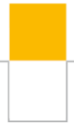


sepp.med
Qualität sichert Erfolg

Modellbasiertes Testdesign in der Praxis

200 Testfälle in 20 Minuten

Überblick

- Motivation
- Erstellung eines Testmodells
- Erzeugung von Testfällen
- Zusammenfassung

Eine einfache Rechenaufgabe

- Wenn man zur Erstellung von 100 Testfälle 10 Tage braucht wie viele Tage braucht man dann für 1000 Testfälle?

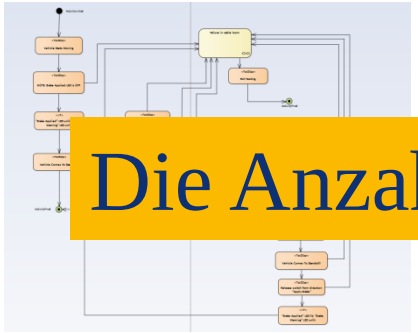
oder

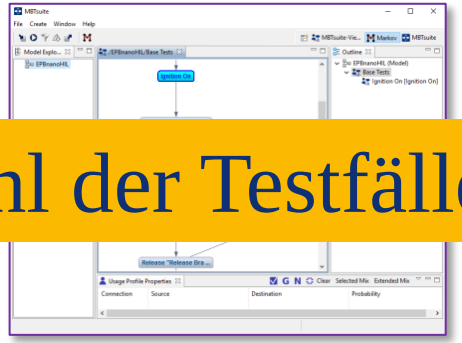
- Wie korreliert der Aufwand im Testdesign mit der Anzahl der Testfälle?
  - linear
  - weniger
  - mehr

Big Picture: Modell – Testfall – Testumgebung

sepp.med
Qualität sichert Erfolg

Test-Design → Testauswahl → Automatisierung → Test-Execution





Die Anzahl der Testfälle spielt keine Rolle mehr

Manuell → automatisch → automatisch → automatisch

Wo steckt der Aufwand im Test

- Testkonzept - 6%
- Testdesign - 54%
- Durchführung - 38%
- Auswertung - 2%

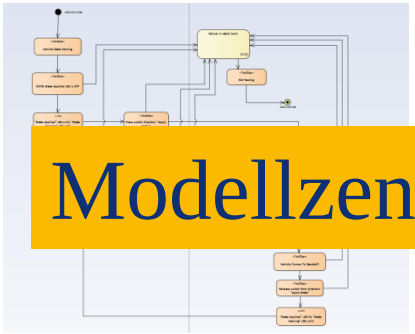
Wie kann man effektiver werden?

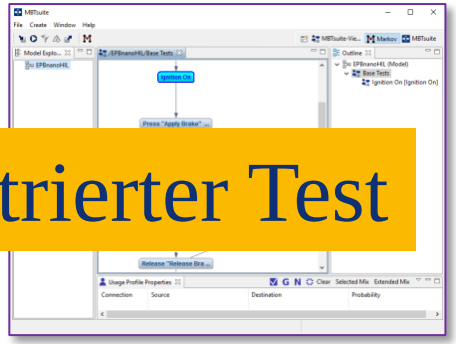
Modellzentriertes Testdesign, Testfallgenerierung und Testautomatisierung

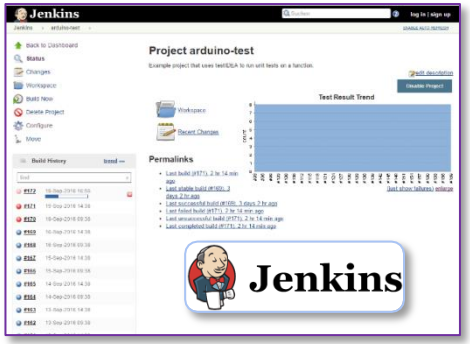
Big Picture: Modell – Testfall – Testumgebung

sepp.med
Qualität sichert Erfolg

Test-Design → Testauswahl → Automatisierung → Test-Execution





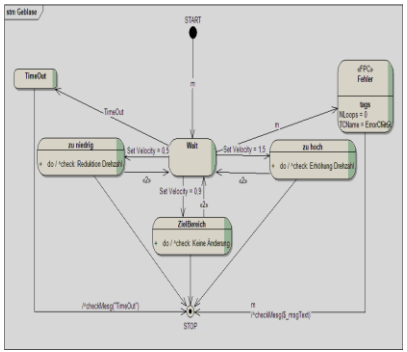


Modellzentrierter Test

Manuell → automatisch → automatisch → automatisch

Modellzentrierter Test

Testmodell



Generator

MBTsuite
THE TESTING FRAMEWORK

TestSpec.

TestScript.

TestData.

automatisch

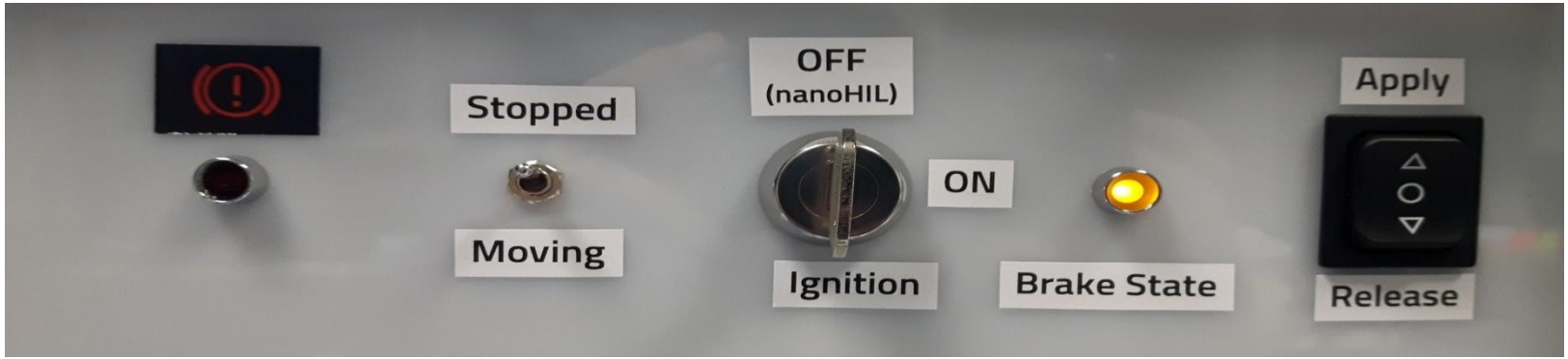
MBT – was werden wir damit erreichen ?

