

Analyse III

1. Modelleren van processen

Processen ontdekken

- Opzetten van het project
 - samenstellen van het team
- Informatie verzamelen
 - het bekomen van een goed begrip van het proces
 - elicitatietechnieken
- Begeleiden van de modelleertaak
 - Creëren van het procesmodel
 - Modelleermethode
- Kwaliteitsborging
 - De procesmodellen beantwoorden aan de vooropgestelde kwaliteitscriteria
 - Vertrouwen creëren in de procesmodellen voor alle stakeholders

! Proces-instance = een unieke passage doorheen een proces: voor één bepaald klant, leverancier, artikel, enz. VB. Proces "boeken van een reis"

- "Elke reis is verschillend"
- "Er zit niet echt een lijn in: onze klanten gaan naar verschillende bestemmingen, in verschillende seizoenen met verschillende transportmiddelen"
- "We doen nooit iets op dezelfde manier; er zijn zoveel speciale voorwaarden"

Expertise van procesanalisten

- Problemen begrijpen
 - Kennis van het probleemdomein
 - De kennis van de organisatie helpt om het probleem te structureren.
- Problemen oplossen
 - Identificatie van de procestriggers
 - Hypothese-beheer: formuleren en testen van hypothesen
 - Uitzetten van doelstellingen: wat is de volgende stap
 - Top-down strategie gebaseerd op de doelstellingen
- Modelleerskills
 - Goed gestructureerd, duidelijke layout
 - Systematische labels
 - Expliciete start- en eindpunten
 - Gepaste granulariteit en decompositie (subprocessen)

Elicitatietechnieken

- **Creatieve technieken**

- Brainstorm
 - **nadelen**
 - ervaren moderator nodig
 - sommige deelnemers veel dominanter
 - **alternatief** brainwriting 6-3-5
 - iedereen komt aan bod
 - Werkt verder om andere ideeën
 - Duurt niet lang
- Invalshoek veranderen
 - Je gaat in een soort van 'rol', je denkt niet aan je eigen mening maar aan de mening van de 'hoed'.
- Gebruik maken van een analogie
 - Je komt los van de huidige situatie
 - stimuleren creativiteit

- **Uitvraagtechnieken**

- Interviewen
 - Gestructureerd vs. ongestructureerd
 - Combinatie van open & gesloten vragen
 - Veronderstelling: analist en stakeholder verstaan elkaars terminologie
 - **LSD** Luisteren, Samenvatten en Doorvragen
 - Luistervaardigheden:
 - Empathie
 - Analyse
 - Synthese
 - Geen manipulatieve vragen
 - Goed voorbereiden
 - Informatie nodig over persoon, bedrijf, ...
 - Begin met neutrale vragen (afwijken mag, maar hou de focus)
 - **Veel prater** gesloten of half open vragen
 - **Niet-prater** open vragen
 - **Nadelen**
 - Neemt veel tijd in beslag door feedbacksessies (LSD)
- Enquête houden
 - fysieke of digitale vragenlijst
 - Open of gesloten vragen (meerkeuzevragen) of combinatie
 - eventueel anoniem
 - **Voordelen**
 - Veel informatie in korte tijd
 - Snelle verwerking bij gesloten vragen
 - Geen remmingen indien anoniem (maar: anonimiteit is moeilijk in de praktijk!)
 - **Nadelen**
 - deelnemers kunnen vragen verkeerd interpreteren
 - Je mist 'body language' = non verbale communicatie

- **Observatietechnieken**

- Veldobservatie
 - Gebruiker werkt in bijzijn van analist
 - varianten

- geen interactie
 - gebruiker legt uit wat hij doet
- Werkstage
 - Analist voert zelf het werk uit
 - zeer tijdrovend maar levert diepgaande kennis op
- **Documentatie-georiënteerde technieken**
 - Systeemarcheologie
 - Lezen vanuit specifiek oogpunt
 - Documenten verwijzen naar rollen, activiteiten business-objecten
 - Formele documentatie
 - Organigram van de organisatie
 - Tewerkstellingsplannen
 - Kwaliteitsrapporten (bij. bij certificaties), auditrapporten
 - Woordenlijsten en handboeken
 - Workinstructies
 - Hergebruik van requirements
 - eerder uitgevoerde analyses
 - kunnen tijd en kosten voor analyse sterk reduceren
- **Ondersteunende technieken**
 - Mind mapping
 - Workshops
 - breng alle key-stakeholders samen
 - Deelnemers discussiëren om zo een gedeeld begrip te creëren
 - Dikwijls software-ondersteund
 - Afzonderlijke rol naast moderator
 - Tijdens workshop worden modellen getekend
 - Model dient als referentie voor verdere discussies
 - CRC-kaarten
 - **Class Responsibility Collaboration**
 - Tijdens workshop worden relevante business-objecten op **kaarten** geschreven
 - Voorbeeld: bestelling, product, klant
 - Workshopdeelnemers voegen daar **eigenschappen** aan toe
 - Kaarten worden gebruikt om processen en requirements in kaart te brengen
 - Audio en video opnamen
 - Gebruikt bij veldobservatie, interviews en workshops
 - **Nadeel**
 - deelnemers gaan zich misschien anders gedragen
 - Use cases
 - Zijn elementaire bedrijfsprocessen: **1** persoon, **1** tijdsspanne, **1** plaats
 - Maken deel uit van een groter geheel
 - Helpen om het elicitatieproces te **structureren**
 - Prototypen
 - = Werkende software van kritieke delen van een toekomstig systeem

Techniekkeuze

- Elke techniek heeft voor- en nadelen
- **! Combinatie** van technieken is nodig
- Houd bij de keuze rekening met
 - Menselijke aspecten

- Communicatieve en persoonlijke vaardigheden van stakeholders
- Ervaring met bepaalde technieken
- De mate waarin stakeholders zich beust zijn van requirements
- Organisatorische aspecten
 - **Beschikbaarheid** belanghebbenden
 - Weinig tijd => verkies veldobservatie boven interviews
 - Beschikbare **budget** en **doorlooptijd**
 - workshop is tijdsbesparen
 - Moeilijk bij geografische spreiding van stakeholders
 - Creatieve technieken minder gangbaar bij fixed price / fixed date projecten
 - kies bij vervanging van bestaand systeem voor documentatiegebaseerde technieken
- Vakinhoudelijk aspecten
 - vereiste detailniveau beïnvloedt keuze
 - ervaring analist met bepaalde technieken

Invloed van de bedrijfscultuur

Open Cultuur

Waar alle medewerkers aangemoedigd worden om hun ideeën en kritiek te uiten

maak gebruik van workshop aangezien de deelnemers gewoon zijn om hun ideeën te spuiens

Strict-hiërarchie organisaties

- draag er zorg voor dat iedere stakeholder in gelijke mate aan bod komt
- zorg ervoor dat ideeën en kritiek niet achtergehouden worden. Kies bijvoorbeeld voor anonieme enquêtes als aanvullende techniek.

2. Documentatie technieken

Functionele requirements beschrijven

- Use Cases (Analyse II)
- User stories
- Use cases 2.0 (use case slices)

User Stories

- Korte beschrijving
- Wat een gebruiker wil
- Gewone spreektaal
- Past op een post-it
- Begrip binnen agile software development
- 'wie', 'wat', 'waarom'

As a <role>
I want to <goal/desire>
So that <benefit>

Als een <rol>
Wil ik <doel>
Zodat <voordeel>

Waarom user stories?

- Verbale communicatie stimuleren/forceren
- Snelle feedback
- Face-to-face communicatie
- Geen technisch jargon
- Planning faciliteren
- Detailleer waar en wanneer nodig (korte termijn versus midellange termijn)

User stories schrijven

Een goede user story is

- Independent (Onafhankelijk)
- Negotiable (Onderhandelbaar)
- Valuable (Waardevol voor gebruiker en opdrachtgever)
- Estimatable (Schatbaar)
- Small (Klein)
- Testable (Testbaar)

! INVEST

Tips

- Identificeer de stories
 - Start met vastleggen van de doelstellingen
- "Slice the cake"
 - Bij het schrijven (opsplitsen) van user stories aandacht dat alle lagen van de applicatie voorkomen
- "gesloten" stories
 - Gesloten story eindigt met een user's goal
- Leg beperkingen vast
- Focus op de belangrijke zaken in de nabije toekomst
- GUI zo lang mogelijk uitstellen
- Gebruik "user roles" in de story
- Schrijf voor 1 user
- Schrijf in actieve taal
- Laat de gebruiker/opdrachtgever meeschrijven
- User stories worden niet genummerd
- Zijn niet gebonden aan IEEE-guidelines
- Zijn geen use cases
 - Scope
 - Volledigheid
 - Levensduur
 - Doel

