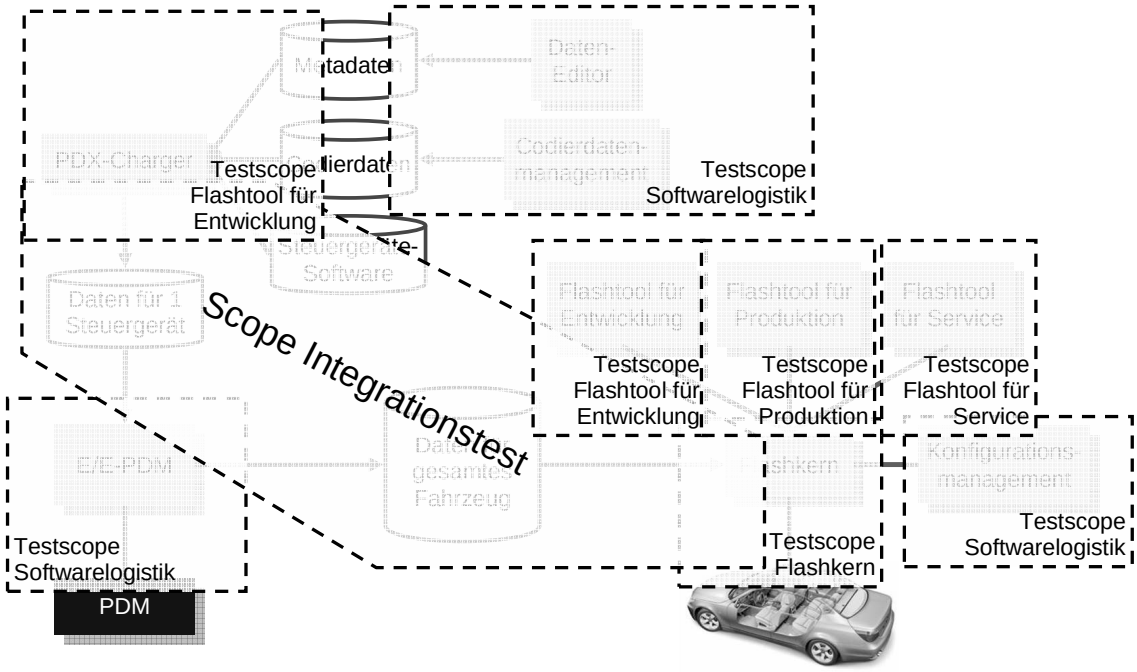

Abbildung 3: Scope der Tests der einzelnen Projekte und des Integrationstests

Wie aus Abbildung 2 ersichtlich, ist die zu testende Systemlandschaft sowohl durch Onlineschnittstellen (insbesondere die Schnittstellen zwischen Programmierkern und den Programmierapplikationen) als auch Offlineschnittstellen (insbesondere die Schnittstelle zwischen Softwarelogistik und Programmierung) geprägt. Diese mussten beim Integrationstest unterschiedlich gehandhabt werden:

- Bei Onlineschnittstellen musste dafür gesorgt werden, dass beim Test des importierenden Systems eine geeignete Version des exportierenden Systems mit benutzt und getestet wurde.
- Bei Offlineschnittstellen war es möglich, die Testschritte mit dem erzeugenden bzw. nutzenden System zu entzerren. Die Idee hierbei war, die beim Test des erzeugenden Systems erzeugten Daten für den Test des nutzenden Systems als Input zu benutzen. Dazu musste das Integrationstestteam für die Festlegung der Testdatenübergabe (wer, was, wann, wie) sorgen.

Im Integrationstest haben wir angestrebt, den normalen Prozess von der Erstellung der Steuergeräte-Software und der zugehörigen Metadaten über deren Einstellen ins System, Freigabe und Verteilung bis hin zur Steuergeräteprogrammierung durchzuspielen. Um den Aufwand für die Erstellung der Testdaten zu minimieren, wurden zum größten Teil Daten aus den Produktivsystemen benutzt. Um die

Produktivdaten jedoch nicht zu beeinflussen, wurden sie nach den ersten Prozessschritten in entsprechende Testumgebungen gespiegelt und dort weiter verarbeitet (siehe Abbildung 4).

