

Inhoudsopgave: Datum:

[1 Hoofdstuk 1: Inleiding 2](#_Toc186125411)

[2 Hoofdstuk 2: Functional Requirements 3](#_Toc186125412)

[3 Hoofdstuk 3: Niet-Functionele Requirements 8](#_Toc186125413)

[4 Hoofdstuk 4: Usability testing 14](#_Toc186125414)

[5 Hoofdstuk 5: Ontwikkelstrategieën 22](#_Toc186125415)

[5.1 Inleiding 22](#_Toc186125416)

[5.2 Scrum 23](#_Toc186125417)

[5.3 Kanban 28](#_Toc186125418)

[6 Sjabloon van een use case: 29](#_Toc186125419)

[6.1 Use Case 29](#_Toc186125420)

[6.2 Niet-Functionele requirements 29](#_Toc186125421)

# Hoofdstuk 1: Inleiding

Hoe bouw je een succesvol ICT project?

* Binnen tijd
* Binnen budget
* Binnen de juiste scope

Functional requirements:

* Zitten in de scope
* Wordt via use cases opgesteld

Non-functional requirements:

* Zijn de scope
* Wordt via SMART opgesteld:
  + Specifiek
  + Meetbaar
  + Acceptabel
  + Realistisch
  + Tijdsgebonden

# Hoofdstuk 2: Functional Requirements

Functionele Requirements:

* Het vastleggen en éénduidig definiëren van de functionaliteit (taak van de functioneel ontwerper)
  + Het maakt niet per se uit hoe je schema er uit ziet, zolang het gehele plaatje maar klopt
* Volgens “best practices” en/of gestructureerde methoden

Afbeelding met schermopname, diagram, lijn, Perceel

Automatisch gegenereerde beschrijvingUse Case Diagram:

* Verzameling alle FR
  + :
* Overzicht alle rollen
  + Bollen zijn de Use Cases
* Overzicht alle frs
  + Actoren zijn de rollen

Use Case Diagram - Include

Afbeelding met cirkel, diagram, Lettertype, tekst

Automatisch gegenereerde beschrijvingWanneer Use Case A wordt uitgevoerd wordt Use Case B **automatisch** **ook uitgevoerd**

Afbeelding met cirkel, Lettertype, symbool, wit

Automatisch gegenereerde beschrijving

Afbeelding met cirkel, Lettertype, symbool, wit

Automatisch gegenereerde beschrijvingUse Case Diagram - Extends

* Wanneer Use Case A wordt uitgevoerd wordt Use Case B **niet altijd uitgevoerd**
* Iemand die bv niet weet hoe hij moet inloggen kan hij Use Case B gebruiken als hulp

Regels bij Use Cases:

* Your use case is not my use case
* Manage your energy
  + Schrijf niet je use case in 1 keer
  + De belangrijkste dingen eerst (later de inhoud/stappen van de use case)
* Keep It Simple
* Communicatie: duidelijkheid

Hoe een use case opstellen:

* Geef een globaal beeld van de functies van het systeem (low-precision)
* Het verhaal wordt voorgesteld door opdrachtgever(s)
* Ieder schrijft eigen verhaal, ieder moet weten wat het systeem doet [groep]
* Scope en domein bepalen [groep]
* Verschillende functionaliteiten worden uitgeschreven [individueel of kleinere groep]
* Bespreking van de verschillende individuele verhalen [groep]

Conclusie:

* Visie op het systeem
  + Alle use cases samen = het systeem
* Scope
* Situering van het systeem
* Belanghebbende partijen (Primary actors, Stakeholders)
* User’s goals / doelen
* Alle verhalen (elk verloop max. ½ pag. Lang )

Geef een voldoende gedetailleerde visie:.

* Brainstorming [groep]
* Elementaire business processen = use cases
  + Leg een zelfde manier af van schrijven voor consistentie
* Primaire actoren
* User goals
* Prioriteiten

Enkele tips:

* Blijf in dezelfde kamer
* Juiste info door de juiste mensen
  + Vraag info aan de juiste mensen met de gekende informatie
* Kleinere groepen werken efficiënter
* Spendeer max. ½ dag met de eindgebruiker
  + Houdt je ondervragingen max 4u per dag (1/2 dag) voor meer productiviteit
* Management (opdrachtgever) mee in je boot
  + Zorg dat je opdrachtgever mee is met het verhaal
* Let op actor <> jobtitel
* “Use cases bij de kg” vermijden
  + Geen overbodige use cases maken
* Je zal zeker vastlopen
* Use cases uitwerken is een sociale activiteit …
* Wees een minimalist
  + Schrijf enkel wat nodig is (bv. Zonder lidwoorden)

Een use case bevat volgende elementen:

* Primary Actor
* Stakeholders
* Precondities
* Postcondities
* Normaal Verloop
* Alternatief verloop
* Domeinspecifieke regels
* Op te klaren punten (optioneel)

Primary actors/stakeholders:

* Stakeholders zijn belanghebbende / deelnemende partijen
* Primary actor is de belanghebbende partij die het systeem vraagt een taak (service) uit te voeren
* Primary actor activeert de use case/ functionaliteit

Elementair business proces/ use case:

* User’s goal
  + Primaire actor moet dat doel kunnen bereiken
* Proces door 1 persoon
  + Het moet uitgevoerd kunnen worden door 1 persoon
* 2 à 20 min
  + Uitvoering van een use case is meestal 2-20 min

Use case level rating:

* Afbeelding met tekst, schermopname, lijn, Lettertype

  Automatisch gegenereerde beschrijvingWe proberen een level bij te houden wat een bepaalde use case is
* Zo kunnen we inzien of het wel nut heeft om de use case wel of niet te maken:
  + **Hoe**?
    - **Low level** (heel kort in stappen + extreem lage uitvoertijd)
  + **Wat** wil de primary actor?
    - **User goals** (Duurt 2 -20 min)
  + **Waarom** doet de primary actor dit?
    - **Higher level** (heel veel stappen + extreem lange uitvoertijd meestal in dagen)
* Een goede use case is 3 tot 15 stappen