Story map

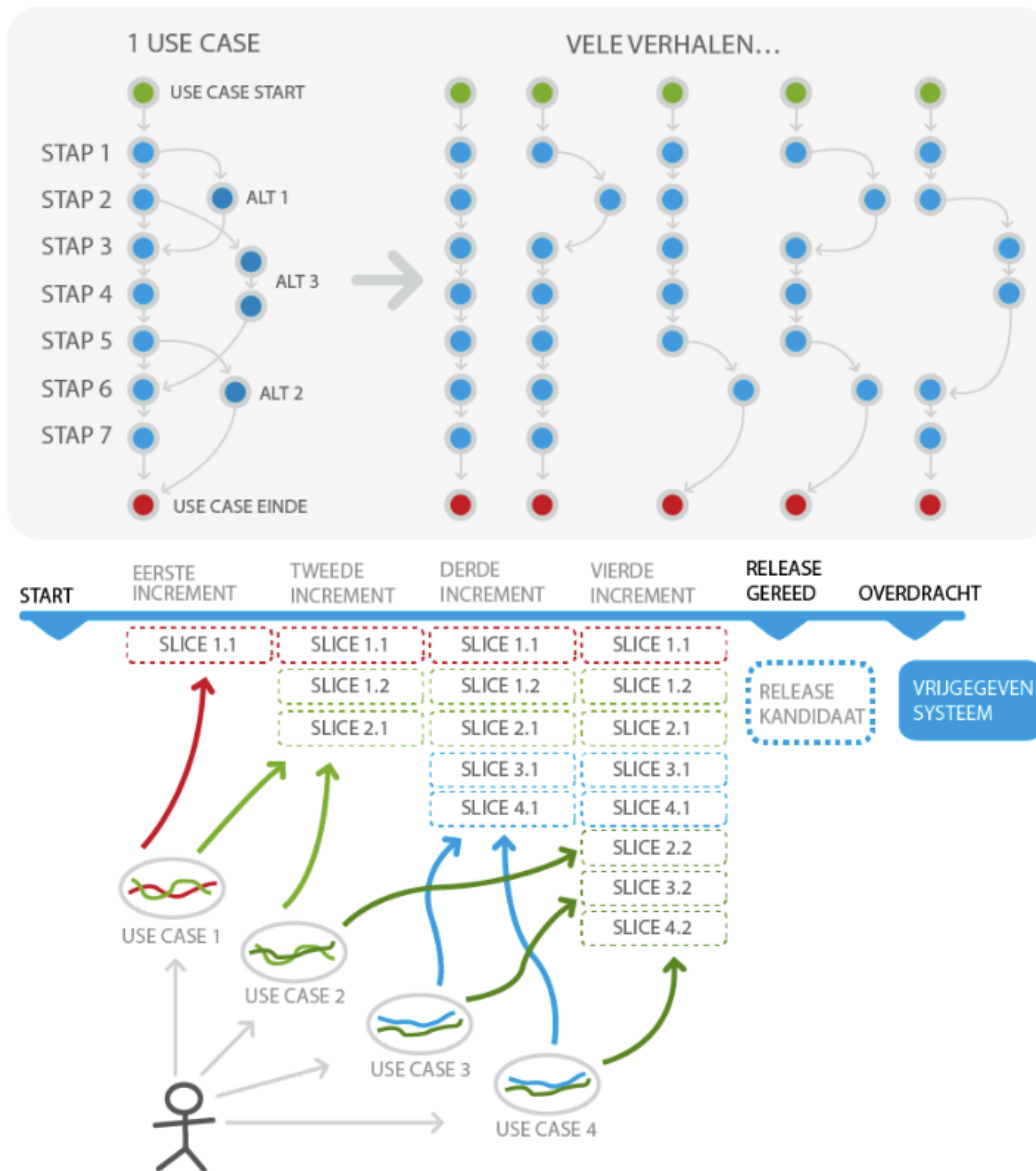
Een story map is een voorstelling van de user stories volgens prioriteit (hoog , laag, verticale as) en functionaliteit (van links naar rechts, horizontale as)

Story map: voorbeeld webshop



Use Cases 2.0 (Use case slices)

- Houd de informatie-uitwisseling **simpel** door het vertellen van verhalen (gebruik van stories)
- Overzie en begrijp het **grote geheel**
- **Focus** op de **(business) value**
- Bouw het systeem in **use case slices**
- Lever het systeem **incrementeel** op
- **Flexibiliteit**, ieder project kent eigen karakteristieken en vergt daarom een eigen aanpak



Stappenplan

- Beschrijf de **actoren** en de **use cases**
- **Verdeel** de use case in **use case slices**
- Voorbereiden van de use case slice
- Analyseer de use case slice
- Implementeer de software (voor een slice)
- **test het systeem** (voor een **slice**)
- **test het systeem in zijn geheel**
- **inspecteer en wijzig de use cases**

3. Wat is een proces

- 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

Business Process Management vs Business Analyse

BPM

- Gericht op bedrijfsorganisatie
- Meestal voortraject van IT-project

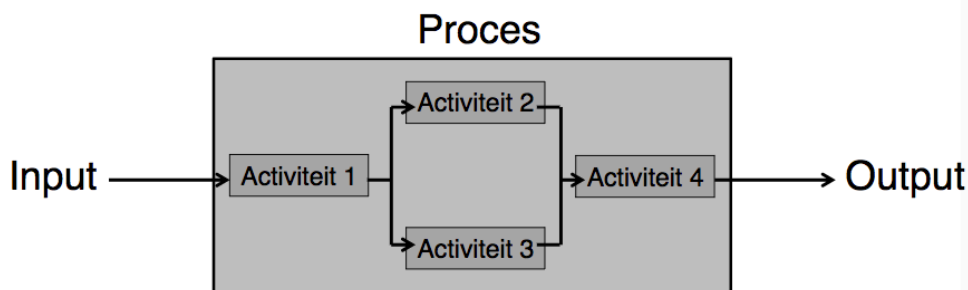
Business Analyse

- Vertrekt van resultaat BPM-oefening
- Gericht op IT-project
- Tijdens Business Analyse blijkt vaak nood aan procesoptimalisatie

Wat is een proces? Waarom BPM?

1. Definitie van een proces

- Transformatie van input naar output
- Creëren van waarde voor afnemer of klant (= belanghebbenden) **BUSINESS VALUE**
- Realiseren van doelen
 - Bepaald door verwachtingen van de afnemers of belanghebbenden
-



opmerkingen:

- Input in een proces is in het algemeen output van andere processen.
- Processen in een organisatie worden in het algemeen gepland en uitgevoerd onder beheerste omstandigheden om waarde toe te voegen.

2. BPM (Waarom? Hoe?)

Waarom?

- Om je proces goedkoper te doen
- Om je proces sneller te doen
- Om je proces beter te doen

Hoe?

- Continuous Process Improvement (CPI)
 - Stelt de huidige processtructuur niet in vraag
 - Identificeert problemen en lost ze één voor één op. Stap voor stap
- Business Process Re-Engineering (BPR)
 - Stelt de fundamentele veronderstellingen en principes van de huidige processtructuur in vraag.
 - Gericht op het realiseren van een doorbraak, bijv. door dure taken zonder directe toegevoegde waarde te elimineren.

- "Hoe zit een proces in elkaar" momenteel -> **AS-IS**
- Reengineering Process -> **TO-BE**

3. Verklaring van enkele begrippen

Begrip	Uitleg
input	Iets dat getransformeerd, gebruikt, verwerkt wordt (klantenvraag, te behandelen dossier, ...)
output	Iets dat geproduceerd wordt. (attest, antwoord, behandeld dossier, ...)
beheersing/besturing	Hoe en wanneer een proces/activiteit plaatsvindt; wordt niet verwerkt of gebruikt (een instructie, richtlijn, doelstelling, ...)
middelen	Personen, systemen, tools, uitrusting, activa, ...; worden niet verwerkt of gebruikt -> worden <i>gebruikt</i>

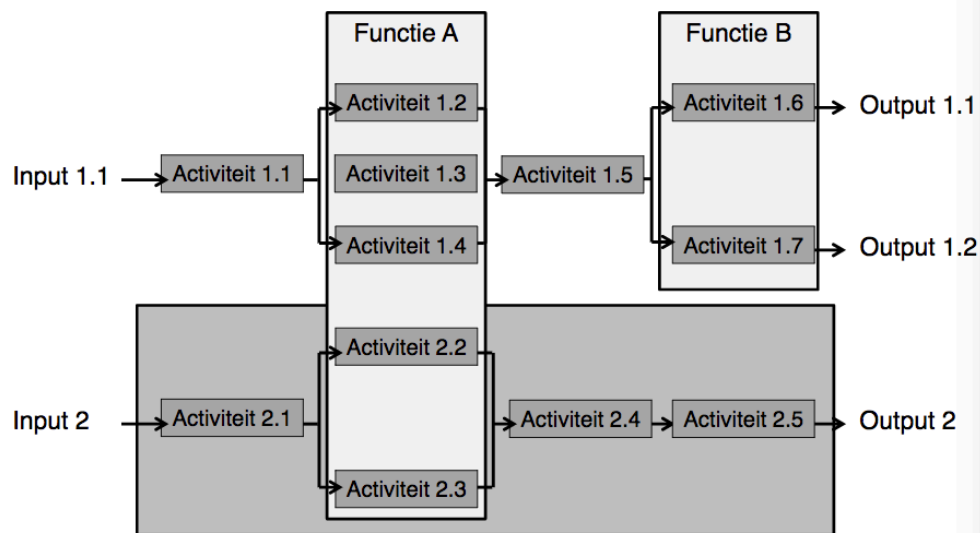
4. End-to-End

- Proces begint bij de behoefte van externen klant of afnemer
 - Klant of afnemer triggert het proces
- Proces eindigt bij dezelfde klant/afnemer
 - Trigger is volledig beantwoord

