# Vorgehensweise JUNIT-Test SWE11.5 LosC3 VRZ3 – Engstellenverwaltung

**Überblick:**
Die SWE Engstellenverwaltung überwacht die Engstellen (Baustellen und Unfälle) von Verkehrsmodellnetzen und führt periodisch 4 verschiedene Prüfungen für die vorhandenen Engstellen durch:

- Für Baustellen:
  - Prüfung Verkehrsstärke auf gesperrten Fahrstreifen
  - Prüfung Engpasskapazität (vs.Verkehrsstärke hinter Baustelle)
  - Prüfung Staulänge über Baustellenende
- Für Unfälle
  - Prüfung Ablaufzeit

Diese Prüfungen werden über Parameter der Attributgruppe EngstellenVerwaltungParameter des Typs EngstellenVerwaltung parametriert.
Wenn eine Prüfung verletzt ist, wird eine Betriebsmeldung generiert.

**Die Junit-Tests sind folgendermassen implementiert:**

1. Am Beginn jedes Tests wird ein Testobjekt (Baustelle oder Unfall) erzeugt, das Testobjekt wird am Ende des Tests wieder gelöscht.
2. Für das jeweilige Testobjekt werden Testdaten generiert
3. Im eigentlichen Test wird das Testobjekt mit den Testdaten versorgt und die zu prüfende Methode aufgerufen, deren Ergebnis mit Assertions ausgewertet wird.
4. Folgende Methoden werden aufgerufen:
   - Baustelle
     - pruefeGesperrteFahrstreifen()
     - pruefeEngpass()
     - pruefeStaus()
   - Unfall
     - pruefeAblauf()
5. Auswertung: Die 4 Methoden geben jeweils eine Sammlung von Ergebnissen zurück, für die Prüfung verletzt ist (ist die Prüfung nicht verletzt, ist die Sammlung leer).

In der gelieferten Implementierung werden die Testdaten gemäss Prüfspezifikation versorgt: Für jede Prüfung wird je ein verletzender und ein nichtverletzender Fall konstruiert, und für jeden dieser Fälle werden die Parameter variiert, das die Prüfung nicht verletzt bzw. verletzt wird.
Die eingespeisten Testdaten können geändert oder ergänzt werden, wenn die JUnit- Tests mit ihren Quellen in der Eclipse-Entwicklungsumgebung eingerichtet und ausgeführt werden.
Bei den JUnit-Tests werden durch den Aufruf der 4 Methoden die wesentlichen Kernfunktionen abgedeckt, ausdrücklich nicht berücksichtigt werden:

- Die Periodizität der Prüfungen
- Die Generierung der Betriebsmeldungen

**Ausführen der JUnit-Tests:**
Voraussetzungen

1. Das mitgelieferte Test-Kernsystem ist lokal auf dem Rechner zu installieren, auf dem die Tests stattfinden sollen
2. Die DaV-Verbindung der Testsoftware erfolgt mit den Standardeinstellungen 127.0.0.1:8083, die DaV-seitig nicht geändert werden dürfen
3. Für die Authentifizierung erwartet die Testsoftware eine Datei passwd in ihrem Arbeitsverzeichnis (als gültige Authentifizierung-Datei) . Hierfür eignet sich die Datei passwd, die Bestandteil des skript...-Verzeichnisses des Kernsystems ist. Erfolgt der Aufruf der Testsoftware nicht aus dem Skriptverzeichnis, so muss die passwd-Datei entsprechend kopiert werden, dies gilt insbesondere, wenn die Tests aus der Eclipse-Entwicklungsumgebung gestartet werden: passwd wird in diesem Falle ins Eclipse-Projektverzeichnis kopiert.

4. Im entsprechenden Konfigurationsobjekt vom Typ Parametrierung (AOE) müssen in der Attributgruppe Parametrierung folgende Attributgruppen als parametrierend angegeben sein: EngstellenVerwaltungParameter, SituationsEigenschaften, BaustellenEigenschaften, BaustellenEigenschaftenErweitert, UnfallEigenschaften.

Die Tests können innerhalb der Eclipse-Entwicklungsumgebung ablaufen, dazu sind die gelieferten Test-Quellen als Eclipse-Projekt einzurichten.
Um die Tests außerhalb der Eclipse-Umgebung ablaufen zu lassen, ist die Startklasse org.junit.runner.JUnitCore mit der Testklasse (Testsuite) de.bsvrz.vew.engvew.AllTests anzugeben.
Neben den Standard jre-Java-Bibliothekenund den Standard-Distributionspaketen müssen sich in jedem Fall im classpath befinden:

- de.bsvrz.vew.engvew-test.jar
- de.bsvrz.vew.engvew.jar
- de.bsvrz.sys.funclib.dambach.jar
- junit-4.4.jar

Beispiel für Aufruf-Skript (Windows) für die JUnit-Tests

```
@echo off
if x%JAVA_HOME%x == xx ( set java=java) else set java=%JAVA_HOME%\bin\java

title Engstellenverwaltung JUnitTests

%java% ^
-classpath ^
..\..\de.bsvrz.vew.engvew\de.bsvrz.vew.engvew-runtime.jar;^
..\..\de.bsvrz.vew.engvew\test\de.bsvrz.vew.engvew-test.jar;^
..\..\de.bsvrz.vew.engvew\test\junit-4.4.jar;^
-Xmx300m ^
org.junit.runner.JUnitCore ^
de.bsvrz.vew.engvew.AllTests

echo errorlevel %errorlevel%
rem Fenster nicht sofort wieder schließen, damit eventuelle Fehler noch lesbar sind.
pause
```

Beispiel für Aufruf-Skript (Linux) für die JUnit-Tests

```
#!/bin/bash
if test "${JAVA_HOME}" == "" ;then java=java; else java=${JAVA_HOME}/bin/java; fi

$java% -cp ../../de.bsvrz.vew.engvew/de.bsvrz.vew.engvew-
runtime.jar:../../de.bsvrz.vew.engvew/test/de.bsvrz.vew.engvew-test.jar:../../de.bsvrz.vew.engvew/test/junit-4.4.jar
\
-Xmx300m \
org.junit.runner.JUnitCore \
de.bsvrz.vew.engvew.AllTests

# Auf das Ende von allen im Hintergrund gestarteten Prozessen warten
wait
```