**Deelproduct 4c**

**Testontwerp**

**Beroepsproduct 4**

*Blok 4*

*Adam Oubelkas (2150912)*

Avans Academie Associate degree Informatica

**1. Inleiding**

Het eindproduct van een ICT-project moet zorgvuldig mogelijk worden getest. Dit kan pas als het eindproduct getoetst wordt op wensen en eisen van zowel de klant als de stakeholders. Dit gebeurt in de vorm van een toets matrijs waarin het eindproduct wordt beoordeeld op criteria over de tussen de klant en ontwikkelaar opgestelde Requirements [2].

Het eindproduct moet dan als het ware, via diverse testen, worden gevalideerd op haar functionaliteiten, functies en entiteiten om de klant/stakeholders inzicht te geven op de werkelijk status van het eindproduct. Met welke tools verschilt per programmeertaal, maar de algemene richtlijnen van elke soort test, dat in dit document wordt uitgelegd, is altijd van toepassing. Het is in ieder geval aangeraden om, tijdens een van in dit document genoemde testen, gebruik te maken van de bijbehorende sjabloon die te vinden is in hoofdstuk 4.

**2. Testsoorten**

“Realistisch gezien, is testen een afweging tussen budget, tijd en kwaliteit. Het wordt aangestuurd door winstgerichte modellen." - Yang, M.C.K.; Chao, A. (1995)

Testen van het eindproduct in deze context is dus niets meer dan fouten/ongewenst gedrag opsporen en verhelpen in de broncode of programma, waardoor de klant/stakeholders financiële/tijdsgebonden/kwaliteitsgerichte winst van kunnen boeken. Toch zijn er verschillende manieren waarop het eindproduct getest kan worden.

De soorten en volgorde van testen is als volgt:

1. Regressietest
2. Unittest
3. Integratietest
4. Systeemtest

*Regressietest*

Er is sprake van regressie als een aantal bugs opnieuw verschijnen in nieuwere versies van broncode. In een regressietest wordt nagegaan of er bepaalde bugs zijn blijven zitten van bestaande code wanneer iets aan de applicatie/project is aangepast.

De regressietest wordt pas uitgevoerd op broncode als deze getest en gevalideerd is in een unit-, integratie en/of systeemtest. De broncode wordt als het ware opnieuw getest om alle bugs op te sporen en te verhelpen, zodat de klant uiteindelijk met het eindproduct (applicatie) hiervan ongestoord en adequaat mee aan de slag kan.

“In agile softwareontwikkeling — waar levenscyclussen erg kort zijn, middelen beperkt zijn, en veranderingen in de software heel regelmatig plaatsvindt—regressie testen zou heel veel overbodige lasten kunnen veroorzaken.” - Yoo, S.; Harman, M. (2010)

*Unittest*

Unittest is gelijk aan een geautomatiseerde test dat de functionaliteit op de component, klasse, methode of property-niveau bevestigd en moet valideren.

Gedurende de ontwikkeling, mag een softwareontwikkelaar in de test criteria of resultaten reproduceren die vanzelfsprekend goed zijn om de juistheid van een softwaremodule (Unit) te toetsen. Tijdens de uitvoering van een test, is het sterk aangeraden om units te loggen die niet voldoen aan elke criteria waarin ze genoteerd worden in een samenvatting.

*Integratietest*

Componenten/Integratietest is het testen op problemen tijdens het samenvoegen van units.

De integratietest gebeurt achteraf de unittest maar vóóraf de systeemtest.

Deze volgorde is van groot belang omdat in een integratietest units eerst gevalideerd moeten zijn ná het doorvoeren hiervan in een unittest. Vervolgens worden deze units samengevoegd in grotere gehelen/verzamelingen die volgens een integratietestplan getest moeten worden. Zodra deze verzamelingen langs de integratietest zijn gegaan en gevalideerd zijn, dan pas zijn deze klaar voor gebruik in een systeemtest.

Integratietests omvat normaalgesproken heel veel programmeercode, met als gevolg dat er meer nakijkwerk is dan in unittests. Dit heeft invloed op de moeilijkheidsgraad van het opsporen van bugs wanneer een integratietest niet goed is uitgevoerd. Om dit probleem te verhelpen, moet het aantal te testen regels code verdeeld zijn in kleine brokstukken zodat bugs makkelijker opgespoord kan worden.

