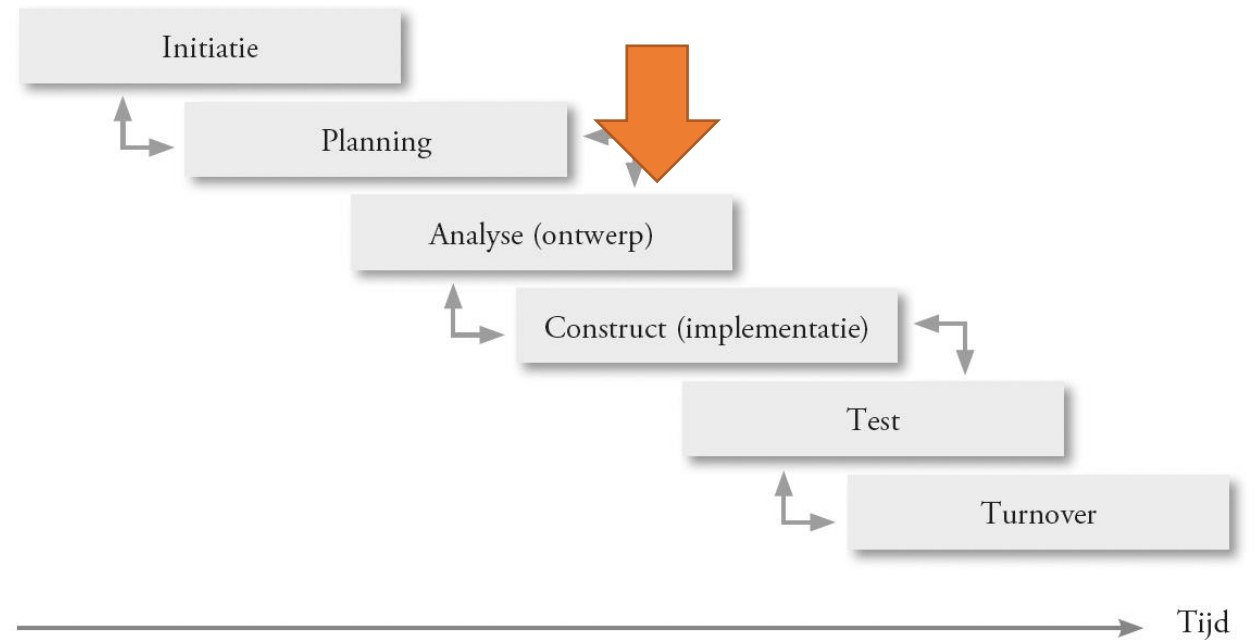


Hoofdstuk 9: Hoe gaat onze oplossing er concreet uitzien?

#analyse

Projectfases



Leerdoelen – Hoofdstuk 9

Op het einde van dit hoofdstuk:

- ... kan ik de verschillende activiteiten benoemen die typisch zijn voor de **analyse- & ontwerpfase**;
- ... onderscheid ik de verschillende **onderdelen van de blueprint** en begrijp ik het belang van elk afzonderlijk onderdeel binnen het project;
- ... begrijp ik het **belang van de blueprint** binnen een project en identificeer ik de verschillende activiteiten die afhankelijk zijn van de blueprint in de verschillende projectfases.

Analysedocument (= blueprint):

- Deeloplossingen
- Vereisten (funct + techn)
- Overeenkomst tss lev-klant (prijs)

Analyse

- Opsplitsen van oplossing in verschillende deeloplossingen
- Opstellen functionele en technische vereisten
- Opmaken van een **analysedocument of blueprint**

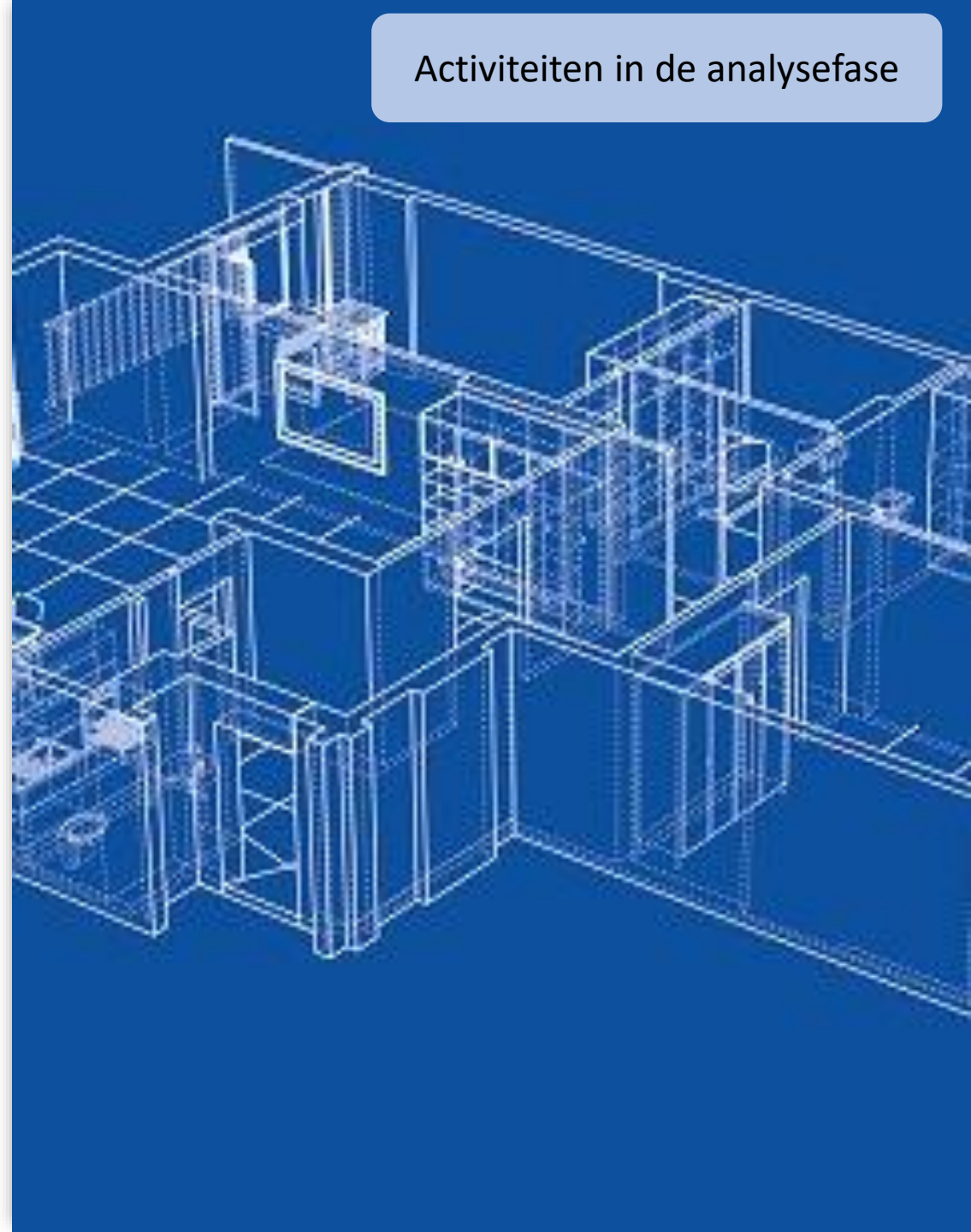
*Het **analysedocument** is het document dat een gedetailleerde beschrijving bevat van de oplossing die de leverancier zal opleveren tegen de afgesproken prijs. Het document bevat dus de definitieve overeenkomst tussen (interne) leverancier en klant.*

Omschrijving en doel

- Uitwerken van de oplossing van de conceptfase in detail
- Er wordt een “blauwdruk” gemaakt van de nieuwe applicatie
- Deze **blauwdruk** wordt gebruikt om de **oplossing te valideren** met de klant en bevat het uiteindelijke resultaat van het project, zowel technisch als functioneel
- Het *analysedocument kan uiteraard nog wijzigen, zeker wanneer er agile gewerkt wordt* zal een deel van de analyse verschuiven naar de implementatiefase
- De analysefase heeft meestal een aantal vaste activiteiten

Workshops

- De (business) analist organiseert een aantal workshops met eindgebruikers om de oplossing in kaart te brengen
- Meestal heeft een workshop een thema: een groep van functionaliteiten of deelprocessen die worden samen genomen
- Indien het over een project gaat in meerdere filialen of landen, kan een workshop per land zinvol zijn
- Voorbereiding is belangrijk: op basis van bestaande informatie tot een verbeteringsvoorstel komen
- Een UX/UI designer kan hier al meegenomen om de visuele voorstelling van de oplossing te bespreken



POC of prototype

- Om de klant al een inzicht te geven in de oplossing wordt er vaak een **prototype of proof of concept (POC)** opgesteld
- Bij een standaardpakket kan er eenvoudig een **demo** opgesteld worden
- Bij maatwerk wordt dit eerder een "light" versie van de applicatie
- Er kan met **mock-ups** gewerkt worden
- Vandaag zijn er heel wat tools om prototypes te maken op basis van screen flows



Mock-up

[Artikel](#) [Overleg](#)

[Lezen](#) [Bewerken](#) [Brontekst bewerken](#) [Overleg](#)

Uit Wikipedia, de vrije encyclopedie

Een **mock-up** is een tijdens de [ontwerp](#)- of [productiefase](#) op [schaal](#) of op ware grootte gemaakt [model](#) van een ontwerp of product. 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. Meestal zijn de mock-ups dus voorbeelden qua uiterlijk.^[1]

wat is een mockup?