Precondities

* Geeft aan wat vervuld moet zijn bij de start van de use case
* Controle in de use case kan/hoeft niet meer!
* In vele gevallen geeft deze aan dat een andere use case reeds uitgevoerd is
* Moeten VOOR de start van een use case kunnen gevalideerd worden
* Kan leeg zijn (bv. Bij de 1e use case)

Postcondities:

* Geeft aan wat vervuld moet zijn na uitvoeren van een scenario
* Bevat wijzigingen van het systeem t.o.v. Het domeinmodel
  + Er kan iets bijgehouden zijn
* Formuleren vanuit standpunt van het systeem
  + Beschrijven als: “Het systeem heeft …”
* Niet elk alternatief scenario bereikt de postcondities

Verloop:

* Normaal verloop: Top-to-bottom beschrijving van een eenvoudige (meest voorkomende) situatie waarbij het doel van de Primay Actor gerealiseerd wordt. = **Main succes story**
* Alternatieve wegen zijn uitbreidingen op het normaal verloop of een andere **alternatieve weg**
* Een verloop is een chronologische ordening van **actiestappen**

Set Actiestappen:

* Volgorde
* Actiestap
  + Interactie tussen primary actor en systeem
    - Klant geeft adres in
  + Een validatie
    - Systeem valideert de PIN
  + Een interne wijziging
    - Systeem vermindert bedrag met hoeveelheid

Tips bij actiestappen:

* **Tip 1: gebruik eenvoudige zinnen**
  + Onderwerp – werkwoord - voorwerp
* **Tip 2: “wie heeft de bal”**
  + Bij elke actie heeft 1 actor een boodschap - Duidelijkheid
  + Je moet weten wie wat doet – wie is in welke stap aan de beurt?
* **Tip 3: Volelperspectief/Bird’s Eye View**
  + Geef bankkaart en pincode. Verminder saldo met opgegeven bedrag
  + Naar
  + De klant plaatst bankkaart in kaartlezer en geeft pincode in Het systeem vermindert het saldo met het opgegeven bedrag
* **Tip 4: toon de vooruitgang in het proces**
  + De grootte van de vooruitgang in 1 stap is gerelateerd aan het niveau van de use case
  + Te kleine stappen, te veel stappen, te lange use case (max . 15 stappen), te laag niveau ( te veel details)
* **Tip 5: toon wat de bedoeling van de actor is, niet de beweging**
  + Beschrijf **GEEN interacties met de GUI (MAAR OOK ECHT NIKS)**
  + De dialoog die wordt beschreven is een veronderstelling van het ontwerp van de GUI Functionele eisen
  + Vermeld NIET hoe iets gebeurt, maar wat er gebeurt, vb:
  + Het systeem connecteert met de databank, voert de query uit en toont de tabellen
  + Naar
  + Het systeem toont het resultaat van de operatie
  + Je zegt **WAT** er gebeurt **niet** **HOE**
* **Tip 6: Aanvaardbare set van actiestappen**
  + Iets opvragen, iets valideren, er wordt iets gewijzigd, resultaat
* **Tip 7: Valideer, controleer niet of …**
  + Vermijd “if statements” (in de use case)
* **Tip 8: User laat systeem A systeem B aansturen**
  + Beschrijf geen interacties met de GUI ( “user hits fetch button “)
  + Mogelijke oplossing :
    - User signaleert het systeem data op te halen van systeem B
    - Systeem haalt achtergrondinfo op bij systeem B Of
    - User laat het systeem achtergrondinfo ophalen bij systeem B
* **Tip 9: Use case roept andere use case op** 
  + Gebruik geen calls
  + Schrijf in de taal van de opdrachtgever
  + Gebruiker betaalt de winkelmand

Alternatieve verlopen:

* Afwijkingen t.o.v. Het normale verloop
* Uitbreidingen eindigen ofwel in succes of maken dat de use case verlaten wordt
* Belangrijk: uitbreidingen niet zomaar naast je neerleggen
  + Het normaal verloop is zeer goed gekend door het team
  + Uitbreidingen vragen dikwijls kennis van “business rules”
  + Soms aanleiding tot nieuwe use case

Use Cases uitbreidingen:

* Brainstorming over alle mogelijke situaties
* Evalueer, elimineer en voeg samen
* Werk uit
* Noteer “**wat** het **SYSTEEM** ontdekt heeft, niet wat er gebeurd is“
  + **Niet** : “klant vergeet pincode in te geven”
  + **Wel** : “tijdslimiet overschreden bij ingave pincode” of “systeem detecteert foute PIN
* Geeft aan wat het systeem doet maar geeft ook de mogelijkheid om **alternatieven** te geven
* Elke uitbreiding heeft eigen omschrijving/titel
* Een uitbreiding is apart scenario met actiestappen
* Alternatief verloop eindigt door:
  + Terug te keren naar stap in normaal verloop
  + Verder te gaan naar een stap in ander normaal verloop
  + Een externe/nieuwe use case op te roepen
    - Impliceert onderhoud, testen, …
  + Stopzetten van de huidige use case

Hoe schrijven we het alternatief verloop:

* 4. Systeem valideert wachtwoord
  + 4a. Fout wachtwoord
    - 4a1. Systeem verhoogt aantal foutieve pogingen met 1
    - 4a2. Systeem toont een gepaste melding
    - 4a3. Keer terug naar stap 2 van het normale verloop
  + 4b. Fout wachtwoord te dikwijls ingegeven
    - 4b1. Systeem verwittigt gebruiker
    - Use case eindigt zonder bereiken postcondities

Domeinspecifieke regens en op te klaren punten (optioneel):

* Domeinspecifieke regels
* Plaats hier alle technische regels voor validaties, logica, … b.v.: DR\_NIEUW\_LID
  + Een nieuw lid is minstens 18 jaar oud
  + Wachtwoord is 8-16 tekens lang en bevat geen spaties
* Op te klaren punten:
  + Vermeld wat nog onduidelijk is (communicatie klant)
  + Optioneel
  + Dit is een notitie bij die bepaalde use case bij het volgende gesprek met de klant

# Hoofdstuk 3: Niet-Functionele Requirements

Requirements:

* Ieder systeem wordt ontwikkeld om een bepaalde reden
  + Beter procesbeheer
  + Lagere kosten
  + Beter benutten van commerciële mogelijkheden
  + Verhogen van de servicegraad
  + Vooraf eisen helder en éénduidig formuleren
    - Business case

Voorbeeld (een systeen heeft een reden voor iedereen):

* **Hotelmanager**: het nieuwe systeem moet heel het primair proces ondersteunen
* **Ontwikkelaar**: belangrijkste requirement is toch dat de klant online een hotelkamer kan reserveren
* **Receptioniste**: als de klanten de info maar in hun moedertaal kunnen lezen
* **Technisch beheerder**: het systeem moet in Java gebouwd zijn, anders kunnen we het niet in beheer nemen
* **Directeur**: Vergeet niet dat het systeem absoluut volgende jaar operationeel moet zijn
* **Hotelmanager**: ik heb positieve geluiden gehoord over agile werken in IT-projecten. Dat moeten we doen , want dan kunnen we tussentijds bijsturen
* **Requirementsanalist**: laten we beginnen met de reden waarom het nieuwe systeem er moet komen nl bezettingsgraad van de hotelkamers moet met 10% stijgen.

Requirement als behoefte? Als eis?

* Requirement = **behoefte** aan geautomatiseerde ondersteuning
  + Een behoefte van een belanghebbende om een nieuw of bestaand proces geautomatiseerd proces te ondersteunen
  + Een behoefte van een belanghebbende om verbeteringen te realiseren, Voorbeeld:
    - De inkoper wil laten controleren of de voorraad het minimum bestelniveau heeft bereikt
    - De systeembeheerder wil vastleggen welke medewerkers toegang tot het systeem mogen krijgen
    - De manager wil de volledige orderverwerking geautomatiseerd laten ondersteunen

Requirement als behoefte? Als eis? Vervolg:

* Requirement = eis gesteld aan het systeem
  + De eis geeft aan wat gedrag of de kwaliteit van het systeem moet zijn
  + Belanghebbende (business)
    - Stelt deze eis aan het systeem (behoefte)
    - Wil een bepaald doel bereiken / probleem oplossen / voordeel behalen

Requirements:

* Vooraf eisen helder en éénduidig formuleren
  + Identificeren van stakeholders
  + Formuleren van KSF (kritische succesfactoren)
  + Formuleren van meetbare acceptatiecriteria

Requirements-KSF:

* **Formuleren kritische succesfactoren (KSF)** Voorbeelden:
* **Business- to- consumer internetsite (grote heterogene groep gebruikers)** 
  + KSF: gemak waarmee de gebruiker het gewenste product kan bereiken
  + KSF: begrijpbaarheid, zelfverklarend
* **Boekhoudpakket (beperkte groep van zeer prof.gebruikers)** 
  + KSF: traceerbaarheid Zelfverklarend is minder belangrijk
* **Grote batchgeoriënteerde administratieve systemen** 
  + KSF: snelheid en “middelen”beslag
* **PC-applicatie** 
  + KSF: werkbaar onder verschillende systeemsoftware

Afbeelding met tekst, schermopname, Lettertype, Elektrisch blauw

Automatisch gegenereerde beschrijvingRequirements: Meetbare acceptatiecriteria:

* Formuleren van meetbare acceptatiecriteria
  + KSF: redelijk abstract
  + Meetbare acceptatiecriteria: concreet
  + Voorbeeld:
    - **KSF**: klanten direct helpen aan de telefoon
* Afbeelding met tekst, schermopname, Lettertype, Elektrisch blauw

  Automatisch gegenereerde beschrijvingFormuleren van meetbare acceptatiecriteria
  + Voorbeeld:
  + **KSF**: bij invoer van gegevens wordt gecontroleerd op juistheid en volledigheid

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijvingSoorten Requirements:

* Businessrequirements
* Gebruikersrequirements
  + Softwarerequirements
  + Functionele requirements
  + Niet-functionele requirements

Niet-functionele requirements:

Afbeelding met tekst, schermopname, Lettertype, ontvangst

Automatisch gegenereerde beschrijving

Soorten niet-functionele requirements (ISO 25010): functionele geschiktheid:

* **Functionele compleetheid** 
  + Het ERP moet aan minimaal 75% van de gespecifieerde functionele requirements voldoen
* **Functionele correctheid** 
  + De loonadministratie moet de berekende lonen naar de juiste rekeningnummers laten overmaken
* **Functionele toepasselijkheid**
  + Het GPS-systeem moet zodanig werken dat de gebruiker in 9 van de 10 gevallen zonder problemen op de gewenste bestemming aankomt

Soorten niet-functionele requirements: functionele geschiktheid:

* **Snelheid** 
  + Het systeem moet 90% van de webpagina’s die klanten zien binnen 1 seconde tonen, voor de andere webpagina’s mag dit niet langer dan 3 seconden zijn
* **Middelenbeslag** 
  + Het systeem mag niet meer dan 1MB aan netwerkcapaciteit en niet meer dan 100GB aan opslagcapaciteit gebruiken
* **Capaciteit** 
  + Het systeem moet 1200 gebruikers tegelijkertijd aan kunnen met een piek van 2500 op de eerste werkdag van de maand. Tijdens de piek mag de snelheid max met 20% dalen

Soorten niet-functionele requirements: Uitwisselbaarheid:

* **Beïnvloedbaar** 
  + Het systeem moet de snelheid van de taken die het op de achtergrond uitvoert verlagen wanneer de servercapaciteit voor meer dan 80% benut wordt (door alle systemen samen)
* **Koppelbaarheid** 
  + Het datawarehouse mag max 1 op de 100,000 berichten vanuit het verkoopsysteem verkeerd interpreteren

Soorten niet-functionele requirements: Beveiligbaarheid:

* **Vertrouwelijkheid** 
  + De geldautomaten moeten afdwingen dat bij het aanvullen van bankbiljetten steeds geautoriseerde medewerkers aanwezig zijn
* **Integriteit** 
  + Het systeem kan aantonen dat een transactie geautoriseerd door Jan effectief werd ingevoerd door Jan
* **Onweerlegbaarheid** 
  + Het systeem moet onvervalsbaar bewijs kunnen leveren van de ontvangst van ieder order
* **Verantwoording** 
  + Het klantbedieningssysteem moet voor elke medewerker vastleggen wat deze met de klant heeft gecommuniceerd.
* **Authenticiteit**
  + Het systeem moet de identiteit van de kassière vaststellen voordat de geldlade opengaat (b.v. In kledingwinkel)