- Anforderungen richtig verstehen
- Das SUT in seinem Gesamtzusammenhang beschreiben
- Testfällen systematisch erzeugen
- Schneller auf Änderungen reagieren
- Bessere Testspezifikationen erstellen

- Effektiver arbeiten
- Die Komplexität besser beherrschen

Testmodellierung am Beispiel Parkbremse

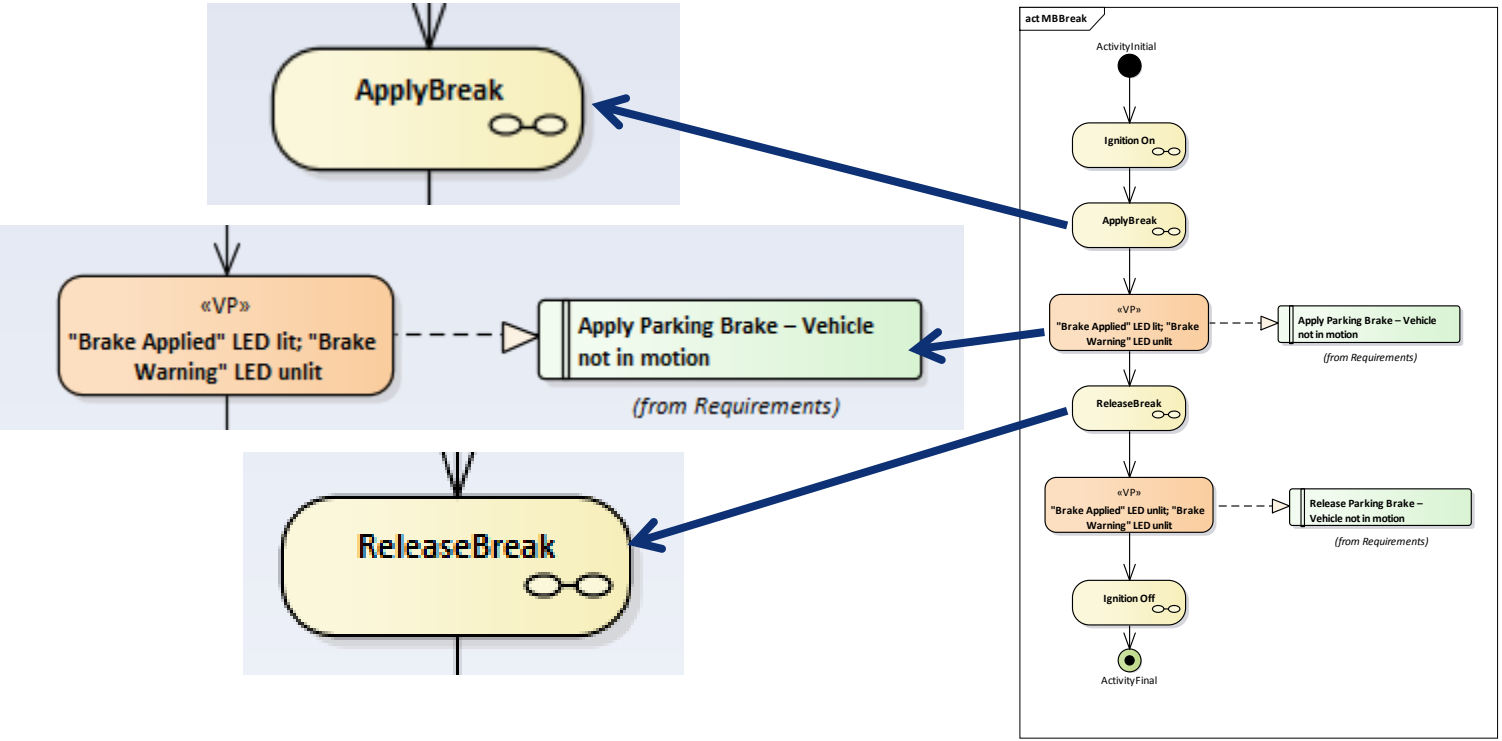
Testmodellierung am Beispiel Parkbremse



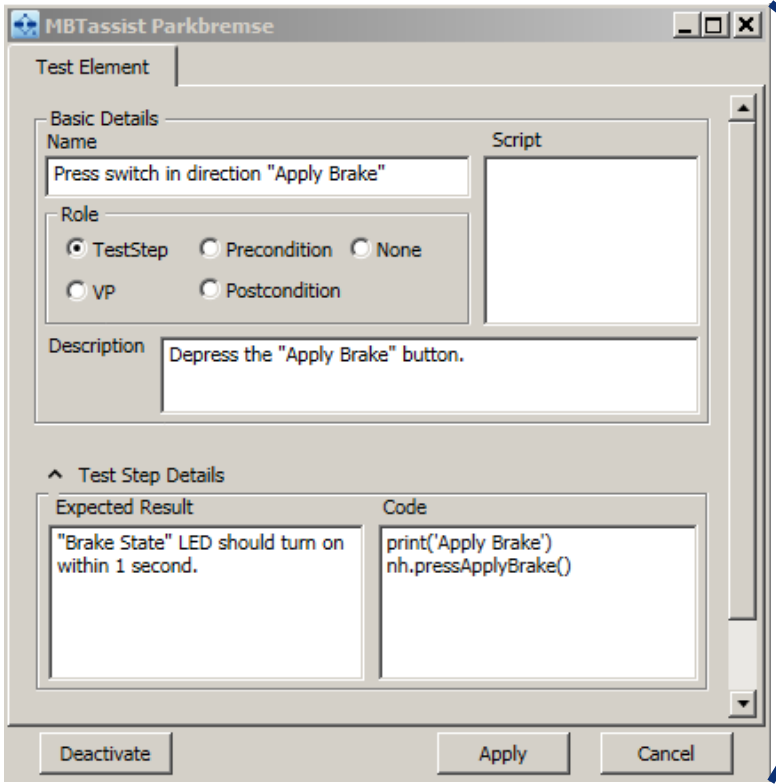
Testmodellierung am Beispiel Parkbremse

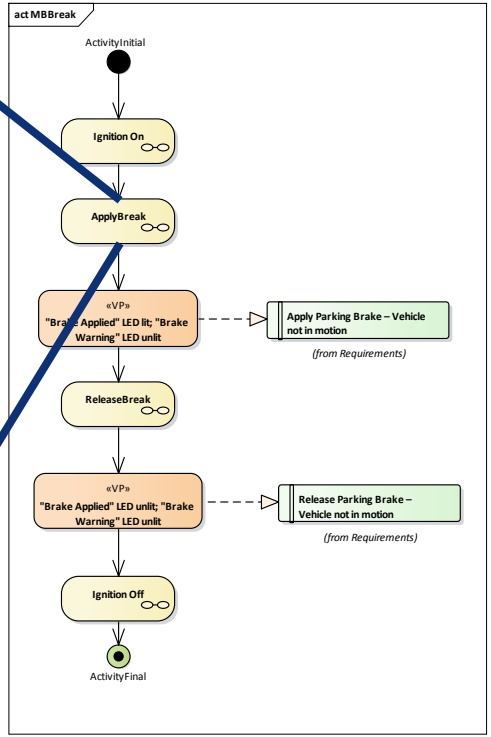
- Anforderungen
  1. Apply Parking Brake – Vehicle not in motion
  2. Release Parking Brake – Vehicle not in motion
  3. System test of all indication LEDs
  4. Automatic brake release upon detection of intended vehicle motion
  5. Application of brake whilst vehicle is in motion; <3s
  6. Application of brake whilst vehicle is in motion; >3s
  7. Detection of a hardware failure in cable loom
  8. Transfer of system into “safe state”