ChatGPT



Een **mockup** is een visuele weergave of een model van een ontwerp, product, of interface dat vaak gebruikt wordt om te tonen hoe iets er uiteindelijk uit zal zien. Mockups worden meestal gebruikt in design- en ontwikkelprocessen, zoals bij het ontwerpen van websites, apps, producten of verpakkingen.

Enkele kenmerken van mockups:

- **Realistische presentatie:** Mockups geven een zo realistisch mogelijke weergave van het eindproduct, inclusief kleur, vorm en layout, maar zonder functionele elementen.
- **Communicatiemiddel:** Het helpt ontwerpers, klanten of stakeholders om een duidelijk idee te krijgen van het eindresultaat voordat de ontwikkeling echt begint.
- **Feedback verzamelen:** Mockups maken het mogelijk om in een vroeg stadium feedback te vragen, zonder dat er veel tijd en middelen aan ontwikkeling zijn besteed.

Een mockup verschilt van een wireframe, dat eenvoudiger en meer gefocust is op de structuur en functionaliteit zonder veel aandacht voor esthetiek.



POC of Prototype

Wat is het verschil? Zoek voorbeelden

- **Proof of concept:**

- Bewijst dat een idee KAN uitgevoerd worden
- Het kan een mini-implementatie zijn van een onderdeel, of een theoretische validatie

- **Prototype:**

- Uitvoering van het idee
- Je bewijst niet alleen dat het idee haalbaar is, je voert het ook uit door een prototype te maken
- Het is nog geen compleet uitgewerkt product
- Een prototype wordt bv. door een deel van de werknemers van een bedrijf getest, en nog niet door klanten
- Het gaat om het testen van een groter geheel in plaats van kleine details

Opstellen van het analysedocument (blueprint)

- De business analist en eindgebruikers stellen samen het analysedocument op
- Wordt gebruikt als basis voor de validatie van het project
- Dient als basis voor de kostenraming in geval van time & material
- Onderscheid tussen “must haves” en “nice to haves”
- Bij Fixed Price zal een duidelijke lijst worden opgesteld van functionaliteiten die inbegrepen zijn



Validatie van het analysedocument

- Op de **validatie meeting** zullen alle betrokken partijen bevestigen dat de beschreven oplossing hetgene is wat opgeleverd zal worden
- Er wordt vastgelegd wat er wanneer wordt opgeleverd en tegen welke kost
- Wanneer er later discussie is, kan er altijd terug gekeken worden naar de blueprint
- De uiteindelijke oplossing zal altijd afwijken, maar de manier waarop met de wijzigingen wordt omgegaan staat in de blueprint

Vorbereiding van de implementatiefase

- Samenstellen finale projectteam
- Vergaderingen vastleggen (stuurgroep, statusvergadering, ...)
- Administratieve en logistieke ondersteuning regelen (badges, computers,...)



Doel van de analysefase



- **Formeel document** dat vastlegt wat de klant kan verwachten en wat de leverancier dient op te leveren
- **Verwachtingen op één lijn krijgen**
- Een goed ontwerp zorgt ook voor een betere uitvoering van het project
 - Duidelijk voor alle teamleden wat ze moeten opleveren
 - Makkelijker testscenario's uit te schrijven
 - Faciliteert de documentatie: al deels in de blueprint beschreven
- Een goed ontwerp vermijdt discussies, vertraging en onevenwichtige verdeling van het werk

Rollen en verantwoordelijkheden

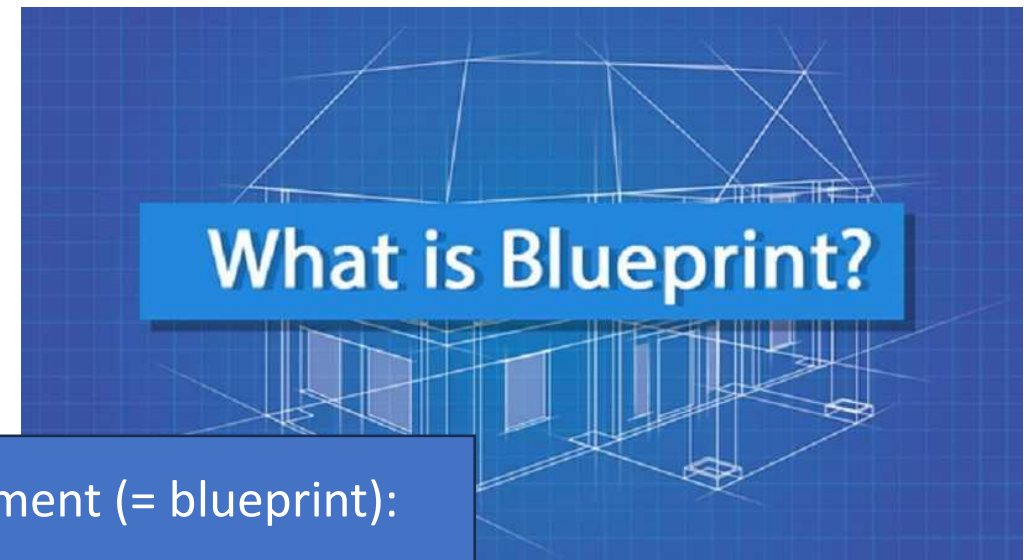
- **Business analist en eindgebruiker**: stellen samen het analysedocument op
 - Business analist organiseert workshops, modereert en synthetiseert in het blueprint document
 - Eindgebruikers nemen deel aan de workshops en leveren extra input wanneer nodig
 - Bij grote, complexe projecten is er vaak een team van business analisten met verschillende expertise
- **Ontwikkelaars**: geven als experts input voor de technische analyse
- **Sponsors**: nemen de finale beslissing en valideren de blueprint
- **Projectmanager**: waakt over de haalbaarheid qua timing en budget
 - Aanwezig bij workshops
 - Zorgt voor de voorbereiding van de implementatiefase

Deliverables - Analysedocument



- De blueprint bevat (meestal) minimaal volgende delen:

1. Procesbeschrijvingen AS-IS en TO-BE
2. Functioneel ontwerp
3. Technisch ontwerp
4. Security, authorisaties en rollen
5. Interfaces
6. Datamigratie
7. Systeemarchitectuur
8. Testplan
9. Assumpties



Analysedocument (= blueprint):

- Deeloplossingen
- Vereisten (funct + techn)
- Overeenkomst tss lev-klant (prijs)

Blueprint – 1. Procesbeschrijvingen

- Zowel de AS-IS (bestaande situatie) als de TO-BE (gewenste situatie) processen worden opgenomen
- Geeft de impact van het project op de organisatie weer

*Een **AS-IS-proces** beschrijft de huidige situatie van een bedrijfsproces en geeft dus een realistisch beeld van de manier waarop er momenteel gewerkt wordt en welke stappen de gebruikers doorlopen.*

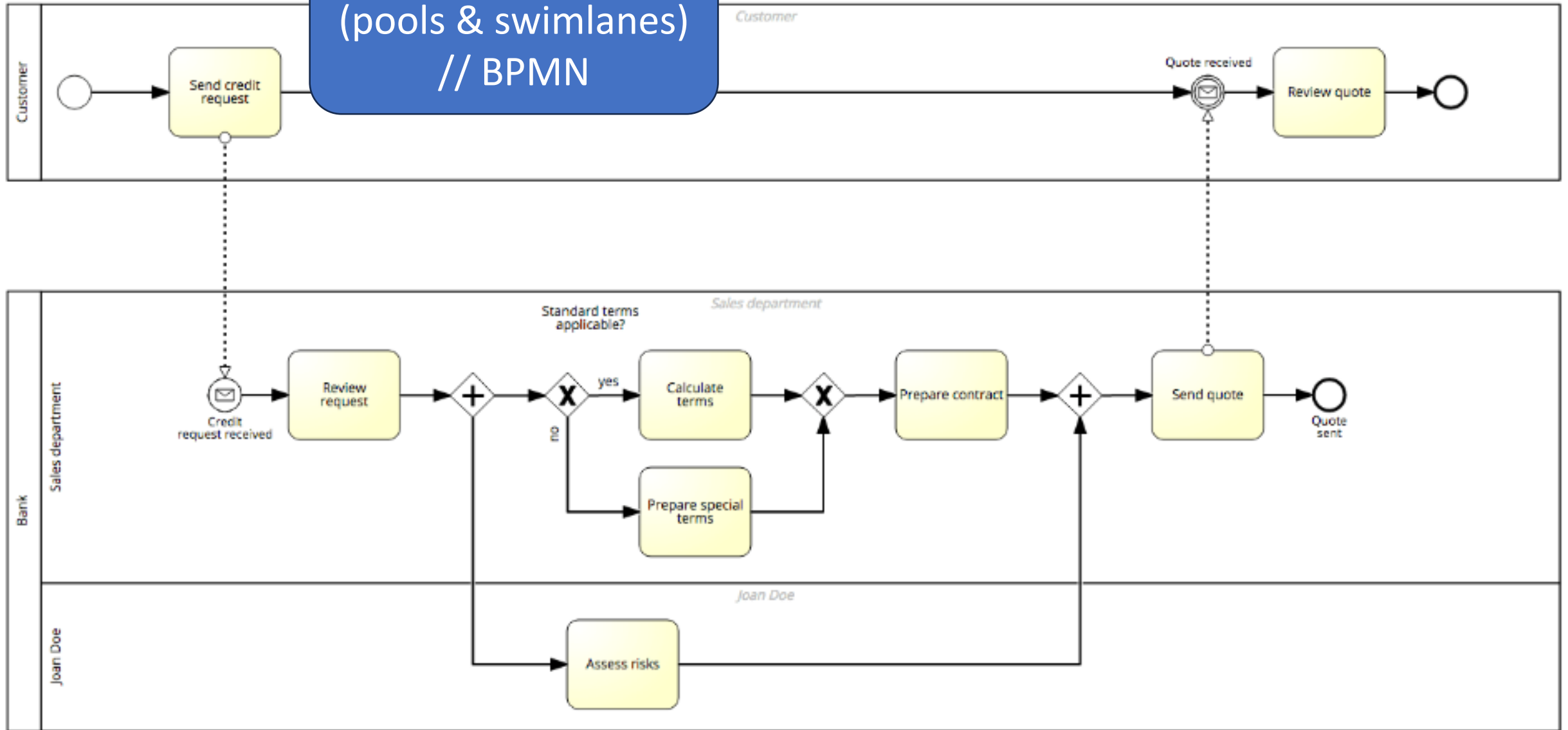
*Een **TO-BE-proces** beschrijft de situatie na de afloop van het project en geeft dus weer hoe de verschillende eindgebruikers zullen werken in de toekomst.*

