

SOFTWARE DESIGN ESSENTIALS

INLEIDING TOT UML
(UNIFIED MODELING LANGUAGE)



WAT IS UML?

- UML (**Unified Modeling Language**) is een visuele modelleertaal die wordt gebruikt om softwarearchitecturen, systemen en processen te ontwerpen.
- Het helpt ontwikkelaars, analisten en andere stakeholders om **complexe software** op een gestructureerde en begrijpelijke manier in kaart te brengen.

WAAROM UML GEBRUIKEN?

- **Duidelijke communicatie** tussen teamleden en stakeholders
- **Visuele weergave** van de structuur en werking van een system
- **Vermindert misverstanden** door een gestandaardiseerde notatie
- **Ondersteunt softwareontwikkeling** in zowel traditionele als Agile omgevingen

VERSCHILLENDEN UML- DIAGRAMMEN

- UML bevat verschillende soorten diagrammen, die kunnen worden onderverdeeld in:
 1. **Structurele diagrammen** – tonen de **statische aspecten** van een systeem (zoals klassen en objecten).
 1. **Class diagram** (klassen en hun relaties)
 2. **Domain Model** (concepten en relaties in het domein, zonder technische details)
 2. **Gedragsdiagrammen** – tonen de **dynamische aspecten** van een systeem (zoals interacties en processen).
 1. **Use case diagram** (functionaliteit vanuit gebruikersperspectief)
 2. **Interaction Sequence diagram** (interactie tussen objecten in een bepaalde volgorde)
 3. **Activity diagram** (stroom van activiteiten en beslissingen)
 4. **State machine diagram** (verschillende toestanden van een object)

HOE STARTEN MET UML?

- De eerste stap in UML-modellering is **het begrijpen van de functionele eisen** van het systeem. Dit gebeurt meestal met een **use case diagram**, waarin wordt weergegeven welke interacties gebruikers hebben met het systeem.



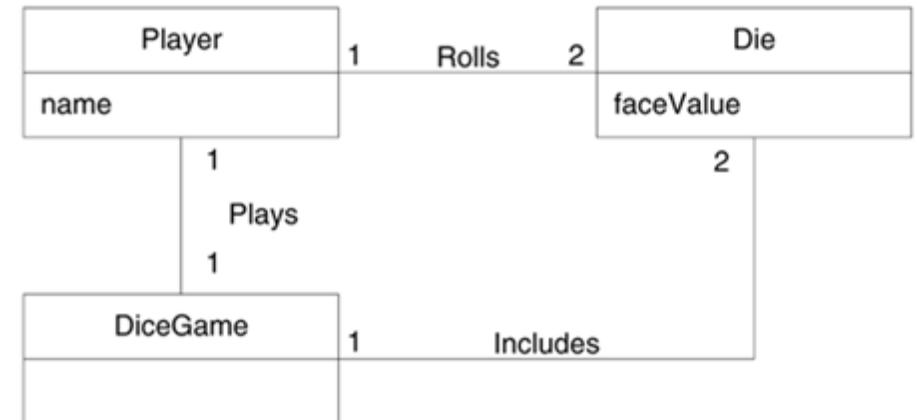
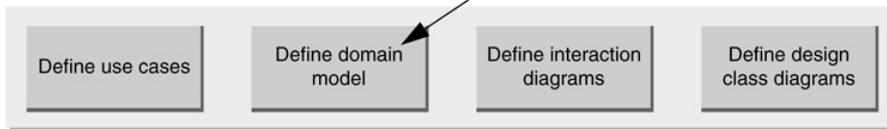
VOORBEELD

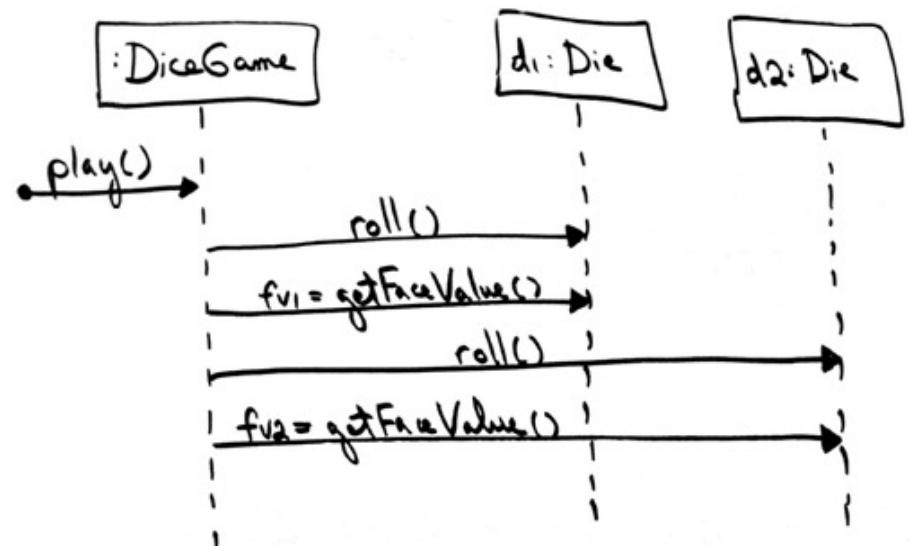


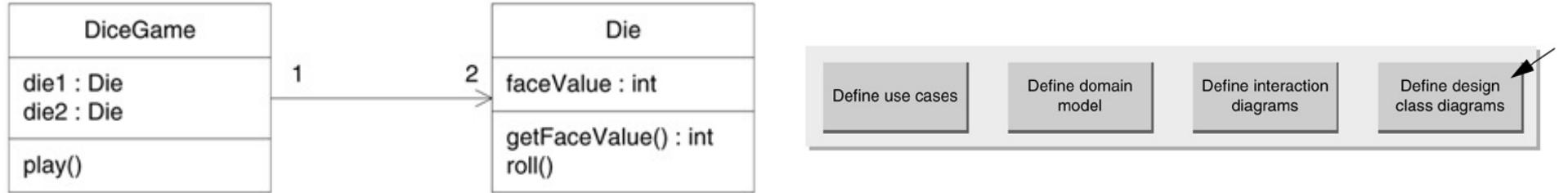
VOORBEELD: DOBBELSTENEN SPEL

- Scenarios over hoe gebruikers de software gaan gebruiken
 - Stories of visuele notatie
-
- *Play a Dice Game: Player requests to roll the dice. System presents results: If the dice face value totals seven, player wins; otherwise, player loses.*









STATISCHE VIEW OP DE KLASSESTRUCTIJR EN RELATIES TUSSEN COMPONENTEN.

SOFTWARE DESIGN ESSENTIALS

HOOFDSTUK 2: USE CASES



USE CASES



USE CASE

Een use case beschrijft een systeem vanuit het gebruikersperspectief.

Het beschrijft de actor, de initiator van de interactie, en het systeem zelf als een opeenvolging van eenvoudige stappen.

Actoren staan in dialoog met het systeem om een bepaald doel te bereiken.

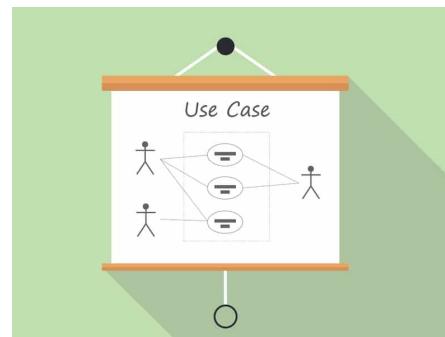
Actoren kunnen eindgebruikers, andere systemen of hardware zijn..

USE CASES: INTERACTIE TUSSEN ACTOREN EN HET SYSTEEM

- Een Use Case beschrijft hoe een actor (gebruiker of extern systeem) interacteert met een systeem om een specifiek doel te bereiken.
- Het is als een kort verhaal dat het functionele gedrag van het systeem uitlegt, zonder technische details.

VOORBEELD: AFREKENEN AAN DE KASSA

- **Situatie:** Een klant wil afrekenen bij de kassa.
- **Actie:**
 1. De kassier scant elk product en registreert het in het systeem.
 2. Het systeem toont de productinformatie en het actuele totaalbedrag.
 3. De klant kiest een betaalmethode en voert de betalingsgegevens in.
 4. Het systeem verwerkt de betaling en slaat deze op.
 5. De voorraad wordt automatisch bijgewerkt.
 6. Het systeem genereert een kassabon.
- **Resultaat:**
 - De klant ontvangt de bon en verlaat de winkel met de gekochte producten.



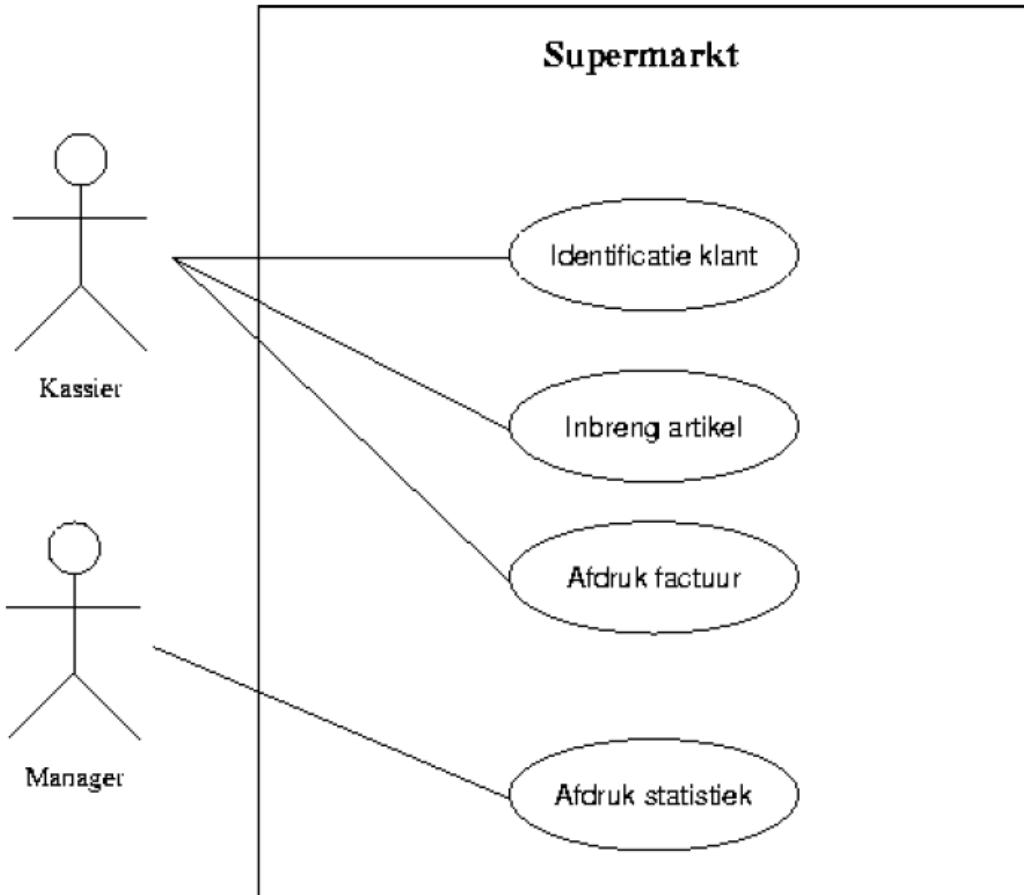
WAAROM EEN USE CASE?

- Maakt functionele vereisten begrijpelijk voor zowel **ontwikkelaars** als **niet-technische stakeholders**.
- Helpt bij het identificeren van **belangrijke interacties** en **uitzonderingen** in het proces.
- Vormt de basis voor verdere modellering, zoals **Use Case Diagrammen** en **Sequence Diagrams**.

UITWERKEN VAN EEN USE CASE

- Bij het uitwerken van een **Use Case** ligt de nadruk op een **duidelijke en gestructureerde tekstbeschrijving** van de interactie tussen de actor en het systeem.
- **Belangrijkste focus:**
 - **Duidelijke tekstbeschrijving** van het scenario, inclusief stappen en uitzonderingen.
 - **Beschrijf het doel en de interactie** van de actor met het systeem.
 - **Vermijd technische details;** focus op wat het systeem doet, niet hoe.
- **Visuele voorstelling (Use Case Diagram):**
 - Een **Use Case Diagram** biedt een handig overzicht van de betrokken actoren en hun interacties met het systeem.
 - Hoewel nuttig, is het **minder gedetailleerd** dan een tekstuele beschrijving en dient het vooral als **aanvulling** op de use case.

USE CASES DIAGRAM



GEBRUIKERSGERICHT DENKEN BIJ USE CASES

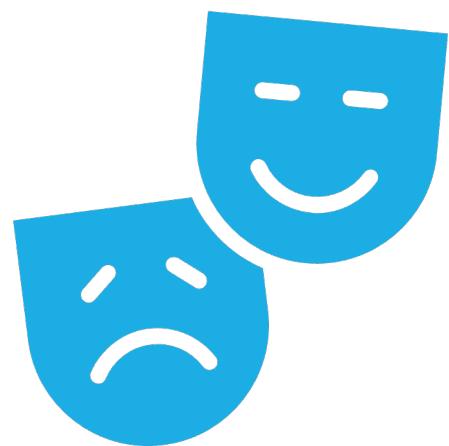
- Focus op de gebruikers en hun behoeften, niet alleen op de functies van het systeem.
- Niet de werking van het systeem centraal stellen, maar hoe het systeem waarde toevoegt voor de gebruikers.
- Benadruk de voordelen voor belanghebbenden: Hoe helpt het systeem hen hun doelen te bereiken?

