Dynamische Ermittlung der Softwarequalität zur Unterstützung der Testfallpriorisierung mit einem agentenbasierten Ansatz

Yang Yang (Logica Deutschland GmbH & Co. KG)
Christoph Malz (Universität Stuttgart)
Prof. Peter Göhner (Universität Stuttgart)
Lothar Beller (ZF Friedrichshafen AG)
Thomas Kerler (ZF Friedrichshafen AG)

TAV 31
03.02.11

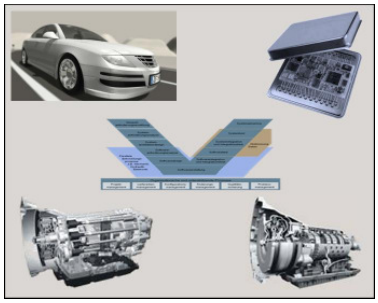
Institut für Automatisierungs- und Softwaretechnik
Pfaffenwaldring 47
70550 Stuttgart

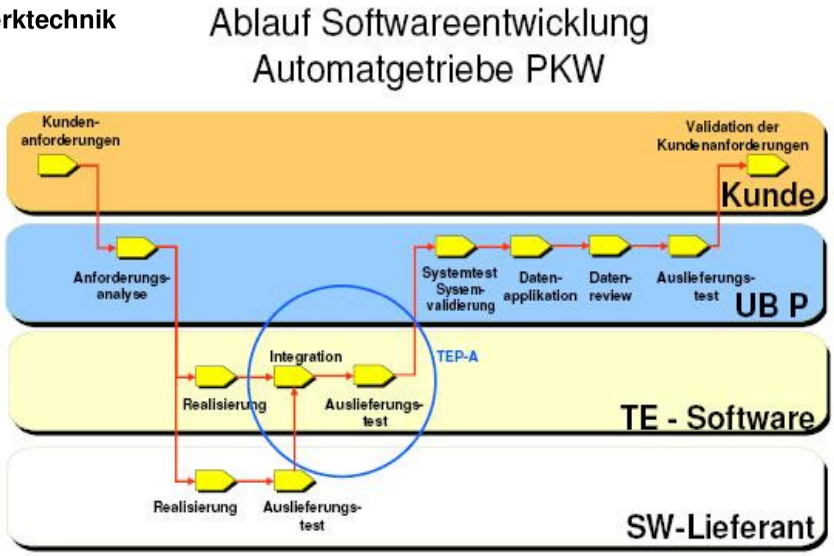
ZF Friedrichshafen AG
Ehlersstraße 50
88046 Friedrichshafen

Motivation

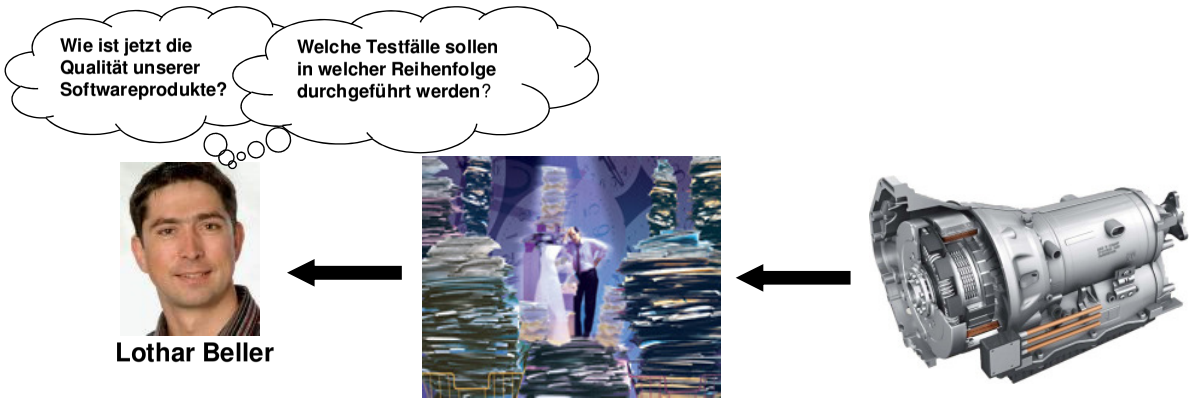
Antriebs- und Fahrwerktechnik

TEP PKW-Antriebstechnik





Motivation



- Management der Testaktivitäten
- Management der Testressourcen
- Bewertung des Testobjekts

