leren. durven. doen.

SYNTRA WEST

SOFTWARE TESTEN

Grondbeginselen / Test Methodes
4 werkuur

west-vlaanderen POM

Hoofdtitel: Software Testen
Subtitel: Grondbeginselen / Basis principes
Auteur / docent: Vincent Van De Walle
Versie: Gebruikt tijdens opleiding

leren. durven. doen.

SYNTRA WEST

VINCENT VAN DE WALLE

Vincent@CopyPaste.be
VincentVanDeWalle@digi4u.be

+32 (0) 475 / 97.63.80

west-vlaanderen POM

Wie bereidt de testen voor?

- **ACCEPTATIE TESTEN**
  - Business / Analisten
- **SYSTEEM TESTEN**
  - Test team / Analisten
- **INTEGRATIE TESTEN**
  - Test team / Analisten
- **COMPONENT / UNIT TESTEN**
  - Analisten / Developers

Wie voert de testen uit?

- **ACCEPTATIE TESTEN**
  - Business
- **SYSTEEM TESTEN**
  - Test team / Analisten / Systeem
- **INTEGRATIE TESTEN**
  - Test team / Analisten / Systeem
- **COMPONENT / UNIT TESTEN**
  - Developers / Systeem

Ad hoc testing

- **WAT IS HET?**
  - Click here, click there and click every where
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Overal
- **TECHNIEKEN**
  - Je doet maar wat
- **VOORDELEN**
  - Je doet maar wat
- **NADELEN**
  - Je doet maar wat

Verkennend Testen (UK Exploratory testing)

- **VERKENNEND TESTEN IS**

  - Een methode van software testing
  - dat persoonlijke vrijheid benadrukt
  - en legt de verantwoordelijkheid bij de tester die het doet

Wat?

  - Met als doel de waarde van het product te optimaliseren

Waarom?

  - Door het test ontwerp,
  - de test uitvoering,
  - de interpretatie van het testen uitvoeren
  - en door constant bij te leren
  - hoe het product werkt

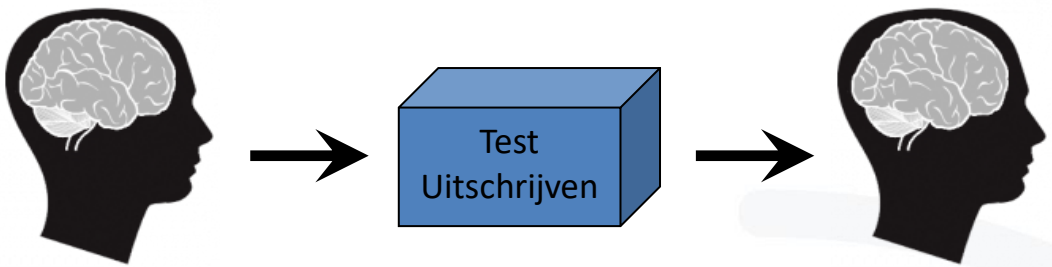
Hoe?

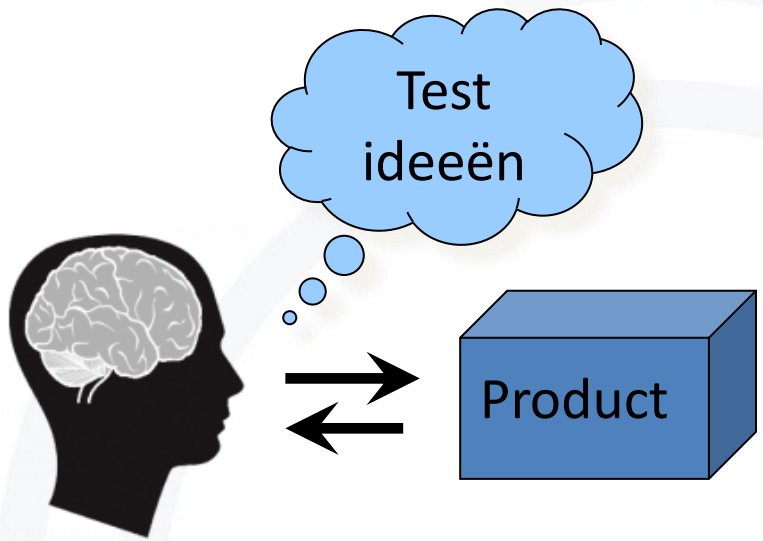
Tegenstrijdige aanpakken

- In testen uitschrijven, de testen zijn eerst uitgedacht, en dan ‘opgenomen’. Dan worden ze misschien later uitgevoerd, misschien door een ander persoon.

- In verkennend testen, is het uitdenken en het uitvoeren tegelijkertijd. Er is een grote kans dat het niet wordt ‘opgenomen’.

- In Ad hoc testen, doe je maar wat.



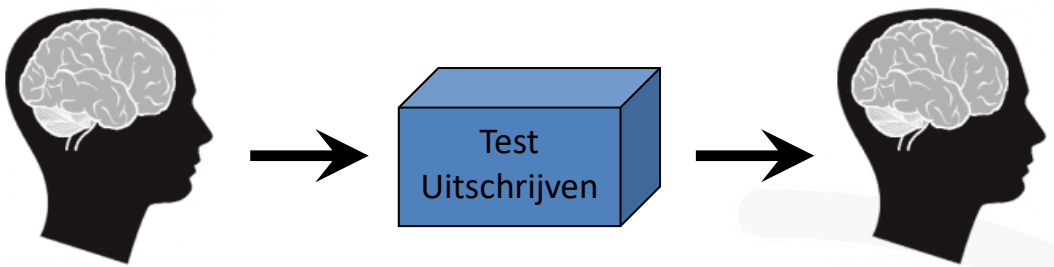


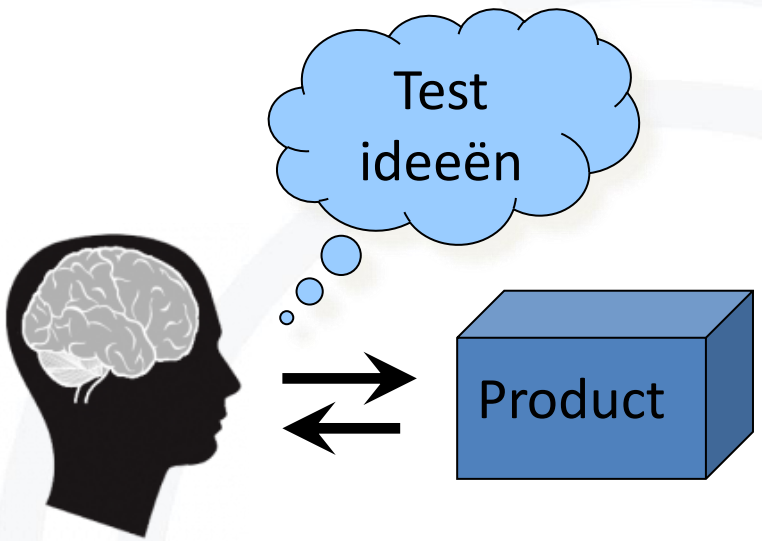
Tegenstrijdige aanpakken

- In testen uitschrijven, krijg je het effect van een voorbereide speech, of het lezen van muzieknoten. Het is een soort begeleiden van vooraf uitgedachte scenario's.

- In verkennend testen is het een conversatie hebben, of ge speelt jazz. Het is een gestructureerd begeleiden.