*Systeemtest*

Een systeemtest is het meten van reactievermogen, verwerkingssnelheid, betrouwbaarheid en of/ schaalbaarheid van een systeem binnen een bepaalde werklast. Hierin wordt het gehele systeem van de klant/stakeholders, bestaande uit vele verschillende hardware- en software modulen, getest op criteria die de klant/stakeholders hebben gesteld over het systeem. In deze criteria staan o.a. de gerelateerde Requirements en verwachtingen van de klant/stakeholders.

**3. Testgrenzen**

*Grenzen van regressietest*

De database, indien van toepassing, moet getest worden op veranderingen in waarde en restricties (nullable, unique of andere voorwaarde) per veld van ieder tabel tijdens de regressietest.

De applicatie moet getest worden op stabiliteit, flexibiliteit en functionaliteit per softwaremodule tijdens de regressietest.

Regressietest moet worden uitgevoerd binnen het kader van de onderstaande kwaliteitseigenschappen, waarin de te testen software hier niet van mag afwijkt:

* Veiligheid - In de geteste software zijn de juiste encapsulaties toegepast voor de modules, zodat er onderscheid is tussen functionele data en gebruikersgevoelige gegevens.
* Controleerbaarheid - Klasses, functies, attributen en variabel zijn duidelijk en herkenbaar toegepast in de programmeercode, met als gevolg dat het opsporen en verhelpen bugs snel en efficiënt verholpen kan worden.
* Zuinigheid - Gevonden bugs worden zodanig verholpen in de geteste software dat er achteraf geen bestaande/nieuwe bugs ontstaan.
* Prestatie - Het opsporen en verhelpen van bugs moet op een dusdanige manier worden uitgevoerd waardoor er zoveel mogelijk tijd en aandacht bespaard kan worden voor de tests die ná de regressietest uitgevoerd moeten worden.
* Functionaliteit - Veranderingen in broncode moeten niet ongedaan worden gemaakt als dit geen bugs (meer) veroorzaakt. Hierdoor kan de door de klant en/of ontwikkelaar gewenste toevoegingen of bijwerkingen blijven functioneren.

*Grenzen van unittest*

De database, indien van toepassing, moet getest worden op veranderingen in waarde en restricties (nullable, unique of andere voorwaarde) per veld van ieder tabel tijdens de unittest.

De applicatie moet getest worden op stabiliteit, flexibiliteit en functionaliteit per softwaremodule tijdens de unittest.

Unittest moet worden uitgevoerd binnen het kader van de onderstaande kwaliteitseigenschappen, waarin de te testen software hier niet van mag afwijkt:

* Bruikbaarheid - De geteste software moet grondig zijn gevalideerd op gebruik van softwaremodules binnen de software die (on)afhankelijkheid verantwoordt aan andere softwaremodules ongeacht de veranderingen in de betreffende broncode.
* Functionaliteit - De geteste softwaremodules moeten gevalideerd worden volgens door de klant en ontwikkelaar opgestelde Requirements.
* Onderhoudbaarheid - Geteste softwaremodules moeten kunnen blijven functioneren ongeacht de veranderingen in de broncode van de gehele software.
* Testbaarheid - De scoop en factoren van ieder geteste softwaremodule is stapsgewijs (in heldere details) te meten.
* Prestatie - Geteste softwaremodule moet voldoen aan de door de klant en ontwikkelaar opgestelde Requirements zodanig dat de module optimaal functioneert volgens de betreffende criteria.

*Grenzen van integratietest*

De database, indien van toepassing, moet getest worden op veranderingen in waarde en restricties (nullable, unique of andere voorwaarde) per veld van ieder tabel tijdens de integratietest.

De applicatie moet getest worden op stabiliteit, flexibiliteit en functionaliteit per softwaremodule tijdens de integratietest.

Integratietest moet worden uitgevoerd binnen het kader van de onderstaande kwaliteitseigenschappen, waarin de te testen software hier niet van mag afwijkt:

* Inpasbaarheid - (De programmeercode van) ieder softwaremodule moet een ander softwaremodule kunnen ondersteunen of respecteren zodat er geen conflicten ontstaan tussen en binnen de modulen met als gevolg dat de samenvoeging van modules ten goede komt.
* Herbruikbaarheid - De geteste software moet grondig zijn gevalideerd op hergebruik van softwaremodules binnen een verzameling van een andere softwaremodules ongeacht de veranderingen in de betreffende broncode.
* Onderhoudbaarheid - Geteste softwaremodules moeten kunnen blijven functioneren ongeacht de veranderingen in de broncode van de gehele software.
* Connectiviteit - Geteste softwaremodules kunnen zonder slag of stoot met elkaar communiceren via gebruikersinterfaces (Front-end) en/of applicatie-interfaces (Backend).
* Testbaarheid - De scoop en factoren van ieder geteste softwaremodule is stapsgewijs (in heldere details) te meten.