Blueprint - 1. Procesbeschrijvingen

- De eerder beschreven processen uit de requirements fase kunnen nog moeten aangepast worden wanneer er nieuwe informatie komt uit de workshops
- Soms wordt er in de requirements fase geen procesbeschrijving gemaakt en moet dit nog gebeuren in de ontwerpfase
- Processen zijn ook van belang voor het bepalen van workflows

*Een **workflow** of werkstroom is de logische volgorde van activiteiten die bepaalde actoren moeten uitvoeren.*

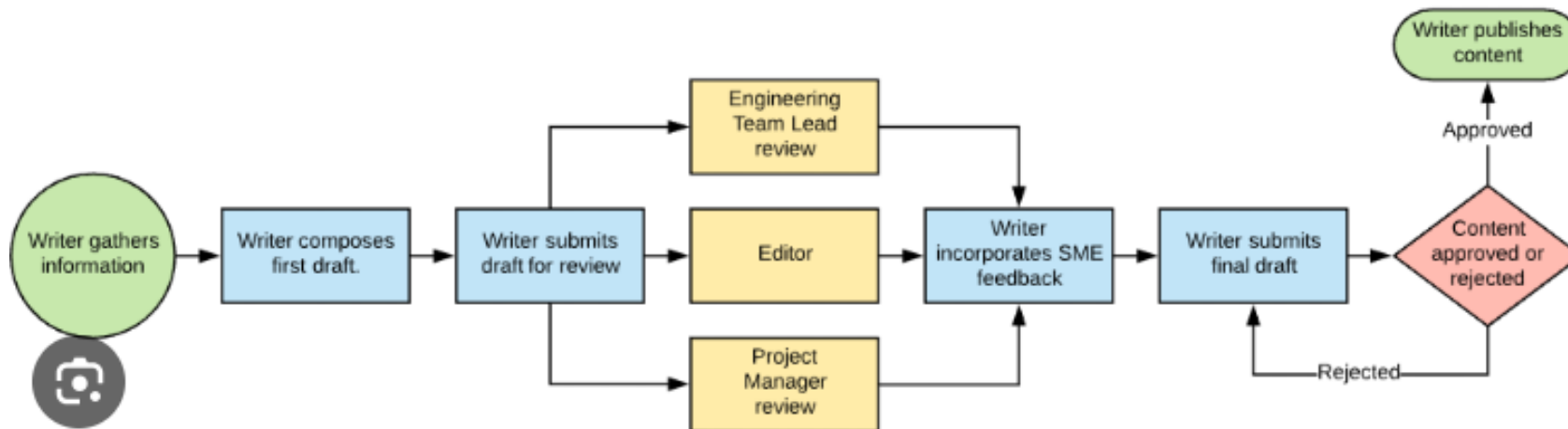
Voorbeeld Proces
(pools & swimlanes)
// BPMN