Testmodell der Parkbremse



Testmodell der Parkbremse





Generierte Testspezifikation

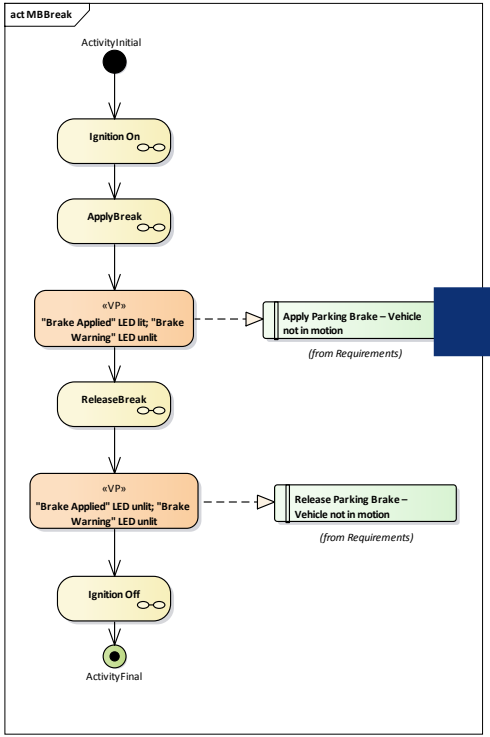
Test Case	MBBreak
Description	
Requirements	System test of all indication LEDs Display of previous brake state Apply Parking Brake – Vehicle not in motion Release Parking Brake – Vehicle not in motion
Precondition(s)	
Postcondition(s)	

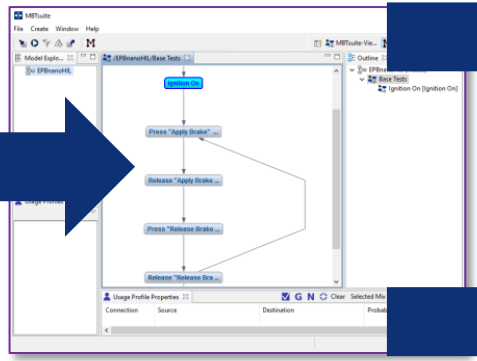
Step	Type	Step Name	Step Description	Expected Result	Requirements
1	Test Step	Turn Ignition On	Switch ignition to "ON"	"Brake Applied" and "Brake Warning" LEDs light for ~1 second.	
2	Verification Point	"Brake Applied" LED lit; "Brake Warning" LED lit	VP: Check "Brake State" and "Brake Warning" LEDs	Both LEDs should be lit (< 1 second)	System test of all indication LEDs
3	Test Step	Wait 2 seconds	Wait 2 seconds	"Brake State" LED should return to previous state (lit if brake is applied, unlit is brake is released). "Brake Warning" LED should be off.	
4	Verification Point	"Brake Warning" LED retains old status	VP: Check "Brake State" LED state.	"Brake State" LED should return to previous state (lit if brake is applied, unlit is brake is released).	
5	Verification Point	"Brake Applied" LED retains old status	VP: Check "Brake Warning" LED state.	"Brake Warning" LED unlit.	Display of previous brake state
6	Test Step	Press switch in direction "Apply Brake"	Depress the "Apply Brake" button.	"Brake State" LED should turn on within 1 second.	
7	Verification Point	"Brake Applied" LED lit; "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs.	"Brake Applied" LED should be lit; "Brake Warning" LED should be unlit.	Apply Parking Brake – Vehicle not in motion
8	Test Step	Press switch in direction "Release Brake"	Depress the "Release Brake" button.	"Brake State" LED should turn off.	
9	Verification Point	"Brake Applied" LED unlit; "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs.	"Brake Applied" LED should be unlit; "Brake Warning" LED should be unlit.	Release Parking Brake – Vehicle not in motion
10	Test Step	Turn Ignition Off	Turn ignition switch to "OFF" position	"Brake Applied" and "Brake Warning" LEDs turn off; brake remains in current position	
11	Verification Point	"Brake Applied" LED unlit; "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs.	Both Brake State and Brake Warning LEDs should be off. Brake should retain its current position.	

Generiertes Testskript

```
void MBBreak(void)
{

    /* Switch ignition to "ON" */
    print('Ignition On')
    nh.turnIgnitionOn()
    /* Expected Result: "Brake Applied" and "Brake Warning" LEDs ligh

    /* VP: Check "Brake State" and "Brake Warning" LEDs */
    print('VP: Brake Applied ON - Brake Warning ON')
    if nh.verifyBrakeLightOn() == False:
        results[key].append(ts.TestStatus(ts.TestStatus.STATUS_FAIL,
        nh.testHasFailed()
        testStatus = "FAIL"

    if nh.verifyWarningLightOn() == False:
        results[key].append(ts.TestStatus(ts.TestStatus.STATUS_FAIL,
        nh.testHasFailed()
```

Was haben wir erreicht?





Test Case	MBBreak
Description	System test of all indication LEDs
Requirements	Display of previous brake state Apply Parking Brake – Vehicle not in motion Release Parking Brake – Vehicle not in motion
Preconditions(s)	
Postcondition(s)	