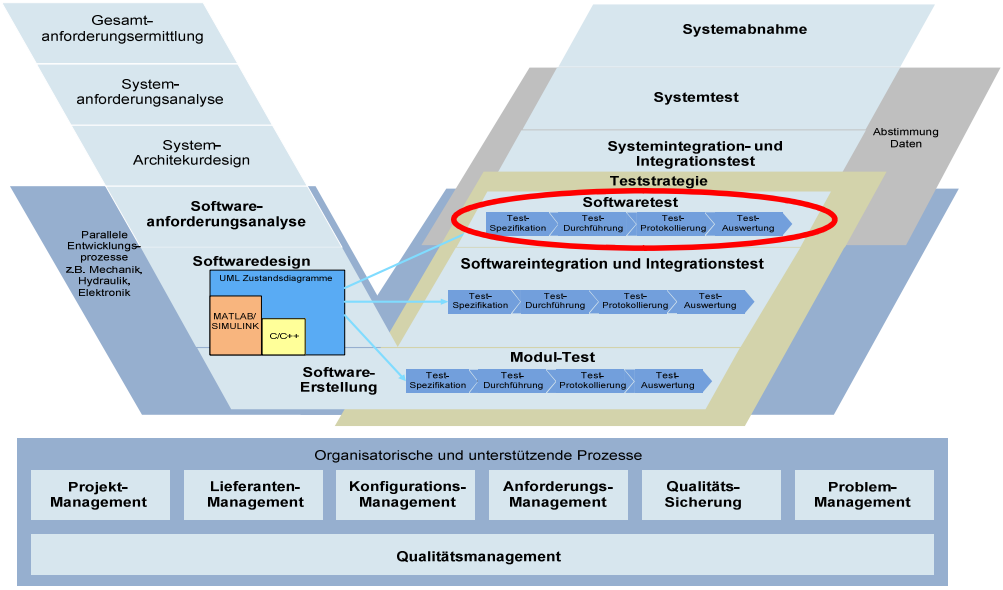
1. Wie kann die Softwarequalität bewertet werden?
2. Wie kann die optimale Reihenfolge durchzuführender Testfälle bestimmt werden?

Gliederung

- Stand der Technik und Problembeschreibung
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Stand des Softwaretests bei TEP, ZF Friedrichshafen AG (1)

- **Softwareentwicklungsprozess**
  - Orientierung am allgemeinen V-Modell
  - Organisatorische und unterstützende Prozesse benögtigt



Stand des Softwaretests bei TEP, ZF Friedrichshafen AG (2)

- **Test von PKW Automatgetriebe**
  - Software- und Systemfunktionstest vom mechatronischen System
  - Hardware-in-the-Loop-Techlonogie
  - Anforderungsbasierter Test

- **Trennung von der funktionalen Sicht und der Softwaresicht**
  - Testmanagement in der funktionalen Sicht
  - Mapping zwischen der funktionalen Sicht und der Softwaresicht

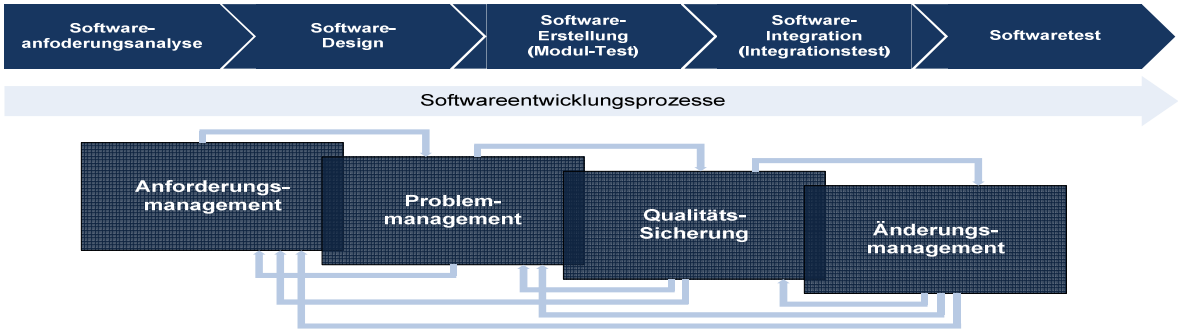
SW-Komponenten / Testthemen	SAB	EH	ATSYS	ASIS	ABK	…
010 SBW	X					
030 FktReg		X				
040 Strat				X		
050 Ersatzfkt			X			
060 10 OBD 05	X	X	X	X	X	
⋮						

Softwaresicht

Funktionale Sicht

Metriken bei ZF Friedrichshafen AG

- **Metriken aus den Softwareentwicklungsprozessen**
  z.B. LOCpro, Komplexität nach McCabe, Komplexität nach Halstead, Testabdeckung Modultest, Testabdeckung Integrationstest usw.
- **Metriken aus dem Anforderungsmanagement**
  z.B. Teststatus der Anforderung usw.
- **Metriken aus dem Problemmanagement**
  z.B. Fehlertyp, Fehlerentdeckung, Status des SW-Fehlers usw.
- **Metriken aus der Qualitätssicherung**
  z.B. Fehlerschwere, Fehler pro Release, Implementierungszeitpunkt des Fehlers usw.
- **Metriken aus dem Änderungsmanagement**
  z.B. Änderungstyp, Komplexität der Änderung usw.