*Grenzen van systeemtest*

De database, indien van toepassing, moet getest worden op veranderingen in waarde en restricties (nullable, unique of andere voorwaarde) per veld van ieder tabel tijdens de systeemtest.

De applicatie moet getest worden op stabiliteit, flexibiliteit en functionaliteit per softwaremodule tijdens de systeemtest.

Systeemtest moet worden uitgevoerd binnen het kader van de onderstaande kwaliteitseigenschappen, waarin de te testen software hier niet van mag afwijkt:

* Beheerbaarheid - Gebruiker van de geteste software en systeem kan zelf deze software op normale wijze (zoals geschreven in de Requirements) aansturen in dit systeem.
* Beveiliging - Geteste software levert geen conflicten op voor andere software en hardware van het systeem, zodanig dat de software geen gevaar vormt voor de particuliere werkzaamheden binnen het systeem van de gebruiker.
* Portabiliteit - De geteste software kan op ieder systeem ongehinderd functioneren, waarvan besturingssysteem door de software ondersteund wordt, ongeacht de hardware- en softwaremodules.
* Continuïteit - De geteste software moet binnen een door de klant bepaalde termijn kunnen functioneren zonder hulp, toevoegingen en bijwerkingen vanuit de ontwikkelaar, ongeacht verandering in de hardware modules van het geteste systeem.
* Gebruikersvriendelijkheid - Software moet ongeacht de hardware-modules van het geteste systeem kunnen functioneren m.b.t. verschillende handelingen die de gebruiker kan uitvoeren. Als gevolg hiervan zal de gebruiker zich niet ergeren aan het gebruik van de software.

**4. Testsjablonen**

Zodra een regressie-, unit-, integratie- en/of systeemtest in de praktijk wordt uitgevoerd, beveel ik de onderstaande sjablonen aan. Dit omdat er zo een duidelijk overzicht gemaakt kan worden waar alle testgegevens op één plek staan en makkelijk terug te vinden is.

Echter moet de gebruikte sjabloon verder nog worden ingevuld, omdat het aantal rijen en gevulde waardes verschillen van context per software/applicatie, zoals:

* Themanaam en eventueel subnaam
* De in samenwerking tussen de klant en ontwikkelaar opgestelde Requirements als testcriteria
* Exacte waardes voor maximale- en behaalde score per test
* Eventueel meer kwaliteitseigenschappen met bijhorende toelichtingen als testcriteria

*Sjabloon voor regressietest*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testnaam | Regressietest op \*Software-/applicatienaam\* | | | |
| Themanaam | \*De thematiek achter de genoemde software/applicatie\* | | | |
| Subnaam |  | | |  |
|  | Testdoelen | Testcriteria | Maximale score | Behaalde score |
| 1 | Software/applicatie testen op Requirements |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | n.t.b. | n.t.b. | n.t.b. |
| b. |  | n.t.b. | n.t.b. | n.t.b. |
|  | Software/applicatie testen op algemene kwaliteitseigenschappen |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | Veiligheid - In de geteste software zijn de juiste encapsulaties toegepast voor de modules, zodat er onderscheid is tussen functionele data en gebruikersgevoelige gegevens. | n.t.b. | n.t.b. |
| b. |  | Controleerbaarheid - Klasses, functies, attributen en variabel zijn duidelijk en herkenbaar toegepast in de programmeercode, met als gevolg dat het opsporen en verhelpen bugs snel en efficiënt verholpen kan worden. | n.t.b. | n.t.b. |
| c. |  | Zuinigheid - Gevonden bugs worden zodanig verholpen in de geteste software dat er achteraf geen bestaande/nieuwe bugs ontstaan. | n.t.b. | n.t.b. |
| d. |  | Prestatie - Het opsporen en verhelpen van bugs moet op een dusdanige manier worden uitgevoerd waardoor er zoveel mogelijk tijd en aandacht bespaard kan worden voor de tests die ná de regressietest uitgevoerd moeten worden. | n.t.b. | n.t.b. |
| e. |  | Functionaliteit - Veranderingen in broncode moeten niet ongedaan worden gemaakt als dit geen bugs (meer) veroorzaakt. Hierdoor kan de door de klant en/of ontwikkelaar gewenste toevoegingen of bijwerkingen blijven functioneren. | n.t.b. | n.t.b. |
| Totaal |  |  | \*totaal max. punten\* =  100% | \*totaal behaalde punten\* = \*Gelijke percentage\*% |