Voorbeeld Workflow



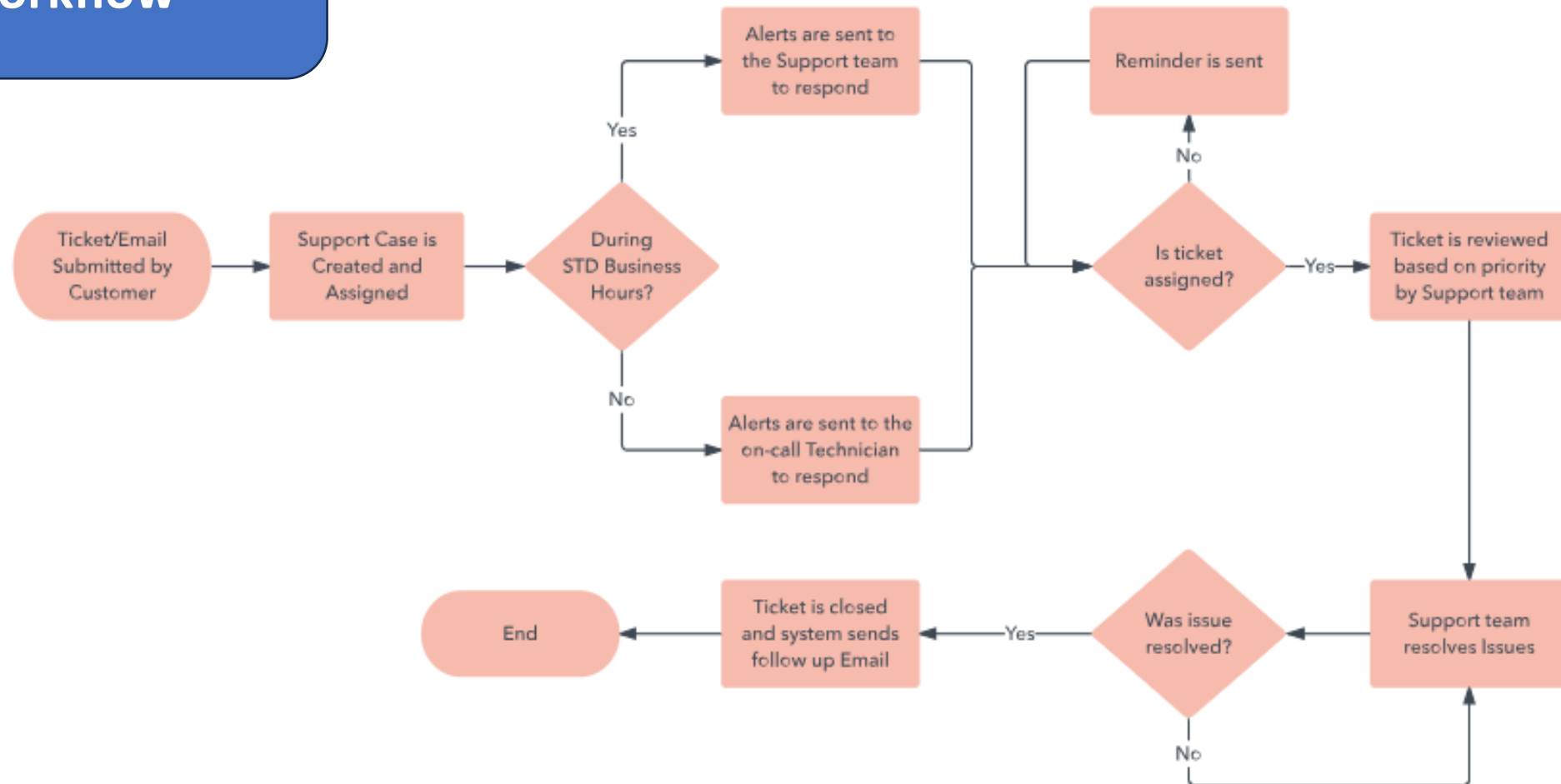
Lucidchart



Workflowmanagement: incl. voorbeelden | Lucidchart
Blog

Bezoeken >

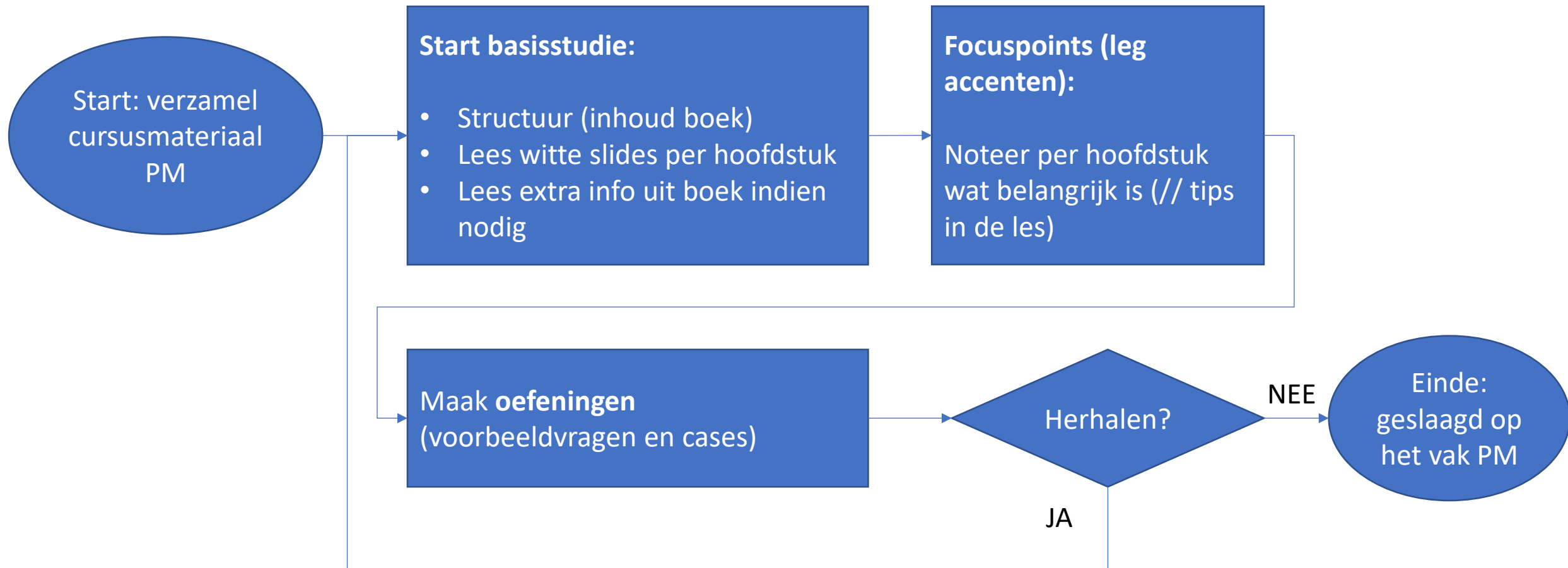
Voorbeeld Workflow



Oefening:

Maak jouw eigen workflow (met opeenvolgende taken of activiteiten) om een succesvol examen PM af te leggen