Problemstellung

- **Auswertung einer großen Menge von Daten notwendig**
- **Mathematische Beschreibung der Zusammenwirkungen unterschiedlicher Metriken schwierig**
- **Nur unscharfes Expertenwissen als verbal formulierte Regeln verfügbar**
- **Verschiedene Faktoren besitzen unterschiedliche Beiträge**

- **Bewertung von Daten schwierig**
- **Bewertung der Qualität von Testthemen schwierig**
- **Bestimmung der Testwichtigkeit von Testthemen schwierig**

Gliederung

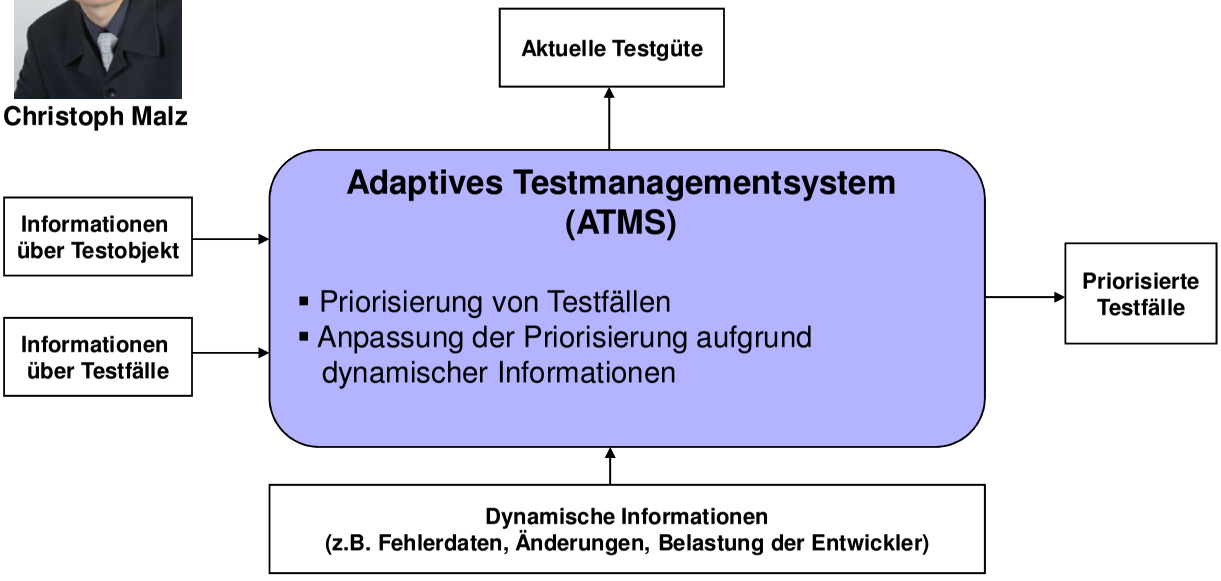
- Stand des Softwaretests bei ZF Friedrichshafen AG
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Adaptives Testmanagementsystem

Christoph Malz

Adaptive Testfallpriorisierung beim Softwaretest

➔ Ziel: Anzahl aufgedeckter Fehler erhöhen



Priorisierung von Testthemen

- **Bestimmung der Testwichtigkeit einzelner Testthemen**

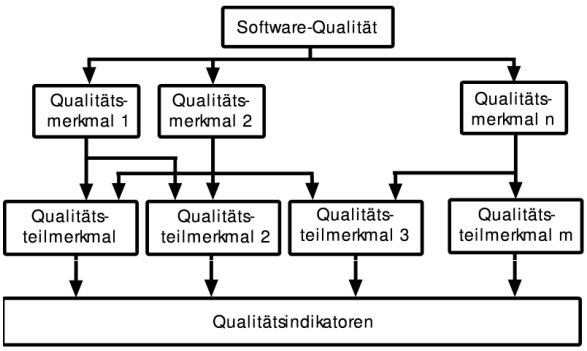
010_SBW
030_FktReg
040_Strat
050_Ersatzfkt
060_10_OBD05
……

- **Expertenwissen als verbal formulierte Regeln**
  - Wo Sicherheitsrelevanz hoch ist, hat es eine große Bedeutung für das System.
  - Wo Implementierungskomplexität hoch ist, könnten viele Fehler auftreten.
  - Wo häufig geändert wurde, könnten viele Fehler verursacht werden.
  - Wo einige Fehler sind, sind auch mehr. [SpLi05]
  - …

- **Einflussfaktoren auf Priorisierung**
  - Sicherheitsrelevanz
  - Implementierungskomplexität
  - Änderungshäufigkeit
  - Fehleranfälligkeit
  - Vertrauensgrad
  - Qualität

Qualitätsbewertung von Testthemen (1)

- **Allgemeines Qualitätsmodell**



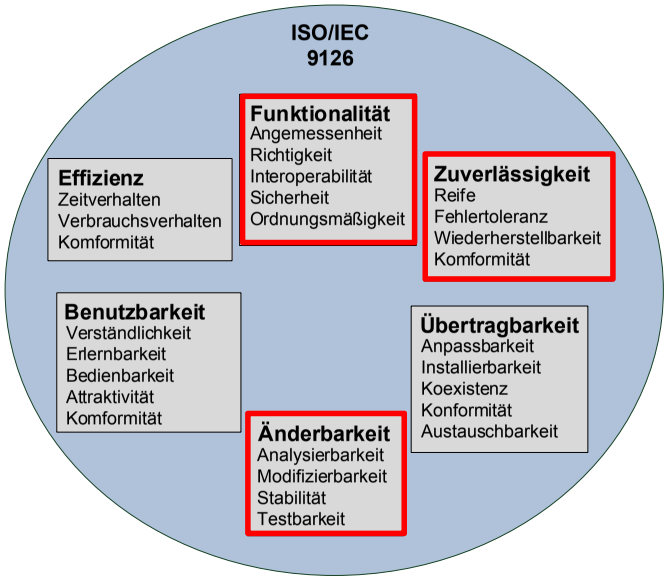
[Balz98]

