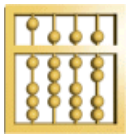
Software Engineering:
Software im Automobil

1. Vorlesung: Einführung und Überblick

Manfred Broy

Technische Universität München
Institut für Informatik
D-80290 München, Germany



Aufgaben des Software Engineerings

Entwicklung umfangreicher Softwaresysteme unter Optimierung der Erfolgsfaktoren

- Qualität
  - Qualitätsbegriff ist nach ISO 8402 definiert als:
    Qualität ist die Gesamtheit von Merkmalen einer Einheit bezüglich ihrer Eignung, festgelegte und vorausgesetzte Erfordernisse zu erfüllen?.
  - nach ISO 9126 (Software):
    Funktionalität,Zuverlässigkeit,Anwendbarkeit,Effizienz, Wartbarkeit und Portabilität
  - Adressierung der Markterfordernisse/Kundenorientierung
- Kosten
  - Beherrschung des Entwicklungsaufwandes
  - Betriebs- und Wartungskosten
- Zeit
  - Termingerechte Entwicklung
  - Time to Market

Beobachtung - Schlüsselfelder in der Softwareentwicklung

- Anforderungen: Requirements Engineering
  - ◇ Die Anforderungen bestimmen den Funktionsumfang eines Systems und damit seinen Wert (Nutzen für den Verwendungszweck) und die Kosten für seine Entwicklung
- Architekturentwurf
  - ◇ Die Architektur bestimmt
    - die Beherrschbarkeit eines Systems aus Entwicklersicht
    - Die Wartbarkeit und Beherrschbarkeit eines Systems aus Betreibersicht
- Qualitätssicherung
  - ◇ Die Qualität der Realisierung eines Systems bestimmt das Risiko und den Mehraufwand bei seiner Nutzung
- Kostenmanagement

Domänenspezifisches Software Engineering

Richtet man das Software Engineering auf bestimmte Anwendungsfelder aus, so sprechen wir von "Domänenspezifischem Software Engineering"

Faktoren für Domänenspezifisches Software Engineering:

- Markt
  - ◇ Anforderungen, Bedürfnisse und Marktpotential
- Technologische Randbedingungen
  - ◇ Besondere Erfordernisse, Vorgaben
- Fachliche Erschließung - Domänenmodell
- Kompetenz
  - ◇ Reifegrad

Markante Anwendungsfelder für Software

Nach historischer Entwicklung:

- Technisch/wissenschaftliche Berechnungen
- Verwaltung großer Datenmengen (Betriebswirtschaftliche Anwendungen)
- Telekommunikation
- Automatisierungstechnik/Prozessleittechnik
- Arbeitsinfrastruktur
  - ◇ Netze
  - ◇ Desktops/Laptops mit Alltags-Software
- Eingebettete Systeme
  - ◇ Haushaltstechnik
  - ◇ Unterhaltungselektronik
  - ◇ Medizintechnik
  - ◇ Verkehr
    - Flugzeug
    - Bahn
    - Schifffahrt
    - Automobil

Software im Automobil

Zur Bedeutung:

- „90 % aller Innovationen im Automobil geschehen im Bereich Elektronik“ (Zitat Piech)
- Der Anteil an Software und der rein softwarebasierten Funktionen nimmt beständig zu
- Der Anteil der Daten im Auto verzehnfacht sich alle 4 Jahre
- Elektronik und Softwareprobleme machen den Herstellern zu schaffen
  - ◇ Mehr als 50 % aller getauschten Steuergeräte sind nicht defekt
  - ◇ 70 % der Fehler im Fahrzeug sind sporadischer Natur
- Aber: Der Anteil der Unfälle und durch Unfälle Getöteten ist rückläufig - bei steigendem Verkehrs-aufkommen - wesentlicher Grund: Elektronik/Software