- Sleutelvraag: "Wat levert dit systeem op voor de gebruiker?" in plaats van "Welke functies heeft het systeem?"

BELANGRIJKE BEGRIPPEN BINNEN USE CASES

- **Actor:** Een entiteit met **gedrag** die interacteert met het systeem. Dit kan een **persoon, een ander systeem of een organisatie** zijn.
- **Use Case:** Een beschrijving van hoe een actor het systeem gebruikt om een specifiek **doel te bereiken**. Dit omvat zowel **succesvolle als mislukte scenario's** die kunnen optreden tijdens de interactie.
- **Scenario:** Een specifieke reeks **stappen en interacties** tussen actoren en het systeem binnen een Use Case. Dit wordt ook wel een **Use Case Scenario** genoemd.

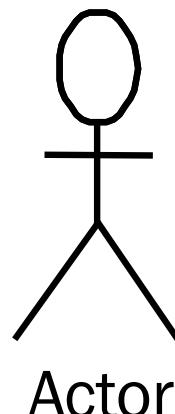
- **Samenvatting:** Een Actor voert een **Use Case** uit, die bestaat uit meerdere mogelijke **scenario's**, afhankelijk van hoe de interactie verloopt.



ACTOR

ACTOR

- **Definitie:**
 - Een **actor** vertegenwoordigt een **rol** die een persoon, een hardware-apparaat of een ander systeem speelt in relatie tot het systeem.
- **Kenmerken:**
 - Een **actor** is **extern** ten opzichte van het systeem.
 - Actoren **interageren** met het systeem, maar maken er geen deel van uit.
 - Eén persoon of systeem kan **meerdere rollen** vervullen (bijvoorbeeld een gebruiker die zowel klant als beheerder is).
- **UML-notatie voor een Actor:**
 - In een **Use Case Diagram** wordt een actor voorgesteld als een **stick figure (poppetje)**.
 - De naam van de actor wordt **onder de figuur** geplaatst.



SOORTEN ACTOREN IN USE CASES



PRIMARY ACTOR (PRIMAIRE ACTOR)

Definitie:

- De **belangrijkste gebruiker** van het systeem, degene die een actie start om een doel te bereiken.

Kenmerk:

- De initiator van de interactie met het systeem.
- Heeft een direct belang bij het eindresultaat van de use case.
- Kan een persoon of een ander systeem zijn.

Voorbeeld:

- Een **klant** die producten bestelt in een webshop.
- Een **kassamedewerker** die een aankoop afrekent.

SUPPORTING ACTOR (ONDERSTEUNENDE ACTOR)

Definitie:

- Een actor die het **primaire proces ondersteunt**, maar zelf geen interactie initieert.

Kenmerk:

- Levert **informatie of diensten** aan het systeem.
- Kan het systeem helpen een actie correct uit te voeren.

Voorbeeld:

- Een **betaalsysteem** dat de betaling van een klant verwerkt.
- Een **voorraadsysteem** dat controleert of een product beschikbaar is.

OFFSTAGE ACTOR (INDIRECTE ACTOR)

Definitie:

- Een actor die **niet direct** betrokken is bij de interactie, maar **wel belanghebbende is** bij de uitkomst.

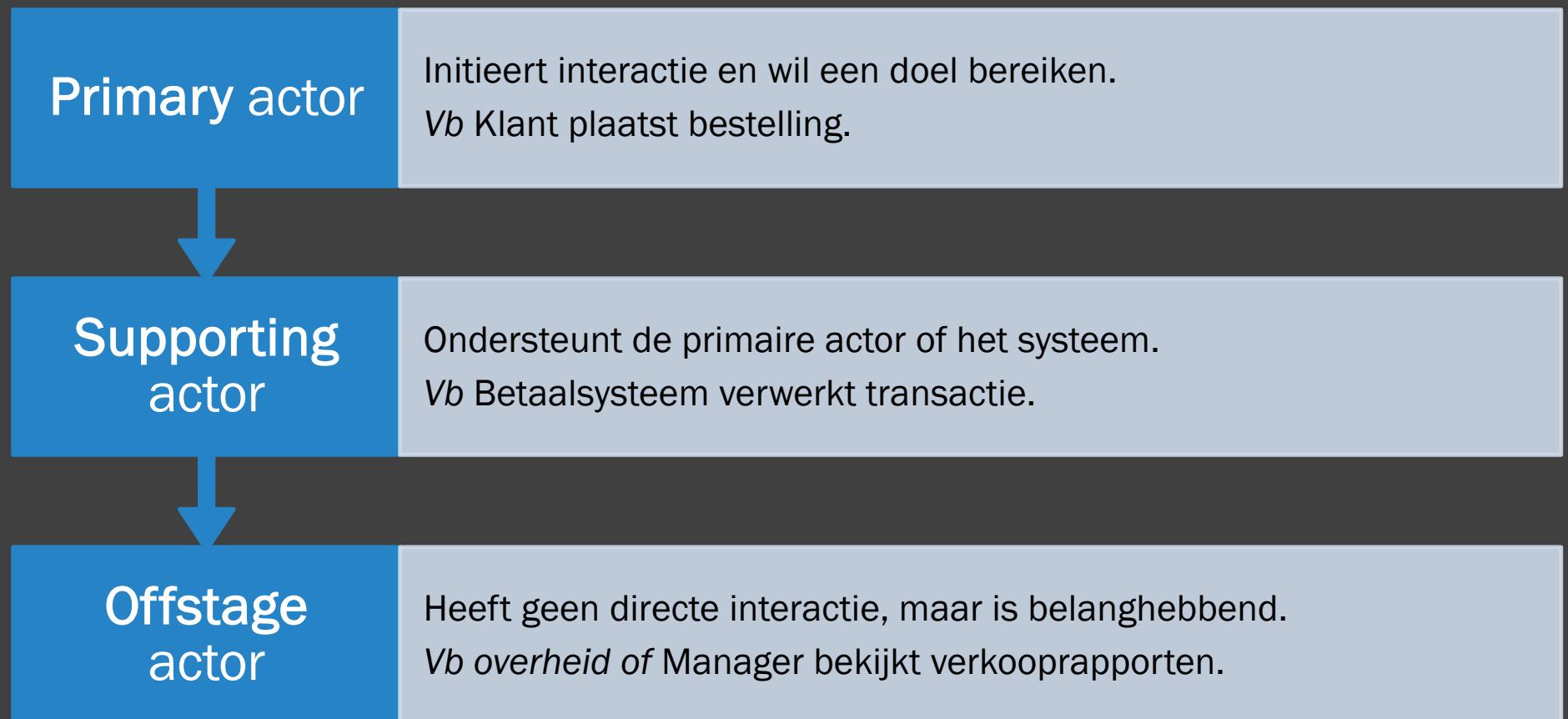
Kenmerk:

- Heeft geen directe interactie met het systeem.
- Kan wel meldingen ontvangen of indirect beïnvloed worden door de uitvoering van de use case.

Voorbeeld:

- **Een manager** die rapporten ontvangt over verkochte producten.
- **Een toezichthouder** die logs controleert op naleving van regelgeving.

SAMENVATTING



A stack of four vintage suitcases of various sizes and materials. The top suitcase is made of brown leather with a large brass handle and a leather strap. The second suitcase from the top is made of a textured fabric with a dark leather strip and a brass lock. The third suitcase is made of a striped fabric with a tan leather handle and a brass lock. The bottom suitcase is made of a similar striped fabric. The word "USE CASE?" is overlaid in white text on the middle suitcase.

USE CASE?

WAT IS EEN USE CASE?

Een **Use Case** specificeert een reeks **acties** die door het systeem worden uitgevoerd om een **concreet resultaat** te bereiken. Dit resultaat is relevant voor **één of meerdere actoren** of andere belanghebbenden.
meerdere actors of andere belanghebbenden.

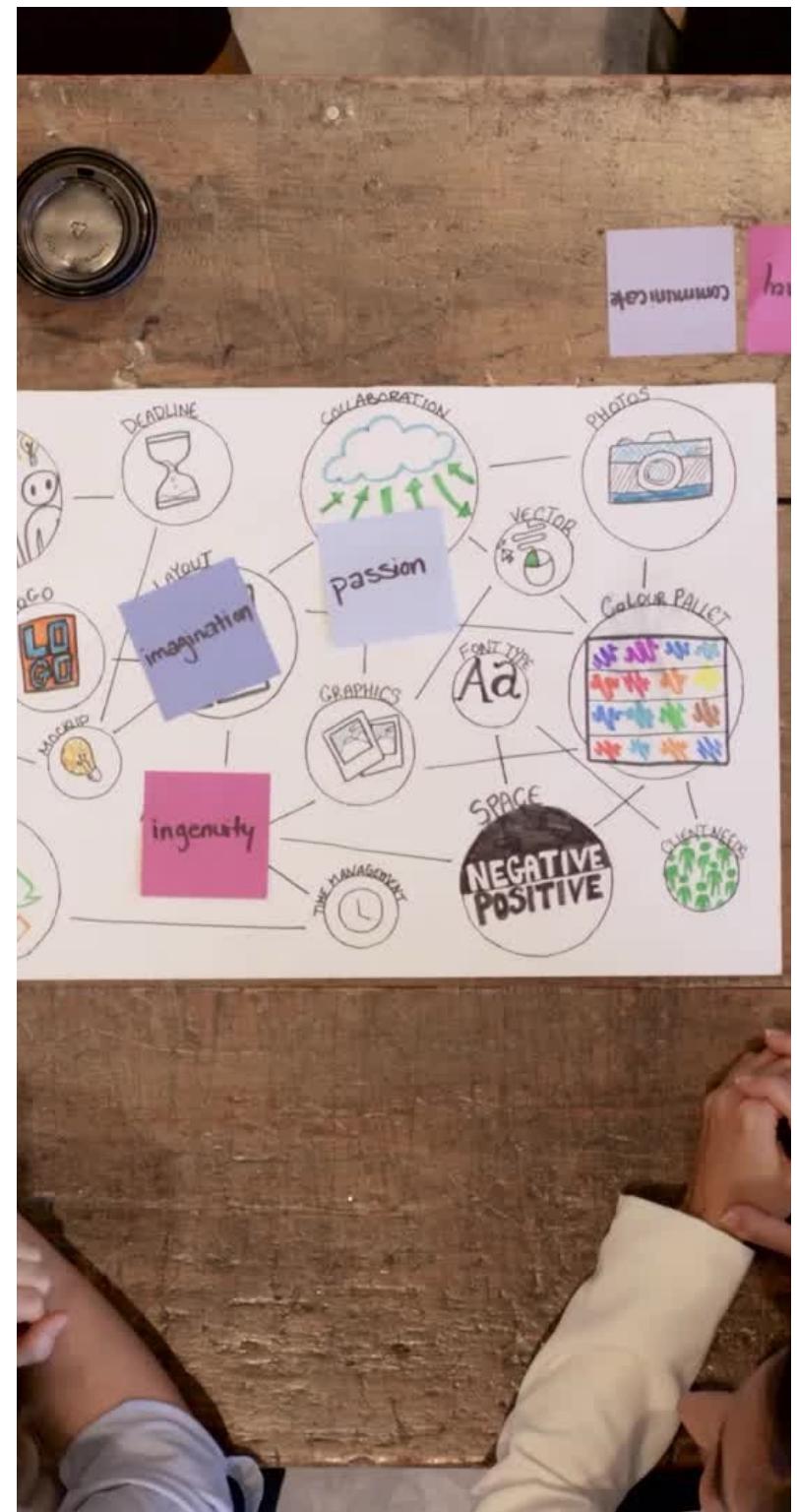
- **Belangrijkste kenmerken:**
 - Beschrijft **hoe** het systeem wordt gebruikt, niet **hoe** het intern werkt.
 - Richt zich op **het gedrag** van het systeem vanuit het perspectief van de gebruiker.
 - Geeft een **duidelijke, doelgerichte interactie** weer tussen actoren en het systeem.
- Met een **Use Case** wordt dus inzichtelijk gemaakt **wat** het systeem moet doen om aan de behoeften van gebruikers te voldoen, zonder in te gaan op technische details



Use Case

VOORDELEN VAN USE CASES

- Use Cases helpen bij het **structureren en verduidelijken** van systeemvereisten door:
 - **Het schetsen van de context** waarin het systeem wordt gebruikt.
 - **Het ordenen van systeemvereisten** in een logische en begrijpelijke volgorde.
 - **Het illustreren van de noodzaak** van het systeem door concrete scenario's.
 - **Gebruiksvriendelijke terminologie** die aansluit bij de taal van gebruikers en klanten.
 - **Duidelijke, realistische scenario's** die tonen hoe het systeem functioneert in de praktijk.
 - **Vergemakkelijking van communicatie** en afstemming met klanten en stakeholders.
 - **Ondersteuning bij test cases, documentatie en design**, doordat de vereisten helder zijn.
 - **Stimuleren van herbruikbaarheid** van vereisten in andere delen van het systeem of toekomstige projecten.



GOEDE KANDIDATEN VOOR USE CASES

Goede kandidaten

- Use Cases zijn geschikt voor systemen die **gedrag vertonen** dat kan worden vastgelegd als een reeks opeenvolgende acties.

Kenmerken:

- Systemen met duidelijk waarneembare **interacties en processen**.
- Systemen waarbij de gebruiker of een ander systeem **stappen doorloopt** om een doel te bereiken.

Voorbeelden:

- **Telefooncentrale** (*Telephone switch*) – gebruikers initiëren oproepen en schakelingen.
- **Cursusinschrijvingssysteem** – studenten en beheerders voeren acties uit om inschrijvingen te beheren.

