

Inhoudsopgave: Datum: 19/12/2023

[1 Hoofdstuk 1: Inleiding 2](#_Toc153874619)

[2 Hoofdstuk 2: Software-ontwikkelingsproces 3](#_Toc153874620)

[3 Hoofdstuk 3 Behoeftenanalyse 5](#_Toc153874621)

[3.1 Vereisten/behoeften 5](#_Toc153874622)

[3.2 Use cases 5](#_Toc153874623)

[3.3 Nut van use cases 6](#_Toc153874624)

[3.4 Use case diagram 6](#_Toc153874625)

[3.5 Activity diagram 7](#_Toc153874626)

[3.6 Testscenario’s 8](#_Toc153874627)

[4 Hoofdstuk 4: Domeinmodel 9](#_Toc153874628)

[4.1 Belang 9](#_Toc153874629)

[4.2 Onderdelen van een domeinmodel 9](#_Toc153874630)

[4.2.1 Associatieklasse 10](#_Toc153874631)

[4.2.2 Associatie 10](#_Toc153874632)

[4.2.3 Associatienaam 10](#_Toc153874633)

[4.2.4 Rolnaam 11](#_Toc153874634)

[4.2.5 Multipliciteit 11](#_Toc153874635)

[4.2.6 Generalisatie/specialisatie 11](#_Toc153874636)

[4.2.7 Aggregatie 12](#_Toc153874637)

[4.2.8 Compositie 12](#_Toc153874638)

[4.2.9 Associatieklasse 12](#_Toc153874639)

[4.3 Domeinmodel opstellen 13](#_Toc153874640)

[4.3.1 Identificeren van de kandidaatsklassen 13](#_Toc153874641)

[4.3.2 Selecteren van de conceptuele klassen 13](#_Toc153874642)

[4.3.3 Associaties leggen en/of aanpassen 14](#_Toc153874643)

[4.3.4 Identificeer attributen 14](#_Toc153874644)

[4.3.5 Optimalisatie 14](#_Toc153874645)

[5 Hoofdstuk 5: System Sequence Diagram (SSD) & Operation Contract (OC) 15](#_Toc153874646)

[5.1 (System Sequence Diagram) 15](#_Toc153874647)

[5.2 Operation Contract (OC) 18](#_Toc153874648)

[6 Hoofdstuk 6: UI en UX 19](#_Toc153874649)

[6.1 Geschiedenis 19](#_Toc153874650)

[6.2 Regels UI en UX 19](#_Toc153874651)

# Hoofdstuk 1: Inleiding

Hoe maken we een succesvol IT/software project?

Door te luisteren naar de klant, die centraal staat.

Met wie communiceren we dan om dit waar te maken?

* De **Klant** / eindgebruiker / opdrachtgever
* **Collega’s**
* Business - Management

Enkel door te vragen en te luisteren, kunnen we kwaliteitsvolle IT-projecten succesvol afleveren.

Dit word gedaan aan de hand van:

* Use cases
* Use case diagrams
* Activity diagrams
* Domeinmodel
* System Sequence Diagram (SSD) & Operation Contract (OC)

# Hoofdstuk 2: Software-ontwikkelingsproces

Bij het aangaan van een gesprek moeten we op drie hoofdzaken letten die de klant zegt:

* Afbeelding met tekst, schermopname, diagram

  Automatisch gegenereerde beschrijvingWat voor **functionaliteiten** wil je in je project hebben? Wie moet wat allemaal kunnen doen?
* Hoeveel **geld** is de klant bereid te investeren?
* Hoeveel **tijd** krijgen wij als developper? Wanneer wil de klant het project in zijn handen krijgen.

Hoe verloopt het ontwikkelingsgesprek tijdens en na het gesprek?

* We **luisteren** grondig naar de klant
* **Analyseer**, hoe en wat gaan we aan de slag
* We kijken naar het **ontwerp** van het project
* Het **implementeren** van de vereisten in het project
* **Testen/controleren** van het gemaakte project. Kijk voor bugs, alle mogelijke zaken die fout kunnen gaan uittesten …
* Regelmatig **overleg** met klant of hij het goed vindt en kijken voor aanpassingen.

Welke onderdelen bevat elk ontwikkelingsproces?

* Requirements (vereisten) verzamelen door te luisteren naar de opdrachtgever
* Requirements analyseren
* Een ontwerp (plan) maken
* Implementeren: het ontwerp uitvoeren
* Afbeelding met tekst, diagram, schets

  Automatisch gegenereerde beschrijvingTesten / controleren

Wat is de waterval en agile methode?

* **Waterval**: We voeren de stappen van ons ontwikkelingsgesprek in zijn volledige uit zonder terug te keren (**zonder iteratiemomenten**) en vanaf het moment dat we de klant zullen zien na ons gesprek, geven wij het eindproduct aan hun.