Qualitätsbewertung von Testthemen (2)

- **Unterschiedliche Qualitätsmodelle**
  - FCM-Modell
  - Modell nach McCall
  - Modell nach Boehm
  - Qualitätsmodell in ISO 9126 √
  - FURPS

- **Betrachtete Qualitätsmerkmale**
  - Funktionalität
  - Zuverlässigkeit
  - Änderbarkeit

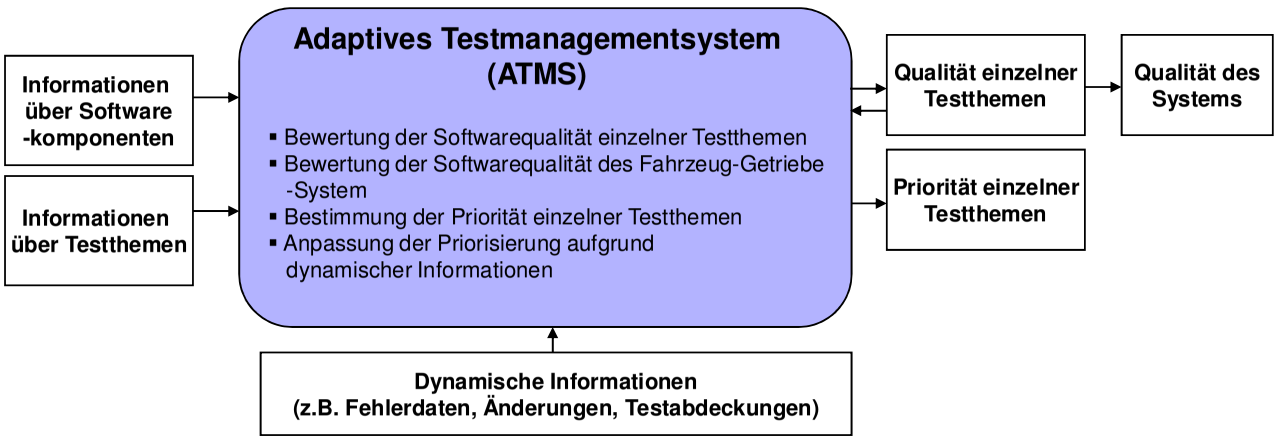
- **Qualität im Sinne der Arbeit**
  - Qualität einzelner Testthemen
  - Qualität des Fahrzeug-Getriebe-Systems



Ziel und Konzeptidee

Automatisierte Bewertung der Testthemen beim Softwaretest

➔ Reduktion manueller Detailtätigkeiten bei der Datenbewertung
➔ Unterstützung der Testthemenpriorisierung anhand dynamisch ermittelter Bewertungsergebnisse



- Autonome Agenten bieten ein gutes Potenzial, um die Komplexität zu beherrschen und diese Aufgaben zu übernehmen.

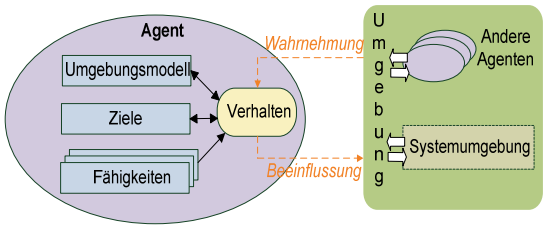
➔ **Agentenbasierter Ansatz mit der Integration von Wissen durch Fuzzy-Logik**

Gliederung

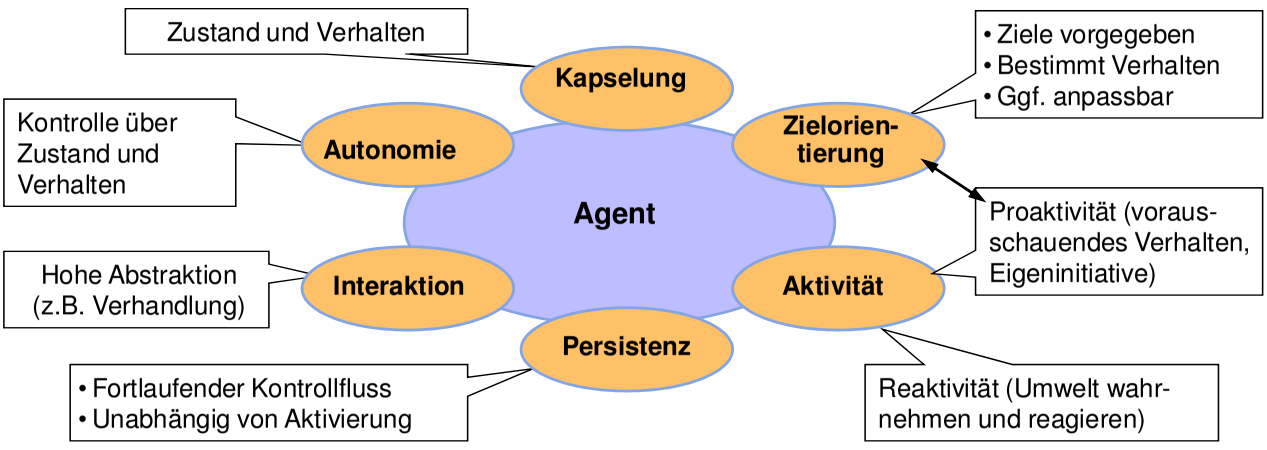
- Stand der Technik und Problembeschreibung
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Agentenorientierte Konzepte (1)