Workflow: taken/activiteiten voor succesvol examen PM

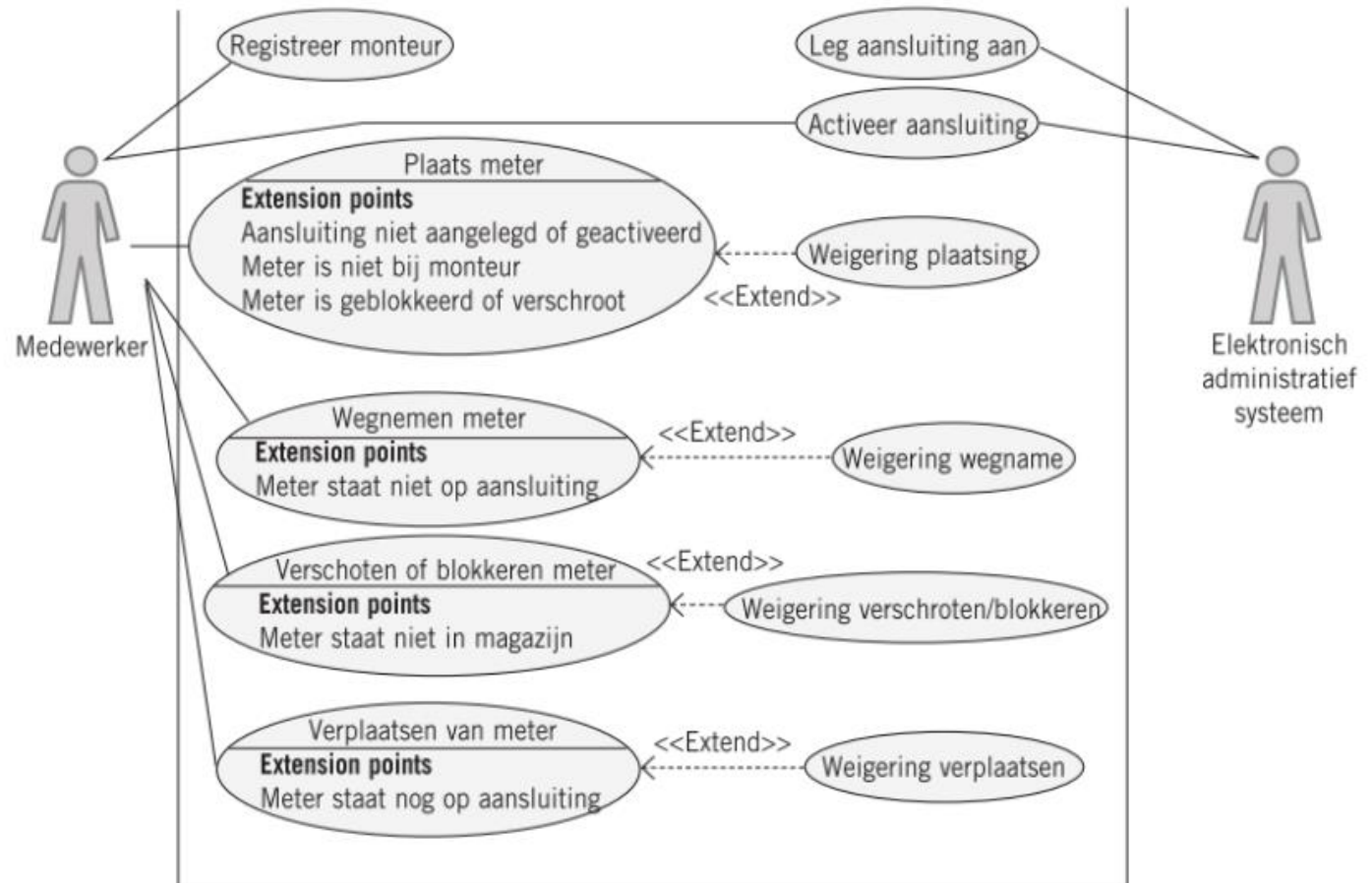


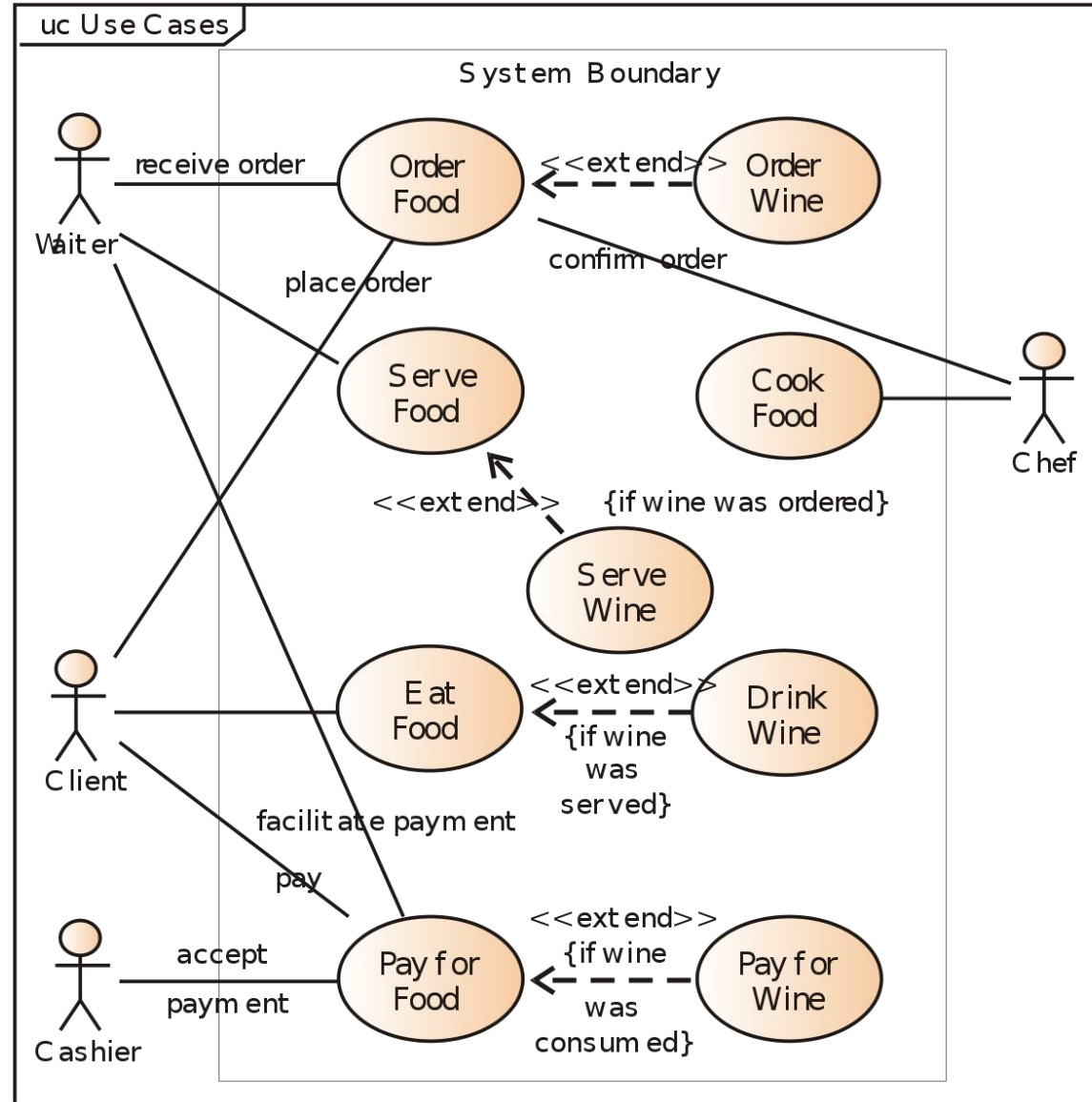
Analyse – 2. Functioneel ontwerp

- Beschrijft de oplossing op het niveau van de eindgebruikers
 - Lijst met alle functionaliteiten die de applicatie zal bevatten
 - Screenshots/Mock-ups die een idee geven hoe de applicatie er zal uitzien
 - Beschrijving van functionaliteiten gebeurt op basis van use cases

*Een **use case** beschrijft hoe de applicatie functioneert op basis van een extern signaal en geeft dus aan wat een bepaalde gebruiker met de applicatie kan doen. Het beschrijft met andere woorden de interactie tussen systeem en gebruiker.*

Use case





Analyse – 2. Functioneel ontwerp

- Oplijsten van mogelijke waarden voor drop-down lijstjes, inhoud van formulieren en mails, foutboodschappen, vertalingen,...
- Beschrijven van business logica voor algoritmen en functies indien nodig

Voorbeeld

Bij een webwinkel die boeken verkoopt, is er onderaan altijd een lijstje boeken te vinden waar kopers ook eventueel in geïnteresseerd zouden kunnen zijn. Welke boeken daar worden getoond, wordt bepaald door een algoritme:

- > Als de klant slechts één boek in zijn mandje heeft, dan zal er een top 3 getoond worden van de boeken die gekocht werden door andere klanten, die ook het boek kochten dat in het mandje van de klant ligt.
- > Als de klant twee boeken in zijn mandje heeft, dan zal er een top 2 getoond worden van de boeken die gekocht werden door andere klanten, die ook het duurste boek kochten dat in het mandje van de klant ligt, aangevuld met het top boek dat gekocht werd door andere klanten, die ook het goedkoopste boek kochten dat in het mandje van de klant ligt.
- > ...

Analyse – 3. Technisch ontwerp

- Beschrijft technische details zonder code op te nemen maar eventuele programmalogica kan opgenomen worden
- Tabelwaarden, parameters, instellingen

Unified Modeling Language (UML):
biedt een manier om het gedrag en
de structuur van een systeem of
proces te visualiseren

Analyse – 3. Technisch ontwerp

- UML – diagrammen

- Klassendiagram: beschrijft de klassen die een softwaresysteem vormen en het statische verband ertussen. De klassen worden gedefinieerd in termen van hun naam, attributen (of gegevens), en gedrag (of methodes).
- Sequence diagram: beschrijft de samenwerking tussen objecten die wordt uitgevoerd als reeks berichten tussen deze objecten. Het sequence diagram beschrijft de gedetailleerde implementatie van één enkele use case (of één variatie van één enkele use case).
- Activity diagram: beschrijft de verschillende activiteiten die binnen (een deel van) een applicatie of systeem uitgevoerd worden.
- Toestandsdigram: beschrijft de toestanden waarin een object zich kan bevinden tijdens de levensloop van dat object.

Unified Modeling Language

(UML) is een taal om objectgeoriënteerde analyses en ontwerpen voor een informatiesysteem te kunnen maken.

(bron: wikipedia.nl)

Logica en gegevens worden georganiseerd rond objecten

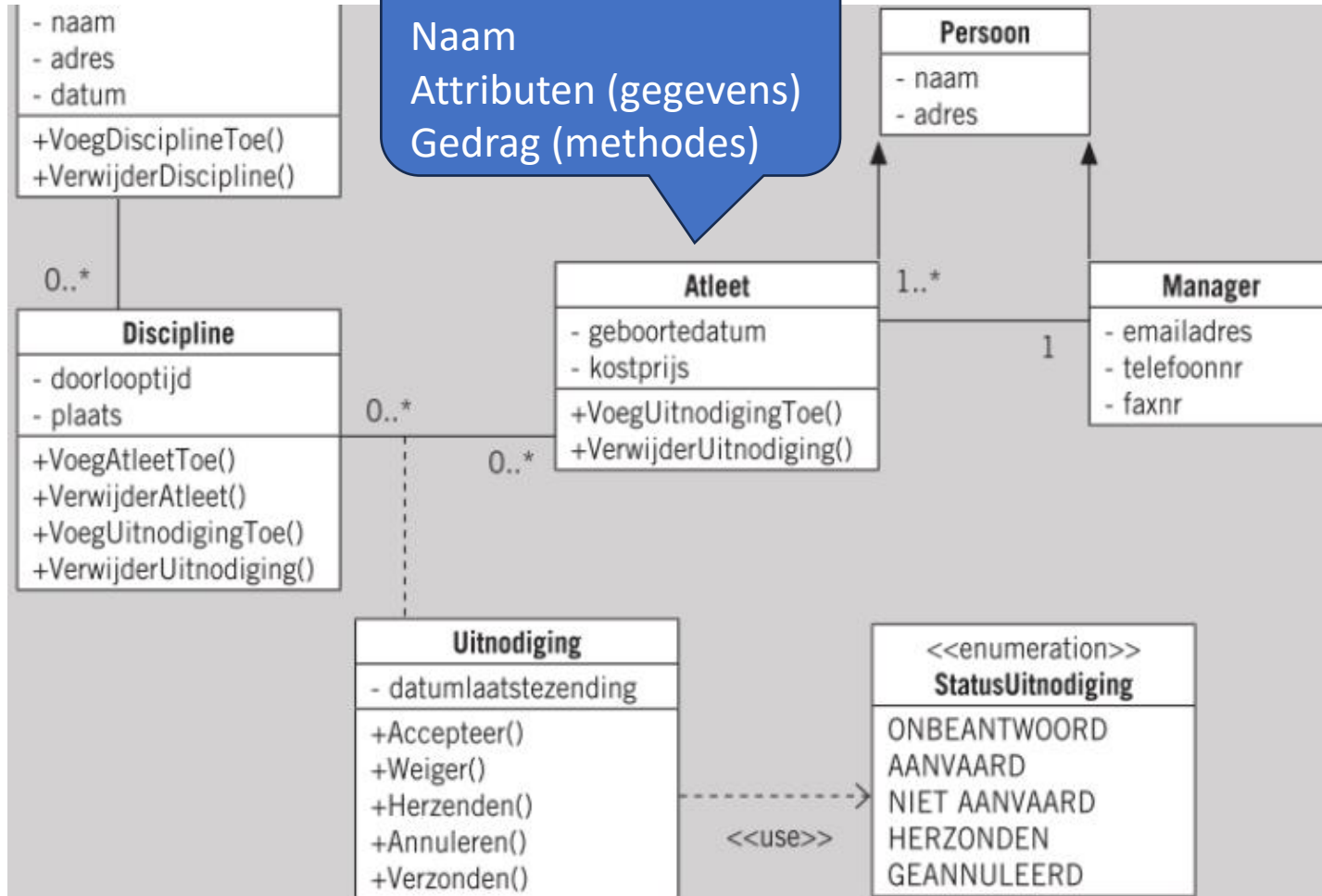


Klasse:

Naam

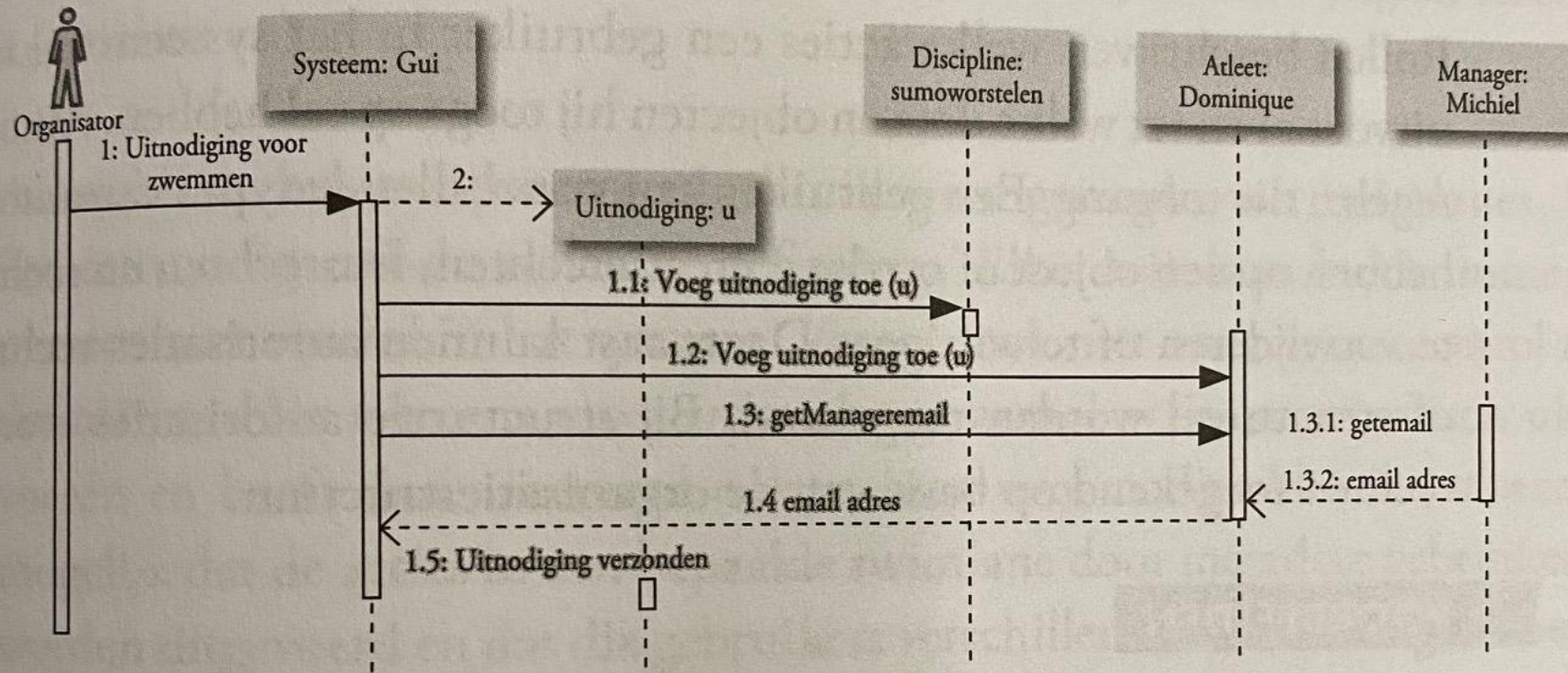
Attributen (gegevens)

Gedrag (methodes)



Voorbeeld
klassendiagram

Figuur 4 Voorbeeld van een sequence diagram.



Voorbeeld
sequence
diagram

Analyse – 4. Security, autorisaties en rollen

- Beveiliging van de applicatie is belangrijker dan ooit: privacy, gevoelige data, concurrentiële data,...
- Zowel bij externe als interne applicaties van belang
- Ook **autorisaties en rollen** moeten worden opgenomen:
 - Autorisaties: een gebruiker krijgt enkel toegang tot die gegevens en transacties die hij nodig heeft om zijn job te kunnen doen
 - Rollen: set van autorisaties die regelt tot welke acties, data en objecten een gebruiker toegang zal hebben
 - Rechtstreeks of structureel (op basis van de organisatiestructuur)
 - Eenvoudig te bepalen wanneer er gebruik werd gemaakt van swimlanes




Analyse – 4. Security, autorisaties en rollen

Voorbeelden

- 1 Een HR-medewerker uit de afdeling in Brussel kan enkel gegevens van medewerkers uit Brussel raadplegen en verzorgen. De HR-medewerker uit de afdeling in Antwerpen enkel de gegevens van Antwerpen. De HR-medewerker uit de centrale diensten kan echter wel alle gegevens van zowel Antwerpen als Brussel beheren. De centrale dienst bevindt zich in het organigram immers boven de business units in Antwerpen en Brussel.
- 2 In een webshop kunnen volgende typische rollen gedefinieerd worden:
 - > Gebruiker: de klant, die de webshop gebruikt om items aan te kopen en te betalen. Hij heeft typisch toegang tot de frontoffice en krijgt enkel schrijfrechten voor de persoonsgegevens, het winkelwagentje en eventueel een wenslijstje. De gebruiker kan geen gegevens van producten aanpassen en heeft geen toegang tot de backoffice.
 - > Verkoper: de verkoper beheert de backoffice en kan dus bestellingen beheren, maar ook producten toevoegen en aanpassen. Daarnaast zal de verkoper waarschijnlijk ook de persoonsgegevens van de klanten kunnen raadplegen en aanpassen. Een verkoper kan echter geen instellingen wijzigen en geen tabelwaarden toevoegen.
 - > Beheerder: de beheerder zorgt voor het onderhoud van het systeem, het aanmaken van gebruikers en het wijzigen van instellingen. De beheerder heeft meestal een quasi onbegrensde account zodat hij problemen kan oplossen.
 - > Verzender/order picker: de persoon die de bestelling moet klaarmaken en verzenden heeft vaak ook een beperkte toegang tot de applicatie nodig. Zo zal hij bestelbonnen kunnen afprinten en de status van een bestelling kunnen wijzigen naar 'verzonden'.

Analysedoc – 5. Interfaces

- **Connectie** tussen twee systemen waarbij data van het ene naart het andere systeem wordt gestuurd
- Beschrijving van **conversieregels** is noodzakelijk
- Gegevensuitwisseling moet **gemonitord** worden via een procedure van opvolging
 - Als er iets misloopt in de connectie moet dit gedetecteerd en opgelost worden!
- Definiëren van **master systeem**:
systeem dat de wijzigingen in data doorvoert en doorstuurt



Een communicatie of besturing waarbij een apparaat of proces (= de master) een of meer andere apparaten of processen (= de slaves) bestuurt

Bron: wikipedia.nl

Analyse – 6. Datamigratie

- Wanneer er een nieuwe applicatie wordt ontwikkeld om de oude te vervangen, moet de oude data vaak mee overgezet worden:
- **Welke gegevens** moeten overgezet worden?
 - Welke periode?
 - Wordt alle data nog gebruikt in het nieuwe systeem?
 - Conversieregels
- **Wie zal de gegevens aanleveren en in welk formaat?**
 - Meestal dient de klant de data aan te leveren en te controleren
 - Marge inbouwen om data te controleren en te verbeteren
 - Eventuele download uit het oude systeem



Analyse - 6. Datamigratie

- **Wanneer** worden de data overgezet en hoeveel testruns worden er voorzien?
 - Data overzetten vergt een goede voorbereiding en wordt best ook getest
 - Kwaliteitscheck kan ook via testloads in testsysteem
 - Meest voorkomende problemen komen uit niet compatibele formaten
- **Hoe** worden de gegevens overgezet?
 - Bij voorkeur automatisch
 - Kost voor ontwikkelen automatische upload
 - Soms interessanter om het manueel over te zetten
- Cruciale activiteit bij go-live