*Sjabloon voor unittest*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testnaam | Unittest op \*Software-/applicatienaam\* | | | |
| Themanaam | \*De thematiek achter de genoemde software/applicatie\* | | | |
| Subnaam |  | | |  |
|  | Testdoelen | Testcriteria | Maximale score | Behaalde score |
| 1 | Software/applicatie testen op Requirements |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | n.t.b. | n.t.b. | n.t.b. |
| b. |  | n.t.b. | n.t.b. | n.t.b. |
|  | Software/applicatie testen op algemene kwaliteitseigenschappen |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | Bruikbaarheid - De geteste software moet grondig zijn gevalideerd op gebruik van softwaremodules binnen de software die (on)afhankelijkheid verantwoordt aan andere softwaremodules ongeacht de veranderingen in de betreffende broncode. | n.t.b. | n.t.b. |
| b. |  | Functionaliteit - De geteste softwaremodules moeten gevalideerd worden volgens door de klant en ontwikkelaar opgestelde Requirements. | n.t.b. | n.t.b. |
| c. |  | Onderhoudbaarheid - Geteste softwaremodules moeten kunnen blijven functioneren ongeacht de veranderingen in de broncode van de gehele software. | n.t.b. | n.t.b. |
| d. |  | Testbaarheid - De scoop en factoren van ieder geteste softwaremodule is stapsgewijs (in heldere details) te meten. | n.t.b. | n.t.b. |
| e. |  | Prestatie - Geteste softwaremodule moet voldoen aan de door de klant en ontwikkelaar opgestelde Requirements zodanig dat de module optimaal functioneert volgens de betreffende criteria. | n.t.b. | n.t.b. |
| Totaal |  |  | \*totaal max. punten\* =  100% | \*totaal behaalde punten\* = \*Gelijke percentage\*% |

*Sjabloon voor integratietest*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testnaam | Integratietest op \*Software-/applicatienaam\* | | | |
| Themanaam | \*De thematiek achter de genoemde software/applicatie\* | | | |
| Subnaam |  | | |  |
|  | Testdoelen | Testcriteria | Maximale score | Behaalde score |
| 1 | Software/applicatie testen op Requirements |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | n.t.b. | n.t.b. | n.t.b. |
| b. |  | n.t.b. | n.t.b. | n.t.b. |
|  | Software/applicatie testen op algemene kwaliteitseigenschappen |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | Inpasbaarheid - (De programmeercode van) ieder softwaremodule moet een ander softwaremodule kunnen ondersteunen of respecteren zodat er geen conflicten ontstaan tussen en binnen de modulen met als gevolg dat de samenvoeging van modules ten goede komt. | n.t.b. | n.t.b. |
| b. |  | Herbruikbaarheid - De geteste software moet grondig zijn gevalideerd op hergebruik van softwaremodules binnen een verzameling van een andere softwaremodules ongeacht de veranderingen in de betreffende broncode. | n.t.b. | n.t.b. |
| c. |  | Onderhoudbaarheid - Geteste softwaremodules moeten kunnen blijven functioneren ongeacht de veranderingen in de broncode van de gehele software. | n.t.b. | n.t.b. |
| d. |  | Connectiviteit - Geteste softwaremodules kunnen zonder slag of stoot met elkaar communiceren via gebruikersinterfaces (Front-end) en/of applicatie-interfaces (Backend). | n.t.b. | n.t.b. |
| e. |  | Testbaarheid - De scoop en factoren van ieder geteste softwaremodule is stapsgewijs (in heldere details) te meten. | n.t.b. | n.t.b. |
| Totaal |  |  | \*totaal max. punten\* =  100% | \*totaal behaalde punten\* = \*Gelijke percentage\*% |