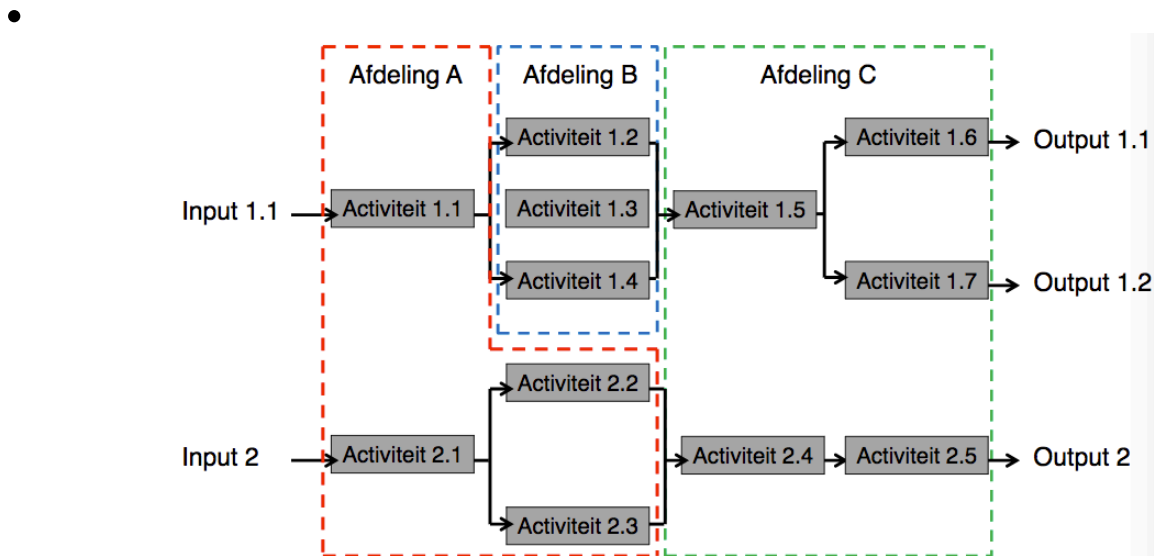
- Één of meerdere outputs

Proces vs functie

- Proces
 - Verzameling activiteiten
 - Gericht op het bereiken van één of meerdere outputs
- Functie
 - Verzameling activiteiten
 - Gegroepeerd volgens competentie
-



Proces vs. afdeling



Belang van processen

- Processen = hart van een organisatie
- Processen lopen vaak fout bij transfer tussen
 - Andere persoon

- Andere afdeling
- Andere instantie

2 belangrijke dimensies in organisatie

1. Strategische dimensie (de juiste dingen doen) **Effectiviteit**
 - Inspanningen van een organisatie moeten gericht zijn op het verbeteren van het vermogen om waarde te creëren
 - Kosten vs baten van het resultaat afwegen
 - 1e stap (in bpm): **identificeren van de 'waarde creërende stromen'**
2. Operationele dimensie (de dingen goed doen) **Efficiëntie**

Creëren van waarden

- Gebeurt via **processen**
- Niet alle processen gericht op waarde creëren
 - Niet alle processen zijn evenwaardig
 - Vaak sappen zonder toegevoegde waarde
 - identificeren van deze processen = 1e stap

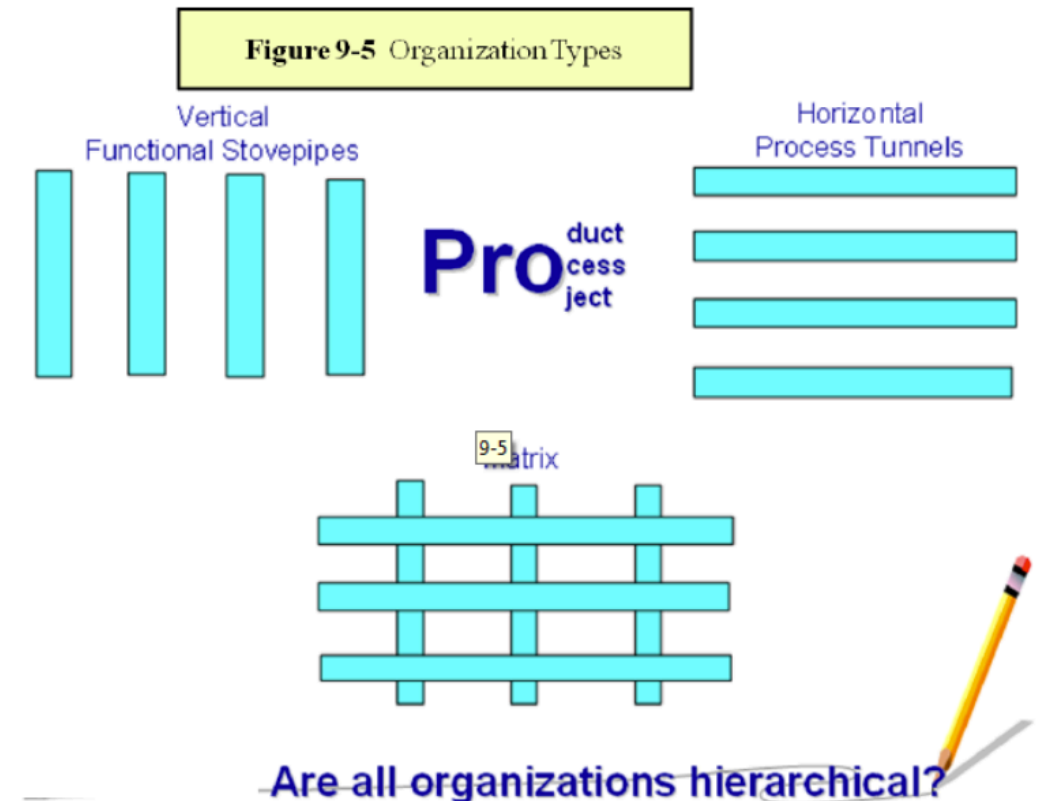
3 clusters van processen

1. **Primaire processen (core business processen)**
 - Processen die nodig zijn om de behoeften van de externe klanten te vervullen
2. **Ondersteunende processen (support business processen)**
 - Processen die de primaire processen ondersteunen
3. **Sturende processen (management processen)**
 - Processen die nodig zijn om de organisatie te besturen teneinde te voldoen aan de doelstellingen en aan de wet- en regelgeving.

	Sturend	Primair	Ondersteunend
Afhandelen van een klacht van een klant (van ontvangst tot antwoord)			x
Goedkeuren van binnenkomende facturen bij een stadsbestuur			x
De aanwerving van nieuwe medewerkers in een call center			x
Organisatie van de bachelorproef aan een hogeschool		x	
Agile project management	x		

Problemen bij processen:

- Functioneel georganiseerde organisaties
 - Functie/Afdeling = verticaal
 - Proces = horizontaal
- Dalende doeltreffendheid door groei
 - Groei leidt tot specialisatie: men wordt efficiënter op een bepaald domein maar men verlist het einddoel, de klant, uit het oog
- Gewoontevorming
 - Men stelt niet meer in vraag wat men doet en hoe men het doet



Vericale functionele "kachelpijpen"

- Elke kachelpijp stelt een **specifieke functie of dienst** voor:
 - IT, marketing, financiën, HR, ...
- Komt voor in **hierarchische, gecentraliseerde organisaties**
- **Expertise** wordt **gedeeld** over **gans het bedrijf**
- **Duidelijke carrière-pade en opleidingsprogramma's**
- Voor elke functie bestaan **backups**
- Managers zijn **vertrouwd** met het werk van hun ondergeschikten
- **Standaarden** kunnen gemakkelijk gehanteerd worden

NADELEN

- Een **eenheid op zich**
- **Focus** op de individuele dienstverlening (bv.: IT) i.p.v. op de dienstverlening aan de klant
- Processen zijn gericht op efficiëntie (de dingen juist doen), niet op effectiviteit (de juiste dingen doen voor de organisatie)
- **Communicatieproblemen** (eigen jargon)

- Business prioriteiten kunnen **afwijken** van de functionele prioriteiten
- Opgeleverde projecten **voldoen niet aan de noden** van de business units