- **Agent:**
  - Softwareeinheit mit definiertem Ziel
  - Autonomes Verhalten zur Erreichung des Ziels
  - Interaktion mit Umgebung und anderen Agenten
  - Dauerhafte Beibehaltung des inneren Zustandes



- **Grundkonzepte der agentenorientierten Denkweise**

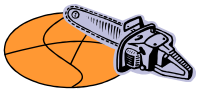


Agentenorientierte Konzepte (2)

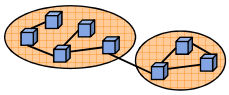
- **Unterschied zur Objektorientierung (höhere Abstraktionsebene)**

Objekt 	Agent 
▪ Kapselung von – Identität („wer“) – Zustand („was“) – Verhalten („wie“) ▪ Keine Kontrolle über Aufruf	▪ Zusätzlich Kapselung von – Freiheitsgrade der Aktionswahl und Interaktion ▪ Anfrage statt Aufruf **➔„wann“** **➔„mit wem“** **➔„ob überhaupt“**

- **Wie entwickelt man Software agentenorientiert?**

1. Zerlegung und Abstraktion 

– Zerlegung in autonome Agenten
– Abstraktion auf Basis agentenorientierter Konzepte

2. Strukturierung 

– Strukturen werden nicht explizit vorgegeben
– Beschreibung möglicher Beziehungen durch Interaktionsmodelle

➔ Agentenorientierte Softwareentwicklung bietet
– Entwurf einer flexiblen Lösung
– Mögliche Strukturen und Verhalten im Rahmen festgelegter Variationen
➔ Reaktion auf Situationen, die nicht konkret im Entwurf vorgesehen sind

Fuzzy-Logik

- **Fuzzy-Logik**

– Übersetzt: „Unscharfe Logik“
– Erweitert die klassische Mengenlehre durch „unscharfe“ Fuzzy-Mengen
– Erlaubt verbale Formulierung von Expertenwissen

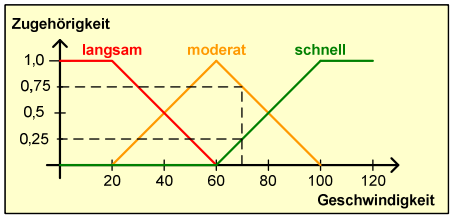
➡ bietet eine gute Möglichkeit, unscharfe menschliche Bewerbungsmaßstäbe und Schlussfolgerungsabläufe nachzubilden. [Lippe06]


Lotfi A. Zadeh

- **Grundlage der Fuzzy-Logik**

– Darstellung von Fuzzy-Mengen über Zugehörigkeitsfunktionen

Fuzzy-Menge „Geschwindigkeit“:



Klassisch: 100 km/h = schnell
60 km/h = moderat
70 km/h = ?

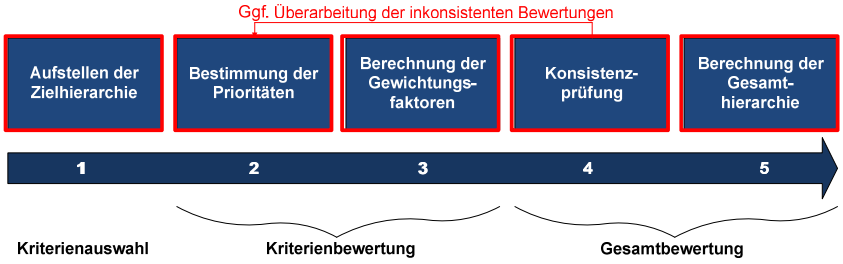
Analytic Hierarchy Process

- **Analytic Hierarchy Process (AHP)**
  - Zur Überprüfung und Ergänzung von subjektiven „Bauch-Entscheidungen“
  - Zum Herausarbeiten von qualitativen Gewichtungsentscheidungen

  ➡ ermöglicht es, komplexe Gewichtungsentscheidung systematisch zu vereinfachen und dadurch rational und transparent zu treffen.