Soorten niet-functionele requirements: Bruikbaarheid:

* **Herkenbare geschiktheid** 
  + De 1ste keer dat iemand onze nieuwe app opstart moet hij binnen de 3 sec kunnen beoordelen of hij voldoet aan zijn behoefte zonder dat hij de info in de Play Store heeft gelezen
* **Leerbaarheid** 
  + 95% van de beginnende- studenten moet zijn oplossingen zelfstandig kunnen indienen via Chamilo ook als ze voor het eerst met het systeem werken
* **Bedienbaarheid**
  + Een ervaren internetgebruiker moet in 2 min op basis van zijn selectiecriteria een hotelkamer kunnen boeken, ook als hij de reserveringssite voor de 1ste keer bezoekt
* **Voorkomen gebruikersfouten**
  + Het boekhoudpakker met altijd voorkomen dat er door foutieve invoer boekhoudkundige onjuiste journaalposten ontstaan. De gebruiker moet gewaarschuwd worden
* **Volmaaktheid gebruikersinteractie** 
  + Het datawarehouse mag max 1 op de 100,000 berichten vanuit het verkoopsysteem verkeerd interpreteren
* **Toegankelijkheid** 
  + Het in- en uitchecken via een abonnementskaart (trein-tram-bus) moet ook mogelijk zijn voor reizigers met volgende beperkingen (doofheid, blindheid, niet bekend met de Nederlandse taal, in een rolstoel zitten)

Soorten niet-functionele requirements: Onderhoudbaarheid:

* **Modulariteit** 
  + 98% van de functionele wensen moet te realiseren met code-aanpassingen in basismodule en /of in max 1 andere module (customising ERP-pakket)
* **Herbruikbaarheid** 
  + Mintens een derde van de code moet herbruikbaar zijn door andere systemen (opgenomen in herbruikrepository)
* **Analyseerbaarheid** 
  + Als het systeem geen elektronische berichten meer kan ontvangen, moet de systeembeheerder binnen de 10 min nadat hij hier werd geattendeerd, de oorzaak van de fout gevonden hebben.
* **Wijzigbaarheid** 
  + De bankbediende moet zelf eenvoudige fiscale wijzigingen (bvb bedragen en percentages) binnen de 10 min wijzigingen kunnen aanbrengen in zijn lokaal systeem
* **Testbaarheid** 
  + De beoogde werking van een nieuwe release van het systeem moet binnen 24u kunnen gevalideerd worden, spoedreparatie binnen de 2u

Soorten niet-functionele requirements: betrouwbaarheid:

* **Volwassenheid** 
  + De spraakherkenningssoftware moet de 5000 meeste gebruikte NL-talige woorden in 98% van de gevallen correct verstaan
* **Beschikbaarheid** 
  + De webshop mag niet meer dan 1 keer per half jaar langer dan 2 minuten uitvallen tussen 8u00 en 22u00
* **Foutbestendigheid** 
  + Het boekhoudsysteem mag niet uitvallen door foutief ingevulde gegevens of verkeerde verwerkingen (bvb deling door 0)
* **Herstelbaarheid** 
  + Wanneer loonverwerkingssysteeem uitvalt in de week voor de loonuitbetaling moet het binnen de 4 u weer operationeel zijn.
  + De geldautomaat mag in geen enkel geval (ook niet bij stroomuitval of panne) volgende gegevens niet kwijtraken (opgenomen bedrag, rekeningnummer, datum+tijdstip geldopname)

Soorten niet-functionele requirements: Overdraagbaarheid:

* **Aanpasbaarheid**
  + Een desktopapplicatie moet in 2 maanden kunnen overgezet worden naar Android 10.0 of hoger
  + De software moet van Windows naar Linux kunnen overgezet worden zonder dat meer dan 15% van code aangepaste hoeft te worden
* **Installeerbaarheid** 
  + Het voorraad- en bestelsysteem moet door de winkelier zelf kunnen geïnstalleerd worden binnen een halfuur
  + De klant moet de software (gedownload via internet) in 3 klikken kunnen installeren
* **Herbruikbaarheid** 
  + Het nieuwe agendaprogramma moet zonder problemen MS Outlook kunnen vervangen en de gegevens vanuit outlook kunnen importeren

Sjabloon niet-functionele requirements:

Afbeelding met tekst, schermopname, Lettertype, logo

Automatisch gegenereerde beschrijving

Niet-functionele requirements – SMART:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

Niet-functionele requirement – Voorbeeld:

Afbeelding met tekst, schermopname, Lettertype, nummer

Automatisch gegenereerde beschrijving

# Hoofdstuk 4: Usability testing

Verschillende modellen

* Conceptueel Model:
  + Acutal model that is the given to the user trought the design and interface of the product
* Mentaal Model
  + Representation a user has in his mind about the object he is interacting with
* Systeem Model
  + How the system actually works

Afbeelding met symbool, Lettertype, logo, Graphics

Automatisch gegenereerde beschrijvingUser experience vs. Usability:

* **User experience 9241-11** 
  + “all aspects of the user’s experience when interacting with the product, service, environment or facility”
* **Usability 9241-210** 
  + “effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments”

User experience vs. Usability (Very simply put…) :

* User Experience
  + “Did the user have as a delightful experience as possible?”
* Usability
  + “Can the user accomplish his goal?”

Usability testing: ISO norm

* Norm
* Meting
* Afbeelding met cirkel, Lettertype, diagram, tekst

  Automatisch gegenereerde beschrijvingEvaluatie (SUS)

Usability testing: norm ISO 9241-11 (Product):

* Opgegeven **gebruikers**
* Opgegeven **doelen**
* Gespecificeerd **context**

Afbeelding met Hoofdtelefoon, cirkel, oortelefoon, internet

Automatisch gegenereerde beschrijvingUsability testing: norm ISO 9241-11:

* Effectiviteit
* Efficiëntie
* Tevredenheid

Afbeelding met tekst, stoel

Automatisch gegenereerde beschrijving

Hoe meten we usability?

