

Telling Test Stories

Modellbasierte Qualitätssicherung komplexer kooperativer Systeme

Michael Felderer

SoftNet 2008

Innsbruck, 8.Mai 2008

Überblick

- **Grundbegriffe**
- **Telling TestStories**
 - Überblick
 - Fragestellungen
- **TTS und Softnet**

Grundbegriffe

- **Systemtest**

- Test des gesamten Systems gegen seine Anforderungsspezifikation (Akzeptanztest)

- **Modellbasiertes Testen**

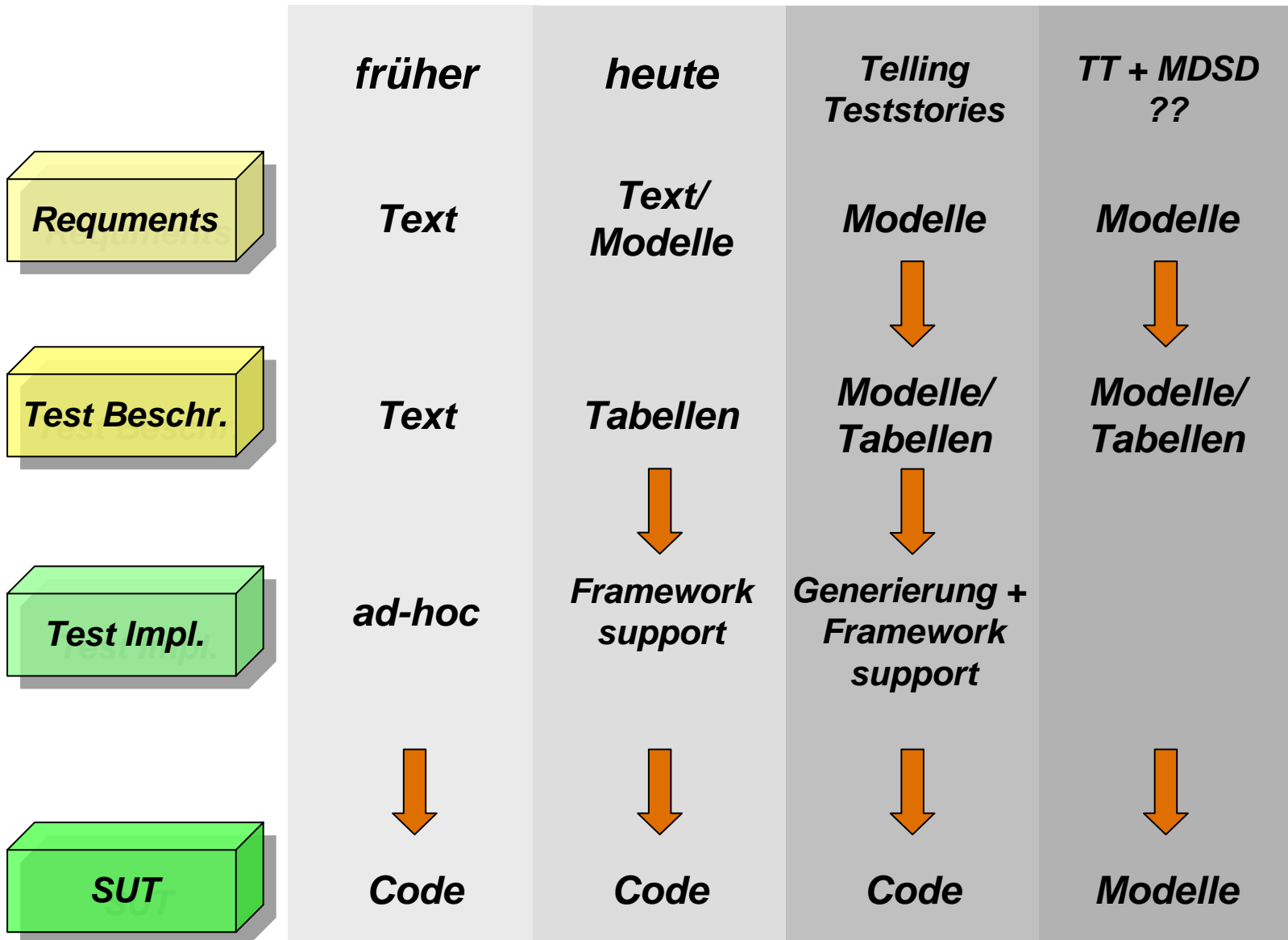
- Generierung bzw. Definition von Testfällen basierend auf einem Modell des zu testenden Systems (SUT)