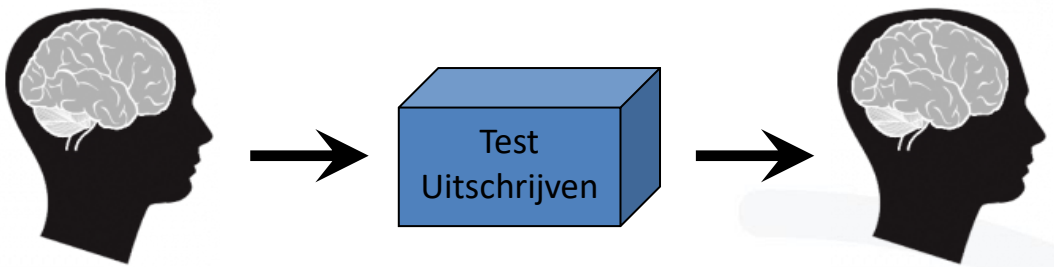
- In Ad hoc testen, doe je maar wat.

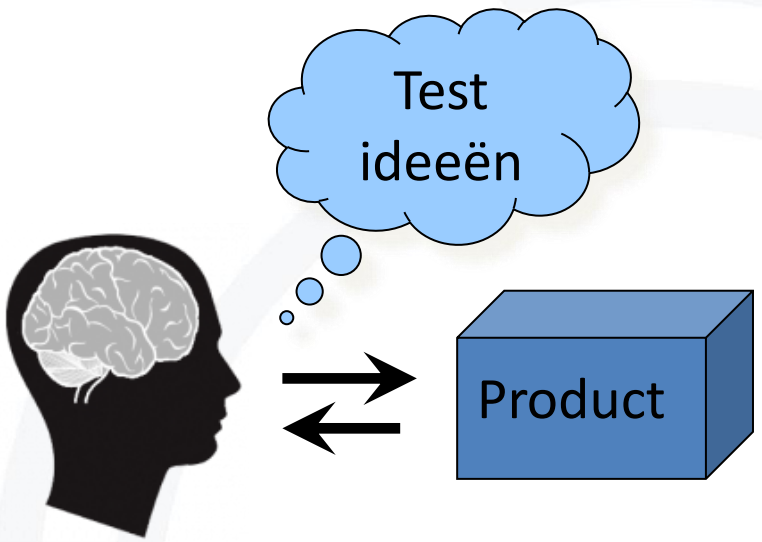




Tegenstrijdige aanpakken

- In testen uitschrijven, benadruk je beslissingen en wie verantwoordelijk is.
- In verkennend testen benadruk je het product leren en aanpasbaarheid.
- In Ad hoc testen, doe je maar wat.





Samenvatting

- **UITGESCHREVEN TESTEN**
  - Beslissing
  - Bevestiging
  - Verantwoordelijkheid
  - Herhaalbaarheid
  - Moeilijker aanpasbaar
- **VERKENNEND TESTEN**
  - Product verkennen
  - Product ontdekken
  - Product aanleren en ‘expertise’ vinden
  - Gemakkelijker aanpasbaar
- **ADD HOC TESTEN**
  - Je doet maar wat

Checken versus Testen

- **UITGESCHREVEN TESTEN**
  - Is checken
  - Je volgt een procedure / algoritme (doe dit, doe dat)
- **VERKENNEND TESTEN**
  - Is echt testen
  - Je gaat een product uitproberen om de werking ervan te kennen
    - Je stelt vast
      - Interactie met het product door dingen te doen
    - Je evalueert
      - Je kijkt of het product correct reageert op de interactie
        - (Algoritmisch, je weet wat te verwachten)
    - Je rapporteert
      - De resultaten worden gecommuniceerd

De vraag van 1 miljoen

leren. durven. doen.

SYNTRA WEST

De vraag van 1 miljoen

leren. durven. doen.

SYNTRA WEST

Het hangt ervan af

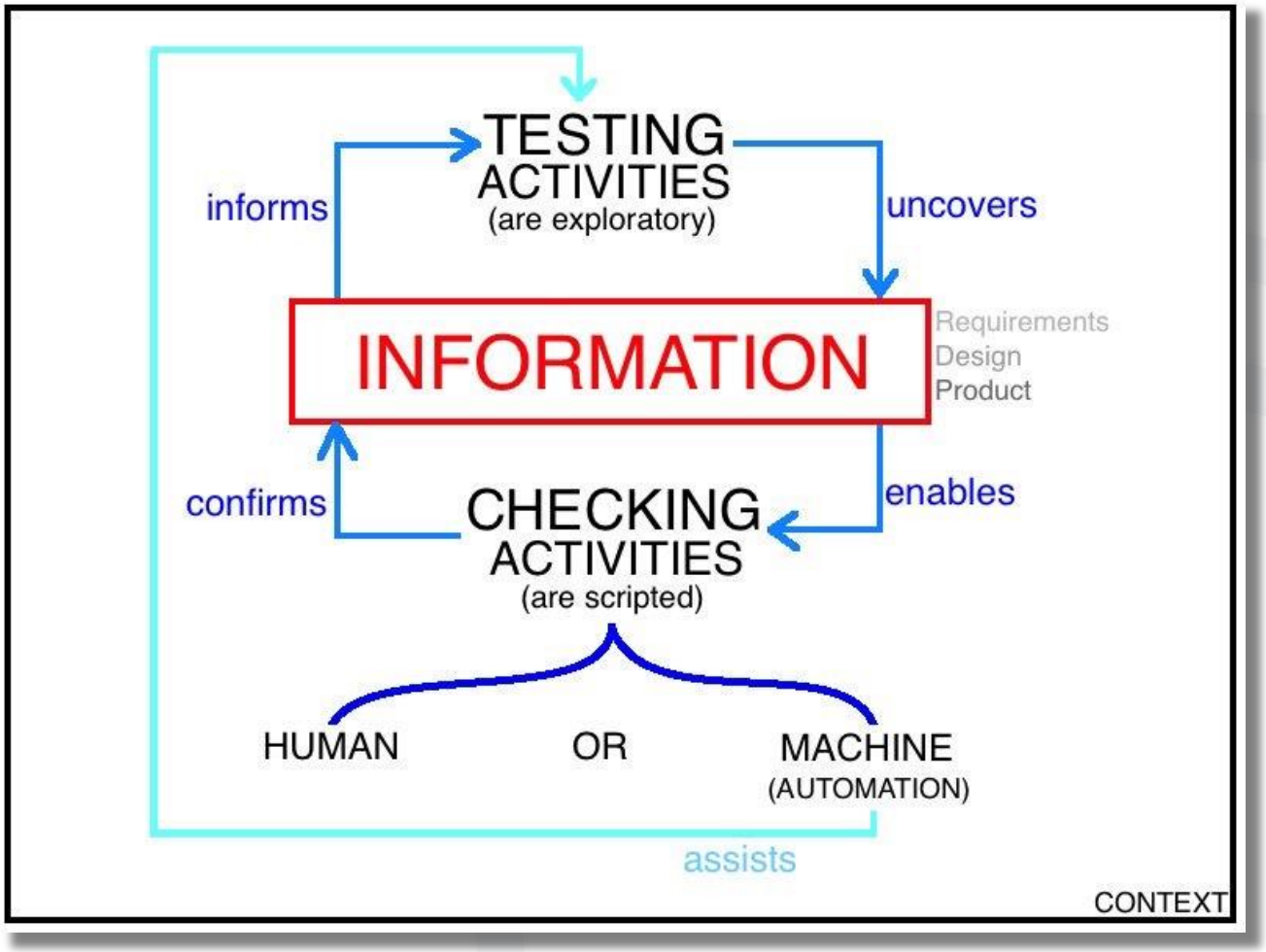
- **VERSCHILLENDE TEST METHODES DIENEN VERSCHILLENDE INFORMATIE OBJECTIEVEN**
- **WAT WIL JE WETEN / DOEN?**
  - Vinden van defecten / issues?
  - Het aantal gevonden fouten maximaliseren?
  - Afblokken van vroegtijdige (niet rijpe) releases?
  - Managers helpen van beslissingen te nemen?
  - Minimaliseren van de technische kost?
  - Voldoet het aan de specificaties?
  - Voldoet het aan de wetgeving?
  - Beperken van de kans op rechtzaken?
  - Vinden of een product correct werkt?