Thomas L. Saaty

- **Ablauf der Gewichtung relevanter Faktoren durch AHP**

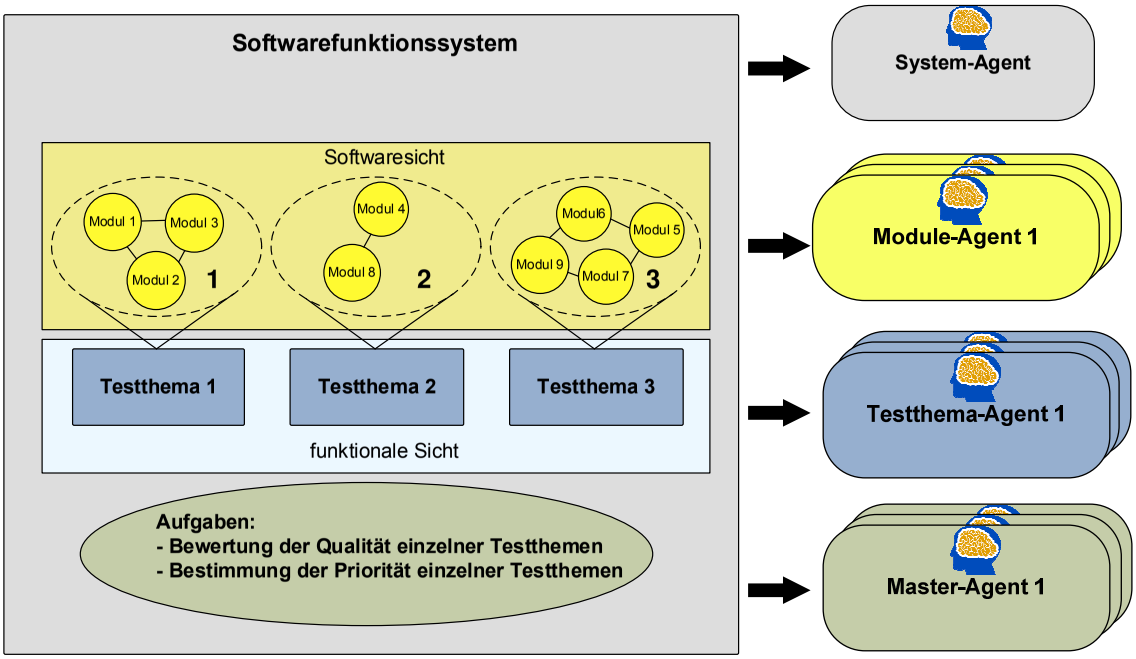


Gewichtungsmatrix	Funktionalität	Zuverlässigkeit	Änderbarkeit	Gewicht
Funktionalität	1	1	2	40%
Zuverlässigkeit	1	1	2	40%
Änderbarkeit	1/2	1/2	1	20%

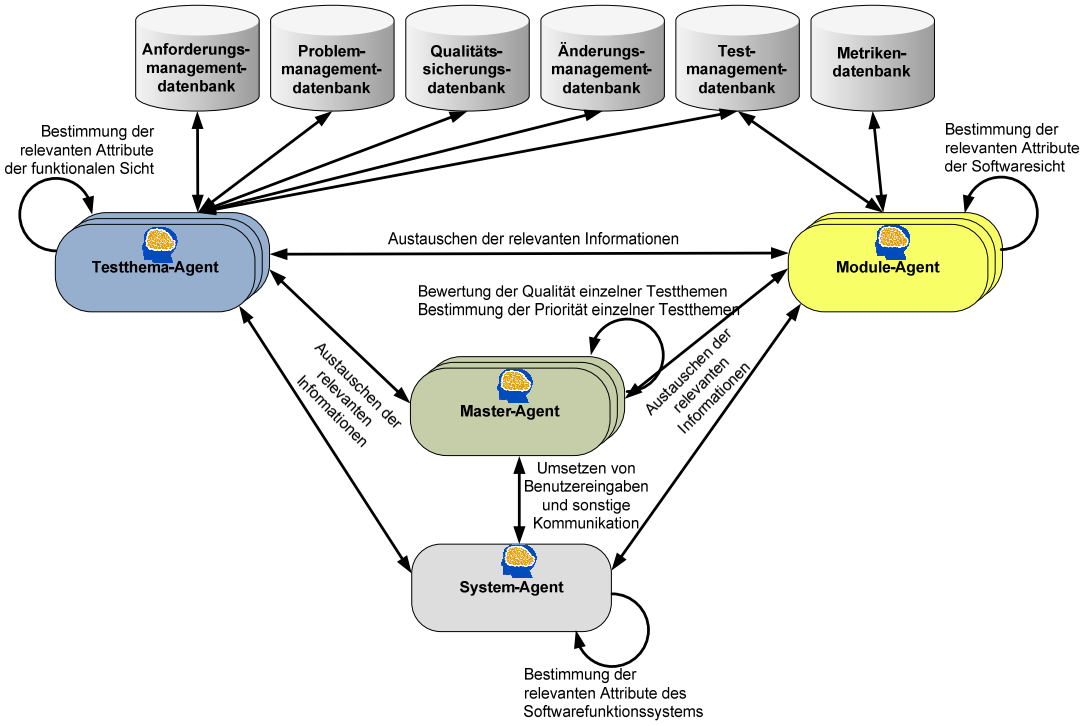
Gliederung

- Stand der Technik und Problembeschreibung
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Agententypen



Interaktionen der Agenten



Qualitätsbewertung (1)

- **Was bedeutet Stabilität?**

– ISO 9126:
Wahrscheinlichkeit des Auftretens unerwarteter Wirkungen von Änderungen.

– Interpretation des Testmanagers:
Stabil, wenn nach dem hinreichenden Test nur geringe Fehlerwirkung entdeckt wurde und nicht mehr so häufig geändert wurde.

- **Verwendung des GQM-Ansatz [EbDu96]**

– *Ziel:* Bewertung der Stabilität eines Testthemas

– *Frage 1:* Wie ist der Vertrauensgrad des Testthemas?