- **Fit/Fitness**

- Automatisierung von Systemtests in Tabellenform (FIT) und Integration mit einem Wiki (Fitnessse)
- Verbindung zum SUT über sog. **Fixtures**
- keine Verbindung zur Spezifikation

CalculateDiscount	
amount	discount()
0.00	0.00
100.00	0.00
999.00	0.00
1000.00	50.00 <i>expected</i>
	0.0 <i>actual</i>
2000.00	100.00

Entwicklung von Systemtests

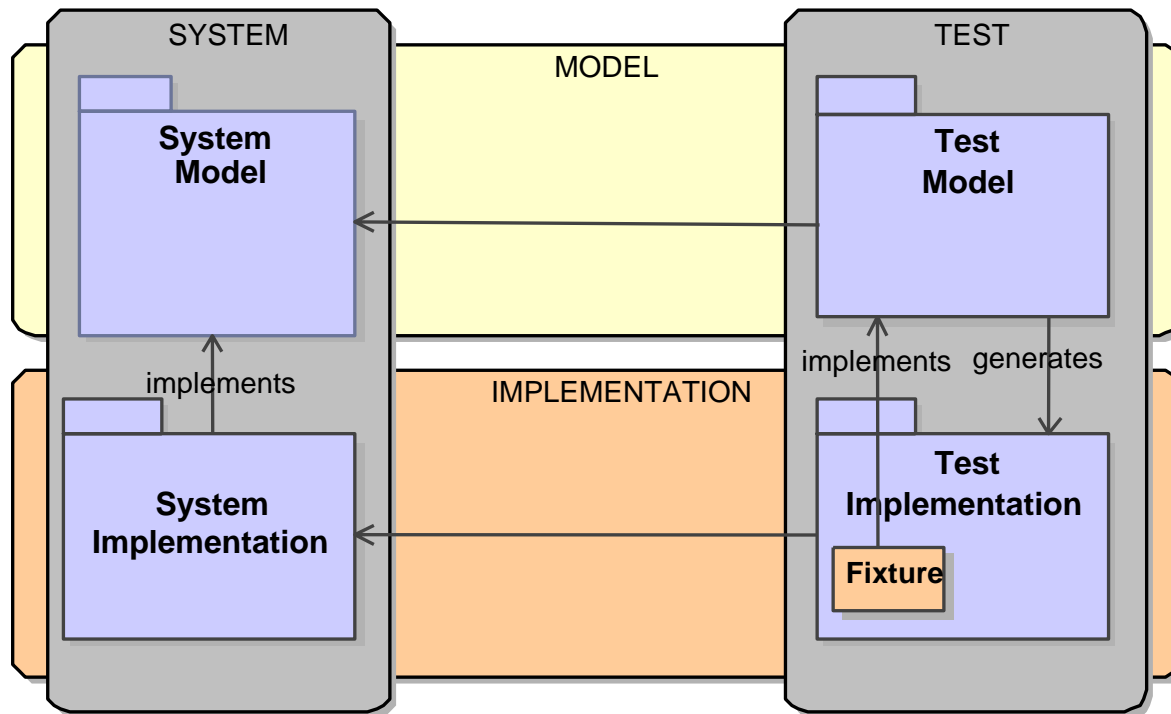


TTS Anforderungen

- **Erstellung eines Testmodells auf Basis der Anforderungen**
- **Ausführbarkeit der Teststories**
 - Teststories sind kontrollierte Abfolgen von Serviceaufrufen auf dem zu testenden System mit Assertions
- **Formales System- und Testmodell**
- **Trennung von Teststories und Testdaten**
- **Testen servicebasierter Systeme**
 - verschiedene Akteure rufen unterschiedliche Services auf