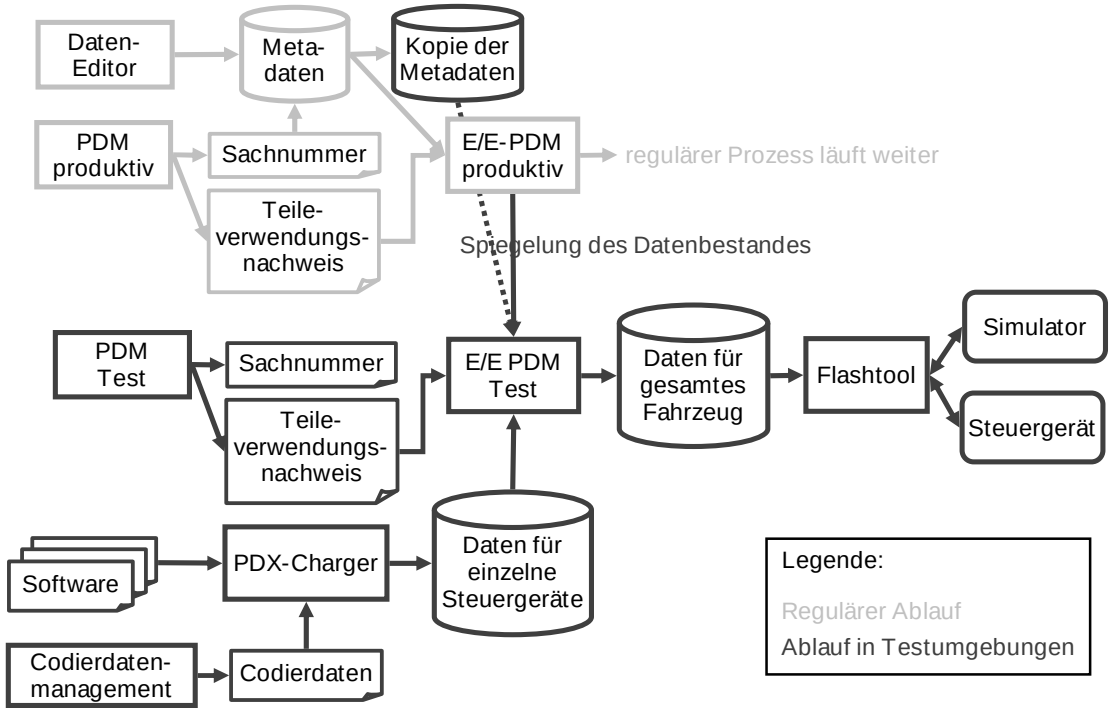

Abbildung 4: Im Integrationstest wurde der reguläre Prozess durchgespielt. Um den Aufwand zu minimieren, wurden die meisten Daten aus den Produktivsystemen übernommen

Die Aufgabe des Integrationstestteams war die Planung, Abstimmung und operative Steuerung des Integrationstests. Die Testschritte selbst wurden von den Testteams der einzelnen Projekte durchgeführt. Das Integrationstestteam hat dafür gesorgt, dass die beim Test in einem Projekt erzeugten Daten an die Testteams der nachgelagerten Projekte übergeben und dort als Input bei den weiteren Tests benutzt wurden. So wurde sichergestellt, dass ein konsistenter Datensatz den Prozess vom Anfang bis zum Ende durchläuft und damit die Durchgängigkeit der Systemlandschaft absichert.