Die Herausforderung: Software im Automobil

- **Vernetzung: Die Systeme wachsen zusammen**
  - neue Funktionen häufig nur noch im Systemverbund
  - Integration von Teilsystemen auch unterschiedlicher Zulieferer
- **Wertigkeit und Charakter der Fahrzeuge wird zunehmend durch komplexe Softwarefunktionen bestimmt**
  - Flexiblere kundenspezifische Gestaltungsmöglichkeiten
  - erforderlich
- **Beherrschung der wachsenden Systemkomplexität wird für Fahrzeughersteller und Zulieferer wettbewerbsentscheidend**
  - Geschwindigkeit, Kosten, Qualität

Automobile sind Systeme mit Software, d.h. isolierten keine Software-Systeme

Ziel: Software Engineerings von Software im Automobil

Ganzheitliche, durchgängige Entwicklung softwareintensiver eingebetteter Systeme

- Wohldefinierte Vorgehensweise („Entwicklungsprozess") und Festlegung der zu erarbeitenden Ergebnisse („Entwicklungsprodukte", „Arbeitsprodukte")
- Schrittweise Erarbeitung aller Anforderungen und Entwurfsfestlegungen („Lösungen") für das Zielsystem
- Strukturierte Sicht auf das Zielsystem („Gesamtsystemarchitektur")
- Darstellung wesentlicher Systemeigenschaften mit standardisierten Modellierungs- und Beschreibungsmitteln
- Integrierte organisatorische, konstruktive und analytische Qualitätssicherung
- Weitgehende, abgestimmte Werkzeugunterstützung

Hauptgesichtspunkte zur Software im Automobil

- Markt
  - Wettbewerbssituation
  - Kostenstruktur
  - Kundenwünsche
  - Stückzahl
- Organisation
  - Kompetenz/Personal
  - Verteilte Entwicklung/Rolle der Zulieferer
  - Produktion
- Vorgehen
  - Arbeitsprodukte
  - Entwicklungsprozess
- Produktstruktur
- Produktlebenszyklus

Schwerpunktthemen der Vorlesung

- Spezifika der Domäne
  - Historische Entwicklung
  - Markt
  - Unternehmensstrukturen
- Der Entwicklungsprozess
  - Struktur Arbeitsprodukte
  - Struktur Aktivitäten
- Die Struktur der Softwaresysteme
  - Funktionen - Nutzersicht
  - Systemarchitektur
  - Software
  - Hardware
  - Deployment

Themen der Vorlesung

- Einführung in das Software Engineering softwareintensiver Systeme im Automobil
- Anwendungsfelder softwareintensiver Systeme im Automobil
- Besonderheiten eingebetteter Systeme
- Plattformen und Standards im Automobil
- Entwurfsprozess
- Management und Projektorganisation
- Requirements Engineering
- Architektur
- Implementierung
- Systemtest und Systemsimulation
- Qualitätssicherung
- Softwarelogistik
- Entwicklungswerkzeuge
- Herausforderungen der Zukunft

Charakteristika der Software im Fahrzeug

- Software ist eingebettet in technische Systeme (Mechatronik)
- Software läuft auf Prozessoren („Steuergeräten")
- Zeitanforderungen (Beispiel Motorsteuerung)
- Software/Steuergeräte sind vernetzt (über Bussysteme)
  - Untereinander
  - Nach außen
- Komplexe, immer stärker von Software bestimmte Mensch Maschine Schnittstelle
- Unterschiedliche Aufgaben/Anwendungsfelder
  - Regelungstechnisch (Motormanagement, ABS, Stabilitätskontrolle, Klima ...)
  - Überwachung (Diebstahlsicherung, Licht, etc.)
  - Infotainment (Telefon, Radio, ...)
- Langlebigkeit

Besonderheiten bei Software im Automobil

- Heterogenität
- Stückzahl
- Arbeitsteilige Entwicklung
- Massenproduktion
- Individualisierung
- Nutzungssituation
  - Nutzungszeitraum
  - Nutzercharakteristik
  - Internationalität
  - Wartung
- Unterschied Zykluszeiten IK/Automobil
- Markt
- Einbettung, Vernetzung, Echtzeit
- Sicherheitsanforderungen