SLECHTE KANDIDATEN VOOR USE CASES

Slechte kandidaten

- Sommige systemen lenen zich **niet goed** voor Use Cases, met name als ze **geen extern observeerbaar gedrag** vertonen in de vorm van gebruikersinteracties.

Kenmerken:

- Systemen die **automatisch processen uitvoeren** zonder directe gebruikerinteractie.
- Systemen waarbij er **geen duidelijke sequentie van handelingen** is.

Voorbeeld:

- **Compiler-software** – ontvangt code en genereert uitvoer, zonder een dynamische interactie tussen gebruiker en systeem.



USE CASES & VEREISTEN

- **FURPS Model (*Functionality, Usability, Reliability, Performance, Supportability*)**
 - **Functionality** → Use Cases beschrijven de functionele vereisten van het systeem.
 - **Usability** → Gebruiksvriendelijkheidseisen vallen buiten Use Cases, maar beïnvloeden de interacties.
 - **Reliability** → Betrouwbaarheidseisen (bv. foutafhandeling) worden soms in alternatieve scenario's opgenomen.
 - **Performance** → Systeemprestaties worden meestal niet beschreven in Use Cases, maar in aparte vereisten.
 - **Supportability** → Onderhoudbaarheid en uitbreidbaarheid zijn geen onderdeel van Use Cases, maar wel van het ontwerp.
- **Use Cases beschrijven functionele vereisten, maar niet alle systeemvereisten!**



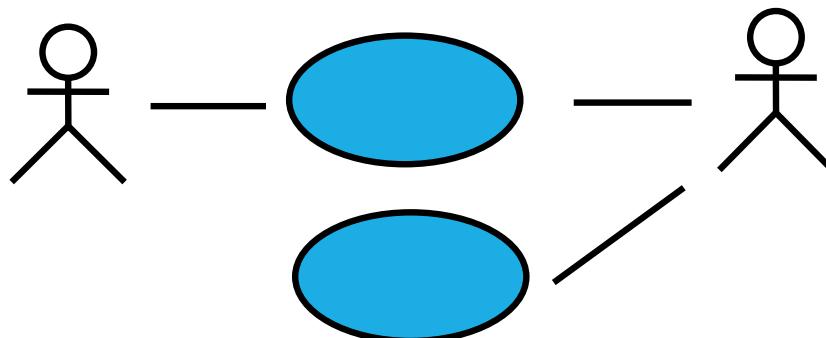
USE CASE MODEL

BELANGRIJKE COMPONENTEN VAN EEN USE CASE MODEL

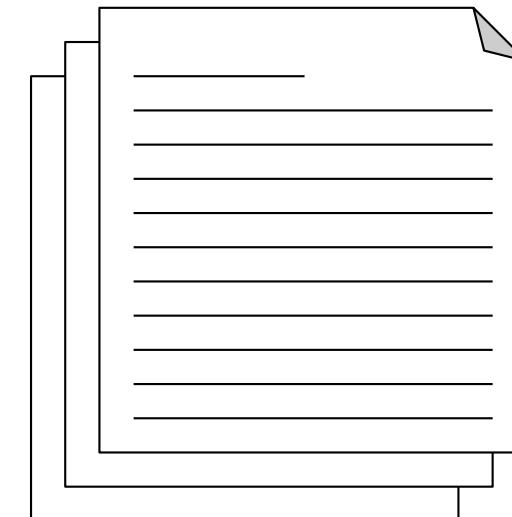
- **Use Case Diagram (Visueel overzicht)**
 - Een **UML-diagram** dat actoren en hun interacties met het systeem weergeeft.
 - Toont de **use cases** (**ovale vormen**) en de **actoren** (**poppetjes**) die ermee interageren.
- **Use Case Beschrijvingen (Gedetailleerde uitleg)**
 - Een **tekstuele beschrijving** van elke use case.
 - Bevat het **stappenplan, uitzonderingen en varianten** van de interactie.

USE CASE MODEL

Use-case diagrams
(visuele voorstelling)



Use-case specifications
(tekstuele voorstelling)

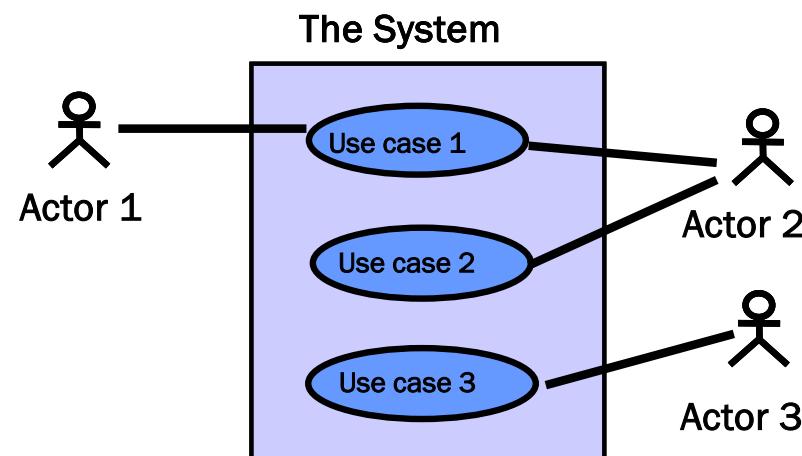


A close-up photograph of a person's hand interacting with a futuristic, transparent touch screen. The screen displays a 3D wireframe model of several rectangular blocks of varying heights, arranged in a descending staircase pattern from left to right. The background is a blurred blue, suggesting a high-tech environment.

USE CASE DIAGRAM

WAT IS EEN USE CASE DIAGRAM?

- Een Use Case Diagram biedt een visuele weergave van:
 - De verschillende **use cases** binnen een systeem en hun interacties.
 - De **actoren** (gebruikers, externe systemen) die met het systeem communiceren.
 - De **systeemgrenzen**, waardoor duidelijk wordt welke functionaliteiten binnen het systeem vallen.
 - Een samenvatting van het **systeemgedrag**, zonder technische details.



The background of the slide features a dense, abstract pattern of small, circular dots arranged in horizontal rows. The dots are primarily white, with some being a darker shade of gray. They are set against a dark, solid background. The pattern creates a sense of depth and texture, resembling a digital or data visualization.

USE CASE SPECIFICATIONS



USE CASE SPECIFICATIES

- Een Use Case Specificatie is een gedetailleerde tekstuele beschrijving van een **use case**, waarin de interactie tussen actoren en het systeem wordt vastgelegd. Dit document vormt een belangrijke referentie voor analisten, ontwikkelaars en testers.
- Wat bevat een Use Case Specificatie?
 - Algemene informatie
 - Flow van gebeurtenissen
 - Voorwaarden en vereisten

ALGEMENE INFORMATIE

- **Use Case Naam** – Unieke naam die de functionaliteit beschrijft.
- **Doel** – Wat wil de actor bereiken met deze use case?
- **Actoren** – Wie of wat interageert met het systeem?
- **Systeemgrenzen** – Duidelijke afbakening van de verantwoordelijkheid van het systeem.

FLOW VAN GEBEURTENISSEN

- **Hoofdflow (Happy Path)** – De standaardstappen die leiden tot een succesvol resultaat.
- **Alternatieve flows** – Afwijkende scenario's, zoals extra keuzes of alternatieve paden.
- **Uitzonderingen (Failure Scenarios)** – Wat gebeurt er bij fouten, zoals ongeldige invoer of systeemuitval?
- **Subflows** – Deelprocessen die binnen de use case kunnen worden uitgevoerd.

VOORWAARDEN EN VEREISTEN

- **Precondities** – Wat moet waar zijn voordat de use case start? (Bijv. de gebruiker is ingelogd).
- **Postcondities** – Wat moet waar zijn na succesvolle uitvoering? (Bijv. bestelling is bevestigd en opgeslagen).
- **Speciale vereisten** – Niet-functionele eisen, zoals beveiliging, prestaties of compliance-eisen.

VOORBEELD USE CASE SPECIFICATIE

"ONLINE
BESTELLING
PLAATSEN"

ALGEMENE INFORMATIE

- **Use Case Naam:** Online Bestelling Plaatsen
- **Doel:** Een klant bestelt producten in de webshop en voltooit de betaling.
- **Actoren:**
 - **Primaire Actor:** Klant
 - **Ondersteunende Actor:** Betaalsysteem
- **Systeemgrens:** De webshopsoftware beheert de bestelling, terwijl de betaling door een extern betaalsysteem wordt verwerkt.

FLOW VAN GEBEURTENISSEN

Hoofdflow (Happy Path)

1. Klant voegt producten toe aan het winkelmandje.
2. Klant bekijkt het winkelmandje en bevestigt de bestelling.
3. Klant voert verzendgegevens in.
4. Klant kiest een betaalmethode en wordt doorgestuurd naar het betaalsysteem.
5. Betaalsysteem verwerkt de betaling en stuurt een bevestiging naar de webshop.
6. Webshop registreert de bestelling en stuurt een bevestigingsmail naar de klant.
7. Voorraad wordt bijgewerkt en de bestelling wordt doorgestuurd naar het magazijn.

ALTERNATIEVE FLOW(S)

- **2a. Winkelmandje is leeg:**
 - Het systeem waarschuwt de klant en voorkomt het plaatsen van een lege bestelling.
- **5a. Betaling mislukt:**
 - De klant krijgt een foutmelding en wordt gevraagd een andere betaalmethode te kiezen of de bestelling later opnieuw te proberen.

UITZONDERINGEN (FAILURE SCENARIOS)

- **Betaalsysteem is niet beschikbaar** → Het systeem toont een melding en biedt de optie om de bestelling op een later moment af te ronden.
- **Verzendadres ontbreekt of ongeldig** → Het systeem vraagt de klant om een geldig adres in te voeren.

VOORWAARDEN EN VEREISTEN

■ Precondities

- Klant heeft een account of bestelt als gast.
- Er zijn producten beschikbaar in de webshop.

■ Postcondities

- De bestelling is geregistreerd en kan worden verwerkt.
- De klant ontvangt een bevestigingsmail.
- De voorraad is bijgewerkt.

■ Speciale Vereisten

- **Beveiliging:** Betaling moet via een versleutelde verbinding verlopen (SSL/TLS).
- **Prestaties:** De bestelling moet binnen 3 seconden worden verwerkt.
- **Compatibiliteit:** De webshop moet werken op zowel desktop als mobiel.

USE CASE SLICE

- Een Use Case Slice is een **klein, zelfstandig implementeerbaar deel** van een Use Case. Dit wordt vaak gebruikt in **Agile-ontwikkeling** om incrementeel functionaliteit op te leveren.
- Voorbeeld:
 - "Bestelling plaatsen" wordt opgesplitst in:
 - "Product toevoegen aan winkelmandje"
 - "Adresgegevens invoeren"
 - "Betaling uitvoeren"