Horizontale process-tunels

- Komt voor in **gedecentraliseerde** organisaties
- Units worden gecreëerd om te focussen op een **bepaald business domein**
- Prioriteiten zijn gebaseerd op de noodzaak van producten en processen
- De **communicatie** tussen de diensten is **veel beter**
- Het personeel is op de hoogte van diverse aspecten van de producten en processen

NADELEN

- De perceptie: opwaartse carrierepaden beperkt
- Jobevaluaties gebeuren vaak door leidinggevenden die de job zelf niet kennen
- Vaak weinig backup-personeel
- Weinig synergie in het gedrag
 - Iedere business unit werkt op zijn eigen manier, wat kan leiden tot redundantie en inefficiëntie

5. Starten met procesverbetering

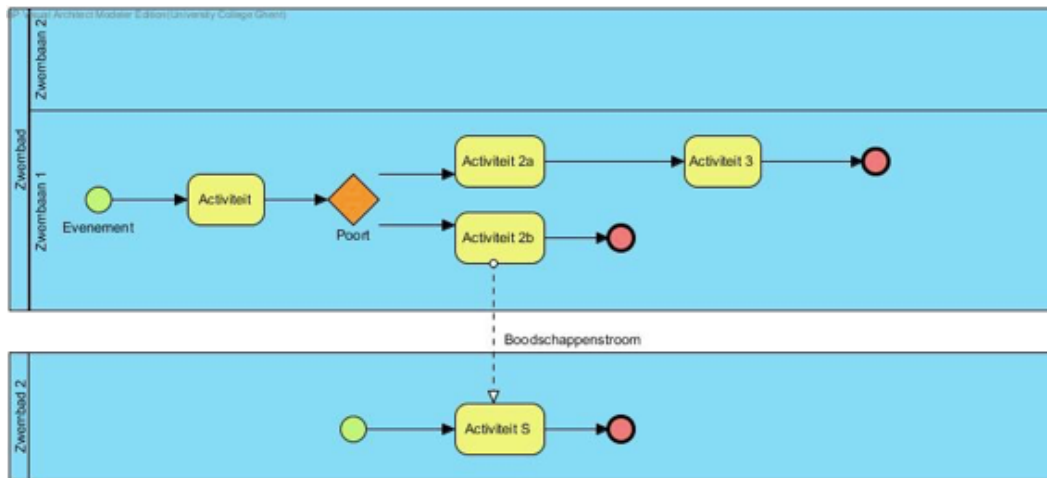
"Kritische" of sleutelprocessen

- Staan in functie van **de strategie van een organisatie**
- Dragen bij tot het **voldoen aan de behoeften van de belanghebbenden**
- Zijn de sleutel tot het **success of continuïteit van de organisatie**
- Worden bepaald op basis van **kritische successfactoren**

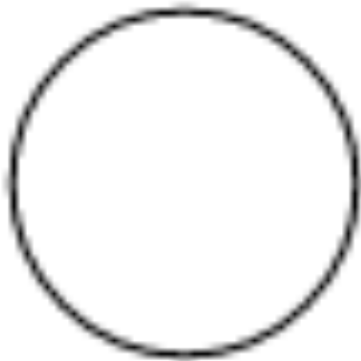
Proces \ Succesfactor	Verhogen klanttevredenheid	Versterken competenties	Verhogen marktaandeel	Verminderen voorraad	Verbeteren van ...
Ontwikkelen vna producten / diensten	x				
Opleiden van personeel		x			
Promotie			x		
Leveren van producten / diensten	x			x	



4. Modelling van processen

4 basiscategorieën van elementen in BPMN





- Stroomobjecten (evenementen, activiteiten en poorten)
 - De grafische elementen die het gedrag van een bedrijfsproces definiëren.
 - Soorten:

Naam	Symbool
evenementen, gebeurtenissen (events)	
Activiteiten (activities)	

	
Poorten (gateways)	

- Verbindingselementen (pijlen)
 - Verbinden stroomobjecten met elkaar of met andere informatie.
 - Soorten:

Naam	Symbool
Sequentiële stroom (Sequence Flow)	
Boodschappenstroom (Message Flow)	
Associatie (Association)	


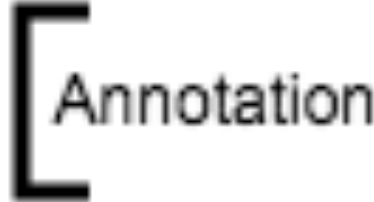


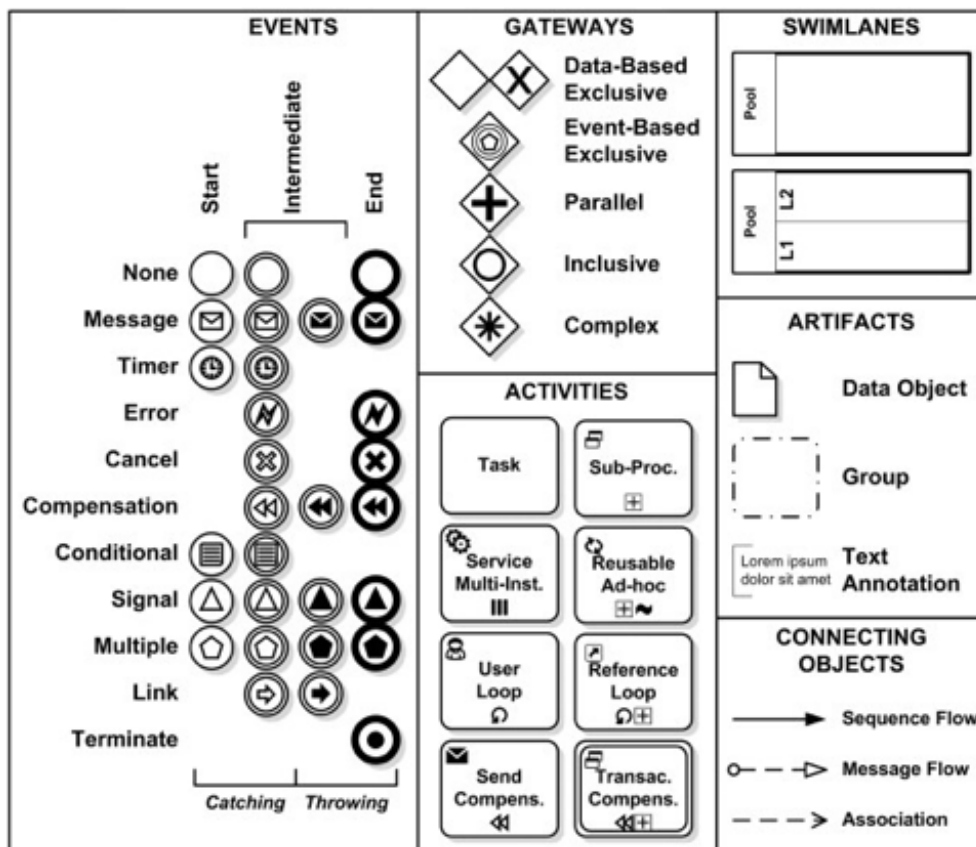
- Zwembaden / zwembanen
 - Groeperen de stroom- en Verbindingselementen
 - Soorten:

Naam	Symbol
Zwembaden (Pools) Zwembanen (Lanes)	

- Artefacten
 - Worden gebruikt om extra informatie over het proces te voorzien
 - Soorten:

Naam	Symbol
Data Object	
Groep (Group)	

	
Annotatie (Annotation)	

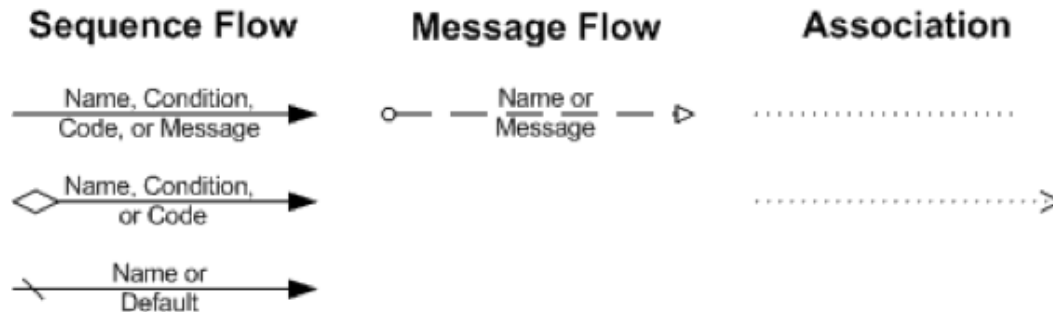


Verbindingselementen

- Sequentiële stroom:
 - Enkel binnen 1 zwembad (met 1 of meerdere zwembanen)
 - Niet tussen zwembaden

- Boodschappenstroom:
 - Enkel tusesn verschillende zwembaden
- Associatie:
 - Verbindt informatie en artefacten met stroomelementen