Da während des Integrationstests die zugrunde liegenden Daten kontinuierlich auf die Serienproduktion hin weiterentwickelt wurden, war es sinnvoll, den Testdatensatz auch Stück für Stück auszubauen. Zudem wurden im Laufe des Integrationstests Fehler in den einzelnen IT-Systemen gefunden, nach deren Behebung ohnehin ein neuer Versuch zum Durchlauf des obigen Prozesses notwendig war. Das hat zu einem iterativen Testvorgehen geführt, in dem immer größere Teile des Prozesses mit einem immer größeren und realistischeren Datenbestand durchlaufen wurden, und somit die Testabdeckung laufend erweitert wurde.

3. Organisatorische Aspekte

Um ein projektübergreifendes Testvorgehen zu erreichen, war es notwendig eine parallele Integrationstestorganisation aufzubauen. Diese war auf der einen Seite eng mit den Projekten verzahnt, musste aber auf der anderen Seite unabhängig davon agieren können. Dazu wurden folgende Maßnahmen ergriffen:

- Intensivierung der Kommunikation mit und unter den Projekten durch die Einrichtung eines gemeinsamen Jourfixes.

- Erstellen eines News-Letters via e-Mail, für die schnelle und regelmäßige Verteilung der Informationen.
- Bericht über die Pläne und Ergebnisse zum Integrationstest in diversen Gremien.
- Organisation von gemeinsamen Fehler-Sessions zur Suche bei mehrfachen Schnittstellenübergängen.
- Reviews der Schnittstellenkontrakte auf Lücken, wenn diese unter dem Multiprojektcharakter betrachtet werden.
- Blockieren von abgestimmten Testslots in den Einzelprojekten für die Durchführung der Integrationstests. Hier gibt es eine hohe Abhängigkeit von den Zulieferungen der Ergebnisse in die Projekte.
- Klärung der verantworteten Testräume: Was wird genau von den Projekten getestet und was nicht? Wo gibt es Überschneidungen und Lücken?
- Ausarbeitung, Abstimmung und Kommunikation eines gemeinsamen Testplans, der festlegt, welches Projekt wann welche Schritte des Integrationstests durchführen soll.
- Etablierung eines Transferlaufwerks für die Übergabe verschiedener Artefakte (Testdaten, neue Toolversionen, Dokumente) zwischen den Projekten.

Die Aktivitäten wurden so frühzeitig wie möglich gestartet, um einen sauberen Ablauf in der „heißen“ Projektphase zu gewährleisten. Ein wichtiger Aspekt war auch die Entlastung der Testdatenzulieferer durch die Übernahme von Testdaten aus Produktivsystemen. Damit konnte einerseits eine Entlastung der Projektteams geschaffen werden, auf der anderen Seite auch die Akzeptanz des Integrationstestteams erhöht werden, da die Projekte einen kurzfristigen Nutzen aus den Aktivitäten ziehen konnten.

Für den Ablauf der Test waren folgende Punkte besonders wichtig:

- Festlegung definierter Übergabeobjekte und Übergabetermine um einen Reibungslosen Ablauf der Testaktivitäten zu gewährleisten.
- Sicherstellen einer engen Zusammenarbeit mit den Testteams der Einzelprojekte, um Synergien zu nutzen und Doppelarbeit zu vermeiden.
- Die Bündelung von Know-how zu allen involvierten Systemen im Integrationstestteam hat die Abhängigkeit von der Verfügbarkeit der Know-how-Träger in den Einzelprojekten minimiert.
- Die Koordination der Terminpläne der Einzelprojekte, um genügend Zeit für Tests mit geeigneten Versionen der Systeme zu haben.
- Ein definierter Prozess für das Handling von übergreifenden Fehlern, die während des Integrationstests gefunden wurden (siehe Abbildung 5). Die Handhabung solcher Probleme erforderte die geordnete Zusammenarbeit mehrerer Projekte, da der Fehler oft nicht im verursachenden sondern in einem nachgelagerten Projekt entdeckt wurde. Das für die richtige Zuordnung des Fehlers notwendige Know-how war im Integrationstestteam vorhanden.