Größenordungen

In heutigen Automobilen finden sich bis zu

- 2500 softwaregestützte Funktionen
- 80 Steuergeräte
- 4 Bussysteme
- 10.000.000 Lines of Code (LOC) Software

Anwendungsfelder von Software im Automobil

- Infotainment
  - Telefon, Navigation, Telematikdienste, Internet, ..
  - Radio, Soundsystem, TV, CD, Video, DVD, ...
- Motormanagement
- Fahrwerks- und Antriebselektronik
  - ABS, ESP, ASR, ABC, BAS, Getriebe, Federung
- Komfort- und Karosserieelektronik
  - Schließung, Zentralverriegelung, Fensterheber, Schiebedach, ...
  - Sitz
  - Klima
  - MMI, Head-Up Display, Sprachsteuerung, ...
  - Servolenkung
  - Einparkhilfe
  - Geschwindigkeitsregelung
- Sicherheitselektronik
  - Airbag, Gurtstraffer, Pre-Crash,
  - Fahrerassistenz,
  - Alarmanlage
- Bordnetz
  - Lichtanlage, Starter, Batterie, Generator, Energiemanagement
- Bussysteme

Arbeitsteilige Entwicklung

Die Entwicklung von der Elektronik im Fahrzeug findet innerhalb der Automobilhersteller aber auch über Firmengrenzen hinweg arbeitsteilig statt.

Dies erfordert besondere Sorgfalt:

- Bei der Festlegung der Zuständigkeiten und Prozesse
  - Saubere Schnittstellen
  - Beherrschung der Architektur
- Bei der Spezifikation der Vorgaben
- Bei der Qualitätssicherung
- Bei den Business-Modellen

Der Markt und die Rolle der Software

- Markt
  - Elektronik im Fahrzeug Teil der Kaufentscheidung
  - Differenzierungsmerkmal
  - After Sales Markt
- Kundenbedürfnisse
  - Basisfunktionen (Sine Qua non)
  - Zusatzfunktionen (Sonderwünsche)
  - Unique Selling Points
  - Innovationsfunktionen
- Kosten
  - Vereinheitlichung der Plattformen zur Kostenreduktion bei Zulieferern
- Zuverlässigkeit
  - Durch den Wettbewerb und die rasanten Fortschritte der Technologie arbeiten die Unternehmen an der Grenze der Beherrschung der Komplexität

Beispiel: „X-by-Wire"

- In der „By-Wire"-Technik werden mechanische oder hydraulische Steuereinrichtungen (Lenkung, Bremse etc.) durch rein elektronische (Mensch-Maschine-Bedieneinrichtung, Steuergeräte, Sensorik, Aktuatorik) ersetzt.
- Vorteile
  - Kosten
  - Höhere bessere Funktionalität
  - Technische Lösung (Gewicht, Packaging)
  - Neuartige Nutzerschnittstellen/Bedienkonzepte
- Nachteile und Probleme
  - Komplexität
  - Beherrschung der Zuverlässigkeit/höhere Sicherheitsanforderungen
  - Gestaltung der Bedienkonzepte (Beispiel: Haptisches Feedback)

Technische und organisatorische Besonderheiten

- Automobilhersteller nicht sehr erfahren in Software
  - Systeme traditionell stark entkoppelt
  - Keine Tradition in Software/Systems Engineering
  - Software wird primär von Herstellern realisiert
  - Zunahme der Bedeutung von Software schlagartig (exponentielles Wachstum)
- Automobilhersteller traditionell stark hierarchisch organisiert
  - Software erfordert kooperative Strukturen
  - Kulturwandel erforderlich
- Softwaremarkt hat seine Besonderheiten
- Hohe Dynamik in der Entwicklung des Einsatzes

Die Mitspieler

- Automobilhersteller
- Klassische Zulieferer
- Technologiespezialisten
  - Werkzeughersteller
  - Methodenberater
  - Softwareinfrastrukturlieferanten
    - Betriebssysteme
    - Bussysteme
  - Systemberater