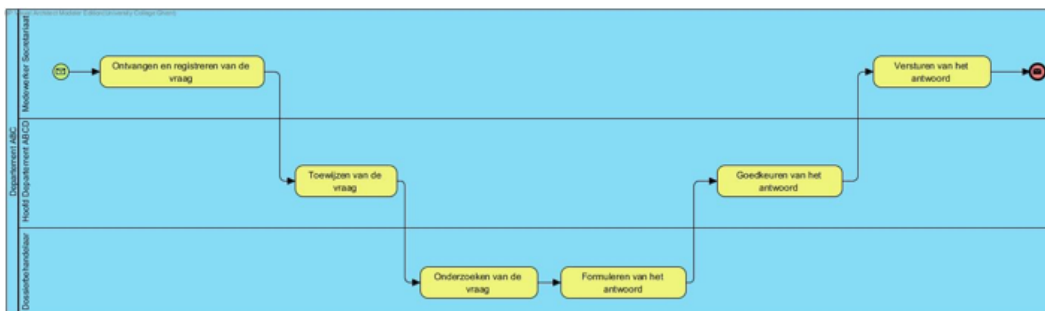
Connections



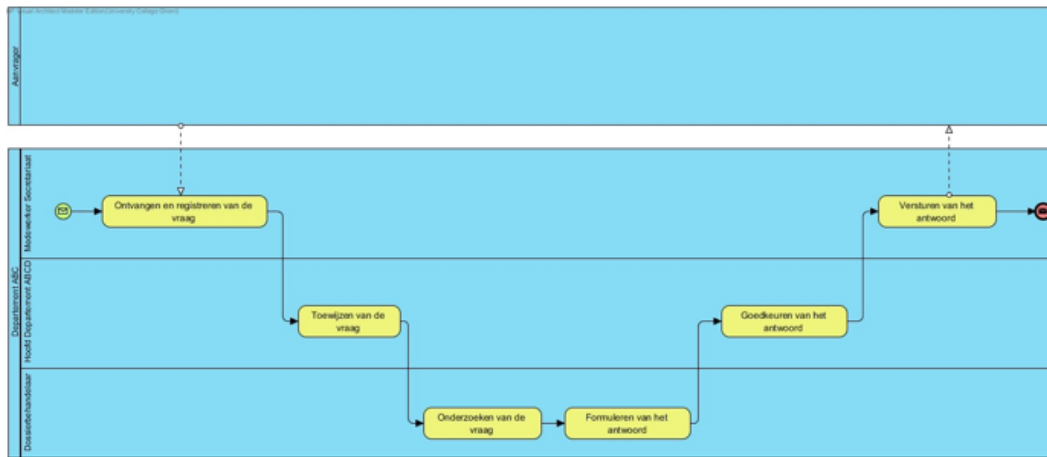
Pools en swimlanes

- Zwembad (Pool)
 - Grafische container voor activiteiten binnen een orkestratie (orchestration)
 - Heeft zicht op haar eigen activiteiten, meestal geen zicht op activiteiten van andere zwembaden (Black Box)
- Zwembaan (Swimlane)
 - Onderverdeling binnen zwembad
 - Over de volledige lengte van het zwembad
 - Horizontaal of verticaal
 - Stellen vaak een rol voor binnen een orkestratie

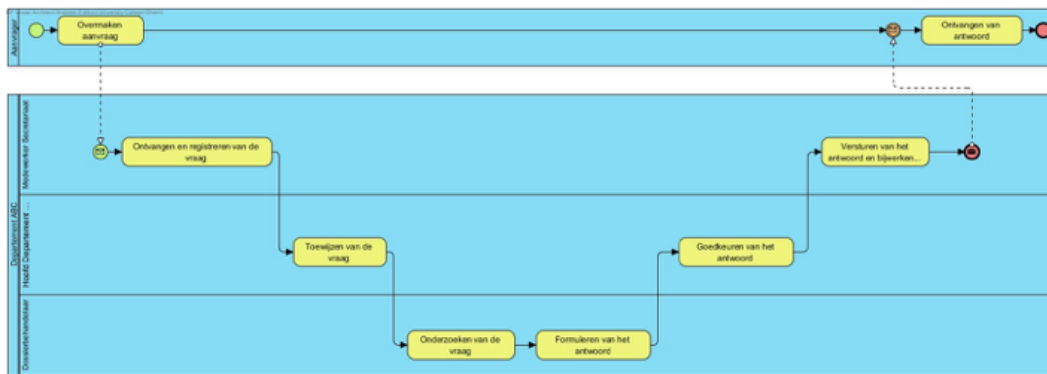
Privaat proces: 1 zwembad



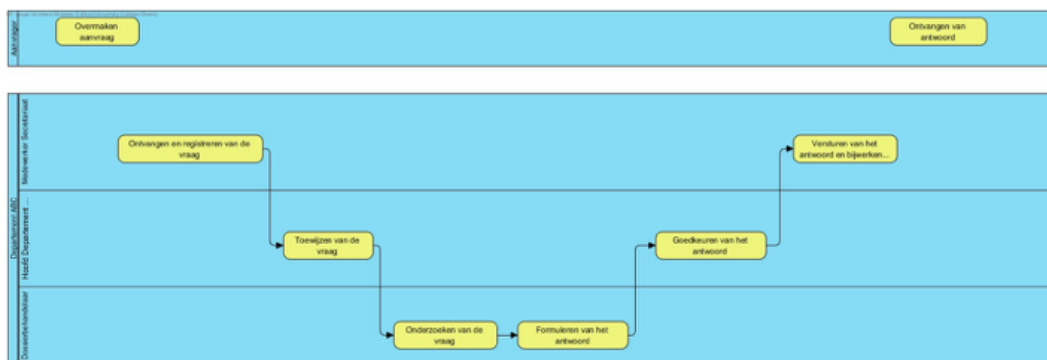
Abstract of publiek proces: lege zwembaden



Globaal proces (collaboration process)

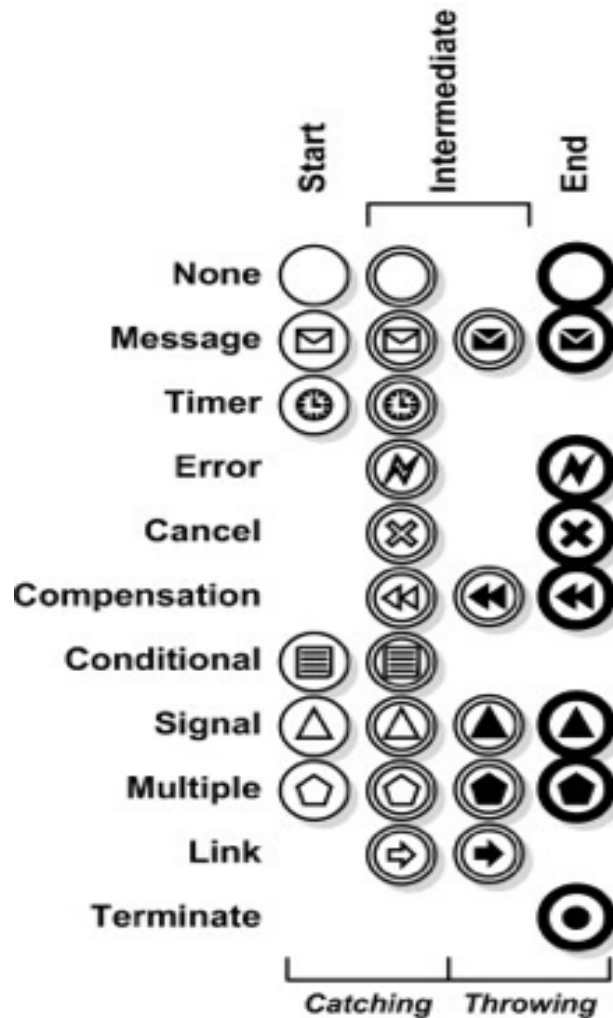


Niet correct gemodelleerd BPMN diagram



Evenementen (Events)

- Gebeurt tijdens verloop van bedrijfsproces
- Beïnvloeden de stroom van het proces
- Hebben een oorzaak (= trigger) of een impact (= resultaat zoals vb. een boodschap)



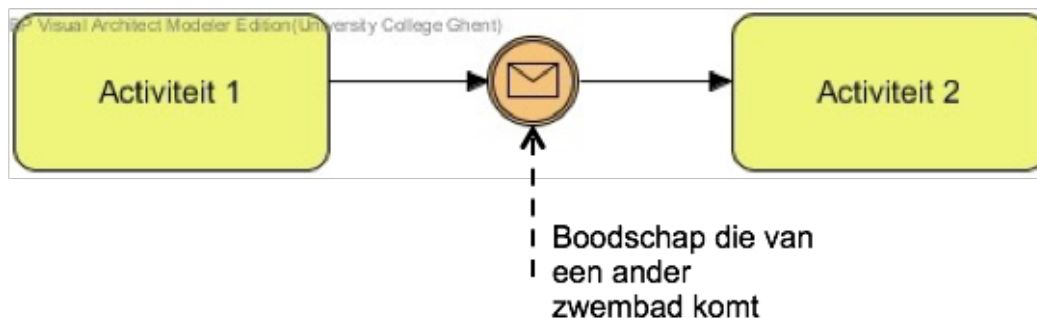
Startevents

Geen		De modelleerder geeft het type startevenement niet weer. Wordt ook gebruikt voor een subprocess dat start wanneer de stroom getriggerd wordt door het moederproces.
Boodschap		Een boodschap (bericht, mail, brief, melding,...) komt van een participant en triggert de start van het proces.
Tijd		Een specifieke tijd/datum of specifieke cyclus (elke maandag om 9.00u) die de start van het proces triggert.
Voorwaar- delijk		Wanneer een voorwaarde (vb. de winst is lager dan x €, indicator hoger dan y) WAAR komt.
Signaal		Een signaal komt aan dat uitgezonden werd door een ander proces. NB: een signaal heeft geen specifiek vooropgestelde bestemming. Een boodschap heeft dit wel.
Meervoudig		Er zijn verschillende manieren om het proces te triggeren. Een van de types is voldoende om het proces te doen starten.