Uitgeschreven versus Verkennend testen

- **BEIDE HEBBEN HUN VOORDELEN EN NADELEN**
- **WELKE JE KIEST HANGT AF VAN WAT JE WILT BEREIKEN**
- **IDEAAL IS HET EEN MIX**
  - Verkennend testen
    - Vinden van risico's en problemen in een vroeg stadium
    - Tonen waar een software product werkt en waar niet
  - Uitgeschreven testen
    - Bevestigen van het gedrag van functionaliteiten (requirements)
    - Een soort checklijst
    - Een soort inventaris van uitvoerbare testen
  - Ad hoc testen
    - Moet leiden naar verkennend testen of uitgeschreven testen

Uitgeschreven versus Verkennend testen

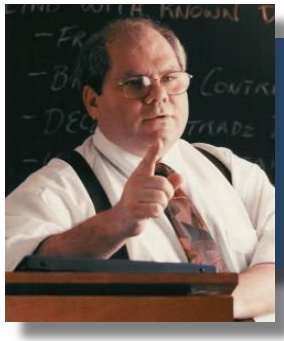
- **BRON: DAN ASHBY**



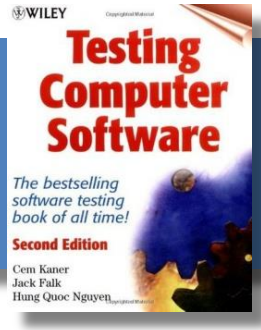
Het ‘probleem’ van verkennend testen

- **VERKENNEND TESTEN**
  - De ‘natuur’ van verkennend testen is het feit dat je als tester je gemakkelijk kan aanpassen. Je ziet als doel dat je de software wilt ‘leren kennen’, maakt dat deze methode heel veel gebruikt wordt, vooral als de tijd kort is.
- **MAAR**
  - Het wordt vaak afgestoten en als een foute methode beschouwd, door deze die ervan uitgaan dat verkennend testen niet reproduceerbaar, niet meetbaar is.

Stap voor stap naar Verkennend Testen



1988



"Brain-engaged testing"

Cem Kaner

1995

"Exploratory testing is simultaneous learning, test design and test execution"

James Bach

2001

"Scientific thinking in real time"

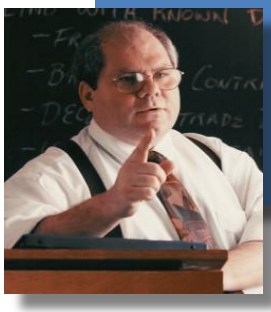
James Bach

Stap voor stap naar Verkennend Testen

”To the extent that the next test we do is influenced by the result of the last test we did, we are doing exploratory testing.“

2001

James Bach

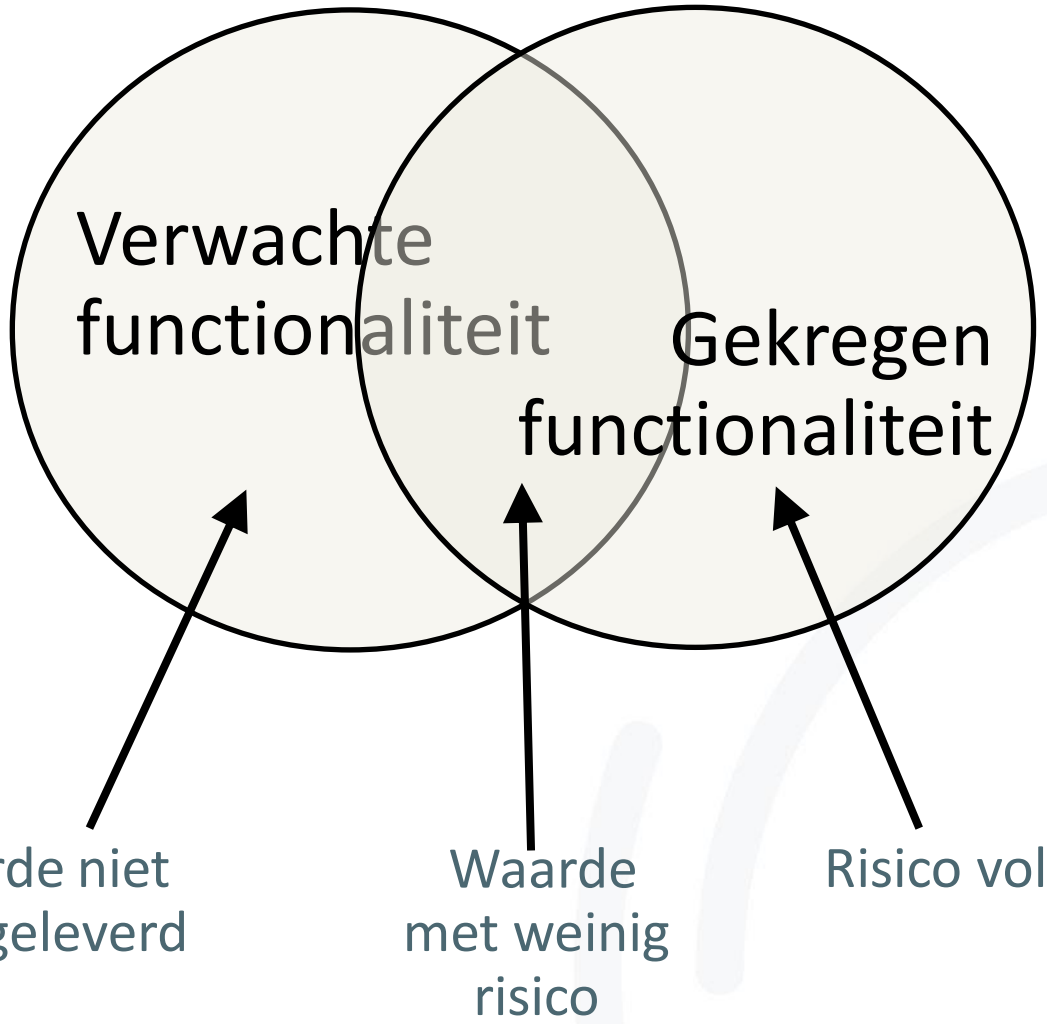


”Simultaneous test design, test execution and learning, with an emphasis on learning”