Bemerkung:

Die Zukunft wird zeigen, welche Rolle die einzelnen Mitspieler langfristig wahrnehmen werden

Komplexitätstreiber

- Vernetzung der Funktionen
  - ◇ Funktionalität durch Kombination von Funktionen
- Heterogene technische Infrastruktur
- Anzahl der Funktionen
- Arbeitsteilige Entwicklung
- Kosten- und Zeitdruck
- Raffiniertheit und Umfang der Funktionen
- Unsystematik im Entwicklungsprozess
- Mangelnde Kompatibilität
- Zusatzanforderungen
  - ◇ Diagnose
  - ◇ Software Download

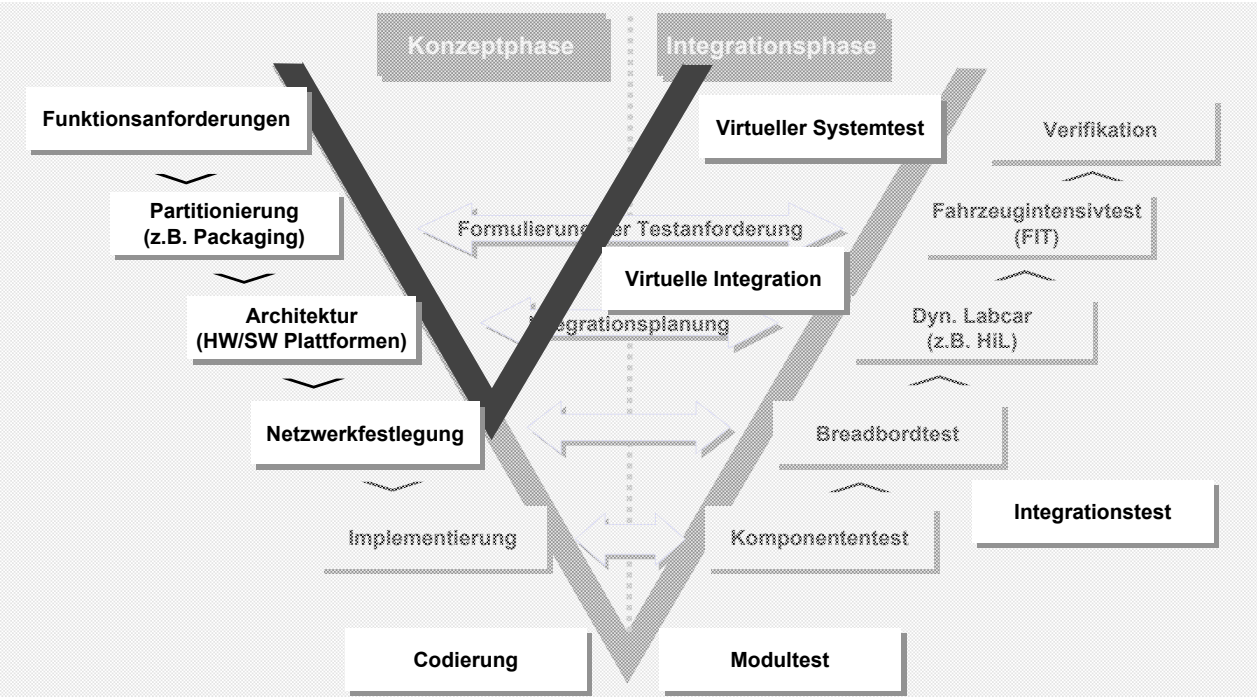
Der Produktlebenszyklus

- Konzeption
- Anforderungen und Spezifikation
  - ◇ Kundenfunktionen
  - ◇ Nichtfunktionale Anforderungen
- Entwicklung
  - ◇ Architektur
  - ◇ Einzelfunktionen
  - ◇ Integration
  - ◇ Qualitätssicherung/Erprobung
- Produktion
- Wartung
- Entsorgung