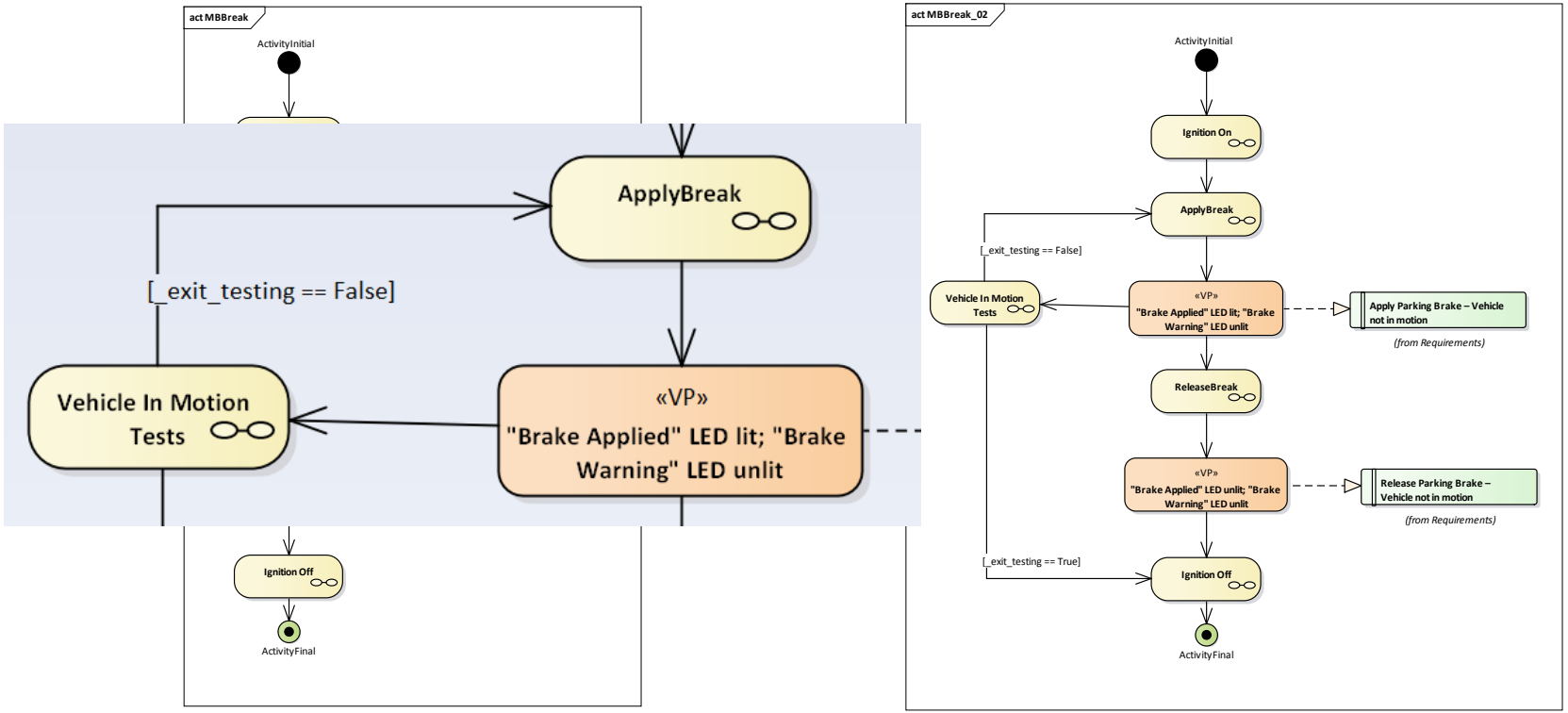
Step	Type	Step Name	Step Description	Expected Result	Requirements
1	Test Step	Turn Ignition On	Switch ignition to "ON"	"Brake Applied" and "Brake Warning" LEDs light for ~1 second	
2	Verification Point	"Brake Applied" LED lit "Brake Warning" LED lit	VP: Check "Brake State" and "Brake Warning" LEDs	Both LEDs should be lit (< 1 second)	System test of all indication LEDs
3	Test Step	Wait 2 seconds	Wait 2 seconds	"Brake State" LED should return to previous state (lit if brake is applied, unlit if brake is released). "Brake Warning" LED should be off	
4	Verification Point	"Brake Warning" LED retains old status	VP: Check "Brake State" LED state	"Brake Warning" LED retains old state (lit if brake is applied, unlit if brake is released).	Display of previous brake state
5	Verification Point	"Brake Applied" LED retains old status "Brake Warning" LED retains old status	VP: Check "Brake Warning" LED state	"Brake Warning" LED unlit.	
6	Test Step	Press switch in direction "Apply Brake"	Depress the "Apply Brake" button.	"Brake State" LED should turn on within 1 second.	
7	Verification Point	"Brake Applied" LED lit "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs	"Brake Applied" LED should be lit. "Brake Warning" LED should be unlit.	Apply Parking Brake – Vehicle not in motion
8	Test Step	Press switch in direction "Release Brake"	Depress the "Release Brake" button.	"Brake State" LED should turn off	
9	Verification Point	"Brake Applied" LED unlit "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs	"Brake Applied" LED should be unlit. "Brake Warning" LED should be unlit	Release Parking Brake – Vehicle not in motion
10	Test Step	Turn Ignition OFF	Turn ignition switch to "OFF" position	"Brake Applied" and "Brake Warning" LEDs turn off. brake remains in current position	
11	Verification Point	"Brake Applied" LED unlit "Brake Warning" LED unlit	VP: Check the state of the "Brake State" and "Brake Warning" LEDs	Both "Brake State" LED and "Brake Warning" LEDs should be off. Brake should retain its current position	

```
void MBBreak(void)
{

    /* Switch ignition to "ON" */
    print('Ignition On')
    nh.turnIgnitionOn()
    /* Expected Result: "Brake Applied" and "Brake Warning" LEDs ligh

    /* VP: Check "Brake State" and "Brake Warning" LEDs */
    print('VP: Brake Applied ON - Brake Warning ON')
    if nh.verifyBrakeLightOn() == False:
        results[key].append(ts.TestStatus(ts.TestStatus.STATUS_FAIL,
        nh.testHasFailed()
        testStatus = "FAIL"

    if nh.verifyWarningLightOn() == False:
        results[key].append(ts.TestStatus(ts.TestStatus.STATUS_FAIL,
        nh.testHasFailed()
```