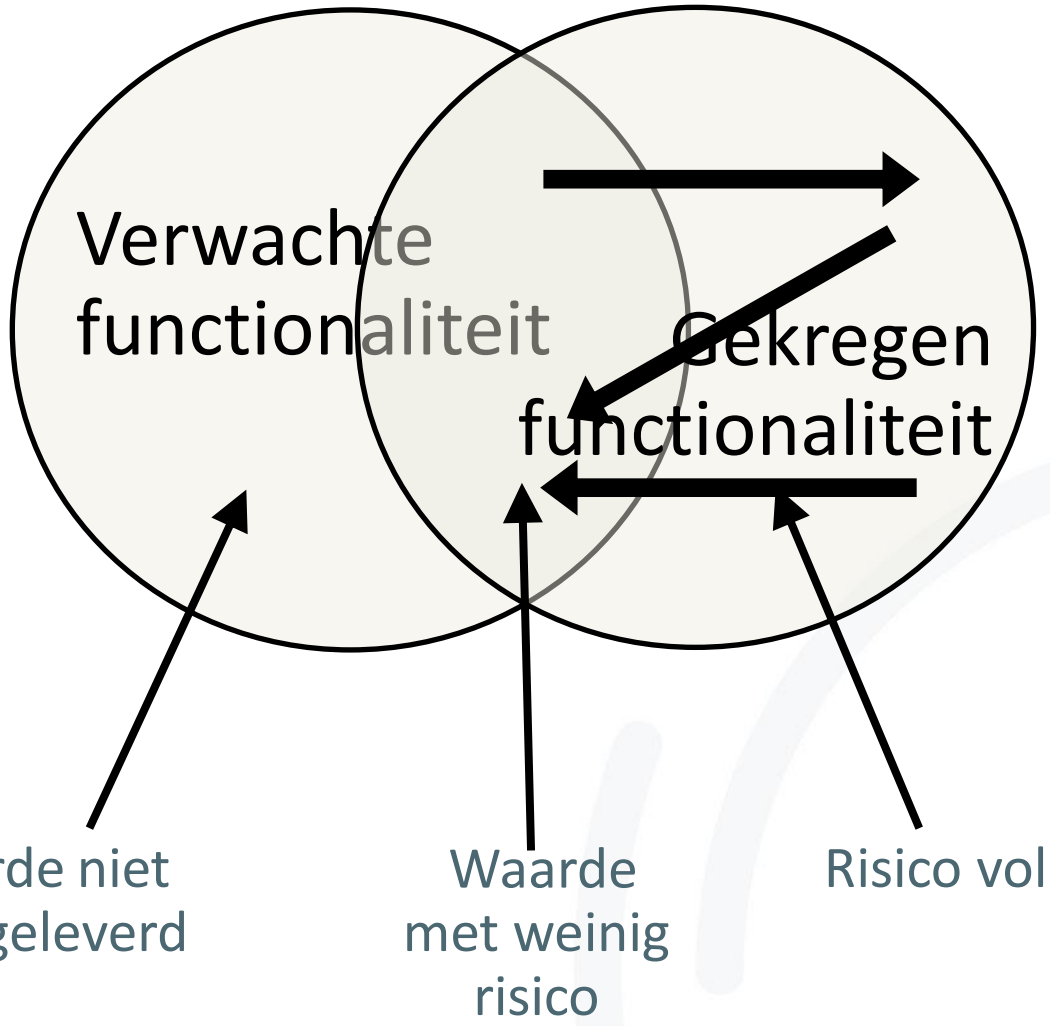
2005

Cem Kaner

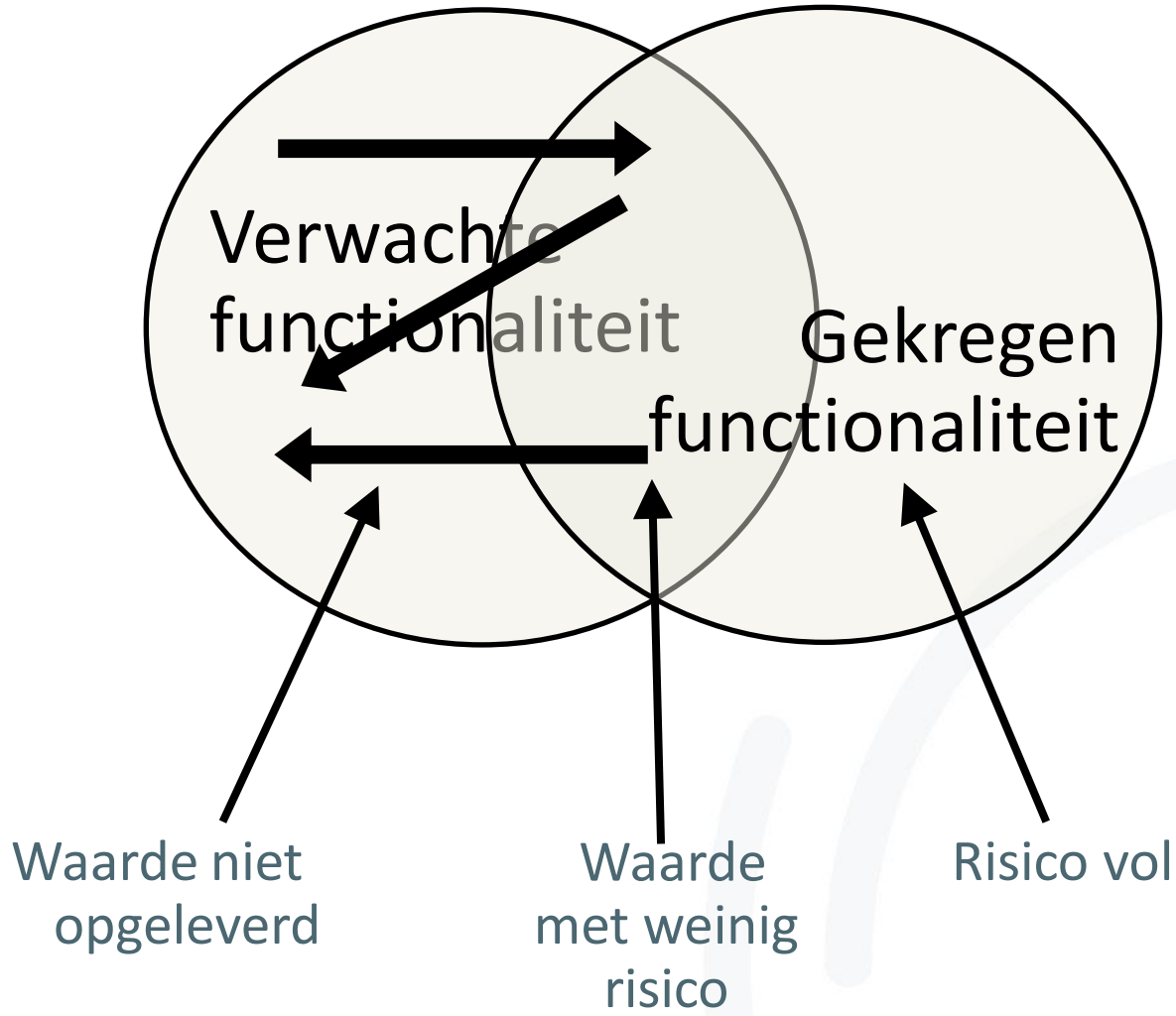
Testen in een notedop



Verkennend testen in een notedop



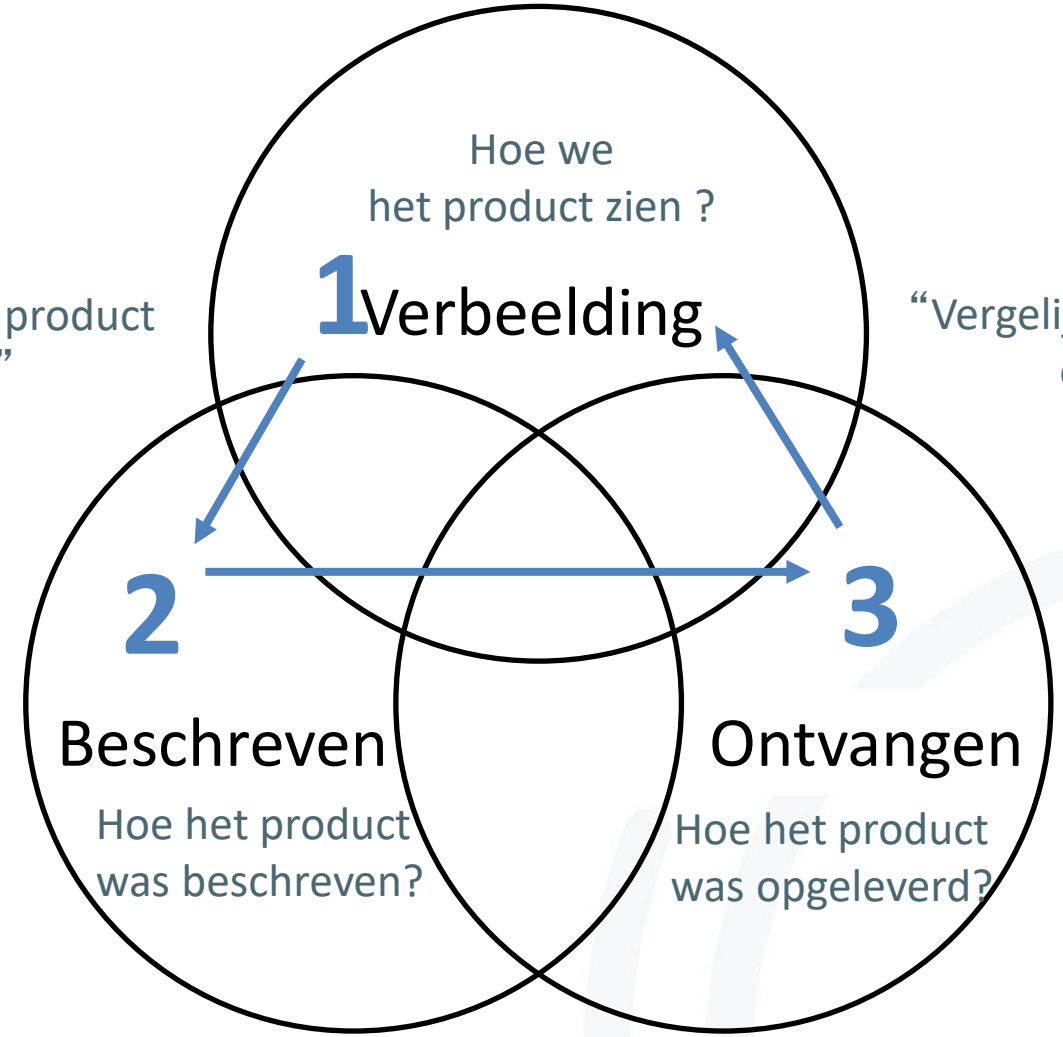
Uitgeschreven testen in een notedop



De waarheid over modellen

Alle modellen zijn fout,
maar sommige zijn bruikbaar
(George Box, 1976)

Een meer realistisch model



"Vergelijk het idee van het product met hoe het beschreven is"