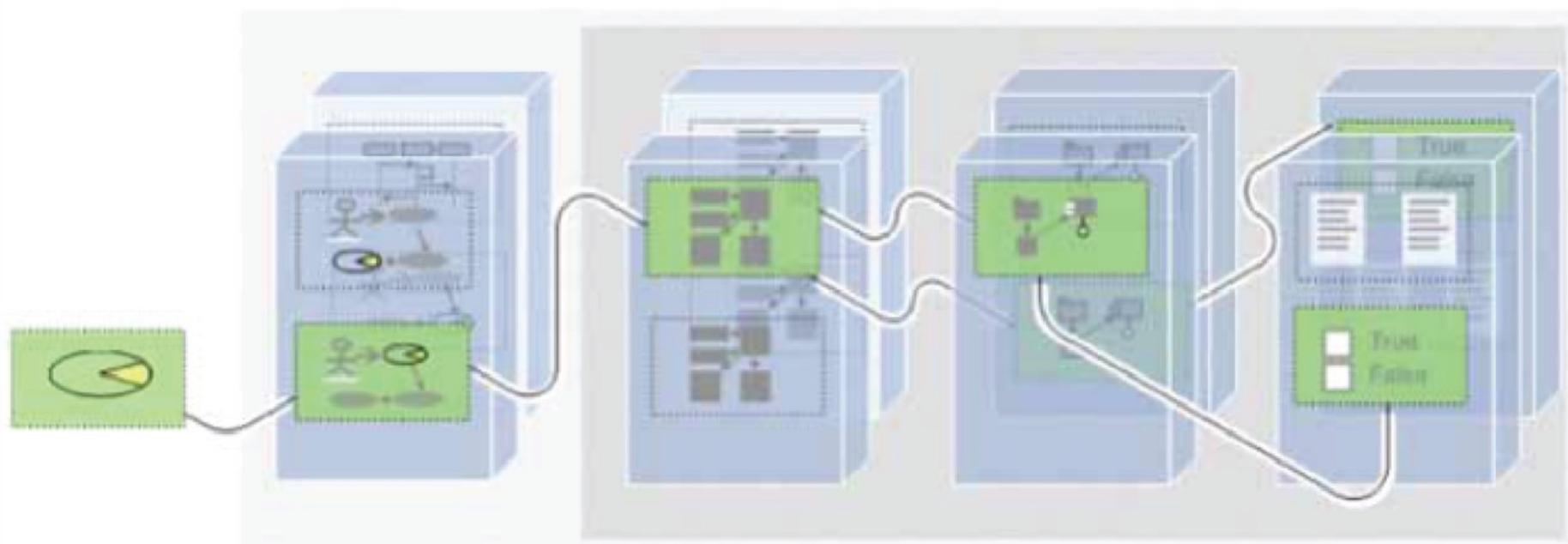
Use-Case Slice

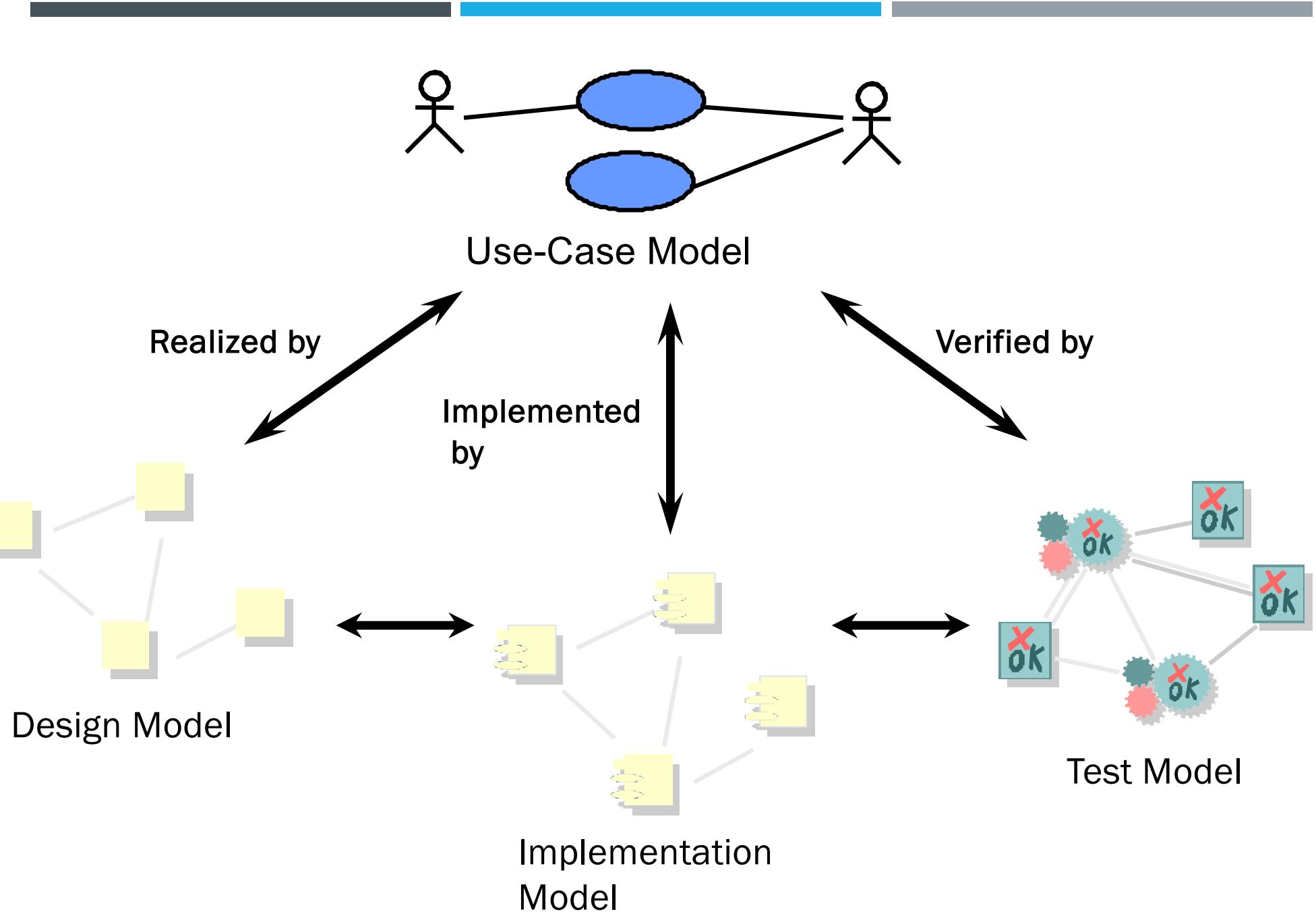
Use Case

Ontwerp

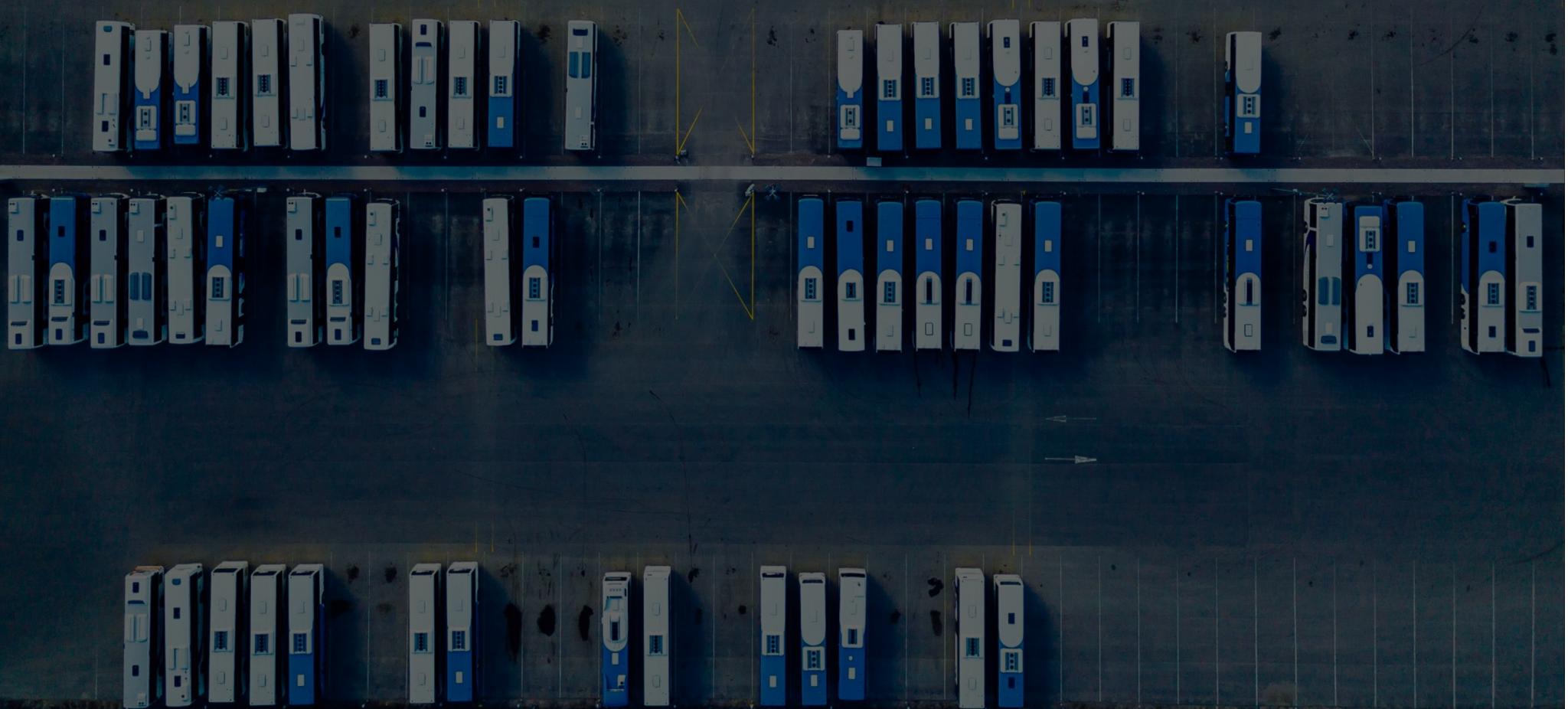
Code

Testgevallen / Scripts &
Testresultaten





VERSCHILLENDEN USE CASE FORMATS



BRIEF FORMAT (KORTE BESCHRIJVING)

- Bestaat uit één paragraaf die enkel het hoofdscenario (Happy Path) beschrijft.
- **Doel:** Snel overzicht van wat de Use Case inhoudt, zonder details.
- **Geschikt voor:**
 - Simpele processen met weinig varianten.
 - Vroege analysefase, wanneer nog niet alles vastligt.
- **Voorbeeld:**

"Een klant logt in op de webshop, voegt producten toe aan het winkelmandje en rekent af met een betaalmethode."

CASUAL FORMAT (INFORMELE, BEKNOpte BESCHRIJVING)

- Beschrijft de Use Case in korte paragraafstijl, inclusief enkele alternatieve scenario's.
 - Bevat meer details dan de Brief Format, maar is niet formeel gestructureerd.
- **Geschikt voor:**
- Middelgrote processen waar enkele uitzonderingen nodig zijn.
 - Documentatie waarbij **geen strikte structuur vereist is**.
- **Voorbeeld:**
- "Een klant bezoekt de webshop en voegt een product toe aan het winkelmandje. Als de klant nog geen account heeft, kan hij inloggen of zich registreren. De klant kiest een betaalmethode en voltooit de bestelling. Als de betaling succesvol is, ontvangt de klant een bevestigingsmail."*

FULLY DRESSED FORMAT ("VOLLEDIG UITGEWERKTE BESCHRIJVING")

- Gedetailleerde, formele documentatie van de Use Case.
- Beschrijft **alle mogelijke scenario's** (happy path, alternatieve paden, fouten).
- Bevat:
 - **Precondities:** Voorwaarden die moeten gelden voordat de Use Case start.
 - **Hoofdscenario:** De standaardstappen die worden doorlopen.
 - **Alternatieve scenario's:** Wat gebeurt er als de normale flow wordt onderbroken?
 - **Uitzonderingen:** Wat als er fouten optreden (bv. betaling mislukt)?
 - **Postcondities:** Wat is de verwachte uitkomst na uitvoering?

FULLY DRESSED FORMAT ("VOLLEDIG UITGEWERKTE BESCHRIJVING")

- Voorbeeld (kort fragment van een volledig uitgewerkte Use Case):
 - Naam: Online bestelling plaatsen
 - Actoren: Klant, Betaalsysteem
 - Precondities: Klant is ingelogd, er zijn producten in de winkelmand
 - Hoofdscenario:
 1. Klant bekijkt het winkelmandje en klikt op "Afrekenen".
 2. Klant vult verzendgegevens in en kiest een betaalmethode.
 3. Systeem verwerkt de betaling via het betaalsysteem.
 4. Bij succesvolle betaling wordt de bestelling geregistreerd en krijgt de klant een bevestigingsmail.
 - Alternatieve scenario's:
 - 3a. Betaling mislukt → Klant krijgt een foutmelding en kan opnieuw proberen.
 - 2a. Adresgegevens onvolledig → Systeem vraagt om correctie.
 - Postconditie: De bestelling is correct verwerkt en kan worden verzonden.

TIPS VOOR HET SCHRIJVEN VAN DUIDELIJKE EN EFFECTIEVE USE CASES

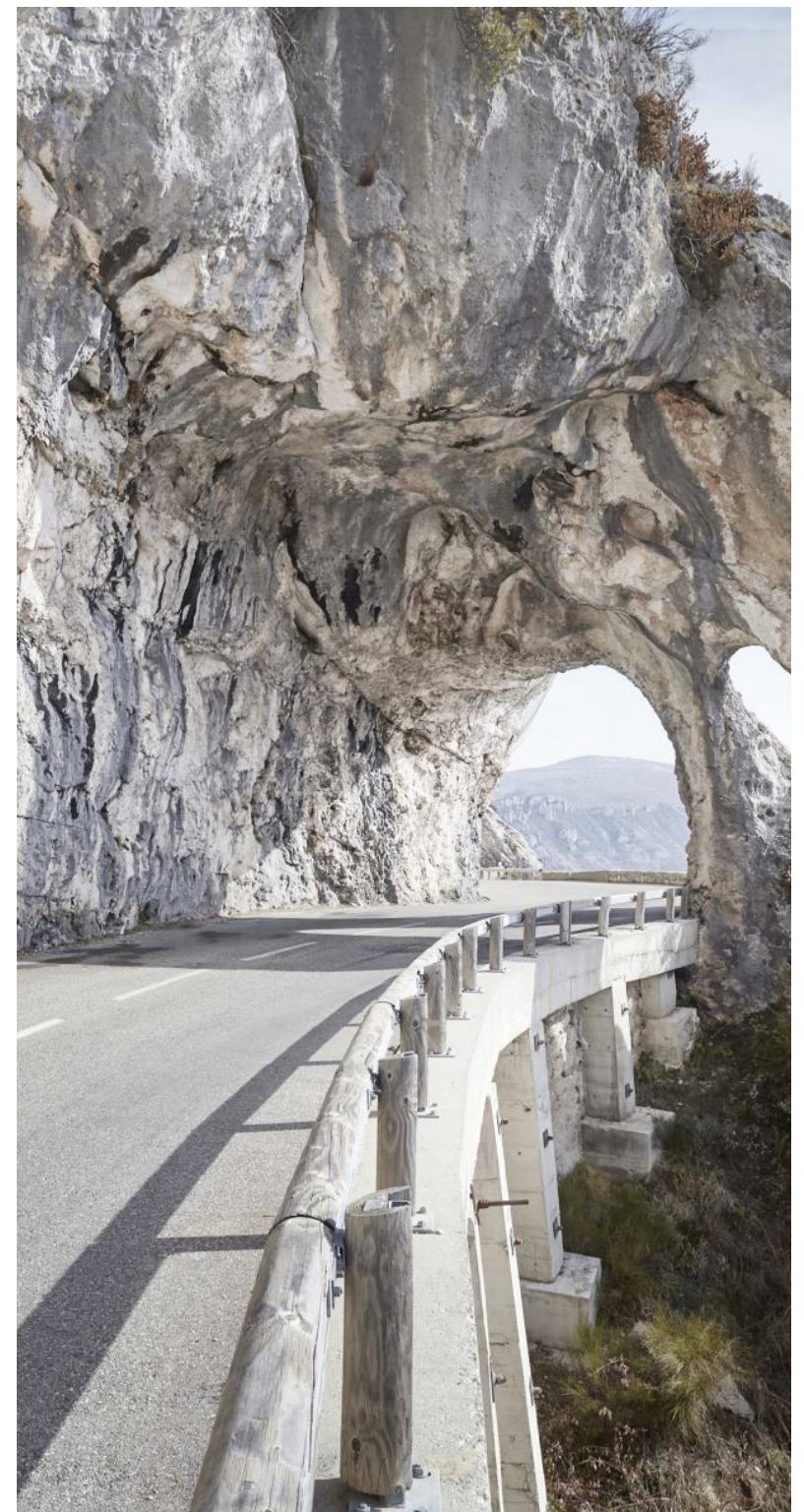


TIP 1: GEBRUIK EEN ESSENTIËLE STIJL

- Schrijf Use Cases **doelgericht**, zonder onnodige details over UI of implementatie.
- **Goed voorbeeld (doelgericht, abstract):**
"De klant plaatst een bestelling."
- **Slecht voorbeeld (UI-afhankelijk):**
"De klant klikt op de knop 'Toevoegen aan winkelmandje', gaat naar de afrekenpagina."
- **Samenvatting:** Schrijf Use Cases die **beschrijven wat de gebruiker wil bereiken**, niet hoe dit technisch of visueel wordt uitgevoerd!