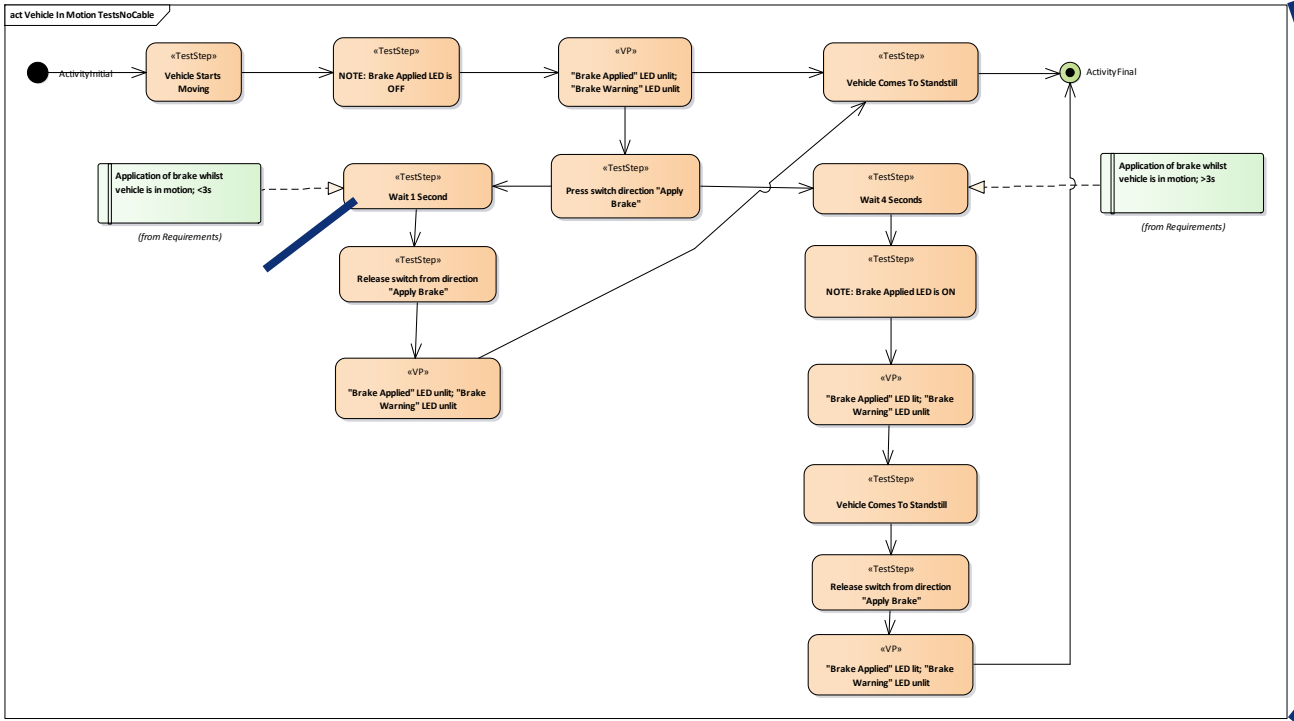
Parkbremse bei bewegtem Fahrzeug

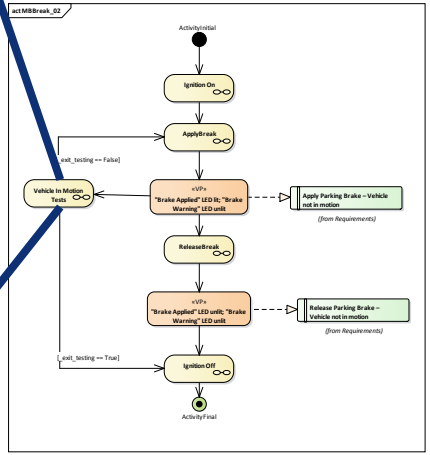
- Anforderungen
  1. Apply Parking Brake – Vehicle not in motion
  2. Release Parking Brake – Vehicle not in motion
  3. System test of all indication LEDs
  4. **Automatic brake release upon detection of intended vehicle motion**
  5. **Application of brake whilst vehicle is in motion; <3s**
  6. **Application of brake whilst vehicle is in motion; >3s**
  7. Detection of a hardware failure in cable loom
  8. Transfer of system into “safe state”

Fahrzeug in Bewegung



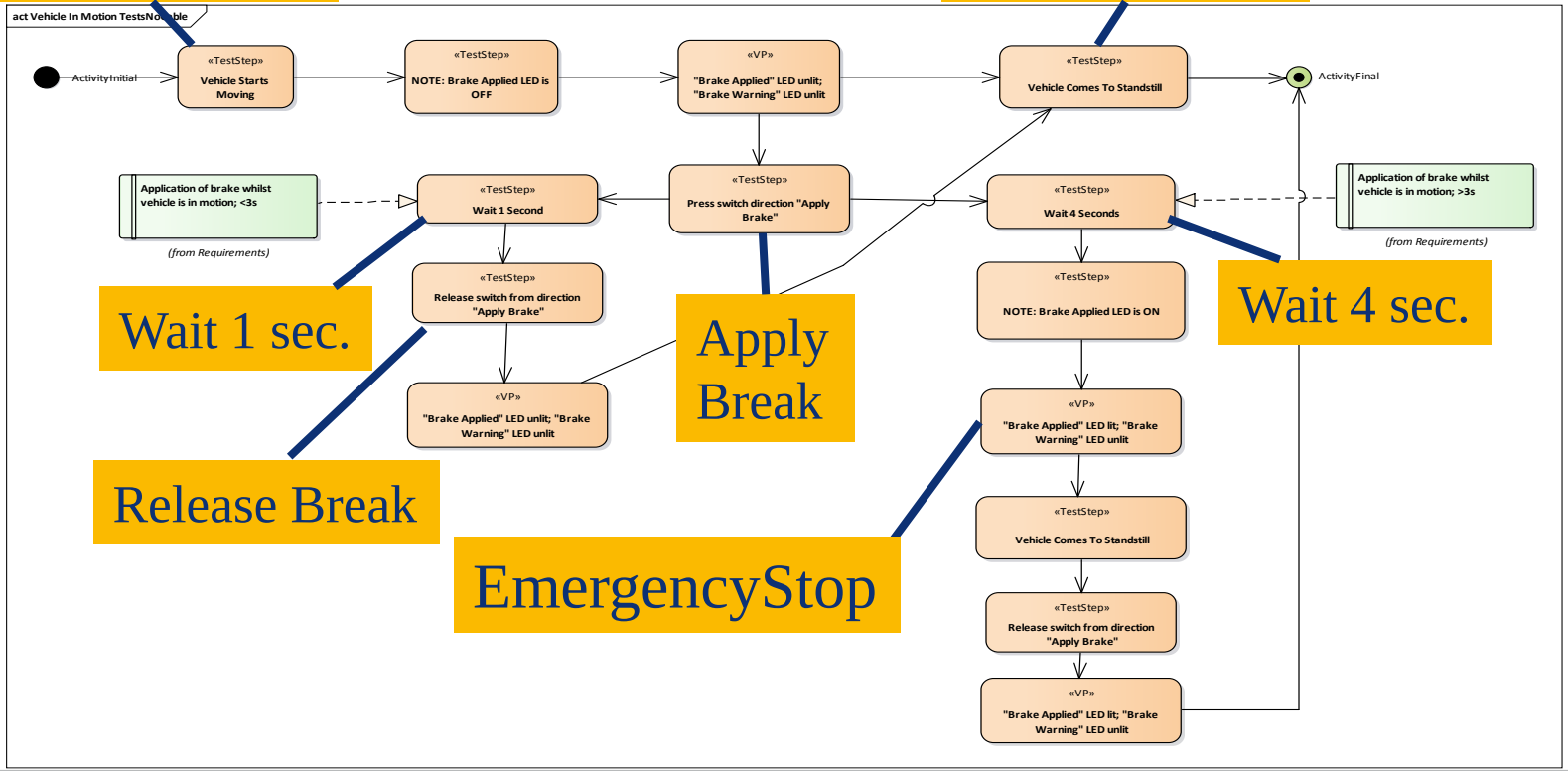
Subdiagramm Fahrzeug in Bewegung





Start moving

Stop moving



Wait 1 sec.

Wait 4 sec.