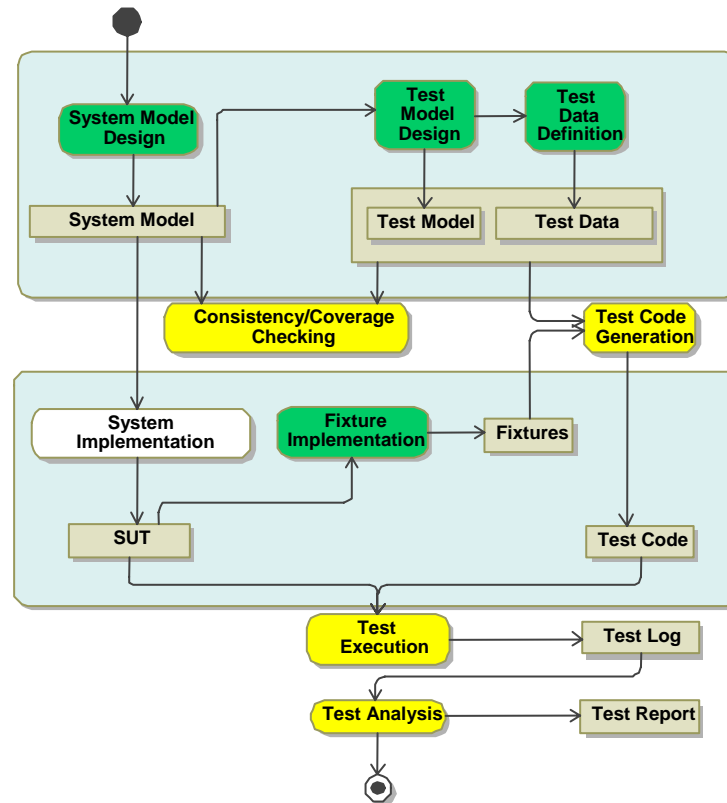
Basiskonzepte

- das Testmodell verwendet Services, Akteure und Klassen des Systemmodells



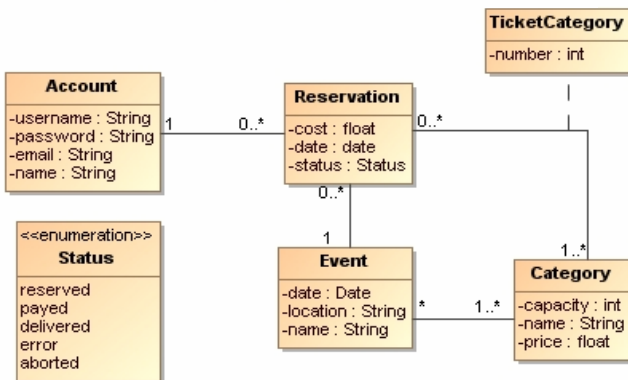
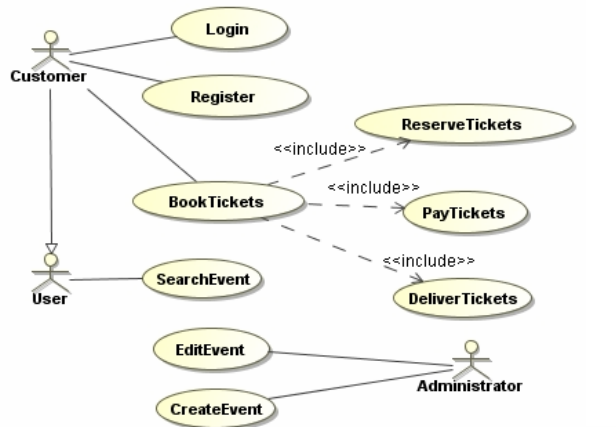
TTS Methode

- Die TTS Methode beinhaltet die Erstellung von System- und Testmodell, Testdaten und Fixtures



Beispiel System Model

- **Grundelemente des Systemmodells sind Services, Akteure und Klassen**
 - zusätzliche Modelle wie ein Anwendungsworkflow erlauben die Generierung von Teststories



```

system test {

  actor User

  service ReserveTickets {

    actors (Customer)
    in (event:Event numPerCategory:Sequence(Integer))
    out reservation:Reservation
    pre "numPerCategory->forall(i|i>=0) "
    post "res.status = reserved"

  }

  class Account {

    attributes ( password:String
                  email:String
                  username:String
                  name:String )

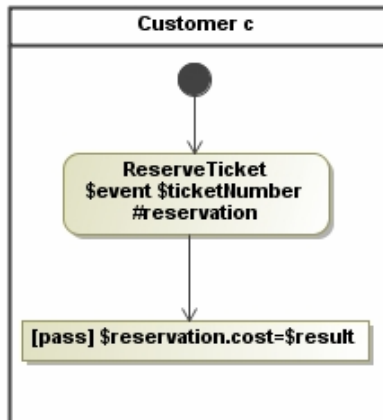
    associations (res:Reservation)

  }

}
  