"Vergelijk het product met het originele idee"

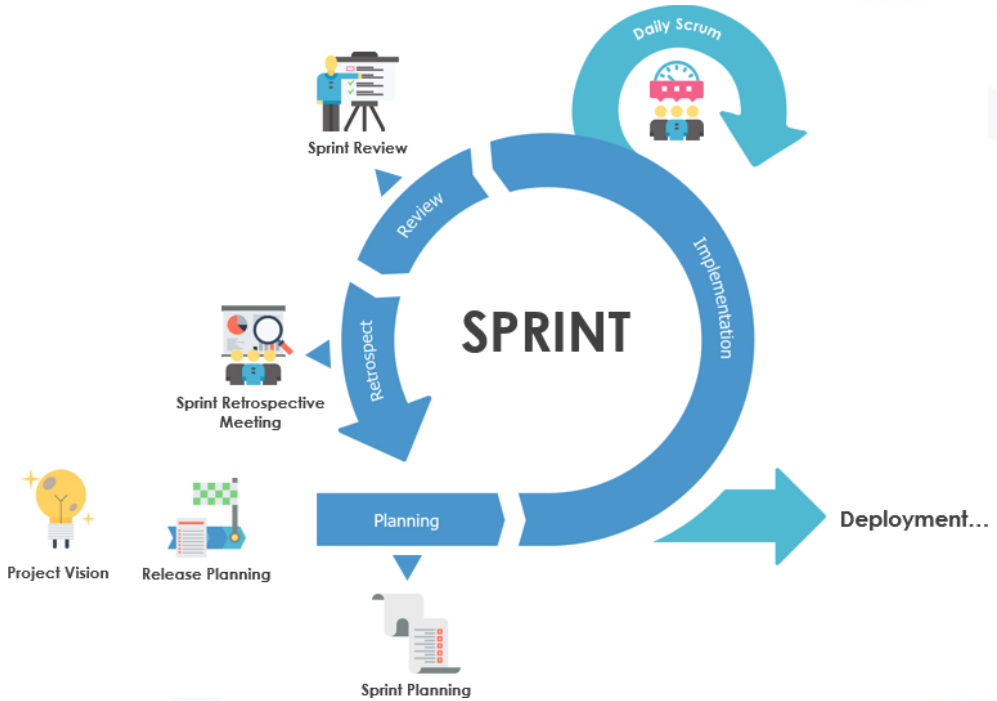
"Vergelijk het product met hoe het is beschreven"

Verkennend testen in een Agile omgeving

- **"AGILE" GAAT OVER HET KRIJGEN VAN VRIJHEID**
  - In het maken
  - In het leren
  - In het aanpassen

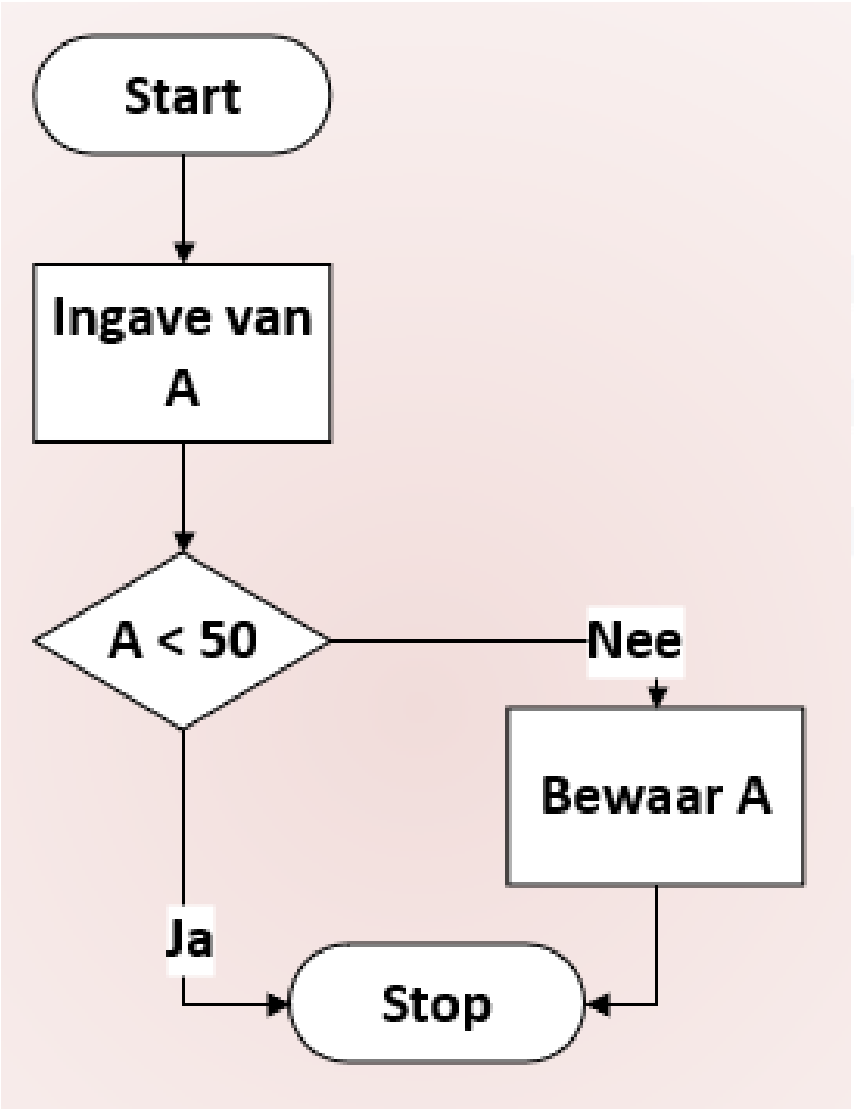
- **OP BASIS VAN SNEL VERKREGEN FEEDBACK**

- **IN TEGENSTELLING TOT HET VOLGEN VAN EEN PLAN**



Oefening – hoeveel test scenario's?