Afbeelding met tekst, schermopname, Parallel, lijn

Automatisch gegenereerde beschrijving

* **Agile**: Er worden een aantal **iteratieperiodes** (meestal 2-6 weken per periode) aangelegd. Dit zijn periodes waarin wij regelmatig langskomen om te zeggen wat hij vindt van de huidige status van het project. Hij geeft dan feedback weer en wij bouwen verder aan het project met feedback.

Wat betekent iteratief?

Je werkt in stappen, na elke stap lever je "werkende software" op, waarmee de gebruiker kan werken en krijg je feedback

Afbeelding met lijn, diagram, Perceel, helling

Automatisch gegenereerde beschrijvingWat betekent incrementeel?

bij elke stap bouw je verder aan de software

Welke tijdperiodes zijn beter voor agile en waterval?

* Kleine projecten (0 – 6 maanden): **Waterval**
* Grote project (6 – n-jaar): **Agile**

Wat zijn de belangrijkste principes achter de agilemethode?

* **Hoogste** **prioriteit**: klant tevreden
* Accepteren dat **gebruikerseisen** en **wensen** **veranderen**, ook later in het project.
* Business en ontwikkelaars werken dagelijks samen **“Face-to-face”** communicatie binnen team
* Levert geregeld **Werkende software** op, want dat is de eerste meting van vooruitgang

Iteratief-incrementele ontwikkeling:

* Afbeelding met diagram, tekst, Lettertype, lijn

  Automatisch gegenereerde beschrijvingThink big, develop small : werk in **iteraties**
* Een iteratie bevat steeds **dezelfde** **activiteiten** De **tijdsbesteding** aan iedere activiteit kan gaandeweg tijdens het project veranderen. (In het begin wellicht meer analyse …)
* Iteraties duren meestal **2 tot 6 weken**.

Agile stappenplan:

* De opdrachtgever **formuleert** het **probleem**.
* Analist noteert het verhaal en de **eisen** of **behoeften**
* Analist stelt een **Use Case Diagram** op met betrokken **rollen** en use cases
* Analist **vertaalt** het verhaal naar **use cases** (komt aan bod in olod Functional analyse)
* Analist stelt aan de hand van use case(s) het **domeinmodel** op
* Analist stelt **per** **use case** een **activity diagram** op met het oog op **testen**
* Analist stelt **per** **use case**, voor minstens het **normaal verloop**, een **Systeem Sequentie** **Diagram** (**SSD**) op met bijhordende **Operation Contracts (OC)**

Wat is UML (Unified modeling language)?

* Een modelleertaal om **objectgeoriënteerde** **analyses** en ontwerpen voor een **informatiesysteem** te kunnen maken
* UML zelf is **geen methode**, maar een notatiewijze die bij verschillende methodes (zoals Iteratief-Incrementeel) kan worden gebruikt.

Dit is een voorbeeld van UML:

Afbeelding met kaart, tekst, atlas

Automatisch gegenereerde beschrijvingWat zijn de voordelen van UML?

* Betere communicatie
* Betere Visualisatie
* Betere Transformatie = UML vergemakkelijkt de overgang:
* Analyse 🡪 ontwerp
* Ontwerp 🡪 programmeren
* Programmeren 🡪 testen

# Hoofdstuk 3 Behoeftenanalyse

## Vereisten/behoeften

Wat zijn functionele en non-functionele requirements?

* **Functionele**: beschrijft wat het systeem moet **kunnen** (starten, vooruit gaan …).
* **Non-functioneel**: beschrijft hoe het systeem moet **zijn** (snel, automatisch …)
  + + (het start met: Het systeem moet …).

## Use cases

Wat is een use case?

* Een **use case** omvat alle manieren waarop het **systeem** gebruikt kan worden om een **bepaald** **doel** voor een **bepaalde gebruiker** te behalen.
* Een complete set van use cases zoals weergegeven in het use case diagram geeft je alle zinvolle manieren om het systeem te gebruiken en illustreert de waarde die dit zal opleveren. Met **testmethodes** kunnen we dan ook uitwerken **hoe goed** onze **software** werkt.

Wat zijn de verschillende actoren van een use case?