TIP 2: GEBRUIK KORTE EN HELDERE BESCHRIJVINGEN

- Schrijf bondige en duidelijke Use Cases door overbodige woorden te vermijden.
- Goed voorbeeld (kort & krachtig):
 - "Het systeem authenticeert de gebruiker."
- Slecht voorbeeld (onnodige woorden):
 - "Het systeem voert een authenticatieproces uit om de gebruiker te verifiëren."
- Samenvatting:
 - Elimineer onnodige woorden en houd de beschrijvingen eenvoudig en to-the-point. Dit maakt Use Cases duidelijker, leesbaarder en efficiënter!



TIP 3: SCHRIJF BLACK-BOX USE CASES

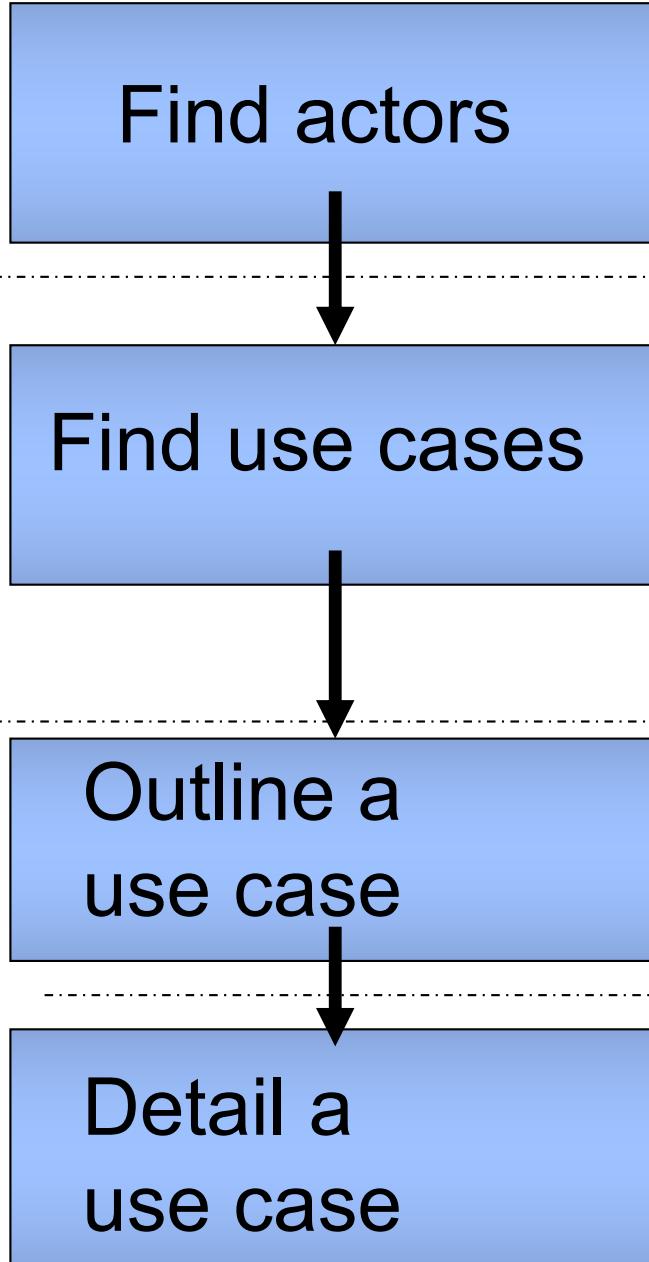
- Focus op **wat het systeem doet**, niet op **hoe** het dat intern uitvoert.
- **Goed voorbeeld (black-box, abstract):**
"Het systeem bewaart de verkoopsgegevens."
- **Slecht voorbeeld (te gedetailleerd, interne werking):**
"Het systeem bewaart de verkoopsgegevens in de database."
"Het systeem genereert een SQL INSERT-statement voor de database."
- **Samenvatting:**
 - Beschrijf **de verantwoordelijkheid van het systeem**, zonder technische details over **opslag, verwerking of implementatie**. Dit houdt Use Cases **generiek, toekomstbestendig en begrijpelijk** voor alle stakeholders!

TIP 4: GEBRUIK HET ACTOR-DOEL PERSPECTIEF

- Schrijf Use Cases vanuit het perspectief van **de actor en zijn doel**, niet vanuit het systeem.
- **Een Use Case is:**
 - "Een reeks interacties waarin het systeem acties uitvoert die leiden tot een waarneembaar en waardevol resultaat voor een specifieke actor."
- **Waarom dit belangrijk is?**
 - Houdt de focus op de gebruiker en wat hij wil bereiken.
 - Vermijdt systeemgerichte beschrijvingen die te technisch of intern gefocust zijn.
 - Maakt Use Cases begrijpelijker voor zowel business als ontwikkelaars.
- **Samenvatting:**
 - Schrijf Use Cases die **duidelijk maken hoe het systeem waarde levert aan de actor**, niet enkel wat het systeem intern doet!

**EN NU STAP
VOOR STAP**





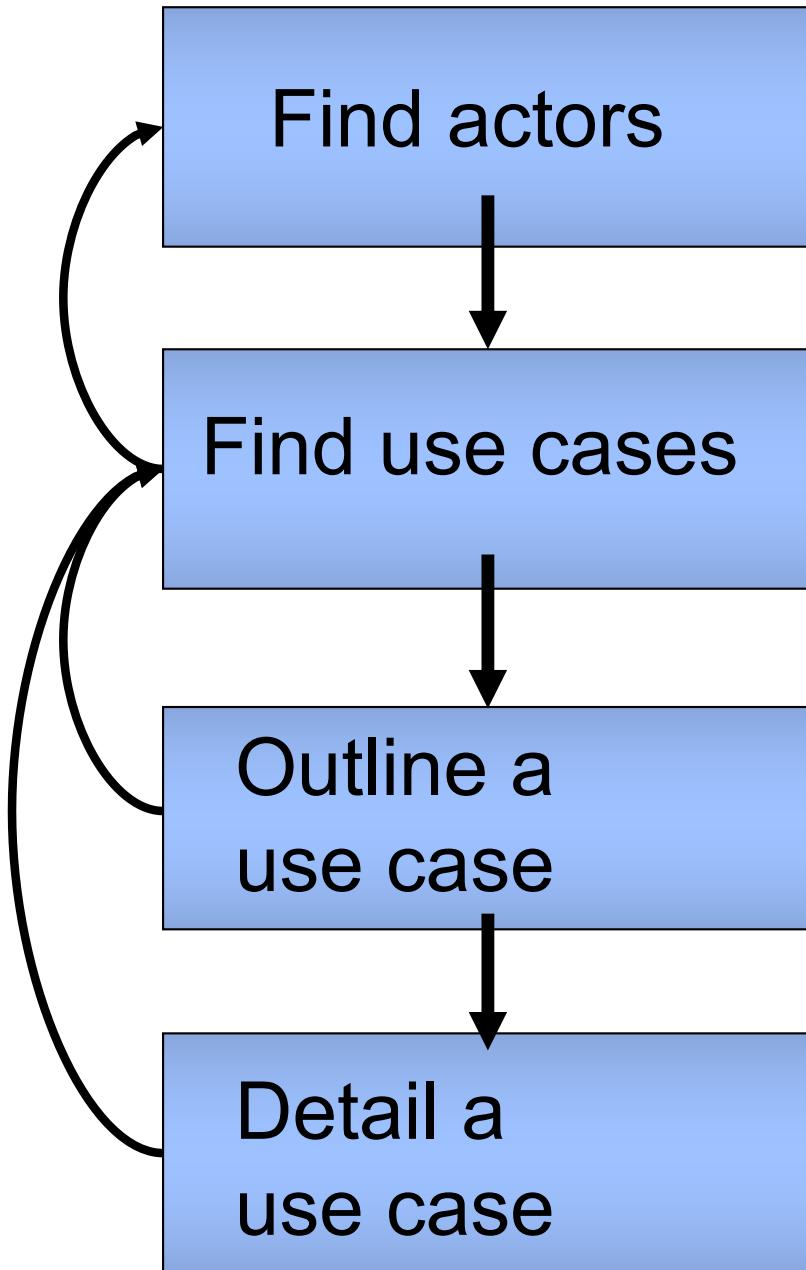
Brief description: This use case allows a Registrar to close the registration process. Courses that do not have enough students are cancelled. The Billing System is notified for each student in each course that is not cancelled, so the student can be billed.

Close Registration Outline

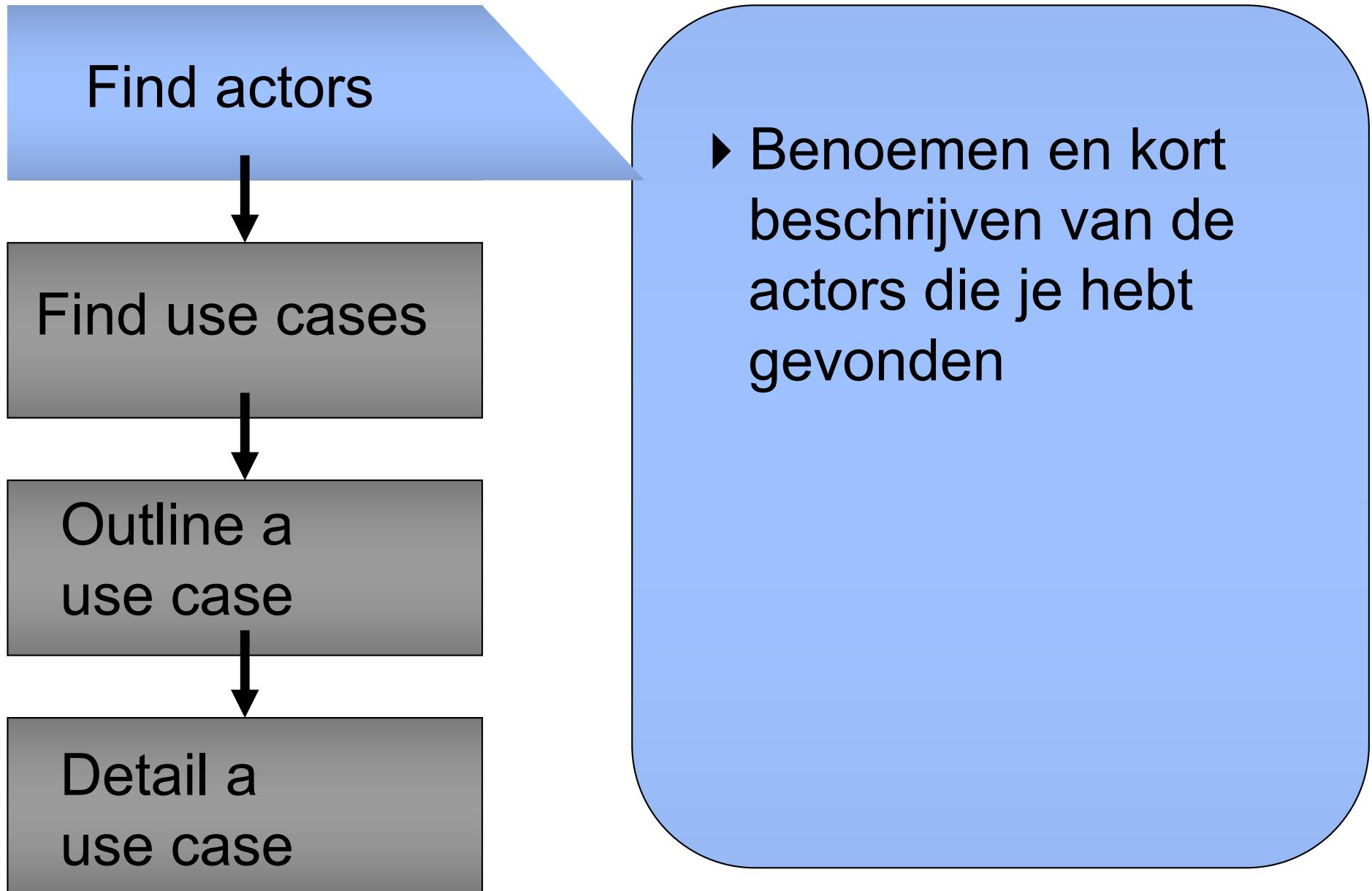
- Flow of events
- Step by step

Close Registration Use-Case Specification

- Detailed Flow of Events
- Special Requirements
- Pre/Postconditions

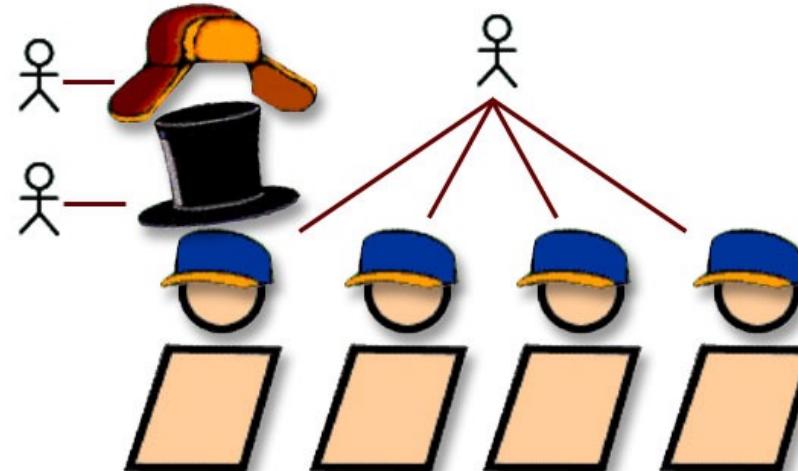


Belangrijk:
Use case uitschrijven
is een *iteratief* proces



VINDEN VAN ACTOREN

- Het identificeren van actoren begint met het bepalen van de systeemgrens:
 - Alles buiten de systeemgrens dat interacteert met het systeem, wordt beschouwd als een actor.
 - Een actor vertegenwoordigt een rol die kan worden vervuld door een persoon, hardwareapparaat of een ander systeem in relatie tot het systeem.