Metrik1: Entdeckte Fehler
Metrik 2: Fehlerschwere
Metrik 3: Anzahl ausgeführter Testfälle
Metrik 4: Testabdeckung

$$Vertrauensgrad_{Test_j} = \left(1 - \frac{\sum_{i=1}^{n} Entdeckter\ Fehler_i \cdot Fehlerschwere}{Anzahl\ ausgeführter\ Testfälle}\right) \cdot Testabdeckung$$

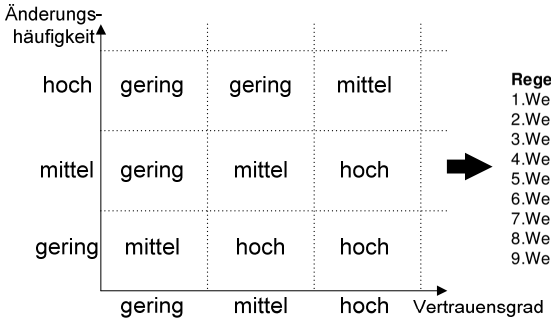
– *Frage 2:* Wie ist die Änderungshäufigkeit des Testthemas?

Metrik 1: Anzahl der Änderungen
Metrik 2: Datum der Änderung

$$Änderungshäufigkeit_{Test_j} = \frac{Anzahl\ der\ Änderungen_{Test_j}}{Datum_n - Datum_1}$$

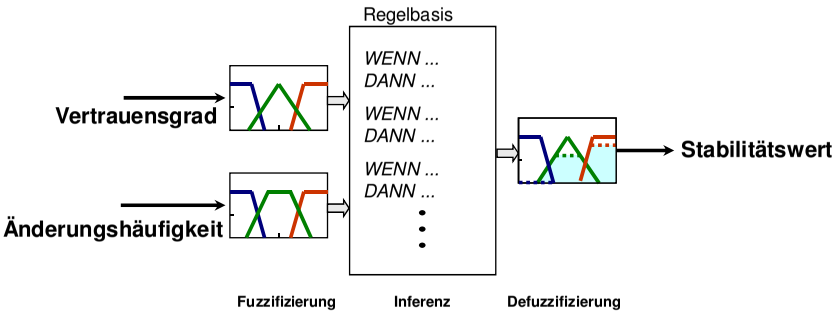
Qualitätsbewertung (2)

- **Aufstellen der Regeln**



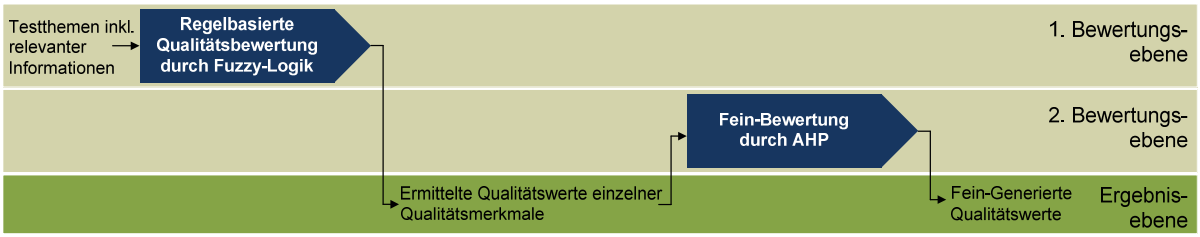
Regelbasis zur Bewertung der Stabilität einzelner Testthemen
1.Wenn (Vertrauensgrad ist hoch) und (Änderungshäufigkeit ist hoch) dann (Stabilität ist mittel)
2.Wenn (Vertrauensgrad ist hoch) und (Änderungshäufigkeit ist mittel) dann (Stabilität ist hoch)
3.Wenn (Vertrauensgrad ist hoch) und (Änderungshäufigkeit ist gering) dann (Stabilität ist hoch)
4.Wenn (Vertrauensgrad ist mittel) und (Änderungshäufigkeit ist hoch) dann (Stabilität ist gering)
5.Wenn (Vertrauensgrad ist mittel) und (Änderungshäufigkeit ist mittel) dann (Stabilität ist mittel)
6.Wenn (Vertrauensgrad ist mittel) und (Änderungshäufigkeit ist gering) dann (Stabilität ist hoch)
7.Wenn (Vertrauensgrad ist gering) und (Änderungshäufigkeit ist hoch) dann (Stabilität ist gering)
8.Wenn (Vertrauensgrad ist gering) und (Änderungshäufigkeit ist mittel) dann (Stabilität ist gering)
9.Wenn (Vertrauensgrad ist gering) und (Änderungshäufigkeit ist gering) dann (Stabilität ist mittel)

- **Bewertung durch Fuzzy-Logik**



Qualitätsbewertung (3)

- **Zweistufige Qualitätsbewertung**
  - Regelbasierte Qualitätsbewertung durch Fuzzy-Logik
  - Fein-Bewertung durch Analytic Hierarchy Process (AHP)



$Angemessenheitswert_{Test_j}$ w_1
$Richtigkeitswert_{Test_j}$ w_2
$Sicherheitswert_{Test_j}$ w_3
$Ordnungsmäßigkeitswert_{Test_j}$ w_4
.....