* **Primaire actor:**
  + de persoon die het verhaal start.
* **Supporting actor:**
  + iemand die je ondersteunt maar niet deelneemt aan het volledige proces.
* **Stakeholder (profiteur):**
  + hetzelfde doel als de primaire actor, maar hij neemt niet meeneemt aan het volledige proces.
  + (vb. iemand rijdt met de auto, andere is passagier – zij gebruikt het systeem)

Sjabloon van een use case:

**Use case**: (Naam van de use case)

**Primaire actor** (de persoon die een **doel** heeft en het **verhaal start**)

**Stakeholder**(s) (**hetzelfde doel** bereiken maar **niet volledig meedoet** met het verhaal)

**Precondities** (Wat moet er gebeuren **voor** het **verhaal start**? – kan leeg zijn)

**Postcondities** (Wat moet er gebeuren **na** het **verhaal** is **beëindigt**?)

**Normaal verloop** (Het **verloopt** volgende oorspronkelijke **bedoelingen**)

**Alternatieve verlopen** (het **verloopt niet** volgens oorspronkelijke **bedoelingen**)

**Domeinregels:** (Nodige **regels** voor het implementeren van de software)

## Nut van use cases

Hoe werkt eenheid van planning?

* Bij het maken van een project alle stappen chronologisch organiseren van **belangrijkste tot minst belangrijke functie**.
* Schat de **ontwikkeltijd** in van elke stap.
* Verdeel die stappen in **iteraties** en zorg er voor dat elke iteratie ongeveer **even lang** is.

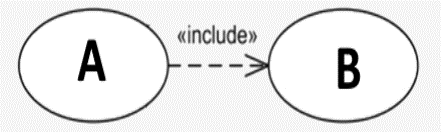
Wat is de rol van de developper en de tester?

* Developper: hoe kan ik het **maken**?
* Tester: hoe kan ik het **breken?**
  + kijken hoe sterk de code is
  + normaal verloop en alle alternatieve verlopen uittesten
  + De hoeveelheid testen staan opgeschreven in het **activity diagram**

## Use case diagram

Nut use case diagram:

Een ruwe schets van hoe de use case er gaat uitzien met **alle actoren** en **functionele requirements**.

Use case diagram: includes:

Use Case A voert ALTIJD use case B uit tijdens **normaal verloop.**

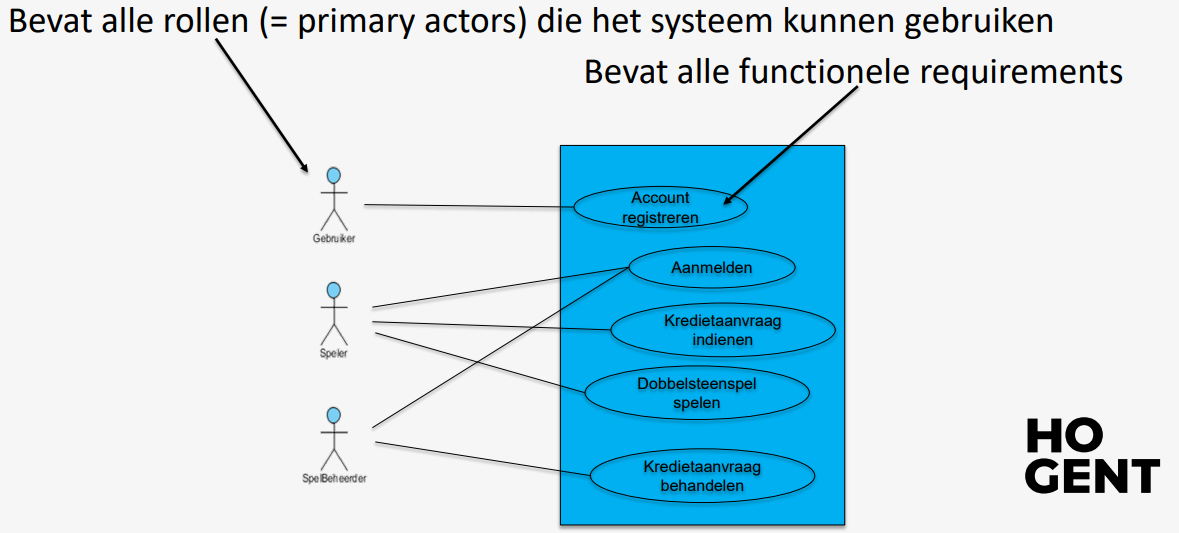
Afbeelding met cirkel, Lettertype, symbool, wit

Automatisch gegenereerde beschrijvingUse case diagram extends:

Use case A voert use case B uit tijdens een **alternatief verloop**

Voorbeeld van een use case diagram (informatie te halen uit de volledige use case):

**Welke actor doet wat?**



## Activity diagram

Wat zijn de onderdelen van een activity diagram (vervolg van een use case diagram)?

Afbeelding met tekst, schermopname, nummer, Lettertype

Automatisch gegenereerde beschrijving

Stappenplan Activity diagram:

* We schrijven **per Normaal Verloop Nummer** een **hokje** met de beschrijving er in.
* We kijken naar de **postconditie** en stoppen dit diagram.
* Vervolgens kijken we naar de **alternatieve verlopen** 
  + We teken de **diamant** **NA** het hokje met alternatieve verloop.
  + Keert het alternatief verloop terug naar een **bepaalde stap**? Dan teken we de diamant **VOOR** het hokje van de stap waar we weer van terugkeren.