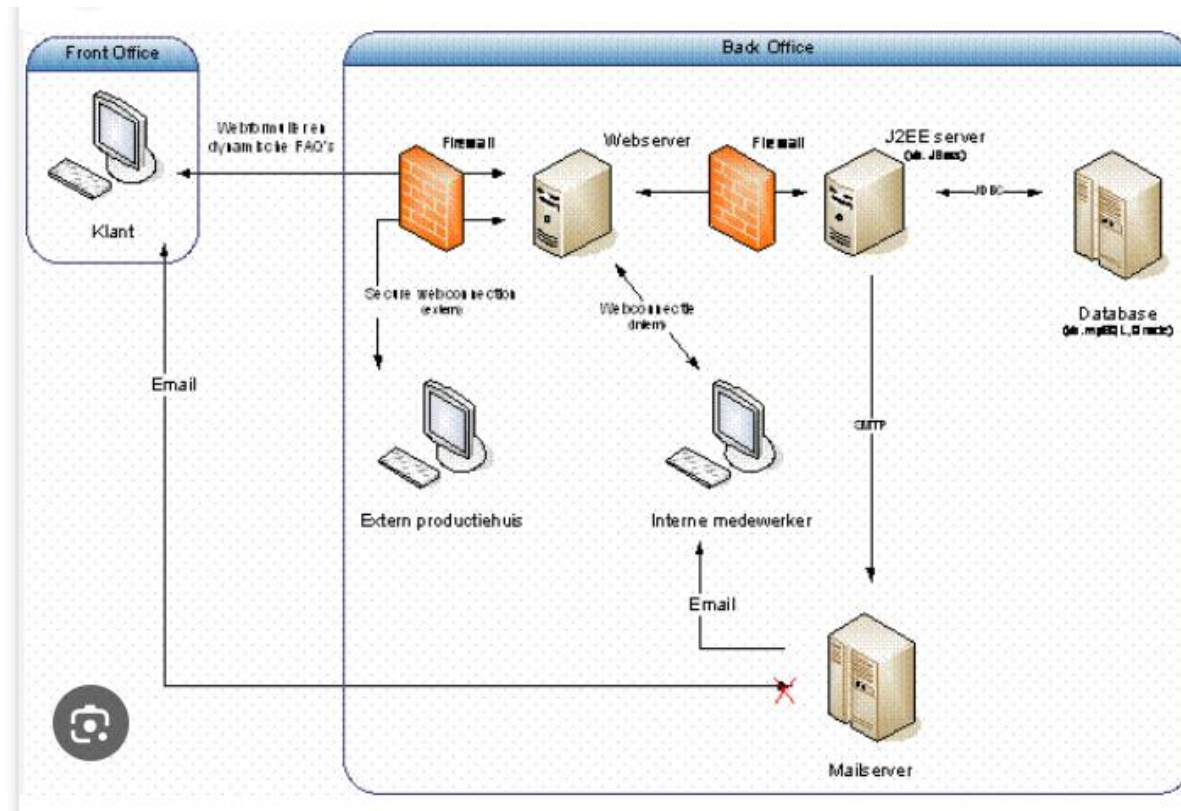
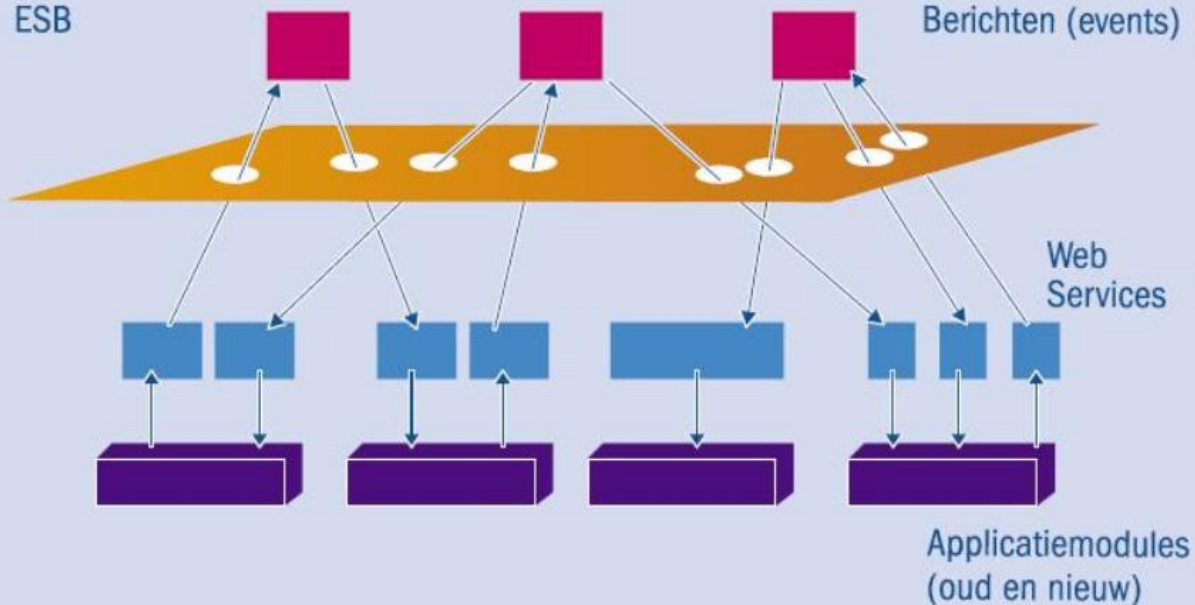
Analyse – 7. Systeemarchitectuur

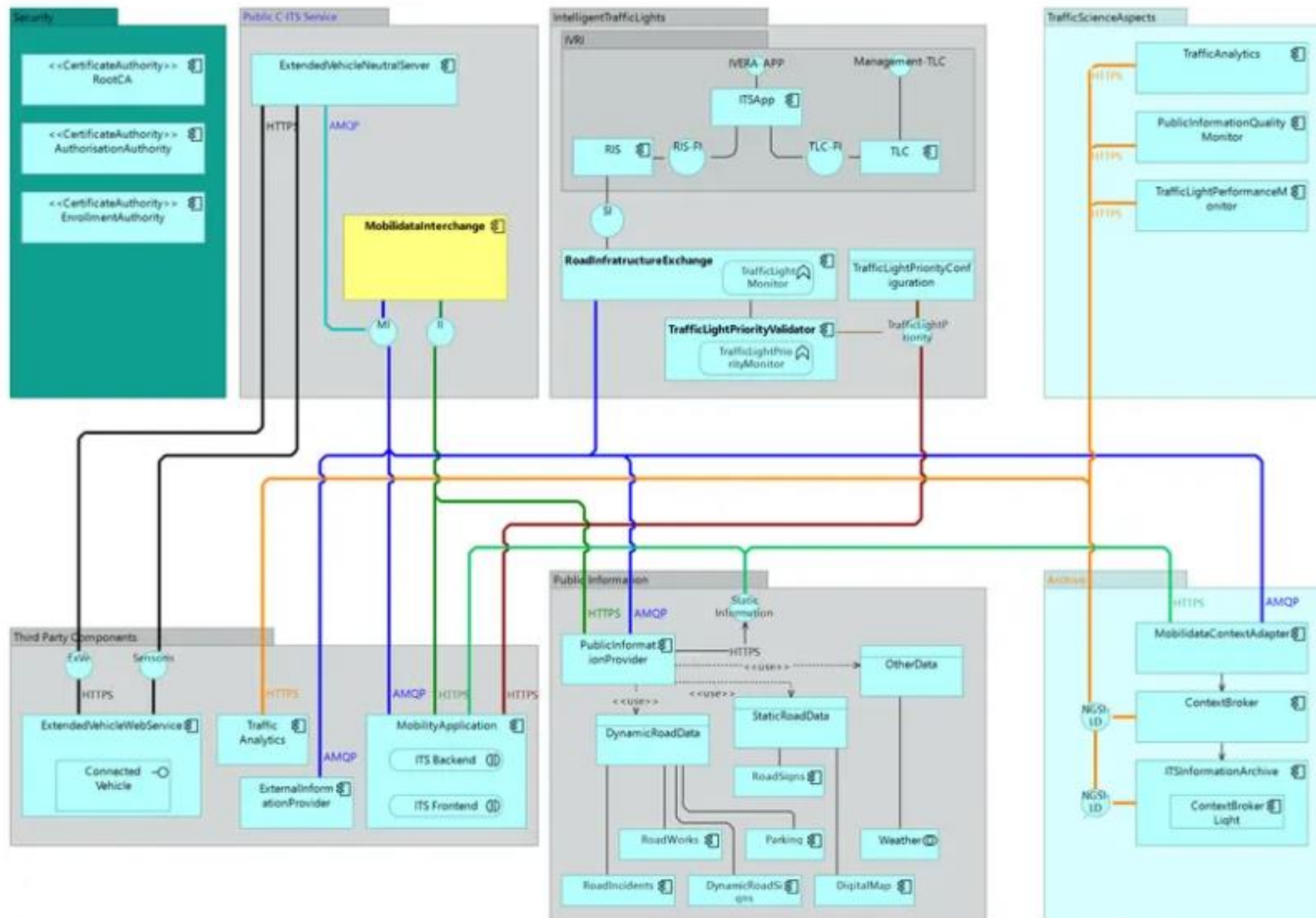
- Opzet van de applicatie en de verschillende randapplicaties en – systemen
- Beschrijving van de impact op de huidige architectuur
- Beschrijving van de benodigde hard- en software
- Opmaken van een architectuurschema zodat de IT-afdeling kan zien of dit past binnen de huidige opzet en de keuzes die het bedrijf maakt op technisch vlak

EVENT DRIVEN ARCHITECTURE

ESB

Berichten (events)





Bron:
<https://www.imec.be/nl/articles/naar-een-uniforme-aanpak-voor-technologie-en-mobiliteit>

Schematisch overzicht van de systeemarchitectuur die Mobilidata ontwikkelde en waarin het – mede dankzij kennisdeling zoals via deze whitepaper – samen met anderen verder wil op bouwen.

Analyse – 8. Testplan

- Definiëren van **de testscenario's** zonder deze al uit te schrijven
 - Vrijmaken van de juiste betrokkenen
 - Planning opmaken van de testen
 - Finale controle op de volledigheid van de blueprint
- De testscenario's worden opgesteld op basis van de business processen
- Testen omvat niet enkel functionele testen maar ook **stress testing, portability testing, performance testing**,... (zie Hoofdstuk 11)

Voorbeeld testscenario's

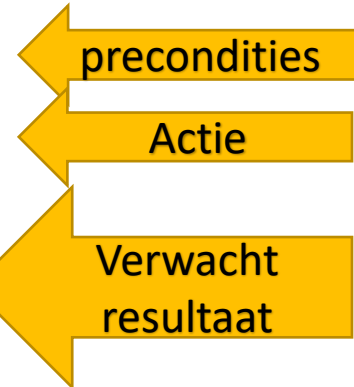
Microsoft Excel - _template Testscenario 0x.0x procesnaam.xls				
Bestand Bewerken Beeld Invoegen Opmaak Extra Data Venster Help				
📄 📁 💾 🖨 🔍 🔗 ✂ 📋 📌 🔄 📶 Σ fx ↕ ↕ 📊 📶 80% ?				
Arial 10 B <i>I</i> <u>U</u> 📏 🔢 📊 % 000 ↑,00 ↓,00 🔍 🔍 🔍				
A128	=			
	A	B	C	D
1	Scenario	0x.0x [procesnaam]		
2	Naam	[naam van tester]		
3	Datum	week xx en verder		
4	Stap	Actie	Uitgevoerd	Opmerkingen
116		Log in als Manager		
117		Kies in het linkermenu voor Takenlijst > Mijn taken		
118		Controleer of de eerder genoteerde aanvraag in de takenlijst staat		
119		Controleer of de reeds eerder ingevulde gegevens nog steeds gelijk zijn.		
120		Controleer of het mogelijk is de gegevens aan te passen/aan te vullen		
121		Controleer of "vorige commentaren" correct worden weergegeven		
122		Controleer of het mogelijk is "Nieuwe commentaren" toe te voegen		
123		Controleer of onderaan het formulier de volgende drie knoppen beschikbaar zijn: 1. Goedkeuren, 2. Afkeuren, 3. Terugsturen		
124	18.2.1	Terugsturen		
125		Controleer of het mogelijk is de aanvraag terug te sturen		
		Controleer of de teruggestuurde aanvraag correct in de takenlijst van de eerdere aanvraag staat		