Eindevenementen

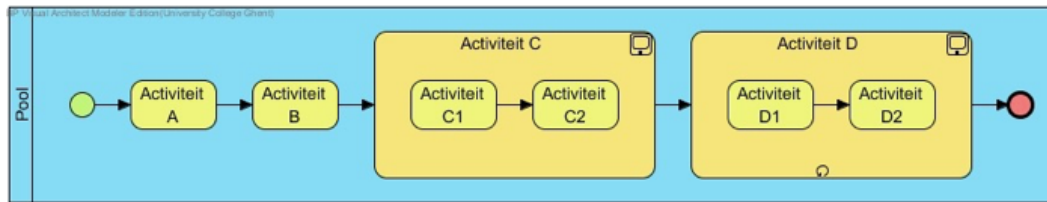
Geen		De modelleerder geeft het type eindevenement niet weer. Wordt ook gebruikt om het einde van een subprocess weer te geven waardoor de stroom naar het moederproces gaat.
Boodschap		Een boodschap (bericht, mail, brief, melding,...) wordt naar een participant gezonden bij het einde van het proces.
Fout (Error)		Een fout wordt gegenereerd en opgepikt door een intermediair fout evenement dat aan de rand van een activiteit weergegeven wordt.
Annulering (Cancel)		Alleen te gebruiken bij transacties (die niet succesvol uitgevoerd zijn)
Compensatie		Geeft aan dat een compensatie nodig is voor een aangegeuide activiteit. Om gecompenseerd te worden moet een activiteit aan de rand van het object een intermediair compensatie evenement hebben.
Signaal		Een signaal wordt uitgezonden wanneer het einde van het proces bereikt is. NB: dit signaal kan uitgezonden worden doorheen de procesniveaus of de zwembaden
Einde (Terminate)		Geeft aan dat alle activiteiten van het proces onmiddellijk moeten beëindigd worden (zonder compensatie of evenementafhandeling)
Meervoudig		Er zijn verschillende manieren om het proces te beëindigen.

Belangrijkste intermediaire evenementen in BPMN



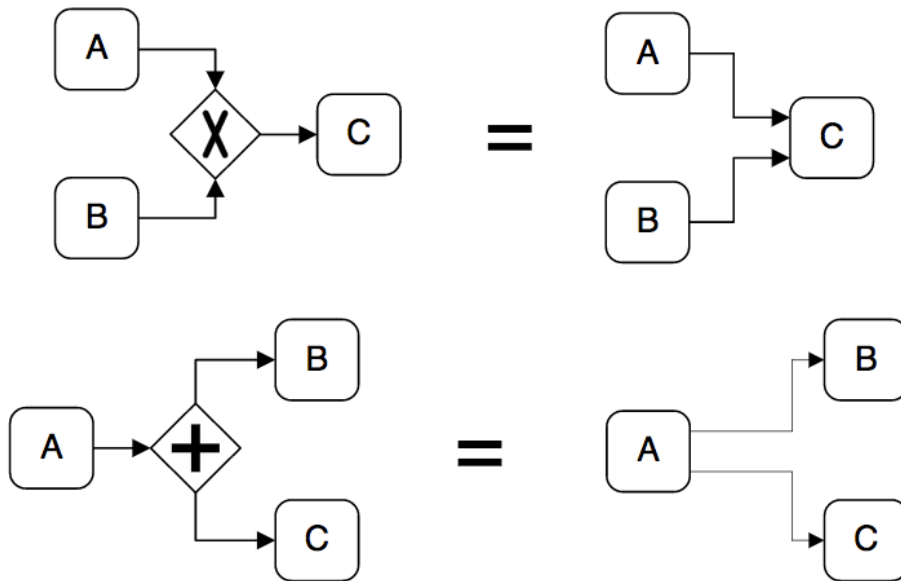
Activiteiten

- Generische term voor werk dat organisatie uitvoert
- Kan atomisch of samengesteld zijn



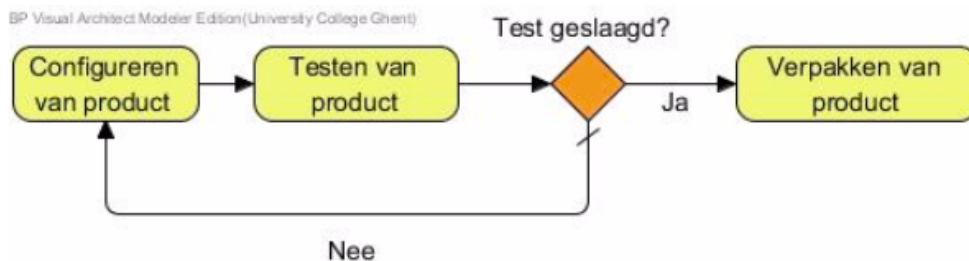
5. Modelling van processen (Deel 2)

Expliciet vs impliciet

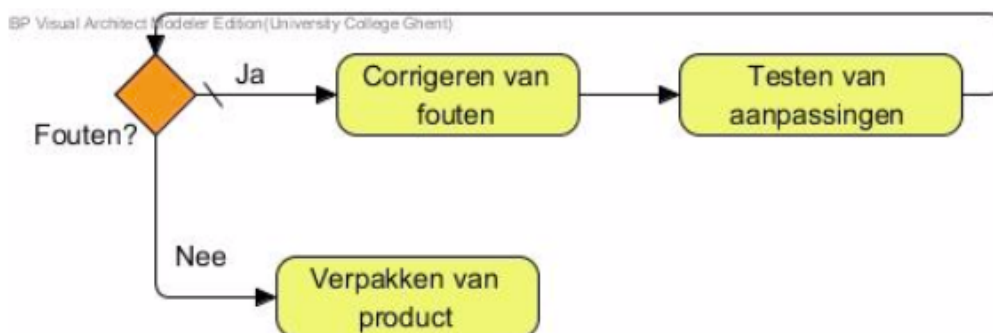


Loops

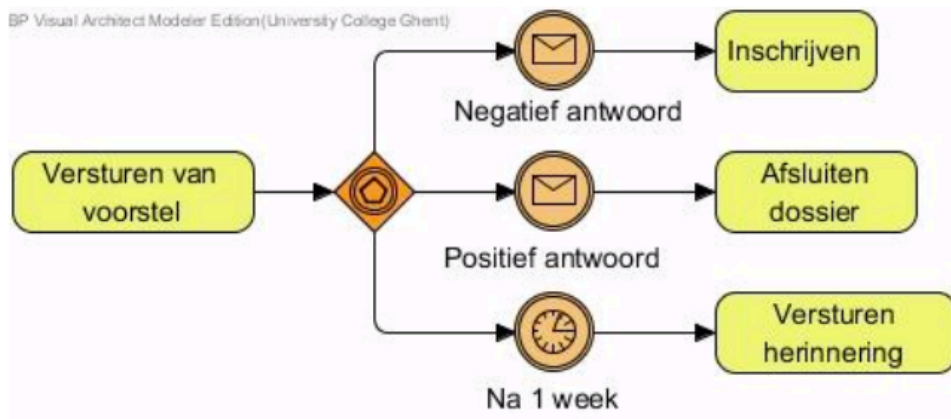
"repeat" loop ("repeat until" or post-test loop)



"while" loop ("while do" or pre-test loop)



Evenement gedreven poort



6. eXtreme Programming

XP is a lightweight, efficient, low-risk, flexible, predictable, scientific way to develop software

Mogelijke risico's

"schedule slips"

XP voorziet:

- Korte "release"-cyclus
- "1 tot 4 weken" -iteraties met de klant
- taken: 1 tot 3 man-dagen

"project canceled"

XP Vraagt de klant om de kleinste release met het grootste zakenaandeel.