Voorbeeld van een Activity diagram:

Starten

Afbeelding met tekst, schermopname, lijn, diagram

Automatisch gegenereerde beschrijving

Normaal verloop

Desecion node

Activity final node (stoppen)

Following Direction

Alternatief verloop

## Testscenario’s

**Je** **test** websites (zoals logins) en doet alle mogelijke pogingen met **normale en alternatieve** **verlopen** om te zien of alles werkt/goed reageert. Je test elke unieke weg.

Wat is een regressietest?

Als je een **nieuwe techniek** **toevoegt**, dan blijft de oude code evengoed werken.

Je werkt in een softwarebureau en werkt mee aan een aantal projecten als programmeur:

* Je moet kunnen tonen dat de functionaliteit werkt
* Je moet tonen dat de opeenvolging van functionaliteiten werken zoals gevraagd.

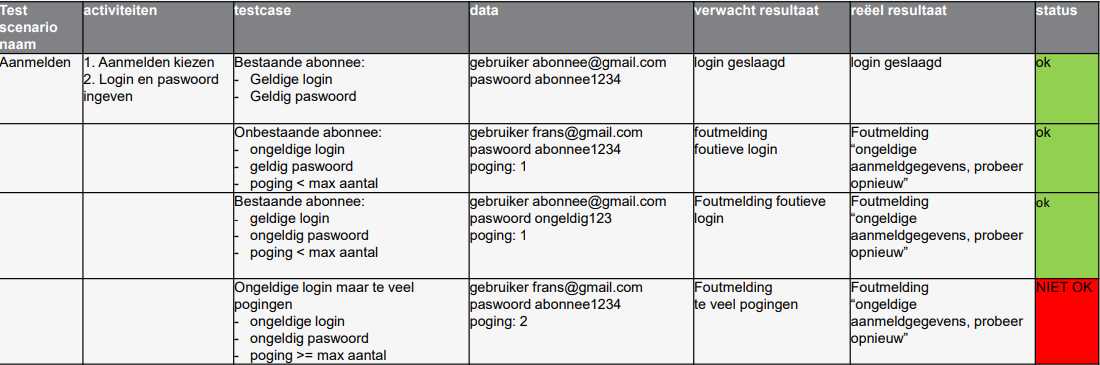
Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving

Testscenariotabel:

Op het voorbeeld van het **activity diagram** zijn 3 testscenario’s:

* Normaal Verloop
* Alternatief verloop 1
* Alternatief verloop 2



# Hoofdstuk 4: Domeinmodel

## Belang

Wat is het belang van een domeinmodel?

* **Domeinklassen** zijn heel belangrijk omdat je daarmee de **kern van het probleem** (en de oplossing) met behulp van een betrekkelijk klein aantal begrippen kunt **beschrijven**.
* Met behulp van het domeinmodel kan je nog eens **afstemmen** **met de klant** of **je** heel goed **begrijpt** hoe alles ineen zit (associaties tussen klassen).
* Afbeelding met tekst, Lettertype, schermopname, groen

  Automatisch gegenereerde beschrijvingConceptuele domeinklassen vormen dan ook een belangrijk uitgangspunt en **inspiratiebron** voor het ontwikkelen van de **softwareklassen** (= klassen met datatypes en gedrag) waaruit een domeinlaag in een applicatie is opgebouwd.

Definitie domeinmodel?

Het domeinmodel is een **visuele representatie** van concepten uit de werkelijkheid en hun onderlinge relatie.

## Onderdelen van een domeinmodel

Afbeelding met tekst, diagram, schermopname, Plan

Automatisch gegenereerde beschrijvingHet domeinmodel is een UML-klassendiagram waarin volgende onderdelen kunnen gebruikt worden:

### Associatieklasse

Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijvingHoe word een klasse beschreven?

**KlasseNaam**: Met een hoofdletter geschreven, enkelvoud en zelfstandig naamwoord.

**KlasseAttributen**: Geen berekeningen.

Bij een domeinmodel moet je kijken naar **één** specifiek voorwerp.

(zoals het vliegtuig waarbij je niet kijkt naar de bestemming)

### Associatie

Een associatie is een verband tussen instantie van klassen

**Regels:**

* Enkel als het verband relevant en **zinvol** is
* Aanduiden door **tussen** **2** conceptuele klassen
  + **Meerdere associaties** tussen 2 conceptuele klassen mogelijk
  + Associatie tussen eenzelfde conceptuele klasse kan (= reflexieve associatie)

Afbeelding met tekst, schermopname, diagram, Lettertype

Automatisch gegenereerde beschrijvingWat is een reflexieve associatie?