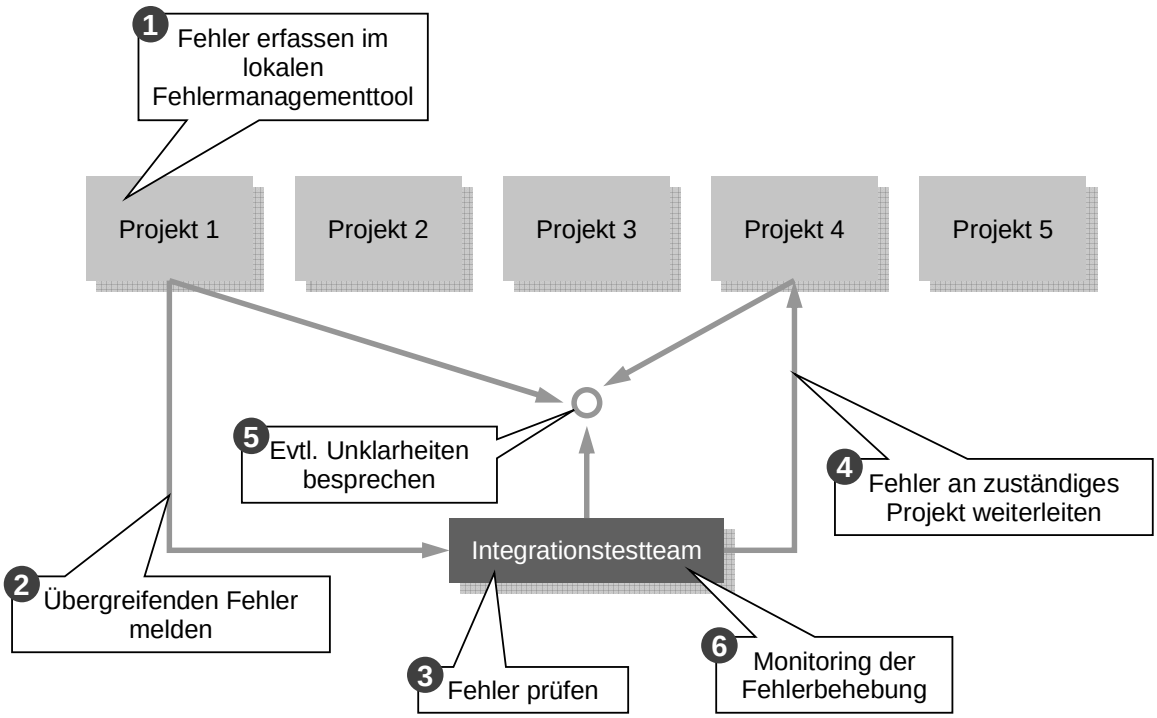

Abbildung 5: Funktionsweise des übergreifenden Bug-tracking und –dispatching Prozesses

4. Fazit

Die in den letzten Jahren stark erhöhte Komplexität von Elektronik und Software im Automobil schlägt sich auch auf die entsprechenden Entwicklungs-, Fertigungs- und Reparaturprozesse sowie die diese abbildenden IT-Systeme durch. Die rasante Entwicklung auf diesem Gebiet hat dazu geführt, dass in kurzer Zeit der Großteil der relevanten IT-Systeme abgelöst werden musste. Das führte zu einer entsprechend umfangreichen Projektstruktur.

Durch die gegenseitigen Abhängigkeiten unter den IT-Systemen, ist die Integration zu einer durchgängigen Systemlandschaft keine triviale Aufgabe. Schnittstellenkontrakte zwischen Paaren von benachbarten Systemen, sowie das Testen der einzelnen Systeme gegen ihre Schnittstellen sind zwar wichtig, reichen aber nicht aus. Deswegen war die Schaffung eines dedizierten Projektes für den Integrationstest ein notwendiger Schritt.