Belangrijke zaken eerst

"system goes sour"

XP Voorziet een batterij testen die bij elke wijziging wordt uitgevoerd (soms verschillende keren per dag)

Testen

"defect rate"

XP stelt dat software ontwikkeld wordt door een team

Pair programming

"business misunderstood"

XP Stelt dat de klant/gebruiker een volwaardig lid is van het team

Feedback

"business changes"

XP stelt dat door korte releases je snel kan inspelen op veranderingen

Flexibiliteit

"false feature rich"

XP Stelt dat enkel opdrachten met hoge prioriteiten uitgevoerd worden

Belangrijke taken eerst

"staff turnover"

XP Vraagt de programmeurs om hun verantwoordelijkheid op te nemen

Motivatie en duidelijkheid

XP Belooft

XP Belooft programmeurs dat zij:

- Kunnen werken aan **belangrijke zaken**
- **Niet alleen** moeten werken
- **Verantwoordelijkheid** krijgen om beslissingen te nemen

XP Belooft gebruikers en managers dat zij:

- **Snel resultaat** zullen zien
- De mogelijkheid krijgen om te **wijzigen** zonder extra financiële middelen

Hoe?

4 waarden:

- Communicatie
- Eenvoud
- Feedback
- Moed

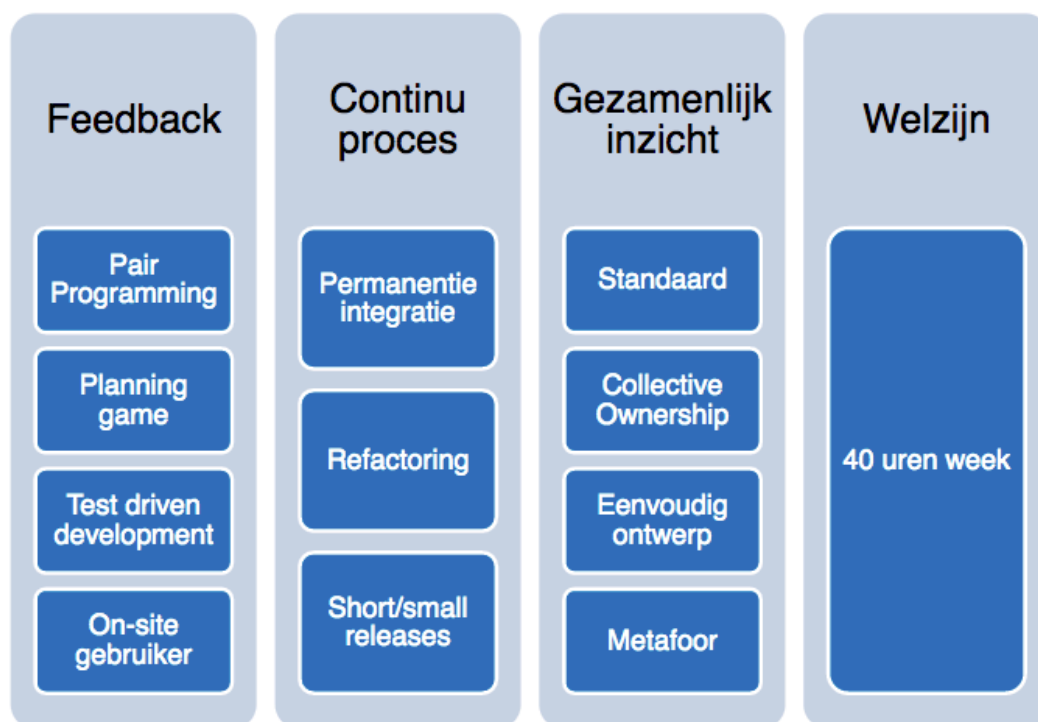
4 Basisactiviteiten:

- Ontwerpen
- Testen
- Luisteren
- Coderen

===

- Planning (Feedback)
- Short releases (Continu proces)
- Metafoor (Gezamenlijk inzicht)
- Eenvoudig ontwerp (Gezamenlijk inzicht)
- Testen (Feedback)
- Refactoring (Continu proces)
- Pair programming (Feedback)

- Collective ownership (Gezamenlijk inzicht)
- Permanente integratie (Continu proces)
- 40 uren week (Welzijn)
- On-site gebruiker (Feedback)
- Standaarden (Gezamenlijk inzicht)



Planning

Software ontwikkeling is een dialoog: wat is gewenst - wat is mogelijk

Zakenlui	Technische mensen
Scope	Schattingen
Prioriteit	Gevolgen
Inhoud van de releases	Proces
Datum releases	Gedetailleerde planning

! Starten zonder plan is onmogelijk ! Permanent planning wijzigen kan niet

- Ruw plan
- Korte releases
- Gebruiker deel van het team

Short releases

Elke release zo klein mogelijk maar met maximaal rendement

- ! Het is onmogelijk om na enkele maanden reeds in productie te gaan
- ! Zeker geen nieuwe releases door voortdurende wijzigingen

- Belangrijkste stories
- Permanente integratie
- Testen (minimale foutenlast)
- Eenvoudig ontwerp

Metafoor

Elk XP-project wordt vergezeld van een metafoor

! Ontwikkeling starten op basis van een metafoor, met te weinig gedetailleerde informatie: Onmogelijk

- Snel concrete feedback
- Communicatie
- Permanente refactoring

Eenvoudig ontwerp

- Voert alle testen uit
- Heeft geen dubbele logica
- Legt elke intentie vast die belangrijk is
- Heeft zo weinig mogelijk klassen en objecten

! Onmogelijk om voldoende ontwerp te hebben voor de code van vandaag: Vastlopen!

- Refactoring is normaal
- erken aan ontwerp in overleg met de collega (vertrouwen)

Testen

Een programma zonder een geautomatiseerde test bestaat niet!

! Al deze testen op voorhand schrijven, is onmogelijk: Tijdsverslies!

- Eenvoudig ontwerp
- Pair programming
- Vertrouwen
- Goed gevoel bij de gebruiker

Refactoring

Refactoring: een **MUST**

! Permanent ontwerp en code wijzigen = tijdsverlies, te weinig controle, crash!

- "collective ownership"
- Eenvoudig ontwerp
- Standaard
- Pair programming / team
- Testen
- Permanente integratie

Pair programming

Met 2 weet je meer dan alleen

! Met 2 achter 1 scherm: rendement zakt, Ruzie...

- Gebruik standaarden
- Minder stress
- Samen testen schrijven
- Same eenvoudig ontwerp uitwerken
- Samen implementeren

"Collective ownership"

Elk teamlid dat een kans ziet een meerwaarde te geven, is verplicht om dit te doen

! Iedereen overal zaken laten wijzigen, je maakt meer stuk dan goed: Onmogelijk!

- Integratie na korte perioden
- Testen
- Pair programming
- Standaard

Permanente integratie

Code wordt geïntegreerd en getest na enkele uren

! Integreren na enkele uren werk: onmogelijk: Tijdsverlies!

- Zeer snel testen
- Voortdurende refactoring
- Je werkt met 2

40 uren week

1 week overuren is toegestaan, de volgende week niet meer

! In een 40 uren week kan je onmogelijk meerwaarde geven

- Planning stelt prioriteiten
- Geen verassingen: testen
- Oefening baart kunst

On-site gebruiker

Een "echte gebruiker" moet in het team zetelen

! Een gebruiker laten zetelen in het team = niet produceren op een andere plaats: Verlies rendement!

- Functionele testen schrijven
- Feedback

Standaard

Noodzakelijk!

- ! Binnen een **team standaarden** gebruiken: **onmogelijk**
- ! Programmeurs zijn **individualisten!**

- XP laat je deel uitmaken van een winnend team!

Enkele regels

- Doel (Maximum economische waarde uit software product)
- Strategie (Zo weinig mogelijk investeren met maximaal rendement)
- Stukken (Story cards, task cards)
- Spelers (Ontwikkelteam, Business team)
- Moves (Exploration / Commitment / Steering)

Exploration

- Schrijf verhaal
- Schat de implementatie van het verhaal in
- Deel verhaal op in meerdere user stories

Commitment

Business team bepaalt scope en datum voor volgende release.
Implementatie team belooft tijdige oplevering

- Sorteer stories op belangrijkheid
- Sorteer stories naar risico
- Geef realistische schatting voor implementatie
- Kies de stories die gerealiseerd moeten worden tegen de volgende release

Steering

- Iteratie (3 weken)

TASK CARDS

- Verdeel story in taken
- Split op of combineer taken
- Accepteer verantwoordelijkheid voor een taak
- Schat taak in
 - Implementeer de taak
 - Zoek partner
 - Schrijf testcases
 - Implementeer
 - Integreer
- Vastleggen vooruitgang

- "Recovery"

Programmeurs soms te optimistisch

- Scope beperken tot enkele taken
- Overleg met gebruiker: scope te beperken van enkele stories
- Niet essentiële taken opschorten
- Betere hulp invoeren
- Sommige stories verdagen in overleg met gebruiker naar latere iteratie

- "New story"

Verifieer story op basis van de functionele testen

- Herplanning

Bij eventuele vertraging na 3 weken:

Alle hens aan dek voor collaboratieve refactoring -> Technical Depth

"Development strategy"

- Permanente integratie
- Leer / test / codeer / release
- "collective ownership"
 - Weinig complexe code
 - Persoonlijke betrokkenheid
 - Kennis verspreiden

Pair programming

- Is niet 1 persoon die werkt terwijl de andere toekijkt
- Is geen tutor sessie
- Is wel een dialoog tussen 2 mensen die proberen simultaan te programmeren (analyse, ontwerp en test)
- Is hulp vragen

Design strategy van XP

Altijd zorgen voor het eenvoudigste ontwerp

- Communicatie
- eenvoud
- feedback
- moed

Randvoorwaarden

- Ondersteuning van het management
 - Gemeenschappelijke werkplaats (colocatie)
 - "Pairing stations"
- XP Team
 - Product manager, on-site customers, testers, programmeurs met sterke design-skills
 - GEEN SOLOSPELERS
 - Team grootte: 4-12 / even aantal

Recap

XP is niet geschikt voor

- Zeer grote projecten
- Mainframe applicaties
- Trage feedback
- Als testen in reële situaties zeer veel kost
- Geen beschikbare ruimte

"XP is a lightweight, low-risk, flexible, predictable, scientific way to develop software"

Terugblik

XP versus Scrum

Waarin verschillen ze?