*Sjabloon voor systeemtest*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testnaam | Systeemtest op \*Software-/applicatienaam\* | | | |
| Themanaam | \*De thematiek achter de genoemde software/applicatie\* | | | |
| Subnaam |  | | |  |
|  | Testdoelen | Testcriteria | Maximale score | Behaalde score |
| 1 | Software/applicatie testen op Requirements |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | n.t.b. | n.t.b. | n.t.b. |
| b. |  | n.t.b. | n.t.b. | n.t.b. |
|  | Software/applicatie testen op algemene kwaliteitseigenschappen |  | \*punten\* ≈ \*percentage afgerond op één decimaal\*% | \*punten\* ≈ \*percentage afgerond op één decimaal\*% |
| a. |  | Beheerbaarheid - Gebruiker van de geteste software en systeem kan zelf deze software op normale wijze (zoals geschreven in de Requirements) aansturen in dit systeem. | n.t.b. | n.t.b. |
| b. |  | Beveiliging - Geteste software levert geen conflicten op voor andere software en hardware van het systeem, zodanig dat de software geen gevaar vormt voor de particuliere werkzaamheden binnen het systeem van de gebruiker. | n.t.b. | n.t.b. |
| c. |  | Portabiliteit - De geteste software kan op ieder systeem ongehinderd functioneren, waarvan besturingssysteem door de software ondersteund wordt, ongeacht de hardware- en softwaremodules. | n.t.b. | n.t.b. |
| d. |  | Continuïteit - De geteste software moet binnen een door de klant bepaalde termijn kunnen functioneren zonder hulp, toevoegingen en bijwerkingen vanuit de ontwikkelaar, ongeacht verandering in de hardware modules van het geteste systeem. | n.t.b. | n.t.b. |
| e. |  | Gebruikersvriendelijkheid - Software moet ongeacht de hardware-modules van het geteste systeem kunnen functioneren m.b.t. verschillende handelingen die de gebruiker kan uitvoeren. Als gevolg hiervan zal de gebruiker zich niet ergeren aan het gebruik van de software. | n.t.b. | n.t.b. |
| Totaal |  |  | \*totaal max. punten\* =  100% | \*totaal behaalde punten\* = \*Gelijke percentage\*% |

**5. Literatuurlijst**

[1] Microsoft Word 2019 - Microsoft Corporation - <https://products.office.com/nl-nl/word>

[2] Swart, N. (2010) - Handboek Requirements - Brug tussen Business en ICT - Uitgeverij: Eburon Business.

Kaner, Cem (November 17, 2006). Exploratory Testing (PDF). Quality Assurance Institute Worldwide Annual Software Testing Conference. Orlando, FL.

Pan, Jiantao (Spring 1999). [*"Software Testing"*](http://www.ece.cmu.edu/~koopman/des_s99/sw_testing/) (coursework). Carnegie Mellon University*. Retrieved November 21, 2017*.

Basu, Anirban (2015). [*Software Quality Assurance, Testing and Metrics*](https://books.google.de/books?id=aNTiCQAAQBAJ&pg=PA150). PHI Learning. [*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier)) [*978-81-203-5068-7*](https://en.wikipedia.org/wiki/Special:BookSources/978-81-203-5068-7).

<https://github.com/junit-team/junit4/wiki/Parameterized-tests>

<https://junit.org/junit5/>

[Martyn A Ould & Charles Unwin (ed), *Testing in Software Development*, BCS (1986), p71](https://books.google.com/books?id=utFCImZOTEIC&pg=PA73&dq=integration+test&hl=en&sa=X&ei=4EpTVOvJMayu7Aak5YCIDA&ved=0CDwQ6AEwAg#v=onepage&q=integration%20test&f=false). Accessed 31 Oct 2014

Xuan, Jifeng; Monperrus, Martin (2014). "Test case purification for improving fault localization". Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014: 52–63. [*arXiv*](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[*1409.3176*](https://arxiv.org/abs/1409.3176). [*Bibcode*](https://en.wikipedia.org/wiki/Bibcode_(identifier)):[*2014arXiv1409.3176X*](https://ui.adsabs.harvard.edu/abs/2014arXiv1409.3176X). [*doi*](https://en.wikipedia.org/wiki/Doi_(identifier)):[*10.1145/2635868.2635906*](https://doi.org/10.1145%2F2635868.2635906). [*ISBN*](https://en.wikipedia.org/wiki/ISBN_(identifier)) [*9781450330565*](https://en.wikipedia.org/wiki/Special:BookSources/9781450330565).

Martin, Robert C. (2000) - Design Principles and Design Patterns (PDF)

<https://en.wikipedia.org/wiki/System_integration_testing>