VRAGEN OM ACTOREN TE IDENTIFICEREN

- Wie gebruikt het systeem?
- Wie ontvangt informatie uit het systeem?
- Wie levert informatie aan het systeem?
- Waar in het bedrijf wordt het systeem gebruikt?
- Wie onderhoudt of ondersteunt het systeem?
- Welke andere systemen interageren met dit systeem?
- Wie start of sluit het systeem af?
- Wie beheert gebruikers en beveiliging?
- Wie bewaakt systeemactiviteit of prestaties?
- Wie controleert de logs?
- Wie ontvangt meldingen bij systeemfouten?

TIPS OM ACTOREN TE IDENTIFICEREN

- Actor-Goal Lijsten (Actor-Doele Methode)
 - Stel een lijst op van mogelijke gebruikers of systemen die interactie hebben met het systeem.
 - Voor elke actor, bepaal welk doel hij wil bereiken met het systeem.
- Waarom nuttig?
 - Houdt de focus op gebruikersperspectief.
 - Voorkomt dat je belangrijke actoren over het hoofd ziet.

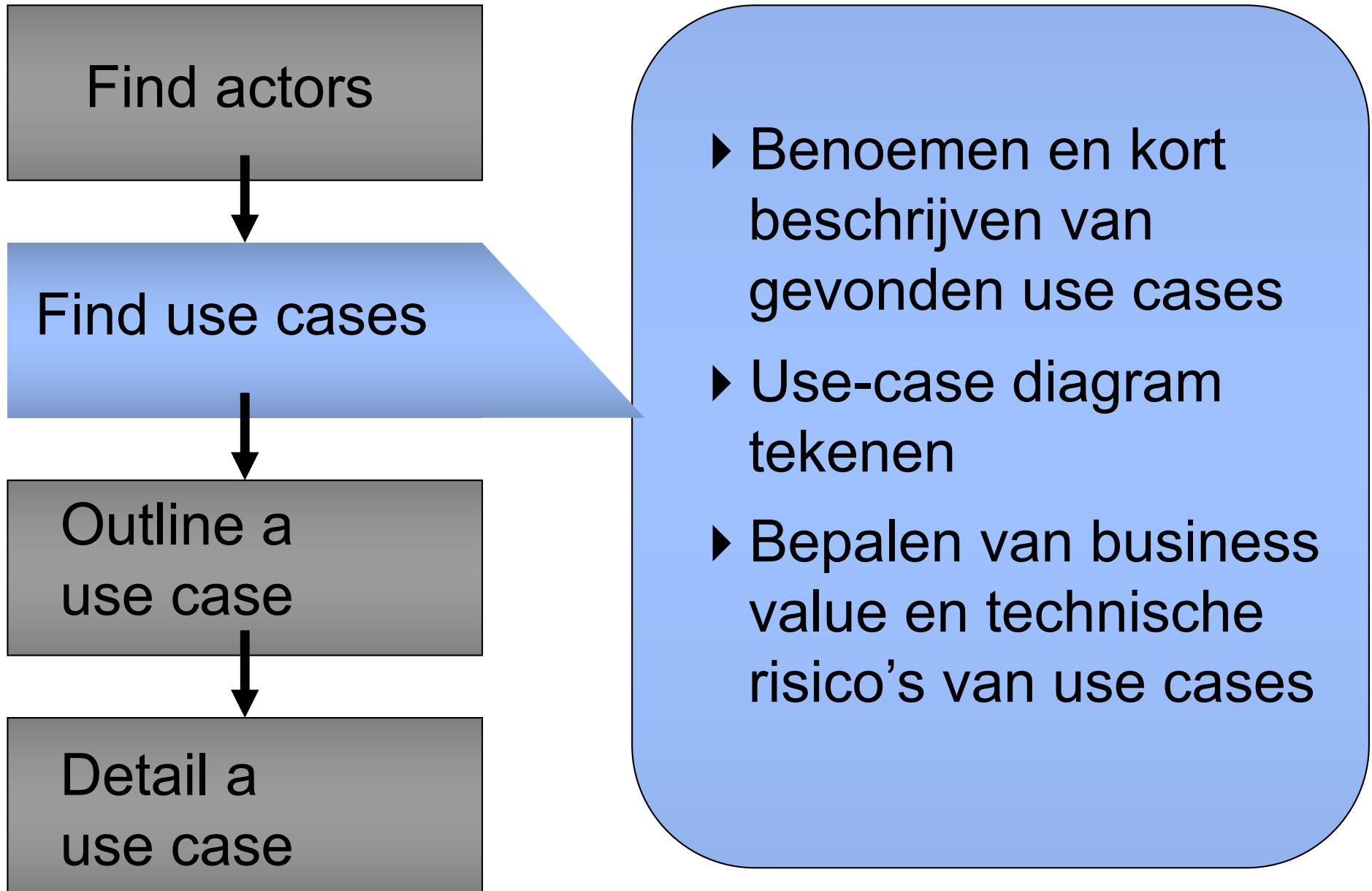
Actor	Doel
Klant	Plaatsen van een bestelling
Magazijnmedewerker	Opvragen van de voorraad
Betaalsysteem	Verwerken van betalingen

VINDEN VAN ACTOREN

Actor	Goal		Actor	Goal
Cashier	process sales process rentals handle returns cash in cash out ...		System Administrator	add users modify users delete users manage security manage system tables ...
Manager	start up shut down ...		Sales Activity System	analyze sales and performance data
...

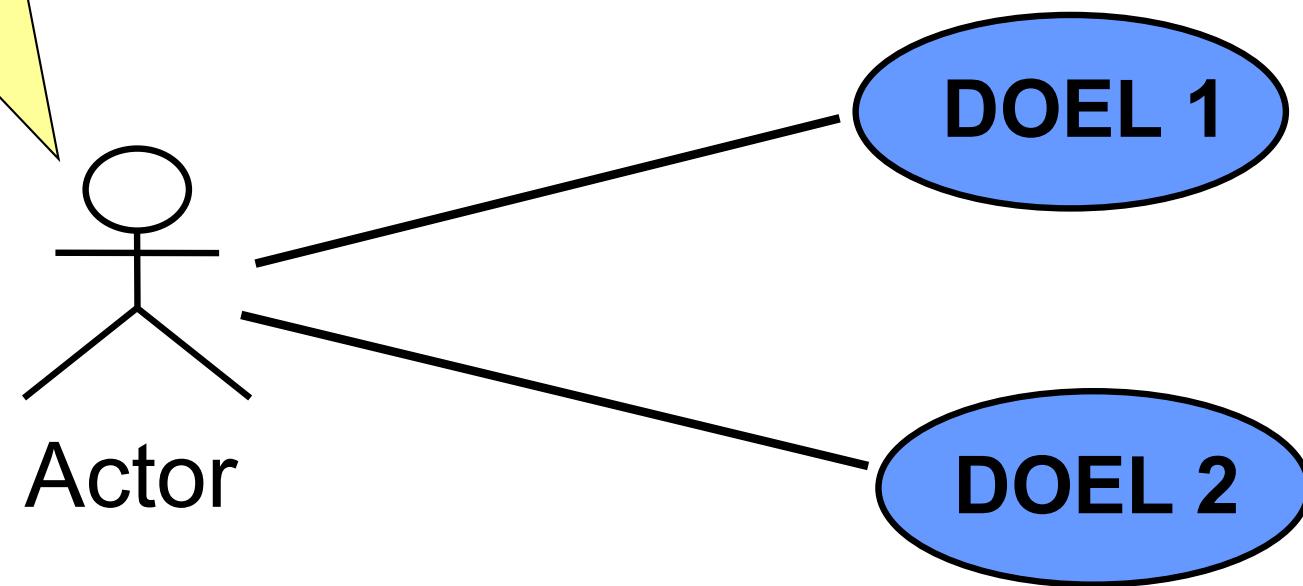
GEEF ELKE ACTOR EEN DUIDELIJKE EN BETEKENISVOLLE NAAM

- Kies een naam die de rol en verantwoordelijkheid van de actor weerspiegelt.
- De naam moet direct duidelijk maken welke interactie de actor met het systeem heeft.
- Vermijd vage of algemene namen zoals "Gebruiker" als er meerdere soorten gebruikers zijn.
- Goed voorbeeld:
 - "Klant" (voor een gebruiker die een bestelling plaatst)
 - "Magazijnmedewerker" (voor iemand die voorraad beheert)
 - "Betaalsysteem" (voor een externe service die transacties verwerkt)
- Slecht voorbeeld:
 - "Persoon" (te algemeen, zegt niets over de rol)
 - "Gebruiker" (onduidelijk bij meerdere soorten gebruikers)
 - "Systeem" (te breed, specificeer welk systeem het is)
- **Samenvatting:** Gebruik specifieke en herkenbare namen die direct aangeven **wat de actor doet** in relatie tot het systeem!



Welk doel probeer ik te bereiken adhv het systeem?

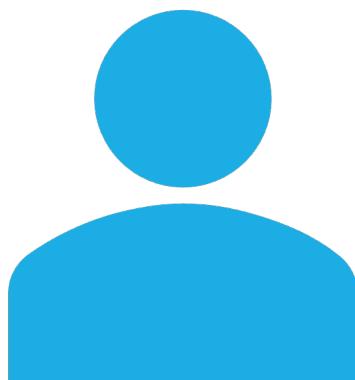
- Find use cases



HULPVragen BIJ HET IDENTIFICEREN VAN USE CASES

- Gebruik deze vragen om **Use Cases te vinden en te verfijnen:**
 - Wat zijn de doelen van elke actor?
 - Waarom wil de actor het systeem gebruiken?
 - Welke acties voert de actor uit?
 - Maakt de actor iets aan, slaat hij iets op, wijzigt hij gegevens, verwijdert hij iets of leest hij informatie uit?
 - Indien ja, waarom? Wat is het doel van deze actie?
 - Moet de actor het systeem informeren over externe gebeurtenissen of veranderingen?

VOORBEELD: "LOGIN" IS GEEN ECHTE USE CASE



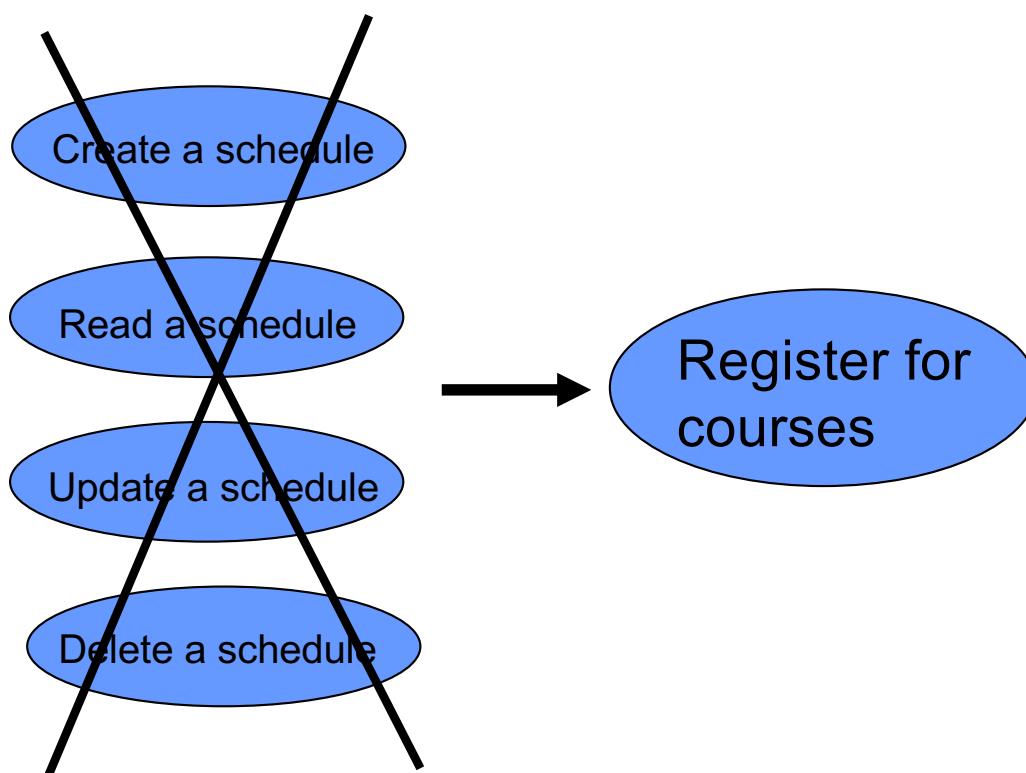
- Volgens de UML-definitie is "Login" geen op zichzelf staande Use Case omdat het geen waardevol resultaat voor de actor oplevert.
- "Login" is vaak een subprocess binnen andere Use Cases, zoals "*Gebruiker beheert profiel*" of "*Toegang verkrijgen tot dashboard*".
- "Login" is geen einddoel → Niemand logt in om alleen in te loggen; de gebruiker wil iets anders bereiken (bijvoorbeeld toegang krijgen tot zijn profiel, een bestelling plaatsen, of gegevens beheren).
- "Login" is meestal een precondition → In plaats van een aparte Use Case is het beter om te vermelden:
 - "*De gebruiker moet ingelogd zijn om deze actie uit te voeren.*"

- Als authenticatie een kernfunctionaliteit is → Bijvoorbeeld in een identity management systeem of beveiligde applicatie waar gebruikersbeheer centraal staat.
- Als het loginproces complex is → Bijvoorbeeld bij tweefactorauthenticatie (2FA), single sign-on (SSO), of biometrische verificatie.