- HOEVEEL TEST SCENARIO'S ZOU JE MAKEN, OM ONDERSTAANDE FLOWCHART TE TESTEN?



Project information

- **EEN PAAR VOORBEELDEN**
  - Deze code is onderdeel van een experimenteel systeem om een ruimteschip te beschermen tegen ruimteafval in de ruimte die het ruimteschip kunnen beschadigen
    Het systeem evalueert de ‘bedreiging’ van het ruimteafval op de route van het ruimteschip.
    Het systeem stelt te ondernemen acties voor.
    50 processors doen dit tegelijkertijd met informatie die komt van een radar netwerk en aan boord van het ruimteschip wordt dit constant ge-evalueerd.
    Wanneer een “dreiging” effectief is, dan wordt de route van het ruimteschip aangepast.
  - Wanneer je jonger dan 50 bent, dan kun je geen lid worden van de OKRA club.

Input van A

- **EEN PAAR SCENARIO'S**
  - Connecteer naar een data server (al dan niet in de cloud)
  - Wacht op autenticatie.
  - Wacht op autorizatie.
  - Maak een nieuw record aan.
  - Handel alle mogelijke fouten af.
  - Voer op het getal groter dan 50 een berekening uit
    - Resultaat is een score van A tot F.
  - Zet de letter in Ascii code om.
  - Bewaar de Ascii code.
  - Alles binnen 25 milliseconden.

A<50?

```
IF ($THREATANALYSISSCORE >= 50)
{
    STORETHREAT($THREATHANDLE);
}
```

Bewaar A

- **EEN PAAR SCENARIO'S**
  - Leg connectie naar de databank.
  - Wacht op autorisatie.
  - Stuur de waarde A als record naar de databank.
  - Ontvang een verificatie sleutel.
  - Check of het correct opgeslagen is.

Stop

- **EEN PAAR SCENARIO'S**
  - Vernietig alle objecten.
  - Geef geheugen vrij.
  - Geef een "succes" code terug.

Static testing

- **WAT IS HET?**
  - Je test een programma zonder de code uit te voeren.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Zo dicht mogelijk bij het tikken van de code
- **TECHNIEKEN**
  - Pair programming (Extreme Programming)
  - Pair testing (Extreme Testing)
  - Code check / review

Static testing

- **VOORDELEN**
  - Problemen worden in een héél vroeg stadium ontdekt.
  - Je vindt ontwerp fouten (design).
  - Programmeurs leren elkaars code en stijl kennen.
  - Analisten leren elkaars analyses en stijl kennen.
  - Documentatie / Requirements worden gelezen.
  - Ontbrekende requirements worden gevonden.
  - Afwijken van de standaard is lastiger.
  - Onderhoudbare code.
  - Interface specificaties die niet consistent zijn worden gevonden.
    - Vooral de dataflow wordt gecontroleerd.
- **NADELEN**
  - Tijdrovend (Je werkt met meerdere op hetzelfde ding)
  - Je test niet de applicatie.

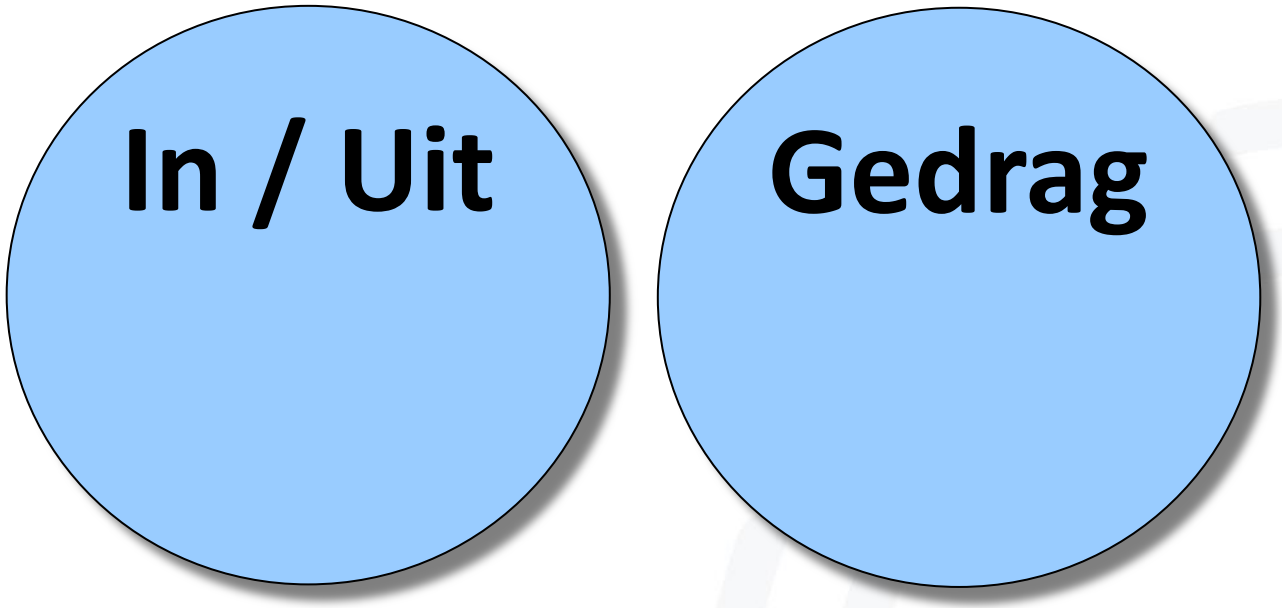
Dynamic testing

- **WAT IS HET?**
  - Je test een programma door de code uit te voeren.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Zo dicht mogelijk na het implementeren van de code
- **TECHNIEKEN**
  - Verkennend testen.
  - Uitgeschreven testen.