Im Integrationstest wurde der reguläre Prozess von der Erstellung der Steuergeräte-Software und nötigen Metadaten bis hin zum Flashen in mehreren Iterationen durchgespielt. Dabei wurde der Umfang und die Reife der Testdaten kontinuierlich erhöht. Beim Test gefundene Fehler wurden den Entwicklungsteams der relevanten Systeme gemeldet; nach der Korrektur wurde der Test wiederholt. Es wurde so lange iteriert, bis der gesamte Prozess mit hinreichend hoher fachlicher und technischer Abdeckung durchgespielt werden konnte.

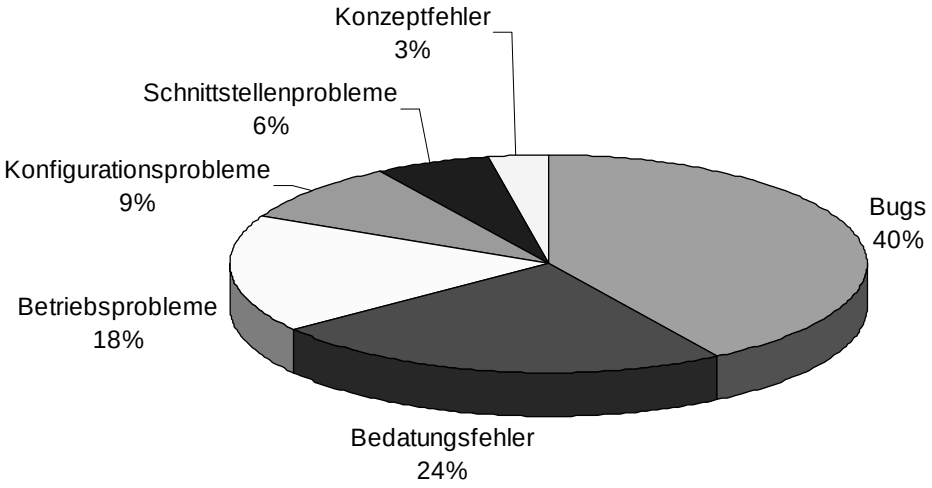

Abbildung 6: Verteilung der beim Integrationstest entdeckten Fehler

Neben der Planung und operativen Steuerung des eigentlichen Integrationstests bestand die Bedeutung des Integrationsteams noch stärker darin, für die Kohärenz des Projektverbunds zu sorgen. Es war anfangs schwierig, Verantwortungen der einzelnen Projektteams über deren Projekt hinaus zu etablieren, da die Einzelprojekte auf ihre Themen fokussiert waren. Doch durch entsprechende organisatorische Maßnahmen ist es gelungen, das Bewusstsein in den Projekten für das Zusammenspiel im Großen zu stärken und gleichzeitig auch für die Akzeptanz des Integrationstestteams zu sorgen. Dies erforderte breitgefächerte Kommunikationsaktivitäten auf allen Ebenen, angefangen von bilateralen Gesprächen mit den Projektsponsoren, über Informationsveranstaltungen für die betroffenen Abteilungen und Einbindung in die Projektmeetings bis hin zur gemeinsamen Fehlersuche und Ad-hoc-Workshops zur Fehlerbehebung. Insgesamt also ein aufwendiger Prozess, der jedoch notwendig war, um den Projektverbund durchgängig abzusichern.

Empfehlung für ähnliche Vorhaben in der Zukunft:

- Frühzeitige Prüfung, wie Projekte im Gesamtzusammenhang verankert sind und zusammenspielen.
- Aufstellen eines unabhängigen und eigenständigen Integrationstestteams. Dies ist an übergeordneter Stelle in der Projektorganisation zu verankern.
- Ausplanung der Integrationstests in enger Abstimmung mit den Projektteams.
- Nutzen der Integrationstests für die einzelnen Projekte darstellen.
- Klare Messkriterien definieren.
- Kommunikation, Kommunikation, Kommunikation ...