* Een associatie met zichzelf.
  + Vb. AssociatieKlasse: Persoon
  + Eén persoon heeft één stel ouders.

Associatie(s) tussen klasse(n):

Afbeelding met tekst, lijn, Lettertype, diagram

Automatisch gegenereerde beschrijving

### Associatienaam

Afbeelding met tekst, schermopname, diagram, Lettertype

Automatisch gegenereerde beschrijvingEen associatienaam is een verduidelijking van de associatie.

**Regels:**

* Werkwoord
* Begint met hoofdletter
* Zorg voor een duidelijke zin
* Lezen van links naar rechts of van boven naar onder
* Indien leesrichting afwijkt, dit aanduiden met of

Afbeelding met tekst, schermopname, Lettertype, diagram

Automatisch gegenereerde beschrijving

### Rolnaam

Een rolnaam is een verduidelijking bij de rol van een conceptuele klasse in een associatie.

**Afbeelding met tekst, schermopname, diagram, lijn

Automatisch gegenereerde beschrijvingRegels:**

* Kleine letters
* Zelfstand naamwoord
* Staat **naast** de conceptuele klasse en associatie bij dewelke de **rolnaam** hoort.
* **Mag weggelaten worden** als de associatienaam duidelijk genoeg is.

### Multipliciteit

**Afbeelding met tekst, schermopname, diagram, Rechthoek

Automatisch gegenereerde beschrijving**Een multipliciteit is een aanduiding van hoeveel instanties van een conceptuele klasse verbonden zijn via de associatie met een andere.

**Regels:**

* Per rol een multipliciteit
* **Minimum** aantal is 0,1 of een exact getal
* **Maximum** aantal is 1, een exact getal of \* (oneindig veel)
* Aangeduid als: min aantal … max aantal bij de rol
* Als minimum = maximum mag je enkel maximum vermelden
* Notatie 0…\* mag vervangen worden door enkel \*
* Opsomming van exacte waarden is toegelaten.

### Generalisatie/specialisatie

= 1 conceptuele basisklasse (=**generalisatie**) met alle gemeenschappelijk attributen die door afgeleide klassen (=**specialisatie**) worden overgeërfd.

**Regels:**

* **Afbeelding met schermopname, tekst, diagram, lijn

  Automatisch gegenereerde beschrijving**Generalisatieklasse bevat alle gemeenschappelijke attributen.
* Specialisatieklasse erft de gemeenschappelijke attributen en associaties over
* Specialisatieklasse kan eigen attributen en associaties hebben
* Aangeduid door generalisatieklasse  specialisatieklasse
* Enkel gebruiken als je kan zeggen: “… **is een …**”.

**Een manager heeft de leiding over minstens 1, max. 5 filialen**

**Afbeelding met tekst, schermopname, diagram, lijn

Automatisch gegenereerde beschrijving**

### Aggregatie

Aggregatie is een deel/geheel relatie tussen 2 conceptuele klassen op een gelijk niveau.

**Regels:**

* Aanduiden met een naast de klasse die het geheel vormt.
* De **minimummultipliciteit** langs de zijde is steeds 0 of 1.
* Een deel kan op zich ook bestaan. Als het geheel verdwijnt blijft het deel bestaan en omgekeerd.
* Enkel gebruiken als je kan zeggen: “… **heeft** …”.

Afbeelding met tekst, diagram, lijn, Lettertype

Automatisch gegenereerde beschrijving

**NIET** met aggregaties overdrijven!

**Tips:**

* Bij twijfel, gewone associatie
* Is er een ‘fysieke” deel/geheel relatie?
  + Indien ja dan eerder aggregatie
  + Indien nee dan eerder gewone associatie.

### Compositie

Compositie is een strengere vorm van aggregatie. Drukt uit dat een instantie van een klasse is opgebouwd uit onderdelen die instanties zijn van andere klassen.

**Regels:**

* Aanduiden met een naast de klasse die het geheel vormt.
* De multipliciteit langs de zijde is steeds **exact** 1.
* Een instantie van het geheel is verantwoordelijk voor de creatie en vernietiging van de delen.
* Enkel gebruiken als je kan zeggen: “… **bestaat uit** …”.

Afbeelding met tekst, Lettertype, lijn, schermopname

Automatisch gegenereerde beschrijving

### Associatieklasse

Een associatieklasse is een conceptuele klasse die attributen bijhoudt die bij een bepaalde associatie horen.