Apply Break

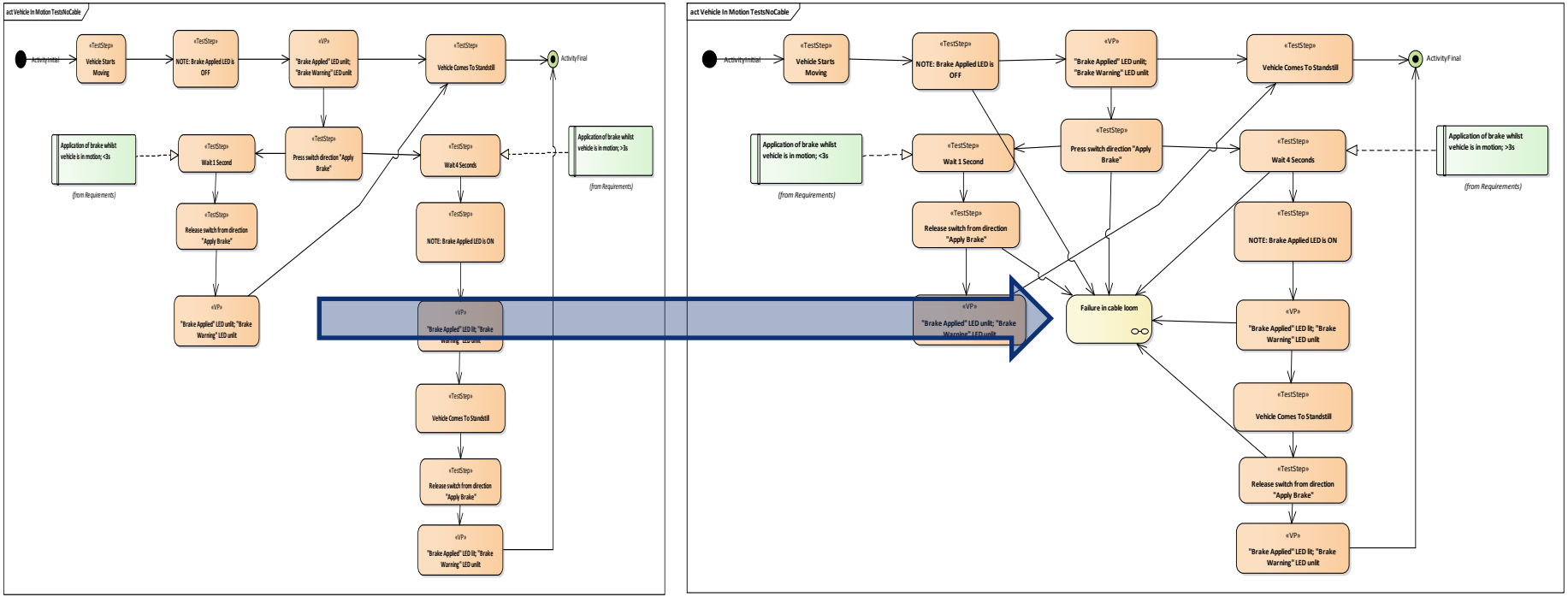
Release Break

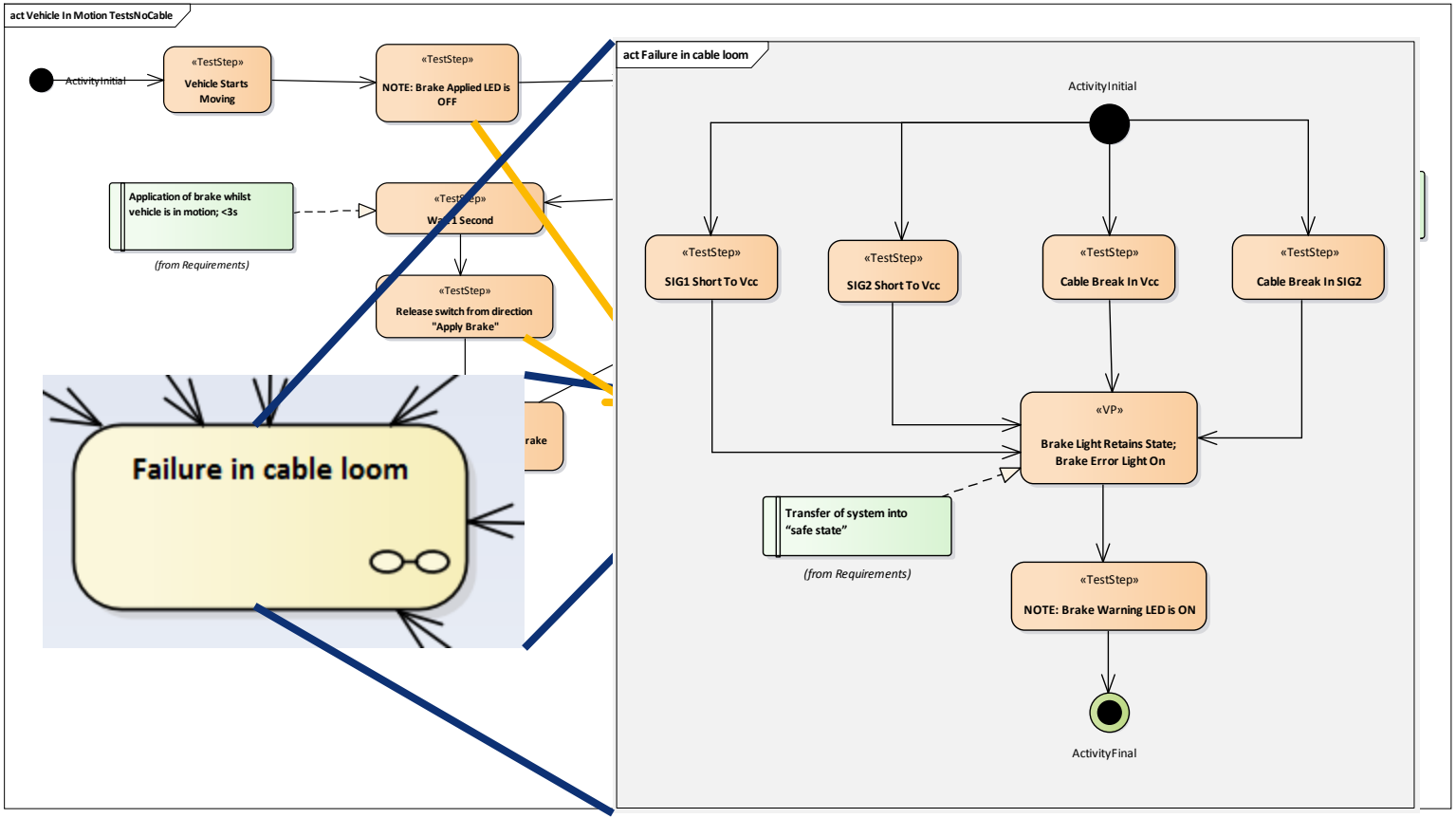
EmergencyStop

Parkbremse bei Fehler im Kabelbaum

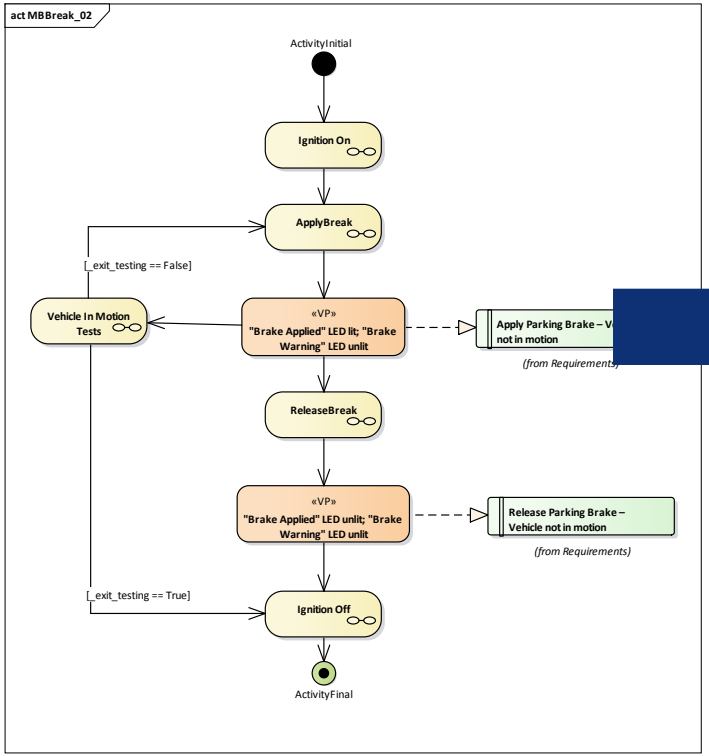
- Anforderungen
  1. Apply Parking Brake – Vehicle not in motion
  2. Release Parking Brake – Vehicle not in motion
  3. System test of all indication LEDs
  4. Automatic brake release upon detection of intended vehicle motion
  5. Application of brake whilst vehicle is in motion; <3s
  6. Application of brake whilst vehicle is in motion; >3s
  7. **Detection of a hardware failure in cable loom**
  8. **Transfer of system into “safe state”**