Dynamic testing

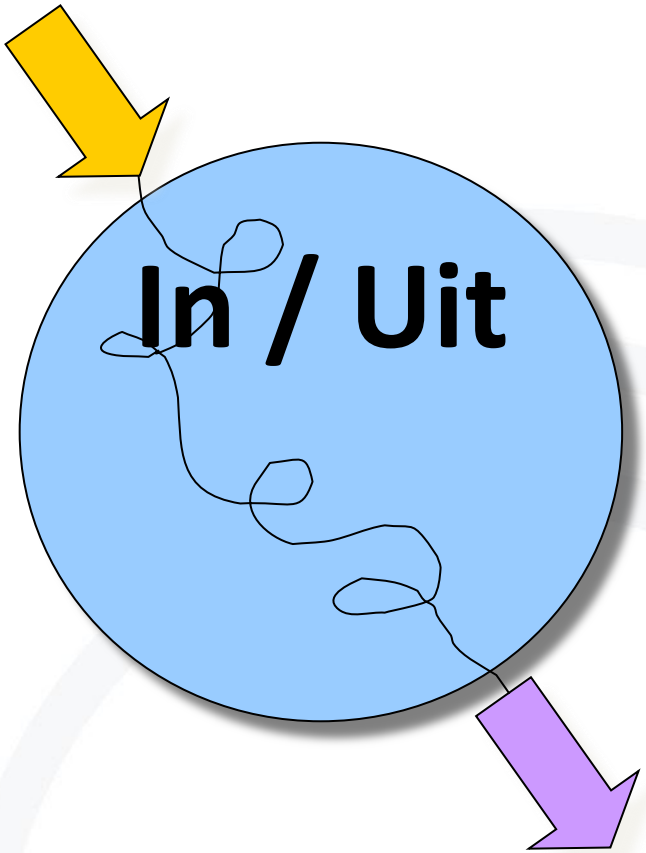
- **VOORDELEN**
  - Je voert de applicatie uit.
  - Je moet abstractie maken van hoe het technisch in elkaar zit, je test de functionele en niet functionele requirements.
- **NADELEN**
  - Je kunt niet alle testen uitvoeren.
  - Je weet niet altijd wat er technisch op de achtergrond gebeurd.
  - Er zijn honderden verschillende type testen uit te voeren.

Twee frameworks voor verkennend testen

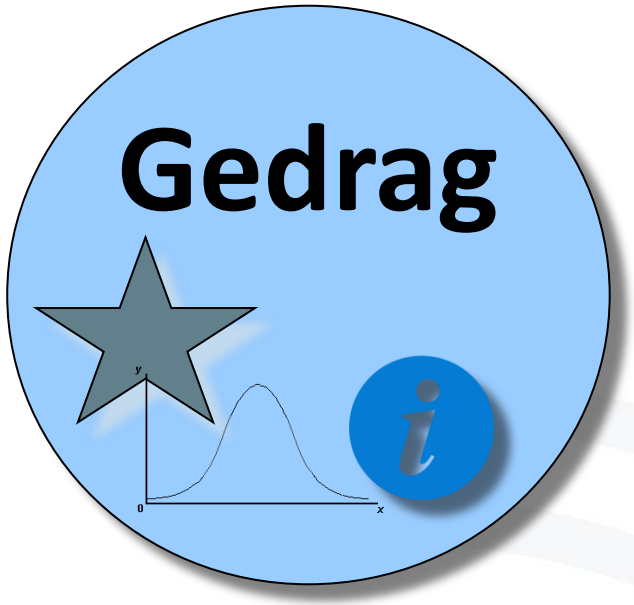


Framework 1 – In / Out

- **IN / UIT**
  - Input
  - Resultaat
  - Koppel input aan resultaat

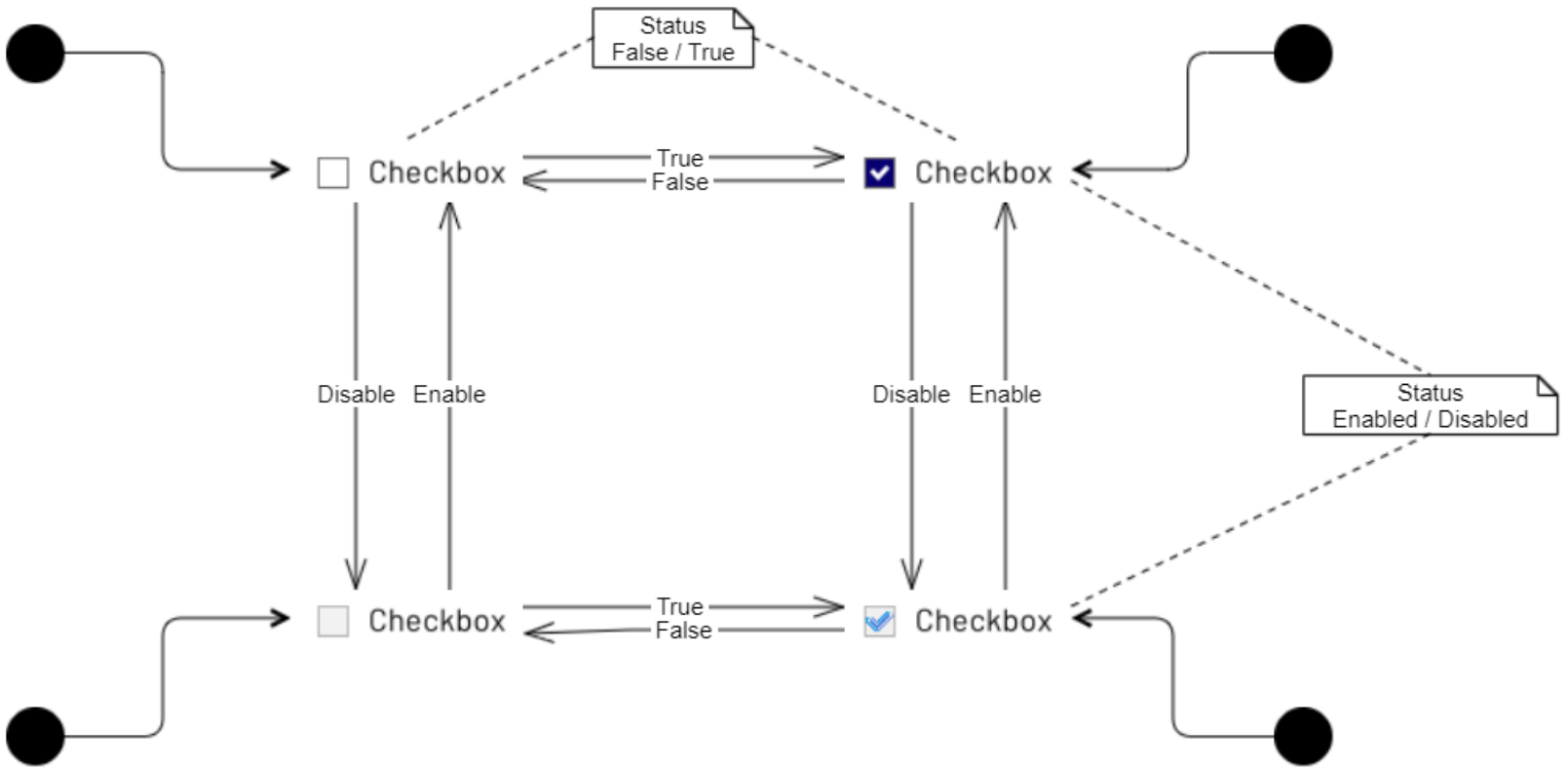


Framework 2 – Gedrag



- **GEDRAG**
  - Actie / Method
  - Gebeurtenis / Event
  - Gedrag
  - Informatie
- **EXTRA INFO**
  - Gebruikers zijn niet altijd de rechtstreekse oorzaak van een observeerbaar gedrag.
  - Verschillend gedrag / antwoord wil zeggen een verschillende “status” (Eigenschap / Property)
    - E.g. State diagrams / state machine

Een voorbeeld van een state diagram



leren. durven. doen.

White Box testing

- **WAT IS HET?**
  - De structuur / ontwerp en uitvoering van het te testen onderwerp is gekend.
    - Met andere woorden, de binnenkant en design kan worden bekeken.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Component testen.
  - Integratie testen.
  - Systeem testen.
  - Unit testen. (Voer voor discussie)
- **TECHNIEKEN**
  - Een beetje code kunnen lezen is handig.
  - Je kijkt dus onder de motorkap van het voertuig.
  - Je kunt echt “bitziften”.