* Meest eenvoudige methode :
* Waarnemen van gedrag van “echte gebruikers” terwijl ze echte taken uitvoeren om problemen rond gebruiksvriendelijkheid te identificeren

Usability testing: meting

* Mogelijke criteria
* Succespercentage
  + Percentage van de deelnemers die correct en volledig elk doel bereiken (zonder hulp)
* Foutenpercentage
  + Percentage van de deelnemers die denken dat ze succesvol waren, maar mislukten
* Aantal fouten per tijdseenheid
* Aantal verzoeken om hulp
* De gemiddelde tijd nodig om elke taak te voltooien (samen met het bereik en de standaardafwijking)
* Tijd nodig voor eerste poging
* Tijd om de taak uit te voeren vergeleken met een deskundige gebruiker
* Aantal aan te nemen klikken om taak te bereiken
* Subjectieve tevredenheid van de gebruiker

Usability testing: meting

* Meten op basis van aantal criteria
* Succespercentage
* Aantal fouten per tijdseenheid
* De gemiddelde tijd
* Subjectieve tevredenheid van de gebruiker
* Minimale criteria voor kwaliteitsvolle meting

Waarom criteria?

* Vooruitgang meten tussen release
  + Methodologie kan enkel aangepast worden als je weet hoe goed je het doet
* Inschatten marktpositie
  + Ben ik beter of slechter dan de concurrentie?
  + Op welk vlak ben ik beter/slechter?
* Sneller kunnen een Stop/Go beslissing nemen
  + Is het ontwerp goed genoeg om gereleased te worden
* Als middel voor bonusverloning projectleiders
  + Bvb: projectleider krijgt bepaalde bonus als het aantal telefoons / e-mails naar de klantendienst voor zijn product onder een grens ligt

Twee methodes:

* **Formatief** testen (b.v. Hardop lezen)
* **Summatief** of benchmark testen

Usability testing: formatief testen

* HOE ?
* Een testdeelnemer maakt gebruik van een prototype om vooraf gedefinieerde taken uit te voeren
* De deelnemers denken hardop terwijl hij of zij deze taken uitvoert
* Product in vroeg stadium: schermafdrukken en papieren mock-ups zijn prima

Deelnemers:

Minstens 5 representatieve gebruikers - elk individueel getest • Testmoderator om de deelnemers te begeleiden door de test, vragen stellen om gedrag te verduidelijken • Observator/notulist (kan gecombineerd worden met moderatorrol)

Usability testing: summatief testen:

* Volledig werkend product
* Gebruikers voeren representatieve taken uit (= productbehoeften)
* Opstellen benchmark of resultaten vergelijken met vorige benchmark(s). Deze bevat o.a.:
  + Succespercentage
  + Aantal fouten per tijdseenheid
  + De gemiddelde tijd
  + Subjectieve tevredenheid van de gebruiker.

Usability testing: evaluatie:

* Gemiddelde score met behulp van een vastgestelde vragenlijst
  + Verhouding van positief tegen negatief meningen die gebruikers hebben gegeven over de product
  + % van participanten die het product zouden aanbevelen aan een vriend
  + % van de gebruikers die de website makkelijker beoordelen dan een expert
* Elke vraag heeft een vaste waardingsschaal:
  + 1 = helemaal oneens
  + 5 = helemaal eens
* = System Usability Scale (SUS)

System Usability Scale (SUS) Voorbeeldvragen vragenlijst:

* Ik denk dat ik het systeem frequent wil gebruiken.
* Ik vond het systeem nodeloos complex.
* Ik vond het systeem gemakkelijk te gebruiken.
* Ik denk dat ik technische ondersteuning nodig heb om het systeem te gebruiken.
* Ik vond de verschillende functionaliteiten goed geïntegreerd in het systeem.
* Ik vond dat het systeem te veel inconsistent was.
* Ik kan me inbeelden dat de meeste mensen snel met dit systeem aan de slag kunnen.
* Ik vond het systeem heel hinderlijk in gebruik.
* Ik was zeer zelfzeker om met het systeem te werken.
* Ik moest veel bijleren voor ik met het systeem kon werken.

SUS waarde berekenen:

* Som van de bijdrage van elke vraag berekenen:
  + Oneven vraag: de score - 1
  + Even vraag: 5 - de score
* Vermenigvulidig de som met 2,5
* Minimale SUS waarde? 0
* Maximale SUS waarde? 100

SUS waarde evalueren:

* <= 51 = F: usability moet topprioriteit worden
* 68 = C: gemiddelde SUS waarde
* > 74 = B: waargenomen bruikbaarheid van 70% van alle geteste producten.
* > 80,3 = A: de top 10% van de waardes. Dit is het punt waar gebruikers het product aanbevelen aan een vriend.

Formatief testen vs Summatief:

* **Formatief** testen: Gericht op het vinden en verbeteren van specifieke ontwerpproblemen
* **Summatief** testen: Product evalueren a.d.h.v. Vastgelegde criteria

Usability testing : in de praktijk:

* 5 deelnemers in een gebruikerstest
* Vind ongeveer 80% van de problemen
* 80% bevat de belangrijkste kwesties
* Usability testing is
  + Gedrag gedreven
* En
  + Gedrag verandert niet veel tussen deelnemers

Afbeelding met tekst, schermopname, Lettertype, Elektrisch blauw

Automatisch gegenereerde beschrijvingUsability testing = iteratief proces:

* Meer iteraties van kleinschalige tests 1ste iteratie: meest significante problemen
  + Fix deze problemen ...
* 2de iteratie: nieuwe problemen ontdekt
  + Fix deze problemen ...
* 3de iteratie : …
  + Fix deze problemen ...
* Aan het einde van het ontwerp zou je kunnen getest hebben met 20 - 30 deelnemers of meer

Usability testing = iteratief process:

* Moderator **verwelkomt** de deelnemer en geeft uitleg over de test sessie en vraagt eventuele pre-test of demografische vragen.
* Moderator verklaart “**hardop te denken**” en vraagt of de deelnemer bijkomende vragen heeft.
* Moderator / Deelnemer leest het **takenscenario hardop** en begint te werken aan het scenario terwijl ze hardop denken.
* Moderator / Waarnemers nemen **notities** van het **gedrag** van de deelnemer, opmerkingen, fouten en afronding (succes of falen) op elke taak.
* Moderator vraagt aan het einde van de sessie **subjectieve vragen** of stuurt de deelnemer door naar een online enquête
* Moderator **bedankt de deelnemer**, geeft de deelnemer de overeengekomen vergoeding, en begeleidt ze uit de testomgeving

Usability testing: meest voorkomende fouten

* Men weet niet wat men wil testen
* Het team is niet betrokken
* De verkeerde groep testers (bv.: mensen met te veel voorkennis)
* Men test het verkeerde
* Inefficiënt organiseren van de test
* Geen planning hoe de testresultaten verspreid worden
* Geen iteraties om oplossingen te testen

Testen voorbereiden:

* Usability Test Plan
* Persona’s
* Key Tasks
* Scenario’s
* Mock-up

Persona’s Helpen om:

* Je in te leven in de gebruiker, zijn doelen en verwachtingen
* Een gemeenschappelijk begrip van de gebruiker met het hele team vast te leggen
* Beslissingen te nemen tijdens het ontwerpproces
* Het ontwikkelingsteam te focussen op de behoeften en verwachtingen van de gebruiker
* Veronderstellingen over gebruikers weg te werken

Persona’s Baseer je op eigen gebruikersonderzoek:

* Observaties, klanteninformatie,…
* Werk je persona’s uit in team vanuit verschillende perspectieven
* Maak persona’s een expliciete stap in het ontwikkeltraject
* Maak persona’s zichtbaar voor heel het team
* Creëer je persona’s specifiek voor je product
* Integreer persona’s in de scenario’s van de gebruiker
* Persona’s zijn zeer krachtig tijdens het ontwerpen en valideren
* Ontwikkelaars zijn gefocust op echte eindgebruikers

Persona’s: Kiosk in de supermarkt:

* Huishoudelijke goederen bestellen die niet onmiddellijk verkrijgbaar in de winkel
* Grote variëteit aan klanten en potentiële gebruikers.

Key tasks:

* Kerntaken die gebruikers absoluut met het product willen uitvoeren om hun doelen te bereiken:
  + Frequente taken identificeren
  + Identificeer kritische / problematische taak
* Voorbeeld : Google maps

Scenario's beschrijven:

* Hoe personages hun doelen bereiken
* Hoe ze erover voelen

Mock-ups:

* In de **software**-industrie komt het begrip tevens voor bij het vroeg in het ontwikkelproces **testen** van de **gebruikersinterface** van het software-ontwerp.
* Een mock-up krijgt pas de term **prototype** als het ontwerp ook echt werkt.

Test:

* Verwelkom tester
* Denk Luidop
  + Demonstreer hardop met een heel makkelijk voorbeeld:
  + Hoogte van de stoel aanpassen
  + Een persoon toevoegen aan uw lijst van contacten op je smartphone
  + Gebruik Google om erachter te komen ... Laat deelnemer oefenen met een ander heel makkelijk voorbeeld
* Lees scenario voor
* Neem notities
  + Noteer alleen wat u ziet of hoort - geen interpretaties:
    - Gedrag van de deelnemer
    - + / - reacties (kijk naar lichaamstaal)
    - Ontwerp ideeën
    - Gebruikersvraagstukken
    - Bugs
  + Scenario Voltooid ? :
    - Met succes
    - Met succes met hulp
    - Mislukking
* Einde van de test
  + Wat vind je van …
* Bedank de tester

Pilot testing stelt je in staat om

* De software te testen
* Handig om foutieve flows op te sporen
* Te ontdekken of de vragen en scenario's duidelijk zijn voor deelnemer (tester)

Rapportering:

* Beoordeel problemen volgens kritiek belang
* Presenteer naar management
* Gouden regels

Beoordeel problemen volgens kritiek belang:

* Problemen die gevolgen hebben voor de **het slagen van de flow** zijn ernstiger dan problemen die de tevredenheid beïnvloeden
* Problemen **die verschillende gebruikers beïnvloeden** zijn erger dan diegenen die alleen een paar beïnvloeden
* Problemen die **blijven opduiken** zijn ernstiger omdat ze een grotere impact hebben op het slagen de en tevredenheid

Maak presentatie (management):

* Focus op moeilijke of onmogelijke te voltooien taken
* Toon gedrag van deelnemers door :
  + Verhalen vertellen
  + Highlight video's weergeven
  + Je laat zien waar testers problemen hadden
* Presenteer gegevens uit de enquêtes : SUS, ...
* Verantwoording van de kosten van de mogelijkse veranderingen

Gouden Regels rapportering:

* Begin door te vertellen wat goed is met het ontwerp
* Baseer je rapport op bewijsmateriaal
  + Citaat van deelnemers
  + Statistieken (geslaagd/mislukt),
  + Gemiddelde taakduur of taaktevredenheid (SEQ)
* Dingen die ze direct kunnen repareren
  + Terminologie, kleuren, enz.
* Toon de ernst van de problemen op
  + Gebaseerd op frequentie, aantal getroffen gebruikers, enz.
* Vertel het ontwerpteam hoe u deze problemen kunt oplossen
  + Geef de oorzaak van het probleem op
  + Een ontwerpoplossing bieden: een mock-up, een ontwerppatroon, een beschrijving of een screenshot van een ander systeem

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijving

Usability testing: in de praktijk:

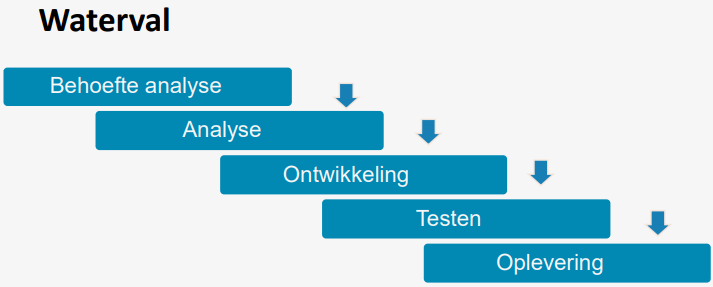
* Betreft **echte gebruikers**
* Testdeelnemers vervullen **echte taken** om hun doelen te bereiken
* **Let op gedrag**: welke test deelnemers doen
* **Observaties** worden gelogd om echte problemen te analyseren en te identificeren
* Formuleer **aanbevelingen** om het product verbeteren

**Usability testing: in de praktijk:**

* Usability testing vereist niet :
  + Usability lab
  + Moderator
  + Waarnemers / notulisten
  + Schermopname
  + Video-opname
  + Enz.