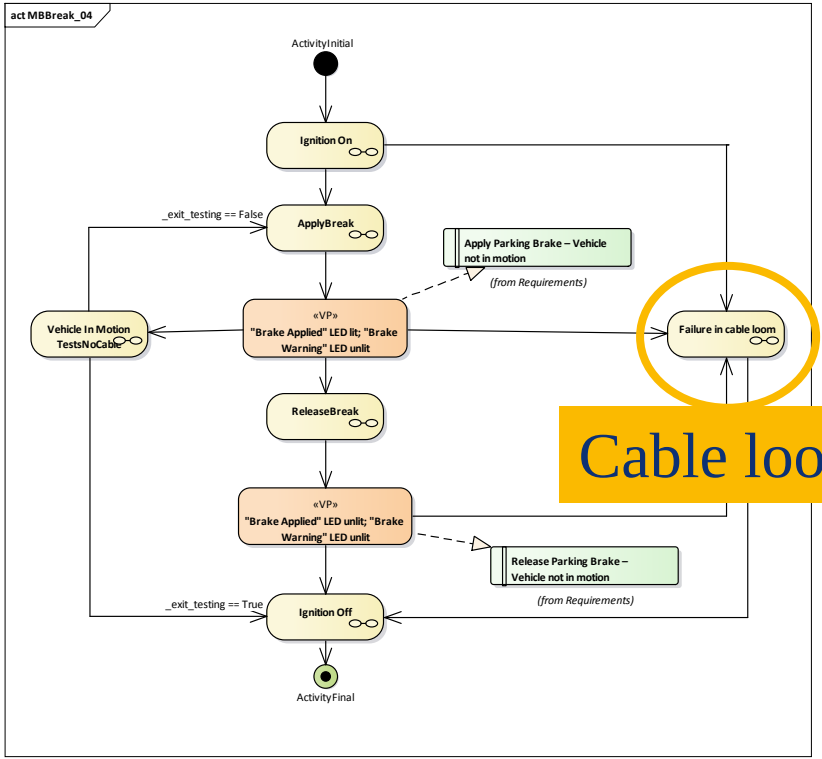
Erweiterung Kabelbruch




act Vehicle In Motion TestsNoCable
ActivityInitial
«TestStep»
Vehicle Starts Moving
«TestStep»
NOTE: Brake Applied LED is OFF
Application of brake whilst vehicle is in motion; <3s
(from Requirements)
«TestStep»
Release switch from direction "Apply Brake"
Failure in cable loom
act Failure in cable loom
ActivityInitial
«TestStep»
SIG1 Short To Vcc
«TestStep»
SIG2 Short To Vcc
«TestStep»
Cable Break In Vcc
«TestStep»
Cable Break In SIG2
«VP»
Brake Light Retains State; Brake Error Light On
Transfer of system into "safe state"
(from Requirements)
«TestStep»
NOTE: Brake Warning LED is ON
ActivityFinal