**Regels:**

* Levensduur van de instanties van de associatieklasse is afhankelijk van de associatie
* Afbeelding met tekst, schermopname, lijn, diagram

  Automatisch gegenereerde beschrijvingKan zelf associaties hebben met andere klassen
* GEEN extra multipliciteiten
* Verbonden met de associatie dankzij - - -

Stappenplan domeinmodel opstellen:

Een domeinmodel wordt iteratief opgesteld aan de hand van een context en/of use case(s). Bij het verwerken van een context of een use case volgen we steeds volgende stappen:

1. Identificeren van de **kandidaatsklassen**
2. Selecteren van de **conceptuele** **klassen**
3. **Associaties** leggen en/of aanpassen
4. **Attributen** toevoegen, verplaatsen en/of schrappen

## Domeinmodel opstellen

Hoe stel je een domeinmodel op? (30% van de examenpunten!!!!!!!!!)

* Identificeren van de kandidaatsklassen
* Selecteren van de conceptuele klassen
* Associaties leggen en/of aanpassen
* Attributen toevoegen, verplaatsen en/of schrappen

### Identificeren van de kandidaatsklassen

Wat is het doel?

* Zo veel mogelijk conceptuele klassen selecteren
* **Afbeelding met tekst, schermopname, lijn, nummer

  Automatisch gegenereerde beschrijving**Inzicht krijgen in het probleemdomein

Wat is de werkwijze?

Een lijst van **ALLE** **zelfstandige naamwoorden** maken

Afbeelding met tekst, Lettertype, schermopname, informatie

Automatisch gegenereerde beschrijving

### Selecteren van de conceptuele klassen

Wat is het doel?

Enkel **conceptuele klassen overhouden** voor het domeinmodel

Hoe gebeurt de werkwijze?

* Stel voor elke kandidaatsklasse volgende 3 vragen:
  + Speelt de klasse een zelfstandige rol in het probleemdomein?
  + Willen we iets van de kandidaat weten of willen we dat hij iets doet?
  + Heeft de klasse een duidelijke verantwoordelijkheid?
    - **Is het antwoord telkens ja, dan is de kandidaatsklasse een conceptuele klasse**

**Afbeelding met tekst, schermopname, Lettertype, lijn

Automatisch gegenereerde beschrijving**

Daarna schrijven we ze ook echt op:

**Afbeelding met schermopname, water, tekst, Lettertype

Automatisch gegenereerde beschrijving**

### Associaties leggen en/of aanpassen

Wat is het doel?

* Vinden van zinvolle associaties tussen de gekozen klassen
* Vinden van bijkomende zinvolle associaties tussen reeds opgenomen klassen (bij iteraties)

Wat is de werkwijze?

* Beschouw alle mogelijke paren van klassen
* Herlees de **context of use-case** en zoek naar **structurele verbanden** die voor een zekere tijd moeten **aangehouden worden**, let vooral op
  + Werkwoorden, MAAR
    - Wijzen soms op gedrag (= operaties)
    - Afbeelding met tekst, schermopname, Lettertype, nummer

      Automatisch gegenereerde beschrijvingZijn soms een synoniem voor bestaande associatie
  + – Bezittelijke voornaamwoorden
    - b.v.: zijn, haar, ...

### Identificeer attributen

Wat is het doel?

• Vinden van relevante attributen voor de gekozen klassen

• Vinden van bijkomende relevante attributen voor reeds opgenomen klassen (bij iteraties)

Wat is de werkwijze?

* Beschouw de lijst uit stap 1 met ‘vermoedelijke attributen’
* Haal uit de tekst:
  + Dingen die voorgesteld worden via een simpele waarde
    - b.v.: datum, getal, ja/nee, karakter, …
  + Woorden waarvoor je ja kan antwoorden op “Is … een eigenschap van een andere klasse” wijzen vaak op een attribuut
  + Bijvoeglijke voornaamwoorden wijzen vaak op attributen

### Optimalisatie

Wat is het doel?

• Verwijderen van overbodige klassen en associaties

• Domeinmodel minder complex maken

Wat is de werkwijze?

* Verwijder klassen zonder attributen **die geen meerwaarde bieden in het schema** Ze hebben ook geen relevante/noodzakelijke associaties met andere klassen
* Afbeelding met tekst, schermopname, diagram, Rechthoek

  Automatisch gegenereerde beschrijvingVorm **specialisatie/generalisatie om naar één klasse** met een attribuut type indien de **specialisaties geen eigen attributen** hebben of aparte associaties met andere klassen. De attributen van de generalisatie worden behouden Resultaat
* Resultaat: Volledig domeinmodel

# Hoofdstuk 5: System Sequence Diagram (SSD) & Operation Contract (OC)

## (System Sequence Diagram)

Wat is een SSD (Systeemsequentiediagram)?

* Toont interactie tussen actor(en) en het systeem
* Bv. het toont hoe de werking gebeurt tussen een klant en een bankautomaat.

Wat leren we er uit?