```

Beispiel Test Model

- Testscenario zur Überprüfung ob die Reservierungskosten stimmen



```

teststory ReserveTickets_Cost {

  actor Customer c
  testtable ReserveTicket_Cost
  initial init_1
  final final_1

  sequence {
    service c:ReserveTicket ($event $ticketNumber) #reservation
    assertion {
      [pass] #reservation.cost=$result
    }
  }
}

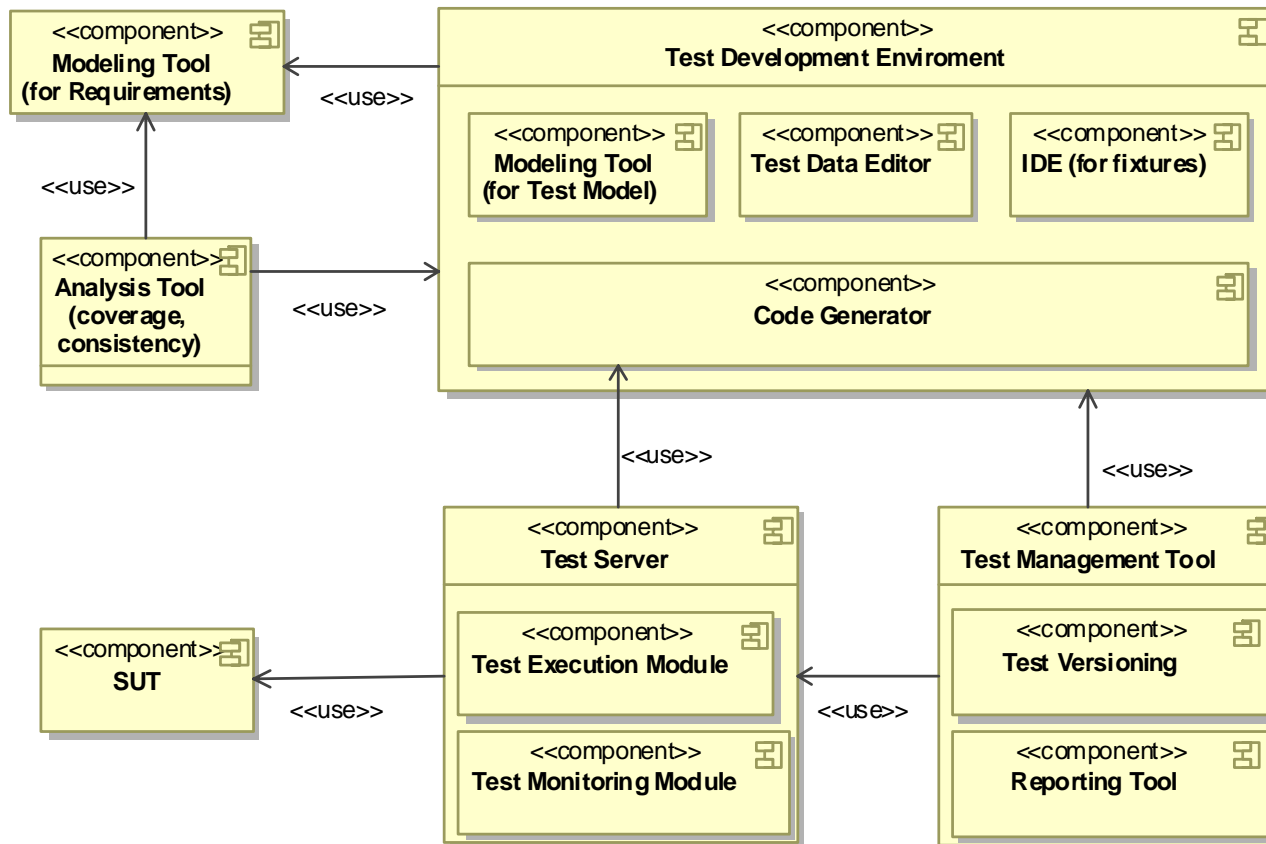
```

id	event	ticketNumber	result
t_1	ref:event_1	[1,0,0]	135,00
t_2	ref:event_1	[0,1,1]	225,00
t_3	ref:event_1	[3,0,1]	650,00

id	date	location	name
event_1	08.06.2008 20:45	Vienna Happel Stadium	EM08 AUT - CRO
event_2	08.06.2008 20:45	Klagenfurt Stadium	EM08 GER - POL
event_3	09.06.2008 20:45	Zuerich Stadium	EM08 NED - ITA