Voorbeeld testscenario's



Testset [standaard]+ [versie]:
[Referentiecomponent]

[Referentiecomponent]				
1	[serviceprovider scenari-naam] (P)			
2	[serviceconsumer scenari-naam] (C)			



Preconditie	Zaakidentificaties die door de TTA worden aangeleverd aan het STP zijn uniek gedurende het doorlopen van de testset voor het Zaaksysteem.			
Volgnummer	Berichttype	Actie	Verwacht Resultaat	Bestandsnaam voorbeeldbericht
1	genererenZaakidentificatie_Di02	Het STP verzendt een genererenZaakidentificatie bericht naar de TTA.	De TTA ontvangt en verwerkt op correcte wijze het ontvangen bericht.	genererenZaakidentificatie_Di02
2	genererenZaakidentificatie_Du02	De TTA verzendt een antwoordbericht met een geldig zaakidentificatie naar het STP.	Het STP ontvangt een antwoordbericht en deze bevat een volgens RGBZ geldige zaakidentificatie	
3	genererenZaakidentificatie_Di02	Het STP verzendt een genererenZaakidentificatie bericht naar de TTA.	De TTA ontvangt en verwerkt op correcte wijze het ontvangen bericht.	genererenZaakidentificatie_Di02
4	genererenZaakidentificatie_Du02	De TTA verzendt een antwoord met een geldig zaakidentificatie naar het STP.	Het STP ontvangt een antwoordbericht en deze bevat een volgens RGBZ geldige zaakidentificatie	
Preconditie	1) De Genereren Zaakidentificatie met volgnummer 1 is succesvol uitgevoerd en heeft een Zaakidentificatie aangeleverd voor het creëren van een zaak. In de overige scenario's zal hiernaar als Zaak1 worden verwezen.			
Volgnummer	Berichttype	Actie	Verwacht Resultaat	Bestandsnaam voorbeeldbericht
1	zakLk01	Het STP verzendt een Creëren Zaak bericht naar de TTA met de Zaakidentificatie uit volgnummer 2 van het scenario Genereren	De TTA ontvangt en verwerkt op correcte wijze het bericht door een Zaak (Zaak1) te creëren op basis van het ontvangen ZaakID uit het	Creëren zaak_zakLk01
2	Bv03Bericht	De TTA verzendt een bevestigingsbericht naar het STP	Het STP ontvangt het bevestigingsbericht bij het creëren van de zaak	
3	zakLk01	De TTA ontvangt een Creëren Zaak bericht vanuit het STP met het ZaakID uit volgnummer 4 van het scenario Genereren	De TTA ontvangt en verwerkt op correcte wijze het bericht door een Zaak (Zaak2) te creëren op basis van het ontvangen ZaakID uit het	Creëren zaak_zakLk01
4	Bv03Bericht	De TTA verzendt een antwoord naar het STP	Het STP ontvangt het bevestigingsbericht bij het creëren van de zaak	

Analyse – 9. Veronderstellingen

- Er zullen ook in deze fases nog onduidelijkheden zijn: er zullen nog steeds veronderstellingen moeten worden gedaan zodat de blueprint sluitend is

Voorbeeld

'Er zullen formulieren voorzien worden' laat te veel ruimte voor interpretatie. Beter is: 'Er worden vijf formulieren voorzien, waarvan drie statische en twee dynamische. Elk formulier bevat maximaal twintig outputvelden en tien inputvelden.' Daarna zou dan nog, indien al bekend, een beschrijving van de verschillende formulieren moeten volgen.

- Ook assumpties over zaken waar de leveranciers geen controle over heeft: licenties, servers,...

Afhankelijkheden

- Het analysedocument omvat de afspraken tussen klant en leverancier en bevat dus precies wat er opgeleverd moet worden.
 - Merk op: in iteratieve projecten wordt er niet altijd een volledige analyse opgesteld
- In de ontwerpfase wordt ook een tijdlijn opgesteld door de project manager
- Een goed analysedocument draagt bij tot een goede documentatie
- Analyse zal gebruik maken van de kostenstructuur uit de eerdere fases

Aandachtspunten

- In de ideale omstandigheden zou de **inhoud van het analysedocument** volledig moeten overeenstemmen met de **oplossing** die opgeleverd wordt
 - In de praktijk zullen er steeds wijzigingen zijn: legale wijzigingen, wijzigingen in de strategie, organisatie,...
 - Kleine wijzigingen kunnen mee opgenomen worden: kleur wijzigen, andere plaats voor een veld,...
 - Grote wijzigingen moeten via de change-request procedure
- Het analysedocument moet een **sluitende overeenkomst** zijn tussen klant en leverancier
 - Juiste bewoordingen gebruiken: vermijd "zou kunnen", "zou moeten", "wij denken",....
 - Functionaliteiten moeten zo gedetailleerd mogelijk beschreven worden

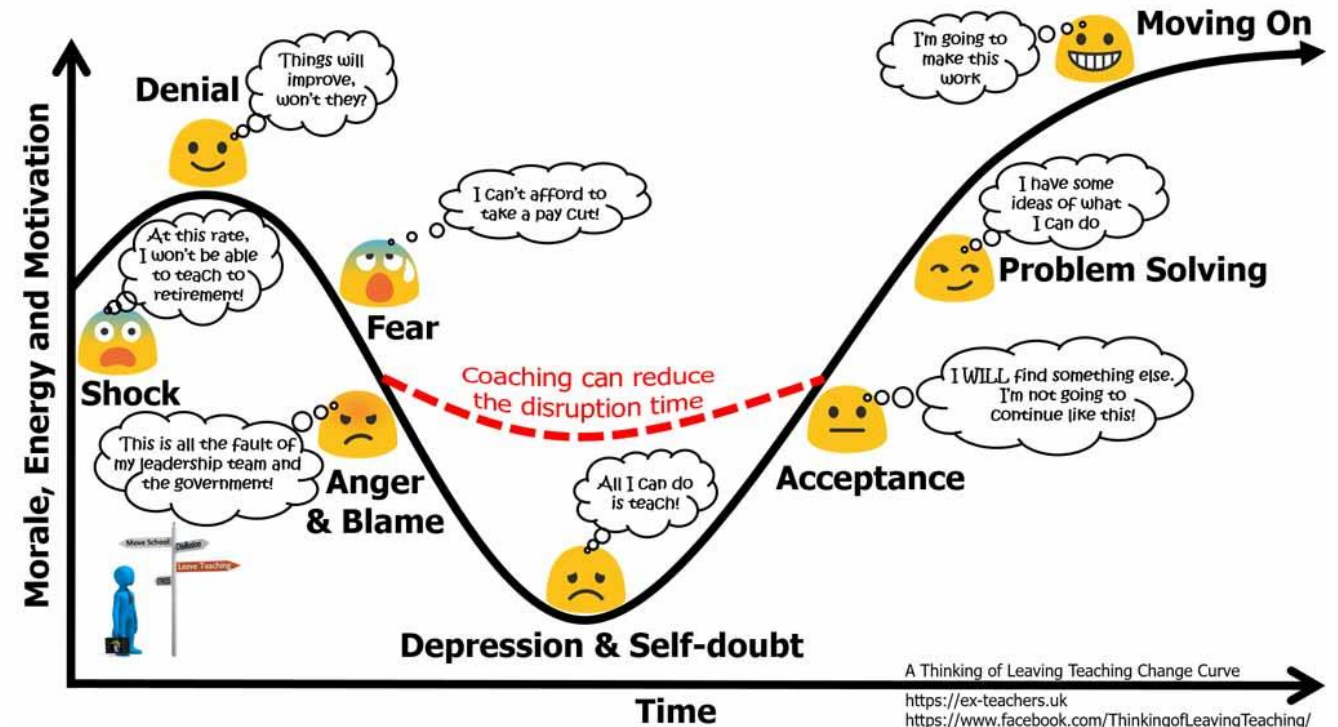
Aandachtspunten

- Het is belangrijk een **gedragen validatie** te krijgen
 - Zorg voor de juiste personen in de validatiemeeting
 - **Betrek de juiste eindgebruikers**
 - De juiste mensen betrekken zorgt voor een lagere weerstand



Aandachtspunten

- Zorg reeds in de ontwerpfase voor een **start van het “change management”**
 - Stel communicatieplan op
 - Organiseer opleidingen



Vragen?