White Box testing

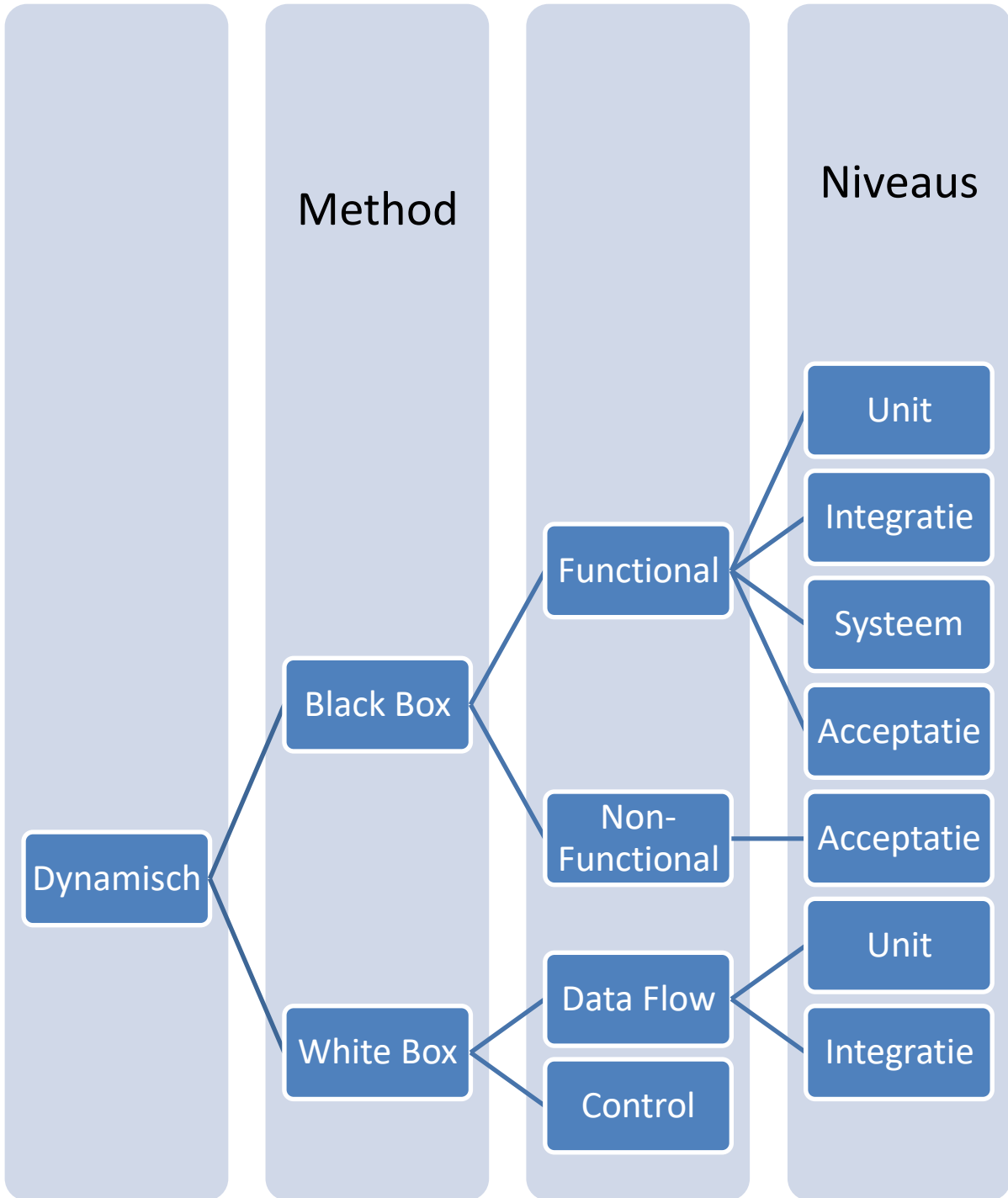
- **VOORDELEN**
  - Ge kunt testen in een vroeg stadium. Niet echt een User Interface nodig om het testen te starten.
  - Doordat je de code en de interne werking kunt zien, kun je ook de mogelijke paden in de code testen.
- **NADELEN**
  - Testen kunnen héél snel, héél complex worden, en dus is een “skill set” nodig.
    - Databanken, Interfaces, Code begrijpend lezen, …
  - Onderhoud van uitgeschreven testen kan veel werk zijn, als de code constant gewijzigd wordt (refactoring).
  - Test methode ligt zeer dicht bij coderen, niet iedere testtool kan dit aan.
  - Er is al iets werkends nodig om testen te kunnen starten.

Black Box testing

- **WAT IS HET?**
  - De structuur / ontwerp en uitvoering van het te testen onderwerp is niet gekend.
    - Met andere woorden, de binnenkant en design kan niet worden bekeken.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Integratie testen.
  - Systeem testen.
  - Acceptatie testen.
- **TECHNIEKEN**
  - Gedrag en performantie fouten.
  - Initialisatie en afsluit fouten.
  - Data structuren, interfaces en ontbrekende functionaliteit.

Black Box testing

- **VOORDELEN**
  - Testen worden uitgevoerd vanuit een gebruikers standpunt.
  - Testers hoeven niet de design te kennen van hoe iets gemaakt is.
  - Testen kunnen uitgevoerd worden door derde partijen.
  - Je ontwijkt het “ontwikkelaars”-vooroordeel.
  - Test scenario’s kunnen geschreven worden, zodra de specificaties / requirements gekend zijn.
- **NADELEN**
  - Er is een grote kans dat niet alle code zal getest zijn. Bijvoorbeeld een randscenario dat ontwikkeld is, maar niet gekend door de testers.
  - Zonder duidelijke specificaties, wordt dit snel Try and Error.
  - Testen kunnen soms overbodig zijn. Iets testen waarvan er een unit test is, is zinloos.


Method
Niveaus
Dynamisch
Black Box
White Box
Functional
Non-Functional
Data Flow
Control
Unit
Integratie
Systeem
Acceptatie
Acceptatie
Unit
Integratie

Gray Box testing

- **WAT IS HET?**
  - Een mengeling van Black Box en White Box testing.
  - Je hoeft niet van alles het detail te weten.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Integratie testen.
  - Systeem testen.

Agile testing

- **WAT IS HET?**
  - De cyclus van het testen wordt verkleind naar een kortere periode.
- **WAAR WORDT HET GEBRUIKT IN HET MODEL?**
  - Dit wordt overal gebruikt in het model.
- **TECHNIEKEN**
  - Test-Driven Development.
    - De input en output zijn gekend op voorhand.
    - Het te verwachten resultaat is gekend op voorhand.
  - Behaviour-Driven Development / Testing.
    - Het gedrag van de applicatie is beschreven op voorhand.
    - De beschrijving van het gedrag laat automatisering van code / testen toe.

Agile testing

- **VOORDELEN**
  - Veel sneller feedback.
  - Alles is timeboxed (afgesproken tijdstippen en tijdsduur).
  - Niet sequentieel.
    - Je test in blokken en werkt met milestones.
  - Testen gebeurt door het team, niet door testers.
- **NADELEN**
  - Je hebt een ander denkpatroon nodig om iets af te werken.
  - Testen gebeurt door het team, niet door testers.
  - Business Driven.
    - En die hebben niet altijd kaas gegeten van software ontwikkeling.
  - Verandering is normaal, maar altijd veranderen is dat niet.
  - Je bent de laatste in de rij, dus vaak kom je in tijdsnood.

Een voorbeeld uit de praktijk

- **EEN COMPONENT**
  - Rijksregister nummer met validatie.
- **EEN COMPONENT GEBRUIKEN**
  - Rijksregister nummer met validatie.
- **INGEVEN VAN EEN GEBOORTEDATUM**
  - Een datum ingave.
- **INGEVEN VAN HET GESLACHT**
  - Een ingave via een keuzelijst.