Entwicklungsmethoden

- Die Entwicklung eines Hardware/Softwaresystems erfolgt in auf einander aufbauenden Entwicklungsschritten.
- Die Entwicklungsschritte dienen der Erarbeitung von Entwicklungsergebnissen, genannt Entwicklungsprodukte.
- Jeder Entwicklungsschritt stützt sich auf einen Satz bereits erarbeiteter Entwicklungsprodukte, genannt Vorgaben, und produziert Entwicklungsprodukte, genannt Ergebnisse.
- Top Down Vorgehen: System wird systematisch ausgehend von einer Gesamtsicht in Teilsysteme zerlegt (vol. Komponentenarchitektur), deren Schnittstellen beschrieben werden
- Bottom Up Vorgehen: System wird - of ad hoc - schrittweise aus Teilsystemen zusammengesetzt

Entwicklung: Vorgehensmodell - V-Modell



Struktur der Arbeitsprodukte der Entwicklung

- Jede Entwicklungsaktivität zielt auf ein Ergebnis: Wir sprechen von einem Arbeitsprodukt
- Die Menge der Arbeitsprodukte hat eine Struktur (Abhängigkeiten zwischen den Arbeitsprodukten)
- Für die Darstellung der Inhalte der Arbeitsprodukte verwenden wir spezifische Techniken (Modellierungstechniken)
- Die Erstellung der Arbeitsprodukte erfolgt durch Aktivitäten des Entwicklungsprozesses
- Für die Produkte und die Aktivitäten sind bestimmte Personen/Organisationen verantwortlich („Rollen")
- Arbeitsprodukte stellen Entwicklungsergebnisse dar, dokumentieren diese und sind zu verwalten.

Maintenance

Elektroniksysteme im Fahrzeug sind über die gesamte Lebensdauer zu warten.

Die Wartung umfasst:

- Überprüfung und ggf. Auswechseln von Hardware (Achtung: U.U. sind die ursprünglich eingebauten Mikrokontroller nicht mehr am Markt verfügbar).
- Korrektur von Softwarefehlern
- Funktionserweiterung von Software

All dies führt auf Kompatibilitätsprobleme

Struktur des Entwicklungsgegenstands Automobil

- Das Ziel des Software Engineerings softwareintensiver Systeme im Automobil ist die Entwicklung der Software in Automobilen
- Diese Aufgabe ist eng gekoppelt mit der Entwicklung der Hardwarestruktur im Automobil
- Es entsteht eine heterogene, komplexe, technische Mechatronik-Struktur:
  - Nutzungssicht
  - Hardware
  - Software
  - Deployment: Abbildung von Software auf Hardware
  - Mechanik: Aktuatorik, Sensorik
  - Mensch-Maschine-Schnittstelle

Sichten auf die softwareintensiven Funktionen

- Um die Komplexität der Mechatronikstruktur im Automobil zu beherrschen, sind unterschiedliche Sichten auf das System sinnvoll
  - Nutzungssicht
  - Logische Lösungsstruktur
  - Hardware
  - Software
  - Verteilung (Deployment) der Software auf die Hardware
- Um so besser es gelingt
  - diese Sichten eigenständig zu halten
  - die Abhängigkeiten zwischen diesen Sichten genau zu verstehen
- um so flexibler und reibungsloser kann die Entwicklung durchgeführt werden

Modelle

- Die Sichten werden in der Regel durch geeignete Modelle dargestellt
- In den unterschiedlichen Phasen werden ganz unterschiedliche Modelle eingesetzt
  - ◇ Diese sind zueinander in Beziehung zu setzen
- Die Modelle sind so zu wählen, dass ein Optimum an
  - ◇ Ausdruckskraft
  - ◇ Abstraktion
  - ◇ Anschaulichkeit
  - ◇ Handhabbarkeit
  - ◇ Automatisierung

  erreicht wird

Modell: Abbild der Wirklichkeit für bestimmten Zweck

Systemarchitektur

- Die Systemarchitektur ist auf eine strukturierte Sicht auf ein System aus unterschiedlichen Perspektiven
  - ◇ Abstraktionsebenen
- Die Systemarchitektur beantwortet alle relevanten Fragen zur Struktur eines Systems
- Es werden spezifische Modelle für die unterschiedlichen Sichten eingesetzt
- Die Sichten sind in Beziehung zu setzen
- Die Systemarchitektur ermöglicht ein strukturiertes Vorgehen
  - ◇ Aufspaltung in Teilaufgaben/Teilentwicklungen
  - ◇ Systematische Integration