* SSD geeft **inzicht** in de **systeemeisen** die een eindgebruiker verwacht
* SSD toont de **systeem boodschappen/operaties** = systeemgedrag
* SSD toont **wat het systeem moet doen**, niet hoe (black box)
* SSD bevat de systeemoperaties. Als een actor communiceert met het systeem gebeurt dit altijd via zo’n systeemoperaties. Dus alle mogelijk communicatie die een actor kan hebben met het systeem, wordt voorgesteld door een systeemoperatie op een SSD
* Als we dan al die gevonden systeemoperaties proberen uit te werken (ontwerp + implementatie) dan krijgt onze software vorm. Het is dus eigenlijk een startpunt voor het ontwerp

Wat is de definitie van een SSD?

Een SSD is een sequentiediagram dat alle interacties tussen de actor en het systeem van 1 use case scenario weergeeft.

Wat zijn de onderdelen van het SSD?

|  |  |
| --- | --- |
| Term | Werking in schema/symbool |
| Deelnemer | de primaire actor  en één systeem |
| Levenslijn | Het moment wanneer het systeem niks doet |
| Systeemoperatie | Lijn waarbij de primaire actor iets doet |
| Antwoord op systeemoperatie | Lijn waarbij het systeem data teruggeeft |
| Herhaling | Één grote doos rond het stuk dat in herhaling komt: |

Extra verklaring onderdelen:

|  |  |
| --- | --- |
| Definitie | Verklaring |
| Actor (deelnemers) | Naam = rol van primary actor uit de use case  Staat steeds LINKS in het SSD |
| Systeem (deelnemers) | Staat steeds RECHTS in het SSD |
| Actor (levenslijn) | – Voorgesteld als blokje  – Is ALTIJD actief! |
| Actor (systeem) | Indien geen systeemoperatie (= niet actief):  Voorgesteld als stippellijn  Indien systeemoperatie (= actief)  • Voorgesteld als blokje  • APART blokje per systeem operatie |
| Systeemoperaties | Werkwoord  Afsluiten met ()  Parameters indien nodig   * Moet duidelijk zeggen wat er gevraagd wordt aan het systeem, wat de bedoeling is dat het systeem doet. (Niet wat de actor doet!) * Moet de bedoeling op een hoog abstractieniveau weergeven – niet in termen van inputtechnologie (Geen datatypes) * Sluit aan bij het gebruikte jargon in het domeinmodel * Formuleert een opdracht « doeIets » (-> starten met de stam van een werkwoord) * Gegevens die meegegeven worden aan het systeem worden tussen ronde haakjes weergegeven. We noemen dit parameters. Deze parameters zijn informatie die het systeem nodig heeft om ‘zijn taak’ uit te voeren. |
| Antwoord op de  systeemoperatie | * ENKEL na communicatie van actor naar systeem als het systeem iets teruggeeft Benoemen wat systeem teruggeeft * Enkel data uit het domeinmodel   + Regels   – Zelfstandig naamwoord  – GEEN operaties  – GEEN datatypes |
| Herhaling | We plaatsen een rechthoek rond het deel dat in herhaling komt |
| Externe use case | Zo weer te geven: |

Wat is het stappenplan om een SSD op te stellen?

* Kies een use case scenario
* Voeg de primary actor toe en het systeem
* Voeg de systeemoperaties toe
* Voeg herhalingen toe

## Operation Contract (OC)

Wat is een operation contract?

* Is onderdeel van de analyse
  + Is een document dat beschrijft wat een operatie moet bereiken
  + Beschrijft de veranderingen/gevolgen in het systeem ten gevolge van die operatie
  + Is een expliciete beschrijving van het effect van die operatie, beschrijft niet hoe de veranderingen tot stand gekomen zijn.
  + **Je gebruikt dit vanaf dat je weet als er iets kan veranderen in je SSD.**

Onderdelen OC:

|  |  |
| --- | --- |
| Definitie | Verklaring |
| Contract | duidelijke naam/omschrijving voor het OC |
| Operation | naam van de systeemoperatie uit het SSD |
| Cross reference | naam van de betrokken use case(s |
| Preconditions | beschrijving van de staat van het systeem VOOR de uitvoering van de operatie. Enkel de niet triviale zaken opnemen |
| Postconditions | gedetailleerde beschrijving van de objecten die gewijzigd zijn NA de uitvoering van de operatie |

# Hoofdstuk 6: UI en UX

## Geschiedenis

Eerste Computer:

De 1e computer was extreem groot en kon de 4 basisbewerkingen.