$Funktionalitätswert_{Test_j}$ w'_1
$Zuverlässigkeitswert_{Test_j}$ w'_2
$Änderbarkeitswert_{Test_j}$ w'_3

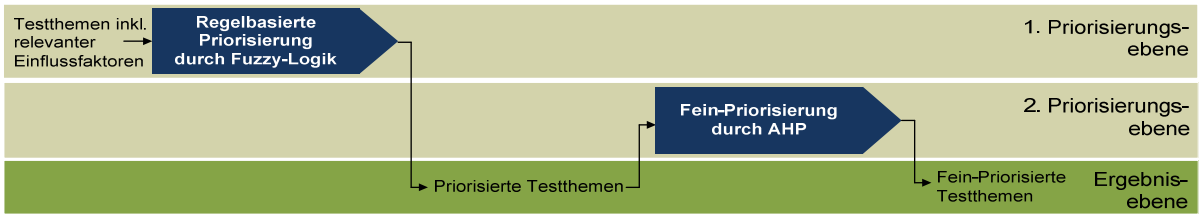
$$Qualitätswert_{Test_j} = \sum_{i=1}^{n} Merkmalwert_{ji} \cdot w_i$$

010_SBW Q_1, w_1
030_FktReg Q_2, w_2
040_Strat Q_3, w_3
050_Ersatzfkt Q_4, w_4
060_10_OBD05 Q_5, w_5
.....

$$Qualitätswert_{System} = \frac{\sum_{j=1}^{n} Qualitätswert_{Test_j} \cdot w_j}{\sum_{j=1}^{n} w_j}$$

Prioritätsbestimmung

- **Zweistufige Priorisierung**
  - Regelbasierte Priorisierung durch Fuzzy-Logik
  - Fein-Priorisierung durch Analytic Hierarchy Process (AHP)



- **Einflussfaktoren auf Priorisierung**
  - Sicherheitsrelevanz w_1
  - Implementierungskomplexität w_2
  - Änderungshäufigkeit w_3
  - Fehleranfälligkeit w_4
  - Vertrauensgrad w_5
  - Qualität w_6

Regelbasis zur Priorisierung von Testthemen

1. Wenn (Sicherheitsrelevanz ist hoch) dann (Priorität ist hoch) (w_1)
2. Wenn (Sicherheitsrelevanz ist mittel) dann (Priorität ist mittel) (w_1)
3. Wenn (Sicherheitsrelevanz ist gering) dann (Priorität ist gering) (w_1)
4. Wenn (Implementierungskomplexität ist hoch) dann (Priorität ist hoch) (w_2)
5. Wenn (Implementierungskomplexität ist mittel) dann (Priorität ist mittel) (w_2)
6. Wenn (Implementierungskomplexität ist gering) dann (Priorität ist gering) (w_2)
7. Wenn (Änderungshäufigkeit ist hoch) dann (Priorität ist hoch) (w_3)
8. Wenn (Änderungshäufigkeit ist mittel) dann (Priorität ist mittel) (w_3)
9. Wenn (Änderungshäufigkeit ist gering) dann (Priorität ist gering) (w_3)
10. Wenn (Fehleranfälligkeit ist hoch) dann (Priorität ist hoch) (w_4)
11. Wenn (Fehleranfälligkeit ist mittel) dann (Priorität ist mittel) (w_4)
12. Wenn (Fehleranfälligkeit ist gering) dann (Priorität ist gering) (w_4)
13. Wenn (Vertrauensgrad ist hoch) dann (Priorität ist gering) (w_5)
14. Wenn (Vertrauensgrad ist mittel) dann (Priorität ist mittel) (w_5)
15. Wenn (Vertrauensgrad ist gering) dann (Priorität ist hoch) (w_5)
16. Wenn (Qualität ist hoch) dann (Priorität ist gering) (w_6)
17. Wenn (Qualität ist mittel) dann (Priorität ist mittel) (w_6)
18. Wenn (Qualität ist gering) dann (Priorität ist hoch) (w_6)

Gliederung

- Stand der Technik und Problembeschreibung
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Evaluierung des Prototyps

- **Evaluierung**

– Entwicklung in Java
– Bibliotheken mittels JADE (Java Agent Development Framework)
– Open Source Bibliothek jFuzzylogic
– Open Source Bibliothek Apache POI
– Verwendung der historischen Daten aus abgeschlossenen Projekten

- **Erstes Ergebnis**

➡ Mithilfe des ATMS können viele Testthemen in folgenden Testzyklen ausgeschlossen werden. Weitere Evaluierung läuft zur Zeit.

Gliederung

- Stand der Technik und Problembeschreibung
- Ziel und Konzeptidee
- Grundlagen des Konzepts
- Konzept des agentenbasierten Testmanagementsystems
- Evaluierung des Prototyps
- Zusammenfassung und Ausblick

Zusammenfassung und Ausblick

- **Zusammenfassung**

– Agentenbasierter Konzept zur Bewertung der Softwarequalität sowie zur Bestimmung der Priorität einzelner Testthemen

– Verlässliche Aussage über die aktuelle Softwarequalität einzelner Testthemen

– Erleichterung der Testthemenpriorisierung

➡ – Fokussierung der Testaktivitäten auf die kritischen Stellen
– Verhinderung vom unnötigen Testen

- **Ausblick**

– Optimierung der Regelbasis

– Weitere Evaluierung des Konzepts bei der ZF Friedrichshafen AG

Vielen Dank für Ihr Interesse!