WANNEER IS "LOGIN" WÉL EEN APARTE USE CASE?

VINDEN VAN USE CASES

- Een CRUD use case is een Create, Read, Update, of Delete use case
- Verwijder CRUD use cases wanneer ze data-management inhouden die geen resultaat opleveren aan een actor
- CRUD use cases zijn meestal mixup van ≠ actors



Niet verwarring met functies!
Focus op resultaat, waarde voor actor

BIJKOMENDE HULPMIDDELEN OM USE CASES TE VALIDEREN

■ The Boss Test

- Stel je voor dat je baas vraagt: "*Wat heb je de hele dag gedaan?*"
- Een Use Case moet een **betekenisvol antwoord** kunnen vormen.
- **Slecht voorbeeld:** "*Ik heb de hele dag ingelogd.*" (geen op zichzelf staande Use Case)
- **Goed voorbeeld:** "*Ik heb klantgegevens bijgewerkt en facturen verwerkt.*" (wel een volwaardige Use Case)

■ The Size Test

- Een **volledig uitgewerkte (fully dressed) Use Case** kan 3 tot 10 pagina's beslaan.
- Als een actie **te klein** is (bijna geen stappen bevat), is het meestal **een aparte Use Case**, maar een **onderdeel van een andere Use Case**.
- **Aanbeveling:**
 - Gebruik deze tests om **te bepalen of een actie een echte Use Case is of slechts een subactie binnen een grotere functionaliteit!**

NAAMGEVING USE CASES

- Uniek, intuïtief en zelfverklarend → De naam moet direct duidelijk maken wat de Use Case doet.
- Beschrijft het observeerbare resultaat → Geeft aan wat de actor bereikt, zonder ambiguïteit.
- Geformuleerd vanuit de actor → De naam weerspiegelt de actie die de actor uitvoert.
- Start met een werkwoord → Gebruik eenvoudige werkwoord-zelfstandig naamwoord combinaties.

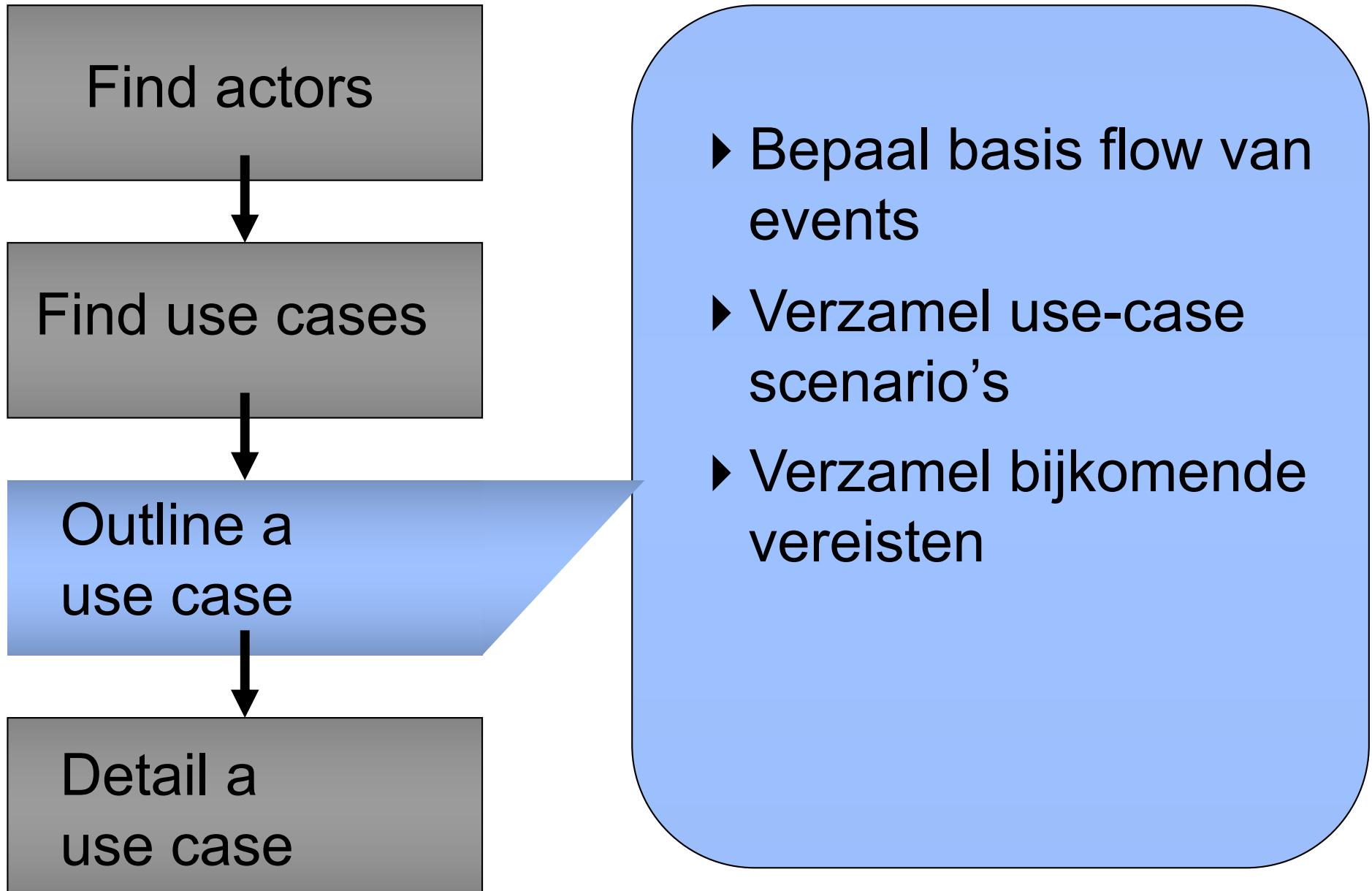
Register for
courses

Select a
course to teach

NAAMGEVING USE CASES

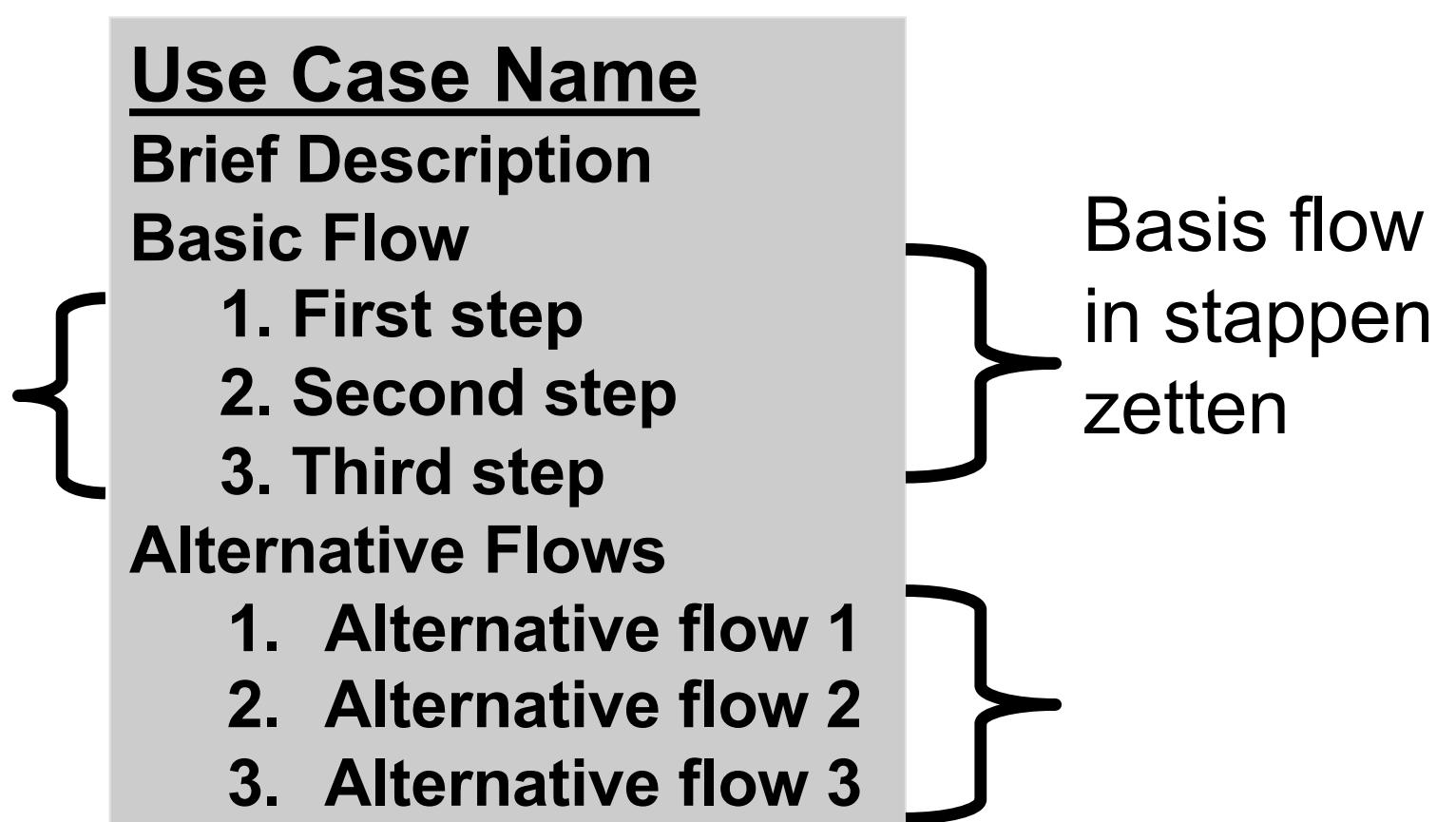
- Goede voorbeelden:
 - ✓ "Student schrijft zich in voor een vak" (duidelijk, actorgericht)
 - ✓ "Klant plaatst een bestelling" (observeerbaar resultaat)
 - ✓ "Beheerder voegt gebruiker toe" (specifiek en actiegericht)

- Slechte voorbeelden:
 - ✗ "Inschrijving" (te vaag, geen actie)
 - ✗ "Bestelling" (geen werkwoord, niet actorgericht)
 - ✗ "Gebruikersbeheer" (onduidelijk welk resultaat bereikt wordt)



EEN OUTLINE BESCHRIJFT DE USE CASE STAPPEN IN ZEER KORTE ZINNEN IN SEQUENTIËLE VOLGORDE

Geef elke
stap een
nummer en
naam



OUTLINE A USE CASE

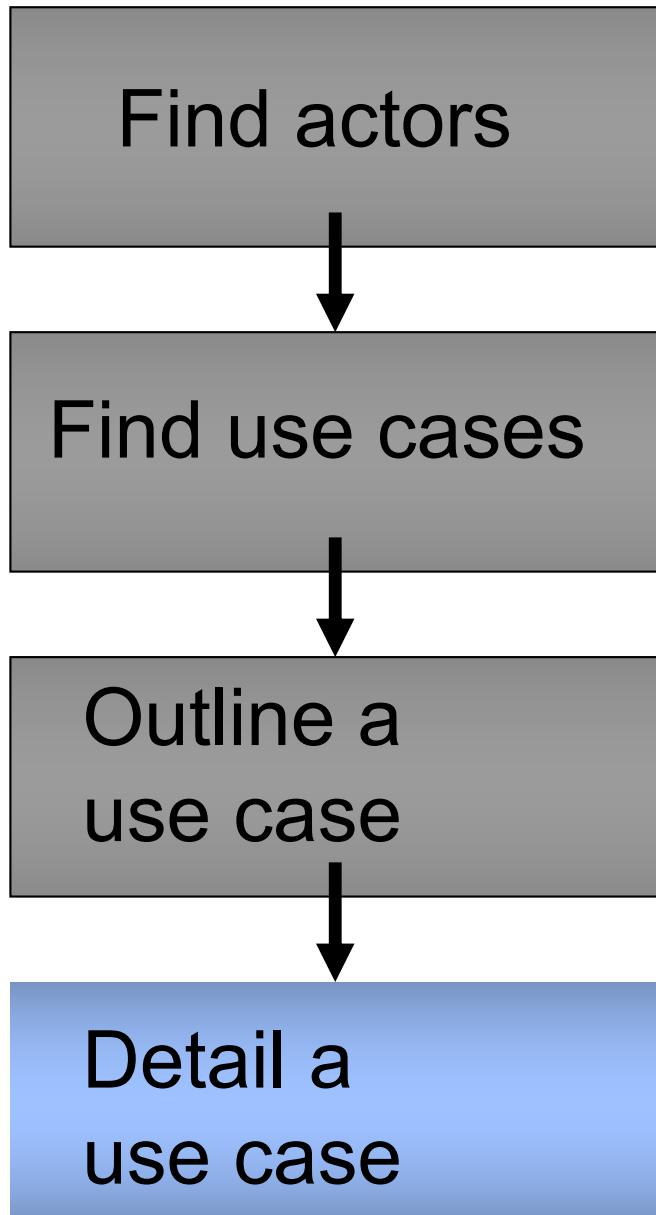
■ Basic Flow: Student Creates Schedule

Student is ingelogd (preconditie).