Nutzersicht: Multifunktionale Systeme

Funktionale Komplexität:

- Typischerweise bieten heute Systeme viele unterschiedliche Funktionen an
- Jede Einzelfunktion dient einem Verwendungszweck (Nutzungsfall) des Systems
- Die für den Nutzer geforderte angemessene Funktionsweise ist alles andere als offensichtlich und bei neuartigen Systemen oft erst nach sorgfältigen Überlegungen und Experimenten zu ermitteln
- Zwischen der Einzelfunktionen bestehen die unterschiedlichsten Abhängigkeiten

Das Requirements Engineering (= Festlegung der Anforderungen) muss auf diesen Umstand ausgerichtet sein.

Multifunktionale Systeme können und müssen in Funktionshierarchien strukturiert werden

Abhängigkeiten zwischen den Funktionen

In Fahrzeugen besteht eine Vielzahl von versteckten, offensichtlichen, gewollten und ungewollten Abhängigkeiten zwischen den Funktionen („Feature Interaction")

Ein wichtiges Ziel der Entwicklung ist es:

- Die Abhängigkeiten eindeutig festzulegen
- Versteckte Abhängigkeiten aufzuspüren
- Unerwünschte Abhängigkeiten zu beseitigen

Hardwarestruktur

Die Hardware umfasst:

- Sensoren und Aktuatoren
- Mensch-Maschine-Schnittstelle (Bedien- und Anzeigeeinrichtung
- Kommunikationsverbindungen
- Steuergeräte („Controller")

Auf der Hardware läuft Systemsoftware.

Softwarestruktur

Die Software umfasst

- Ein Plattform aus Systemsoftware
  - ◇ Betriebssystem (in der Regel Echtzeitbetriebssysteme)
  - ◇ Kommunikationssoftware (Bustreiber)
  - ◇ Gerätetreiber
- Anwendungssoftware
  - ◇ Steuer-, Regelungs- und Kontrollprogramme für technische Komponenten (Motor, Getriebe, Klima)
  - ◇ Bedieneinrichtungen (Schließung, Fensterheber etc.)
  - ◇ Infotainment
- Infrastruktursoftware
  - ◇ Fehlerdokumentation
  - ◇ Software zum Laden von Software

Deployment

- Die Software wird zur Ausführung in die Hardware (Steuergeräte) eingebracht („Deployment“).
- Das Deployment bestimmt, welche Software in welchen Steuergerät zur Ausführung kommt.
- Wir unterscheiden
  - ◇ Statisches Deployment: Die Festlegung, welche Software in welchem Steuergerät zur Ausführung kommt, erfolgt vor Inbetriebnahme und ändert sich über die Betriebsdauer nicht.
  - ◇ Dynamisches Deployment: Die Festlegung, welche Software in welchem Steuergerät zur Ausführung kommt, erfolgt (teilweise) während des Betriebs und ändert sich über die Betriebsdauer.
- Heute herrscht statisches Deployment vor, obwohl es Gründe für dynamisches Deployment gibt („Fehlertoleranz“, „Energiesparen“)

Echtzeit

Der überwiegende Teil der Software im Fahrzeug muss Echtzeitanforderungen genügen:

- Die Reaktionen der Software muss zeitlich angemessen auf die physikalischen und technischen Vorgänge erfolgen (harte und weiche Echtzeitanforderungen).
- Die Reaktionszeiten sind wesentlich von den Ausführungsgeschwindigkeiten („Taktung“) der Hardware (Steuergeräte, Sensoren, Aktuatoren, Bussysteme) abhängig.
- Durch Timesharing und Multiplexing (mehrere Tasks kommen auf einem Steuergerät oder Bus zur Ausführung) versteckte Abhängigkeiten und schwer kalkulierbare Ausführungszeiten.