Waarin verschillen ze niet?

XP versus Kanban

Waarin verschillen ze?

Waarin verschillen ze niet?

Agile = incrementele strategie?

Agile = iteratieve strategie?

Iteratieve en incrementele strategieën?

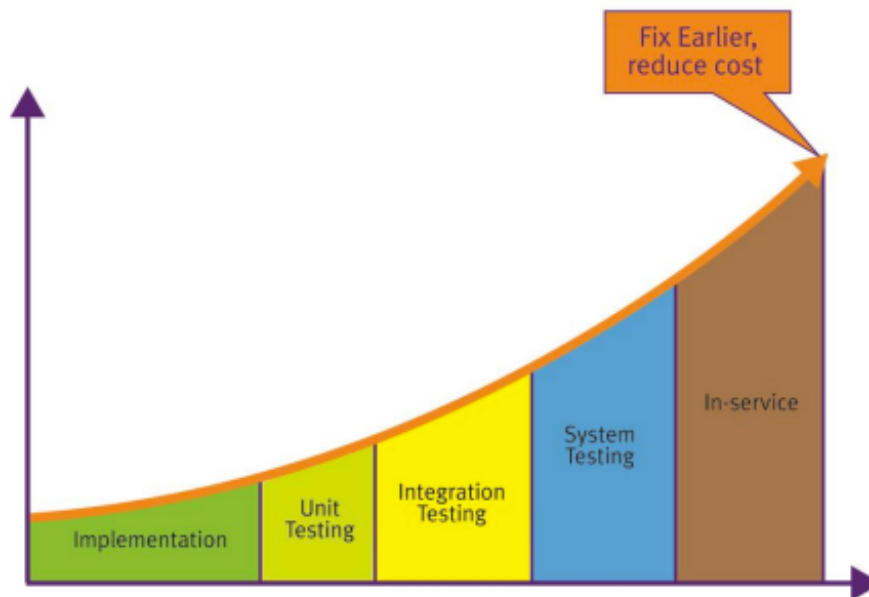
Waarin verschillen ze?

Waarin verschillen ze niet?

Agile Testing

Why testing?

- Kwaliteit
- Reputatie
- Vertrouwen
- Schade beperken
- Cost of defects



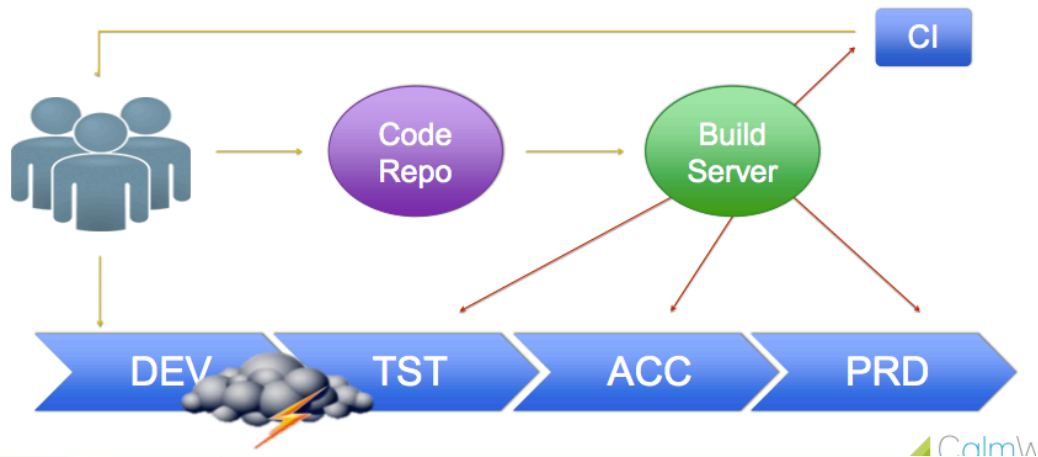
Why aren't we testing?

- Changing development habits is hard
- It takes time
- We love to write new functionality
- The investment value is not visible at the moment

ALM

Application Lifecycle Management

ALM



CONTINUOUS DELIVERY

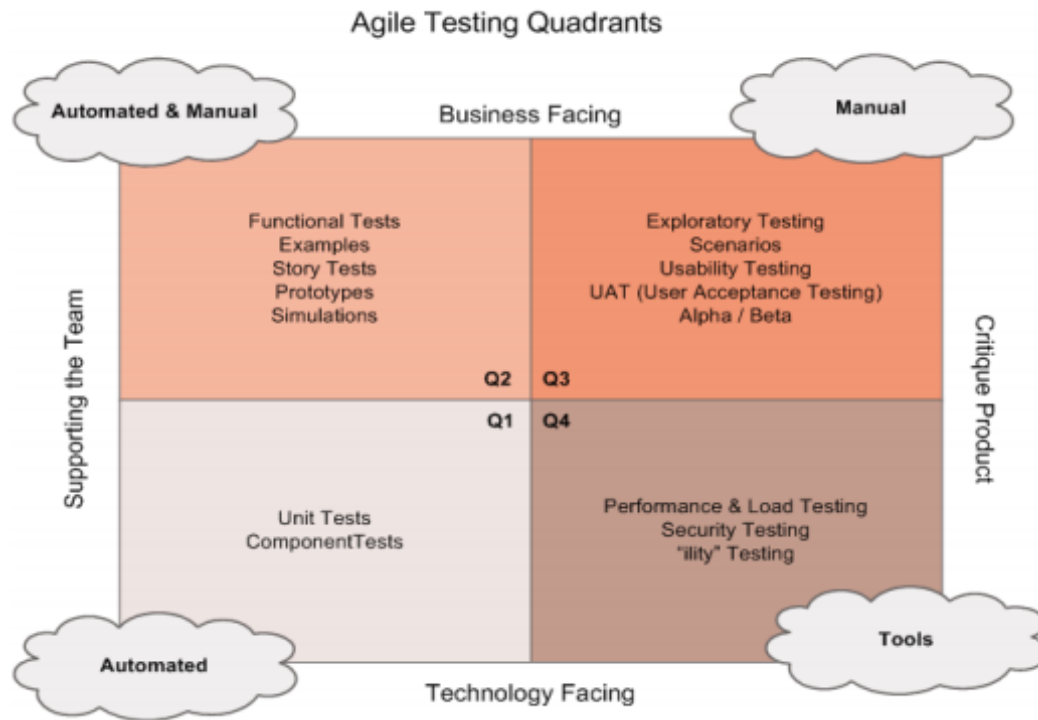


CONTINUOUS DEPLOYMENT



Automation

- Stable user interface
- Time
- Deadline
- Testing Quadrants



Types

- Unit Testing
- Integration Testing
- System Testing
- End 2 End testing

Approach

- User expectations
- Time
- Priorities
- Types of tests
- Can be based on manual test cases
- Regression

Testing Static content

- Existence UI element
- Examples:
 - Page title
 - Footer
 - Heading with specific tag

Testing Links

- Frequent source of error
- Broken links
- Missing pages

Function Tests

- Form based
- Input expects specific result
- Most complex
- Most useful
- Examples: Login, Registration, ...

Load Testing

1. Define critical transactions
2. Define actions for each transaction
3. Combine transaction to Scenario's

Selenium

- Testing Automation Tool
- Web-based applications
 - Selenium IDE
 - Prototyping tool
 - Firefox plugin
 - No programming experience
 - Simple tests
 - Quickly
 - Record and Play
 - No iteration/conditional statements
 - Export functionality
 - Selenium RC
 - Uses javascript
 - Maintenance mode
 - Several programming languages
 - Selenium WebDriver
 - Direct calls to browser
 - Selenium RC underlying technology
 - Unit testing
 - Selenium Grid
 - Multiple environments
 - Parallel
 - Multiple browsers
- Flexible and Rapid feedback
- Open source
- Multi-language backend support
 - Java
 - Ruby
 - Python