# Hoofdstuk 5: Ontwikkelstrategieën

## Inleiding

Ontwikkelingsstrategieën:

* Inleiding
  + Waterfall
    - In één iteratie
  + Agile
* Scrum
* Kanban

Wat is Agile?

* Een filosofie
* Een mentaliteitswijziging (van Command&Control naar Collaboration)
* Geschiedenis
  + Reactie op waterval
  + Gebaseerd op Agile Manifesto

Het Agile Manifesto (2001):

Afbeelding met tekst, schermopname, Lettertype, ontwerp

Automatisch gegenereerde beschrijving

Afbeelding met clipart, tekenfilm, tekening, paraplu

Automatisch gegenereerde beschrijving

Principes achter het Agile Manifesto:

* Hoogste prioriteit: klant tevreden
* Verwelkom wijzigingen
* Lever geregeld werkende software op
* Business en ontwikkelaars werken dagelijks samen
* Bouw producten rond gemotiveerde individuen
* “Face-to-face” communicatie binnen team
* Werkende software is de eerste meting van vooruitgang
* Afbeelding met tekst, schermopname, Lettertype, diagram

  Automatisch gegenereerde beschrijvingEenvoud is essentieel
* Zelf organiserende teams
* Zelf bespiegelende teams

## Scrum

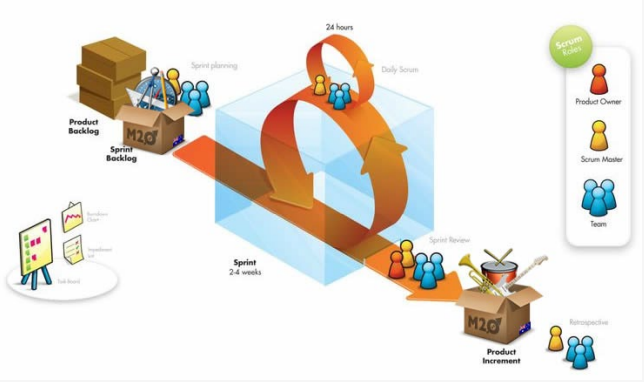
Wat is Scrum?

* Scrum is geen...
  + ...methodologie
  + ...afgebakend en vast proces
  + ...set van procedures
* Scrum is open framework met eenvoudige regels
* Regels gebaseerd op **CAS**
  + Complex Adaptieve Systemen (vb. School vissen)
  + Doel: CAS evolueren naar intelligentie
* Iteratief en incrementeel
  + Iteratief = het zal niet allemaal vanaf de eerste keer correct zijn
  + Incrementeel = verticaal in plaats van horizontaal bouwen

Scrum regels:

* Werken in **gesloten iteraties** (geen scopewijzigingen)
* Elke iteratie **productiewaardige software**
* **Zelforganiserende** en **zelfreflecterende** teams
* Afbeelding met tekst, schermopname, diagram

  Automatisch gegenereerde beschrijvingAlles heeft een bepaalde **prioriteit**



Rollen: Product-eigenaar:

* Vertegenwoordigt de belanghebbende partijen
  + Klant, eindgebruiker, …
* Eén stem, liefst één persoon
* Bepaalt
  + Wat er gebouwd moet worden
  + Wat prioritair is
* Product-eigenaar doet commitment om het team met rust te laten tijdens sprint

Afbeelding met tekst, Lettertype, schermopname

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving**Rollen: Scrum-master:**

* Bewaakt het proces
  + Scrumconcepten toepassen
  + Kwaliteitscontrole
* Zorgt voor continuous improvement

Afbeelding met tekst, Lettertype, schermopname, Elektrisch blauw

Automatisch gegenereerde beschrijving

Wat doet de Scrum-master niet?

* Taken bepalen
* Taken toewijzen
* Prioriteiten bepalen
* Schattingen bepalen
* Problemen oplossen
* Verantwoordelijkheid nemen of ontvangen
* Wie doet dit wel?

Rollen: Team:

* Typisch 5 à 9 mensen
  + Hoe meer mensen hoe groter de communicatie overhead
  + Indien groter team nodig; werken met meerdere teams
* Leden zouden full-time aan het project moeten werken
  + Uitzonderingen kunnen bestaan bij ondersteunende functies (vb. Systeembeheerders, …)
* Teams zijn zelforganiserend
  + Leden brengen elk hun eigen achtergrond aan, maar zijn multidisciplinair, delen hun kennis en werken mee aan domeinen waar zij geen specialist op zijn

Wat is een sprint?

* Scrum projecten worden uitgevoerd in een aantal “sprints”
  + Analoog aan XP iteraties
* Doel = **twee/drie weken** 
  + Een constante duur leidt tot een beter ritme
* Product wordt **ontworpen, geschreven en getest tijdens de sprint**

Overzicht van een sprint:

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving

Voorbereiding sprint:

* Aantal features of taken worden opgemaakt door Product-eigenaar en de klant
* Oplijsting in een document **(product backlog)**
* **Uniek geprioriteerd** (dus niet in blokken)

Sprint planning deel 1:

* **Ruwe inschatting** van aantal items dat opgenomen zal worden tijdens volgende sprint
* Inschatting **door team**, niet door Product-eigenaar
* Product-eigenaar kan antwoorden op alle vragen van het team
  + Indien niet mogelijk => team neemt item niet op
* Bepalen van een sprintdoel

Sprint planning deel 2:

* Backlog items worden opgesplitst **door team**
* Team is zelf-organiserend rond hoe het sprintdoel te bereiken
* **Geen managers** 
  + Manager kent geen taken toe aan individuen
  + Managers maken geen beslissingen voor het team
* **Sprint Backlog** wordt gemaakt
  + Taken van **4-16 uur**
* (Bepalen van een sprintdoel)

Schattingen:

* Planning poker
* Uren of punten?
  + Sprint schatting en/of nieuwe teams: uren
  + Backlog schatting: punten
* Door mensen die het werk zullen gaan doen
  + Niet door de mensen die het werk geven
  + Opgelet voor **anchoring**!
* Liefst verbale communicatie in plaats van gedetailleerde neergeschreven specificaties

Anchoring:

Afbeelding met tekst, schermopname

Automatisch gegenereerde beschrijving

Stand-up:

* Parameters
  + **Dagelijks**
  + **15 minuten**
  + **Stand-up** = rechtstaan !
  + Niet om problemen op te lossen
* **Drie vragen** (rond het bord?)
  + Wat deed je gisteren?
  + Wat ga je vandaag doen?
  + Welke obstakels liggen in je weg?
* Waarom dagelijks?
  + “Hoe kan een project een jaar te laat zijn?”
    - “Een dag per keer.” (Fred Brooks, The Mythical Man-Month)
* Kan Stand-up vervangen worden door status rapporten via mail (**neen**)?
  + Volledig team ziet elke dag volledig beeld
  + Creëer sociale druk: doen wat je zegt
  + Wanneer mensen niet elkaar face-to-face zien, zullen ze hun problemen op elkaar projecteren

Demo/sprint review:

* Team stelt voor wat het bereikt heeft gedurende sprint: **potentieel productie-waardige software**
* Typisch **een demo** van het product of van de onderliggende architectuur
* **Informeel** 
  + 2-uur voorbereidingstijd
* Deelnemers
  + Klanten
  + Management
  + Product-Eigenaars
* Hoe valideren?
  + Door testen opgesteld tijdens product planning
* Wat is af?
  + **0 known defects!**
  + Geef nooit toe op kwaliteit
  + **Klaar voor IT of klaar voor Business?**

Retrospective:

* Deelnemers
  + Team
  + Scrum Master
  + Product-eigenaar (optioneel)
* Vragen
  + **Wat ging goed gedurende de laatste sprint?**
  + **Wat kan verbeteren in de volgende sprint?**
* Resultaat
  + Altijd acties bepalen
  + Indien invloed op Product-eigenaar, toevoegen aan Product Backlog
  + Opgelet: moeten business value hebben

Retrospective: tips:

* Geen beschuldigingen!
* Gebruik nooit het woord: “JIJ”
* Gebruik een “spreekstok”

Documenten:

* Scrum heeft heel weinig documenten
  + **Product Backlog**
  + **Sprint Backlog**
  + **Burndown Statistieken**
* Kan gebeuren door een Excel spreadsheet
* Meer geavanceerde / ingewikkelde tools bestaan
  + Mingle
  + Conchango (Scrum for Team Foundation Server)
  + Trello
  + Scrumwise
  + Jira
  + Version One

Afbeelding met tekst, software, nummer, Webpagina

Automatisch gegenereerde beschrijvingBacklog:

* Een lijst van **al het gewenste werk** van het project
* Meestal een combinatie van
  + Story-based werk (“laat gebruiker zoeken en vervangen”)
  + Task-based werk (“verbeter exception handling”)
* Lijst wordt **geprioritiseerd door Product-eigenaar**
  + Typisch een Product Manager, Marketing, Interne klant, enz.

Sprint backlog:

* Werkdocument van het team
* **Opvolging van individuele taken**
* Invullen van **wat nog te doen valt** om zo te zien en **te corrigeren om het doel te behalen** – en dit **iedere dag** opnieuw
* Groot verschil met de klassieke worksheets waarin na de feiten tijden worden ingevuld en worden vergeleken met de schatting
* Gebruik een **bord** (story/tasks/work in progress/verify/done)

Afbeelding met tekst, lijn, Lettertype, Perceel

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, lijn, Lettertype

Automatisch gegenereerde beschrijving

Afbeelding met tekst, diagram, Lettertype, lijn

Automatisch gegenereerde beschrijvingAfbeelding met tekst, Lettertype, diagram, lijn

Automatisch gegenereerde beschrijvingAfbeelding met tekst, schermopname, diagram, Parallel

Automatisch gegenereerde beschrijving

Afbeelding met tekst, Lettertype, diagram, lijn

Automatisch gegenereerde beschrijving

Afbeelding met tekst, diagram, Plan, schermopname

Automatisch gegenereerde beschrijving

Backlog testen:

* Je moet eerst een witte categorie volledig afwerken
* Je mag kleine taken van andere categoriëen, maar niet allemaal

## Kanban

* Kanban
* Geen rollen
* Visueel bord
* Laagdrempelig
* Optimalisatie (Lean)
  + JIT
  + Aantal bottlenecks beperken
  + WIP limit
* Gebruikt in productie-omgeving/systeembeheer
* Afbeelding met tekst, diagram, schermopname, lijn

  Automatisch gegenereerde beschrijvingAgile

Afbeelding met tekst, schermopname, diagram, lijn

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermopname, diagram, software

Automatisch gegenereerde beschrijving

# Sjabloon van een use case:

## Use Case

**Use case:** <naam van de use case>

**Primaire actor:** <primaire actor>

**Stakeholders:** <stakeholders>

**Precondities**: <preconditie>

**Postcondities**: <postconditie>

**Normaal verloop**

1. <actiestap>

2. ...

**Alternatieveverlopen:**

1A. <alternatieve paden>

1A1

**Domeinspecifieke regels:**

DR\_<naam domeinregel><gedetailleerde omschrijving domeinregel>

**Op te klaren punten:**

• <op te klaren punten >

•...

## Niet-Functionele requirements

|  |  |
| --- | --- |
| **NFR** | *Categorie NFR* |
| **Indicator** | *Indicator* |
| **Meetvoorschrift** | *Meetvoorschrift (hoe wordt het gemeten)* |
| **Norm** | *Norm (waar het moet aan voldoen)* |