Sicherheit

Im Automobil herrschen hohe Sicherheitsanforderungen

- Funktionssicherheit - Safety
  - Das Verhalten eines Systems und die Funktionsweise erfüllen die Erwartungen und Sicherheitsbedürfnisse der Nutzer
  - Zuverlässigkeit
  - IEC 61508 wird angewendet (Funktionssicherheit) mit SIL-Stufen SIL 1 - SIL 4
- Zugriffssicherheit - Security - Es ist sichergestellt:
  - dass Funktionen nicht von Unbefugten benutzt werden,
  - dass Daten nicht von Unbefugten geändert oder gelesen werden,
  - dass das System nicht in unzuverlässiger Weise manipuliert wird,
  - dass keine wertvollen Daten verloren gehen oder Systemfunktionen sich verändern

Funktionssicherheit - Safety

- Angemessenheit der Spezifikation
  - Spezifikation trifft die Erwartungen des Nutzers
  - Wahrscheinlichkeit der Fehlbedienung ist gering
- Korrektheit der Implementierung
  - Implementierung entspricht der Spezifikation
- Zuverlässigkeit
  - Hohe Verfügbarkeit
  - Geringe Fehlerrate
  - Auch beim Auftreten von Fehlern Beherrschbarkeit des Systems gewährleistet - Vermeidung von Katastrophen (Fail Safe)
  - Schnelle Wiederinstandsetzung

Zugriffssicherheit: Security

Wesentliche Aspekte

- Identifikation
- Authentifikation
- Authorisation
- Privacy
- Integrität

Schützenswerte Informationen für Personen

- Fahrer/Beifahrer/Wartung
- Fahrzeug
- Software/Technische Innovationen
- Technische Integrität

Risiken

Die Entwicklung und der Betrieb von Software im Fahrzeug ist mit Risiken verbunden:

- Entwicklungsrisiken
  - ◇ Entwicklungsaufgabe zu komplex
  - ◇ Entwicklung zu teuer
- Marktrisiken
  - ◇ Funktion wird vom Markt nicht nachgefragt
  - ◇ Auswirkungen auf das Markenimage
- Betriebsrisiken
  - ◇ Wartungskosten
  - ◇ Fehler verursachen Unfallgefahr
  - ◇ Nachbesserung
  - ◇ Rückrufaktion

Ein Ziel der Entwicklung: Risikobegrenzung

Gesichtspunkte bei Fehlern

- Fehlervermeidung
  - ◇ Konstruktive und analytische Qualitätssicherung
- Fehlerbeseitigung im Entwicklungsprozess
  - ◇ Fehlermanagement
  - ◇ Änderungsmanagement
- Fehlerbehandlung im Betrieb
  - ◇ Fehlererkennung
  - ◇ Fehlerbegrenzung
  - ◇ Fehlerdokumentation
  - ◇ Fehlerbeseitigung
- Behandlung von Fehlern in der Systemumgebung
  - ◇ Hardware/Mechanik
  - ◇ Nutzer

Qualitätssicherung

Aufgrund der hohen Qualitätsanforderungen kommt der Qualitätssicherung besondere Bedeutung zu:

- Konstruktive Qualitätssicherung: Alle Maßnahmen zur Erzeugung von Arbeitsprodukten hoher Qualität

  Beispiele: Vorgaben für Dokumentaufbau, Codingstandards etc.
- Analytische Qualitätssicherung: Alle Maßnahmen zur Überprüfung der Arbeitsprodukten auf Qualität