Architektur

- Module des TTS Frameworks



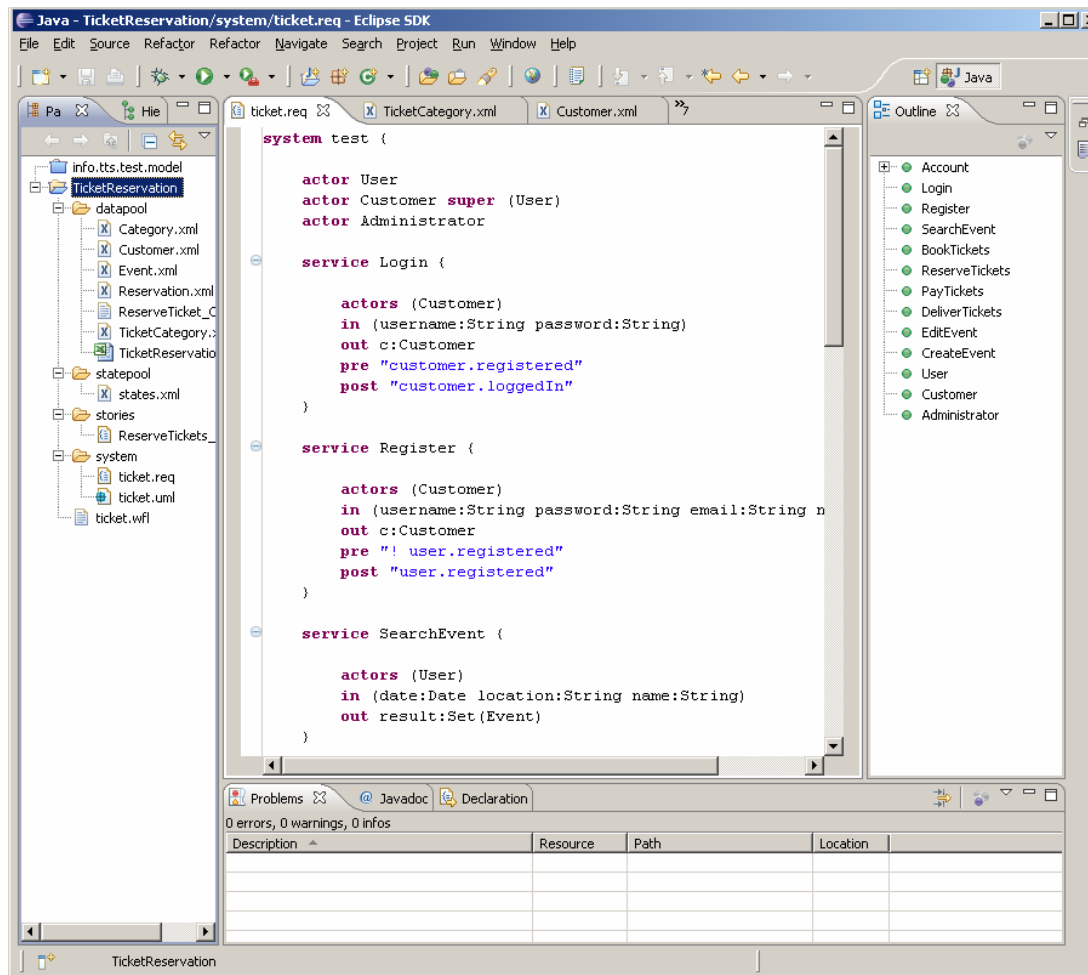
Technologien

- **Eclipse Plugins für die Verwaltung von Testsuiten**
- **ecore basiertes Metamodell für System- und Testmodell**
- **openArchitectureWare für die Generierung von Testcode**
- **Workflow Engine SecServ**



Editor Beispiel

- Editor zur Erstellung von Test Suiten



Vorteile

- **Modellbasiertes Testen**
 - Erstellung von Tests in früher Projektphase,
Codeunabhängigkeit
- **flexibel einsetzbares Systemmodell**
- **Trennung von System- und Testmodell**
- **Trennung von Testlogik und Testdaten**
- **Verbindung zwischen Anforderungsspezifikation und Testmodell**

Wissenschaftliche Fragestellungen

- **Automatische Generierung von Test Stories und Testfällen**
- **Konsistenzprüfung zwischen Testmodell und Anforderungsspezifikation**
- **Definition und Implementierung von Überdeckungskriterien der Anforderungsspezifikation durch Test Stories**
- **Testen Serviceorientierter Anwendungen**
- **Testausführung und -auswertung**
 - Definition komplexer Verdiktfunktionen
 - Formalisierung des Zustandsmodells
 - Workflow für Testkampagnen

Technische Problemstellungen

- **Setzen von Anfangszuständen für Test Stories**
- **Vollständige Testfallgenerierung**
- **Integration und Erweiterung der Workflow Engine**
- **Versionierung von Testfällen**
- **Anwendung auf industrielle Fallstudien**



TTS und Softnet

- Telling TestStories verwendet für die Prüfung von Konsistenz- und Überdeckungseigenschaften das Framework SQUAM aus Softnet