Afbeelding met machine, overdekt, projector, uitrusting

Automatisch gegenereerde beschrijvingEerste evolutie 1970:

* Mainframes
* Focus op snelheid
* Voor experten
* Voor administratie en productie
* Weinig nood aan usability

Tweede evolutie 1980:

* Afbeelding met tekst, computer, Elektronisch apparaat, elektronica

  Automatisch gegenereerde beschrijvingEerste PC (Mac)
* Productivity Tools (vb. tekstverwerkers)
* Minder technisch
* Usability doet intrede

Afbeelding met tekst, grafische vormgeving, poster, boek

Automatisch gegenereerde beschrijvingDerde evolutie 1990:

* Windows 3.0
* Kritische gebruikers
* vanaf 1995: internet

Afbeelding met elektronica, tekst, Uitvoerapparaat, computer

Automatisch gegenereerde beschrijvingVierde evolutie 2000:

* PC doet intrede in de huiskamer
* Eerste laptops

Vijfde evolutie 2020:

Afbeelding met Mobiele telefoon, tekst, gadget, Elektronisch apparaat

Automatisch gegenereerde beschrijving

## Regels UI en UX

UI:

Het moet **efficiënt** zijn (1 keer klikken ipv 5 keer)

Iedereen moet **tevreden** zijn

Het moet **effectief** zijn (het moet vlot gaan)

Wireframes:

* Ruwe schets
* geen kleuren
* geen details
* geen echte dat
* Ruwe plaatsing elementen
  + **Lage getrouwheid** in juistheid

Mock-up:

* Kleuren
* Alle details
* Echte data
* Precieze plaatsing elementen
  + **Hoge getrouwheid** in juistheid. kleuren, foto’s, juiste lettertype …

Concetpueel model vs mentaal model:

Hoe het moet werken volgens systemen en hoe de gebruiker denkt dat het werkt (Windows 8 taakbalk weggehaald). System interface:

Responsive vs adaptive:

* Responsive: **Dezelfde UI steeds kleiner** maken voor verschillende devices.
* Adaptive: **verschillende designs** maken aan de hand van het device.

Device experience – posture:

* **Lean back**:
  + Het is **duidelijk** hoe je je navigeert dus de gebruiker denkt niet na.
  + Volledige scherm wordt benut
  + Grote knoppen en tekst
  + Minimalistische visuele stijl (Less is even more!)
  + Veel visuele feedback
  + Ondersteuning bij gebruikersinvoer
* **Lean forward**:
  + Tekst mag kleiner, want hij is **intensief** bezig met de software.
  + Veel informatie op 1 scherm
  + Weinig visuele feedback
  + Kleinere knoppen en tekst
  + Beperkte mogelijkheden bij gebruikersinvoer
* **Quick bursts**:
  + **Eerst** is hij **intensief** bezig, **daarna** op een **luie** manier (wekker zetten)
  + sporadisch gebruik
  + maar wel intensief

Afbeelding met diagram, schets, lijn, Rechthoek

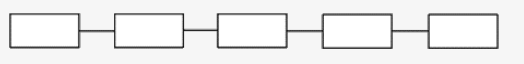
Automatisch gegenereerde beschrijvingBoomstructuur (gebruikt systeem voor een website):

Bij een boomstructuur kom je ergens terecht en binnenin die structuur kan je nog ergens is klikken … maar ook teruggaan.

Afbeelding met tekst, schermopname, Besturingssysteem, multimedia

Automatisch gegenereerde beschrijvingNavigatie:

Een navigatie als dit kan: ‘back’ en ‘next’ zijn:



Sterstructuur:

Afbeelding met tekst, schermopname, software, Computerpictogram

Automatisch gegenereerde beschrijvingApplicatie met centraal scherm en je kan alles instellen aan de hand van opties (PowerPoint, Excel …)Afbeelding met diagram, lijn, Plan, ontwerp

Automatisch gegenereerde beschrijving

Afbeelding met tekst, software, Computerpictogram, Webpagina

Automatisch gegenereerde beschrijvingMultipaneel:

Afbeelding met diagram, lijn, Plan, Technische tekening

Automatisch gegenereerde beschrijving

Richtlijnen 1 pagina:

* **Beperk de mentale belasting:**

(rekenmachine horizontaal en verticaal houden op gsm) – toon niet alles als 1e. Informatie toevoegen wanneer je het nodig hebt. Informatie tonen op de juiste plaats, op het juiste moment.

* **Less is more (meer is beter)**
* **Design for error:**

Toon de correcte melding met een eenvoudige boodschap.

* **Geef de gebruiker de controle:**

pauzeknop tijdens installeren, alles kunnen cancelen, maak duidelijke keuzemenuutjes …

* **Snelheid:**

maak het zo snel mogelijk. Tenzij je het echt nodig hebt, dan moet je helaas de wachttijd wel gebruiken. Vermijd overbodige media. Thumbnail.

* **Duidelijk en bondig schrijven:**

Zorg dat je kop duidelijk is samen met alle geschreven teksten doorheen de pagina’s

* **Gepaste kleuren:**

Drie hoofdkleuren en drie varianten van de hoofdkleuren (dus 6 kleuren maximaal).