Vragen (aan Edwin):

1. Wat is het nut van de Linux GNU build?   
   => dat is een iets strengere compiler dan Intel; daarmee kunnen nog kleine foutjes aan het licht komen. Is vnl voor intern gebruik. Voor uitleveringen is de Intel versie wellicht meer consistent met wat we voor Windows doen (al maakt het in principe niet uit).
2. Waar komt het verschil in aantal tests vandaan?   
   => basis is 32-bit
   1. 64 bit: 5 minder; dat bleken tests waar wat problemen mee waren
   2. Linux Intel: geen tests waarin de dll wordt aangeroepen.
   3. Linux GNU: -
3. Waarom worden de trendtests niet uitgevoerd in de Linux-Intel build?   
   => Worden wel uitgevoerd maar in- en uitvoerbestanden worden kennelijk niet opgenomen in de Artifacts. Zou eenvoudig aan te passen moeten zijn.
4. Hoeven de Linux bestanden niet gesigned te worden?   
   => Nee, dat signen is bij Linux niet aan de orde.
5. Welke bestanden moeten uiteindelijk opgeleverd worden? (zie lijst excel voor vermoeden)
   1. Kan dat vergemakkelijkt worden in TC?   
      => Kan wel, maar is met deze toelichting eigenlijk niet echt meer nodig.
6. Wanneer kan versienr verhoogd worden? Gaat Joël doen, wellicht heeft hij al eerder tijd. Edwin zeker niet.

# Overzicht stappen

1. Aanpassingen afronden
   1. Check SIG score
   2. Check tests
      1. Voorlopige bestanden voor ‘test’ verzamelen
      2. Testrapport (voorlopig) updaten
   3. Indien a en b OK:
      1. Versienummer in code aanpassen
      2. TeamCity: build(s) pinnen
      3. SVN: tag aanmaken
2. Bestanden voor ‘bin’ verzamelen
3. Bestanden voor ‘test’ verzamelen
4. Bestanden voor ‘doc’ verzamelen
   1. Testrapport - (definitief) updaten
   2. Protocol van Overdracht - opstellen
   3. Release notes - opstellen
   4. Functioneel Ontwerp - updaten danwel oude versie overnemen
   5. Technisch Ontwerp - updaten danwel oude versie overnemen
   6. License file - overnemen
5. Opleveren en administratie bijwerken
   1. Opleveren aan RWS, via SURFfilesender
   2. Kopie van oplevering op p-schijf zetten
   3. Jira bijwerken
   4. Documentatie pdf's aan de tag toevoegen
6. PO’s van gebruikende softwareproducten informeren.

# Bestanden verzamelen, pinnen, taggen

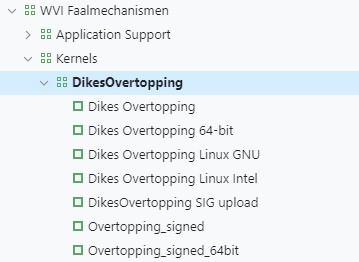
## Doelstructuur voor uitleveren

A screenshot of a computer program

Description automatically generated with medium confidence

## Spullen op TeamCity

Overzicht inbedding en onderliggende builds in TeamCity:



Wat is precies de functie van deze builds?

Inschatting:



Uit te leveren softwareproducten:



# Testrapport updaten

## Testresultaten verzamelen en nader analyseren

### Unittests

Check, bij de verschillende OS-versies:

* Alles groen?
* Alle testen gedraaid?

### Regressietests / Inventarisatie veranderingen

Vergelijkingen checken:

* Vergelijking Windows32-bit\_nieuw met Windows32-bit\_oud
* Onderlinge vergelijking nieuwe OS-versies, vooralsnog alleen   
  Vergelijking Windows32-bit\_nieuw met Windows64-bit\_nieuw

1. Matlab script draaien: "Compare\_overtoppingmodule\_results\_all.m"
   1. Gedeelte ‘input’ aanpassen
   2. Resultaat inspecteren

### Trendtests

1. Matlab script draaien (pdf's genereren): "Analyse\_overtoppingmodule\_results\_all.m"
   1. mappen voor in- en uitvoer aanpassen
2. pdf bestanden bundelen, m.b.v. "pdf Exchange Tools"
3. Resultaat bekijken
   1. Trends visueel checken
   2. Known issues checken

## Document aanpassen

1. Versienummers updaten
2. Uitvoer opnemen
   1. Kies één van de OS-versies; vooralsnog ‘Windows 32-bit’
   2. Output unittests (HTML) van gekozen OS-versie in document opnemen
   3. Output trendtests (pdf grafieken) van gekozen OS-versie in document opnemen
3. Tekst - waar nodig - aanpassen
4. Reviewen
5. Paraferen

===/\===

|  |
| --- |
|  |