  Beispiele: Tests, Inspektionen, Reviews, etc.
- organisatoreische Qualitätssicherung
  - ◇ Prozesse (CMMI, Verfahrensanweisungen, ...)
  - ◇ Weiterbildung
  - ◇ Zertifizierungen
- Maßnahmen der konstruktiven und analytischen Qualitätssicherung müssen ineinander greifen.
- Von besonderer Bedeutung für das Management sind präzise Informationen zum Qualitätsstand

Testfälle

Aus dem RE ergeben sich Testfälle:

- Ein Testfall ist ein Systemablauf (Sequenzdiagramm), der Eingabestimuli und Ausgabereaktionen festlegt
- Zusätzlich wird dokumentiert, welche Eigenschaft ein Testfall überprüft
- Negative Testfälle dienen der Überprüfung von Anforderungen, die festlegen, dass bestimmtes Verhalten auf keinen Fall auftreten darf
- Testfälle sind auf eine bestimmte Abstraktionsebene in der Modellierung eines Systems ausgerichtet

Produktion

Software hat in der Regel nur Entwicklungs- aber keine Produktionskosten

Aber:

- Software im Automobil muss in der Produktion ins Produkt eingebracht werden
- Dies erfolgt in der Regel durch „Flashen“:
  - Flashen ist das Aufspielen von Software in Steuergräte
  - Dabei müssen die vorhandenen Zugänge (Bussysteme) genutzt werden
  - Flashdauer ist kritisch
- Flashen erlaubt auch die Änderung von Software in der Wartung („Software Downloads“)

Kompatibilitätsmanagement

Softwaresystem erfordern in der Regel Änderungen während ihres Betriebs.

Gründe für Änderungen im fertigen System:

- Abkündigung von Steuergeräten
- Fehlerkorrektur
- Funktionserweiterungen

In jedem Fall sind Änderungen an der Software erforderlich

Dies führt auf das Problem der Kompatibilität der modifizierten Softwareanteile mit dem restlichen System

Kompatibilität

- Universelle Kompatibilität:

  Eine Komponente A ist zu einer anderen Komponente B (universell) kompatibel, wenn A als Komponente in beliebigen Gesamtsystemen G das gleiche Verhalten des Gesamtsystems wie B garantiert.

- Lokale Kompatibilität in Bezug auf vorgegebenes Gesamtsystem G

  Eine Komponente A ist zu einer Komponente B in einem Gesamtsystem G kompatibel, wenn A als Komponente im Gesamtsystem G das gleiche Verhalten wie B des Gesamtsystems G garantiert.

Wirtschaftliche Gesichtspunkte

Software im Automobil muss wirtschaftlichen Gesichtspunkten gerecht werden

- Vernünftiges Kosten/Nutzenverhältnis

Kostenkategorien

- Entwicklungskosten
- Produktionskosten
- Wartungskosten
- Gewährleistungskosten
- Entsorgungskosten
- Kosten für weitere Risiken

Schlüssel: Beherrschung der Komplexität

Vielfältige Komplexitätsgesichtspunkte:

- Markt
- Entwicklung
  - ◇ Prozess
  - ◇ Technische Zuverlässigkeit
- Produktion
- Wartung
- Bedienung
- Kosten

Ein Ziel des Software Engineering ist die Beherrschung der Komplexität

Literatur

- J. Schäuffele, Th. Zurawka: Automotive Software Engineering. Vieweg 2003
- M. Broy, M. von der Beeck, I. Krüger: SOFTBED: Problemanalyse für das Großverbundprojekt ”Systemtechnik Automobil - Software für eingebettete Systeme”. Ausarbeitung für das BMBF 1998
- M. Broy: Automotive Software Engineering. In: 25th International Conference on Software Engineering, IEEE 2003, 719-720
- M. Broy: Software. In: Braess/Seiffert (Hrsg.): Handbuch Kraftfahrzeugstechnik. Vieweg 2004
- J. Gausemeier, J. Wallaschek (Hrsg.): Intelligente mechatronische Systeme. 2. Paderborner Workshop, Heinz Nixdorf Institut 2004