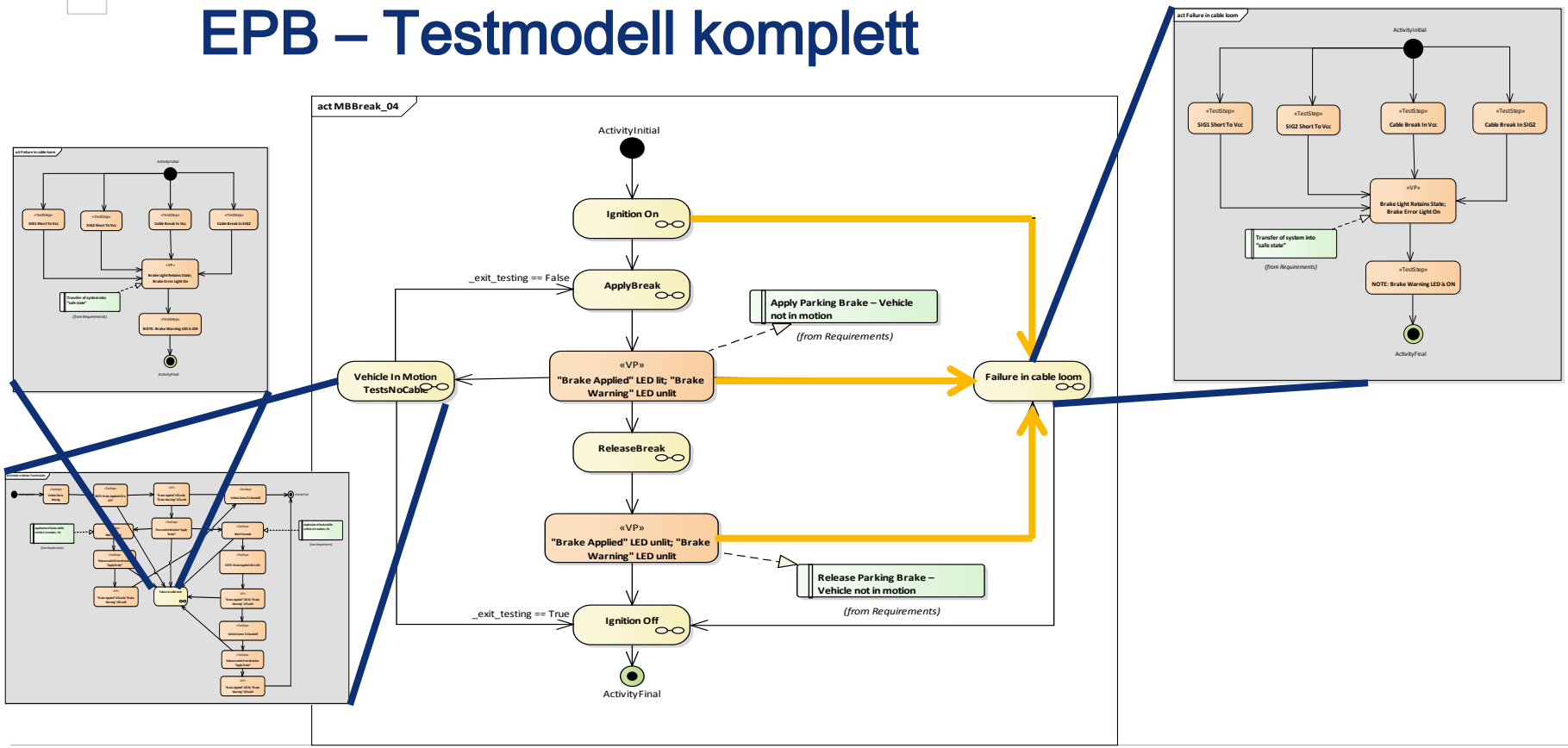
EPB – Testmodell in Ruhe mit Kabelbruch





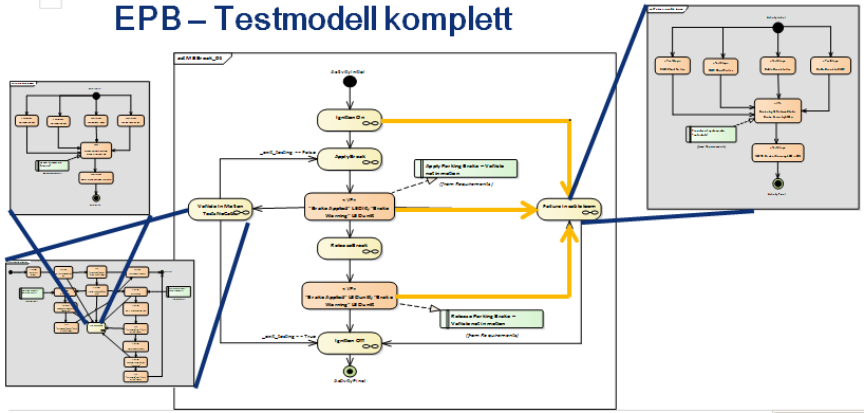
Cable loom

EPB – Testmodell komplett



Nächster Schritt: Testfälle erzeugen

- Wie viele Testfälle enthält das Modell?
- Welche Testabdeckung wollen wir erreichen?
  - 100% Abdeckung
  - Kantenabdeckung
  - Anforderungsabdeckung
  - Pfadabdeckung



Generierungstrategien und -filter

- Strategien
  - Guided Path
  - Shortest Path
  - Random
  - Usage Coverage
  - Full Path

Strategien arbeiten auf dem Model

- Filter
  - Node Coverage
  - Edge Coverage
  - Requirement Coverage
  - Weigth Filter
  - Dynamic Tag Filter

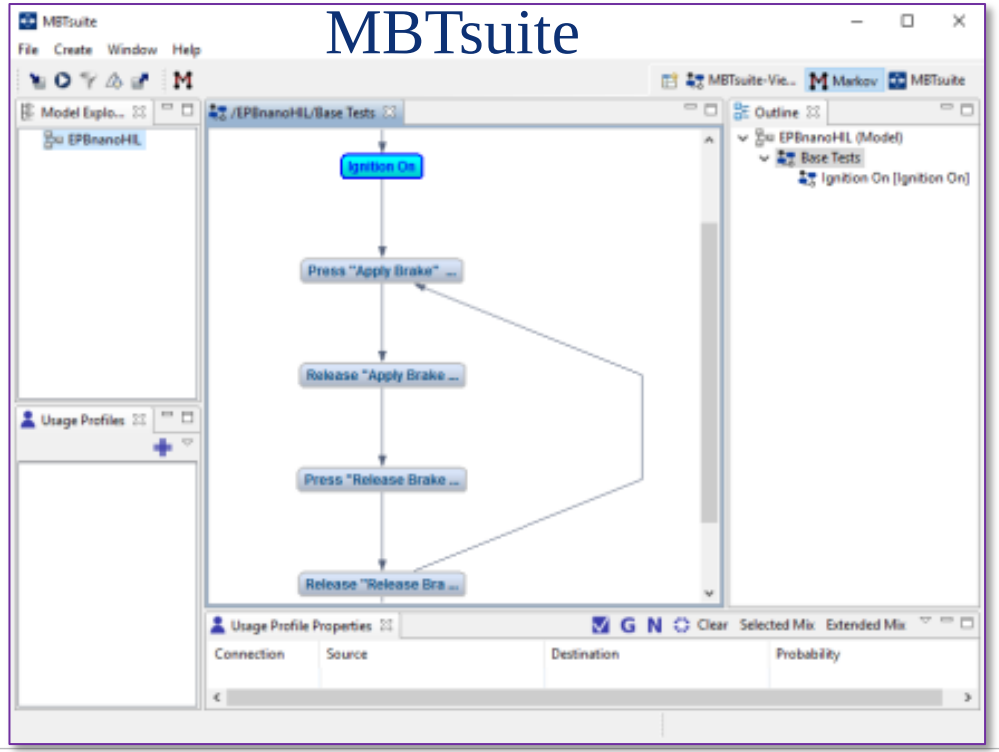
Filter arbeiten auf dem Generat einer Strategie

Testmanagementinformationen

- Prioritäten
- Risikoeinstufung (ASIL, SIL etc.)
- Gewichte
- Kosten
- Dauer
- Benutzerdefinierte Attribute

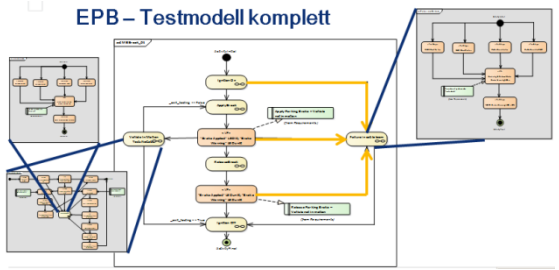
- Beliebige Kombinationen von Attributen unter Verwendung von logischen Ausdrücken

Life-Demo Testfallgenerierung



Nächster Schritt: Testfälle erzeugen

- Wie viele Testfälle enthält das Modell? - 1012
- Welche Testabdeckung wollen wir erreichen?
  - Kantenabdeckung - 2
  - Anforderungsabdeckung - 2 oder 663
  - Pfadabdeckung - 1012
  - zufallsfgesteuert - 200



Zusammenfassung

Wo steckt der Aufwand im Test

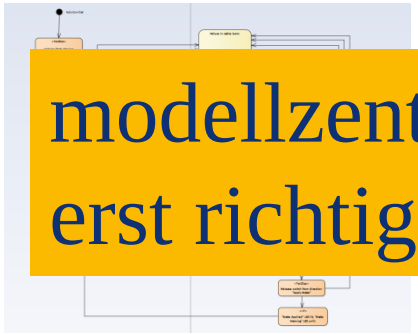
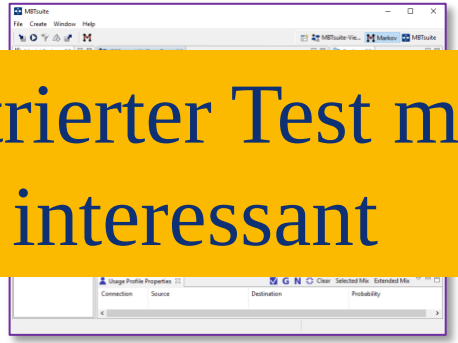
- Testkonzept - 6%
- **Testdesign - 54%** **Reduzierung ~25%**
- Durchführung - 38%
- Auswertung - 2%

- Zusätzliche Reduzierung des Pflegeaufwands der TC: ~80% vor allem im Bereich Testautomatisierung

Big Picture: Modell – Testfall – Testumgebung

sepp.med
Qualität sichert Erfolg

Test-Design → Testauswahl → Automatisierung → Test-Execution

modellzentrierter Test macht Testautomatisierung erst richtig interessant

Manuell → automatisch → automatisch → automatisch

Vielen Dank

Dr. Martin Beißer
Tel.: +49 9195 931-202
eMail: martin.beisser@seppmed.de