1. Student kiest de optie "Maak nieuw rooster".
2. Student bekijkt de lijst met beschikbare vakken.
3. Student selecteert vakken en voegt ze toe aan zijn rooster.
4. Student controleert het rooster en dient het in.
5. Systeem bevestigt de inschrijving en toont het definitieve rooster.

■ Alternative Flows

- **A1. Unidentified Student:** De student is niet ingelogd → Het systeem vraagt om in te loggen.
- **A2. Quit:** De student sluit de applicatie → Niet-opgeslagen wijzigingen gaan verloren.
- **A3. Cannot Enroll:** Student voldoet niet aan de inschrijvingsvoorwaarden → Het systeem toont een foutmelding.
- **A4. Course Catalog System Unavailable:** Studenten kunnen geen vakinformatie opvragen → Kunnen we inschrijving alsnog toestaan?
- **A5. Course Registration Closed:** Roosterwijzigingen zijn niet meer mogelijk → Het systeem toont een melding.



- ▶ Gedetailleerd uitschrijven van event flow
- ▶ Structureren van flow van events
- ▶ Specifiëren van bijkomende use case eigenschappen

USE CASES

- <Use-Case Name>
1. Brief Description
 2. Basic Flow of Events
 3. Alternative Flows
 4. Subflows
 5. Key Scenario
 6. Preconditions
 7. Postconditions
 8. Extension Points
 9. Special Requirements
 10. Additional Information

Detail
toevoegen

Structureer flow
in stappen

Geef elke stap een
nummer

Beschrijf de
stappen

Use Case: Inschrijven voor Cursussen

1. Basic Flow

1.1. Systeemtoegang

- De student opent het **Cursusregistratiesysteem** en kiest "*Inschrijven voor cursussen*".
- Het systeem controleert of de student bevoegd is om zich in te schrijven.

1.2 Nieuw rooster aanmaken

- Het systeem toont beschikbare functies.
- De student kiest "*Nieuw rooster aanmaken*".

1.3 Cursusinformatie ophalen

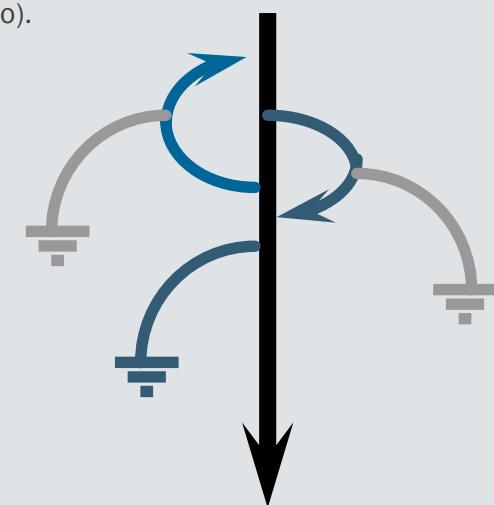
- Het systeem haalt een lijst op van beschikbare cursussen uit het **Cursuscatalogussysteem**.
- De student kan filteren op **departement, professor of onderwerp**.

1.4 Cursussen selecteren

- De student kiest vier **hoofdvakken** en twee **alternatieve vakken** uit de lijst.

STRUCTUUR VAN EEN USE CASE FLOW

- **Basic Flow (Hoofdscenario)**
 - Het "Happy Day" scenario → De standaard, succesvolle uitvoering van de Use Case van begin tot eind.
- **Alternatieve Flows**
 - **Reguliere varianten** → Normale afwijkingen binnen het proces (bv. een andere betaalmethode).
 - **Bijzondere gevallen** → Minder vaak voorkomende maar verwachte situaties (bv. kortingscode gebruiken bij een bestelling).
 - **Exceptionele (error) flows** → Situaties waarin iets fout gaat (bv. betaling mislukt door onvoldoende saldo).



BASISSTROOM / ALTERNATIEVE STROMEN

BASISSTROOM

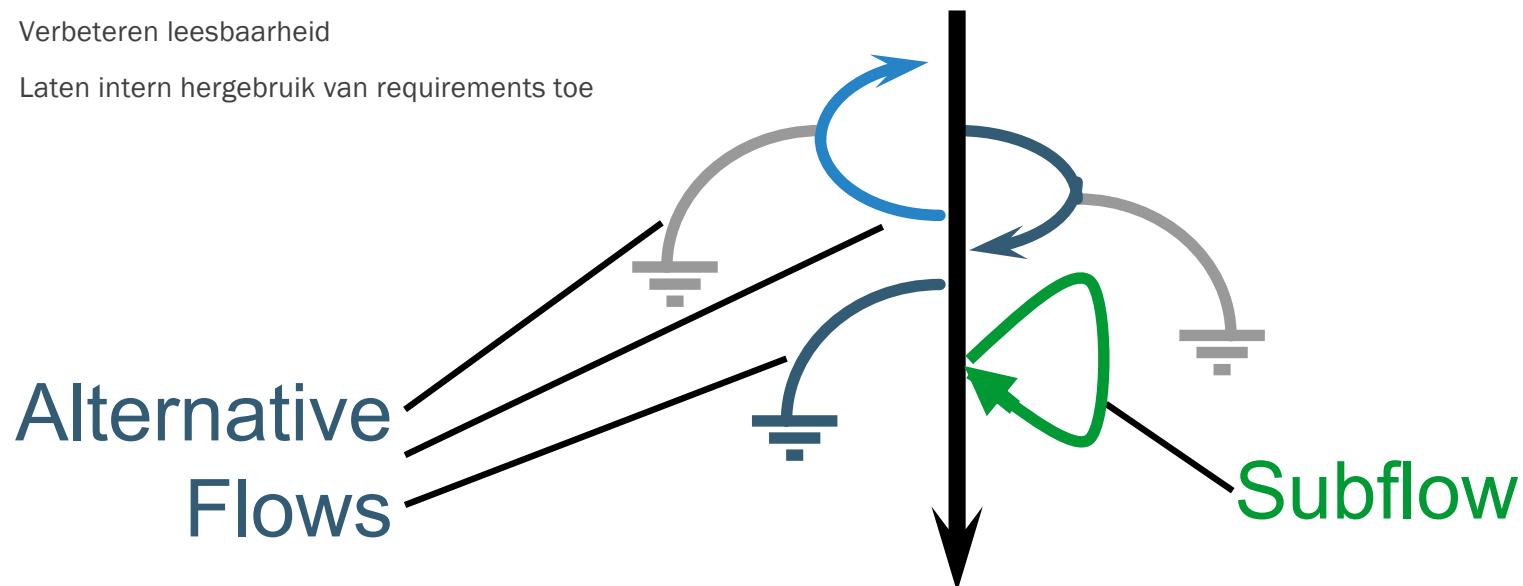
1. Kaart Invoeren
2. Kaart Valideren
3. Selecteer Geld Opnemen
4. Selecteer Rekening
5. Bevestig Saldo
Voldoende
6. Kaart Teruggegeven
7. Geld Overhandigen

ALTERNATIEVE STROMEN

- A1 Kaart Ongeldig
- A2 Niet-Standaard Bedrag
- A3 Ontvangstbewijs nodig
- A4 Onvoldoende geld in machine
- A5 Onvoldoende saldo op rekening
- A6 Leidt tot rood staan
- A7 Vastzittende kaart
- A8 Geld niet uitgenomen etc.

USE CASES - SUBFLOWS

- Indien een flow te lang wordt, breek bepaalde secties op in subflows
- Voordelen
 - Verbeteren leesbaarheid
 - Laten intern hergebruik van requirements toe



USE CASES

- Voorbeeld

1. Use Case Name: Register for Courses

1.1 Brief Description

...

2. Flow of Events

2.1 Basic Flow

1. Log On

...

2. Select 'Create a Schedule'

...

3. Obtain Course Information

Perform subflow S1: Obtain Course Information

4. Select Courses

...

5. Submit Schedule

...

6. Accept Completed Schedule

...

2.2 Subflows

2.2.1 S1: Obtain Course Information

The student requests a list of course offerings. The student can search the list by department, professor or topic to obtain desired course information.

The system retrieves a list of available course offerings from the Course Catalog System and displays the list to the student.

KENMERKEN VAN PRECONDITIONS

- Bepalen de **uitgangssituatie** waarin het systeem zich moet bevinden.
- **Zijn optioneel**, maar nuttig om fouten en onduidelijkheden te voorkomen.
- Worden niet binnen de Use Case zelf afgehandeld, maar als gegeven aangenomen.

VOORBEELD: "INSCHRIJVEN VOOR CURSUSSEN" (REGISTER FOR COURSES)

- **Precondities:**
 - De lijst met beschikbare vakken moet bestaan en toegankelijk zijn in het systeem.
 - De student moet ingelogd zijn in het registratiesysteem.
- **Waarom belangrijk?**
 - Zonder deze precondities kan de Use Case niet correct worden uitgevoerd.
 - De Use Case hoeft zich niet bezig te houden met het **genereren van de vakkenlijst** of het **inlogproces**.
 - Duidelijkheid voor ontwikkelaars en testers over **wat er al moet bestaan voordat de Use Case start**.
- **Samenvatting:**
 - Precondities **definiëren de startvoorwaarden**, zodat een Use Case correct kan verlopen!

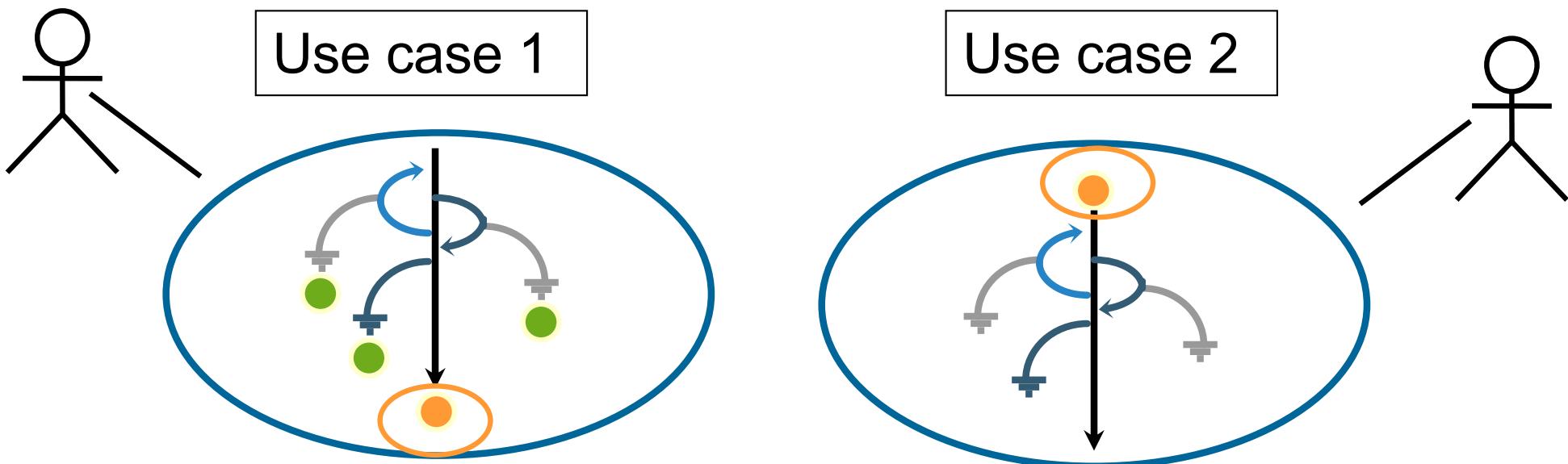
KENMERKEN VAN POSTCONDITIES

- Bepalen de eindsituatie van het systeem.
- Mogen nooit verwijzen naar andere Use Cases → ze moeten op zichzelf staan.
- Moeten eenvoudig en verifieerbaar zijn.

VOORBEELD: "INSCHRIJVEN VOOR CURSUSSEN" (REGISTER FOR COURSES)

- Postcondities:
 - De student is succesvol ingeschreven voor de gekozen cursussen.
 - Als de registratie is mislukt, blijven het rooster en de inschrijvingsstatus ongewijzigd.
- Waarom belangrijk?
 - Geeft duidelijk aan wat het systeem garandeert na de Use Case.
 - Helpt bij validatie en testing → testers kunnen controleren of de verwachte uitkomst klopt.
 - Voorkomt onduidelijkheid over de systeemstatus na afloop van de interactie.
- Samenvatting:
 - Postcondities maken expliciet wat het resultaat van de Use Case is en zorgen voor duidelijke verificatiecriteria!

SEQUENTIES VAN USE CASES MET PRE- EN POSTCONDITIES



Use cases interageren niet met elkaar. Toch kan het zijn dat de postconditie van een use case een precondition is voor een andere use case.

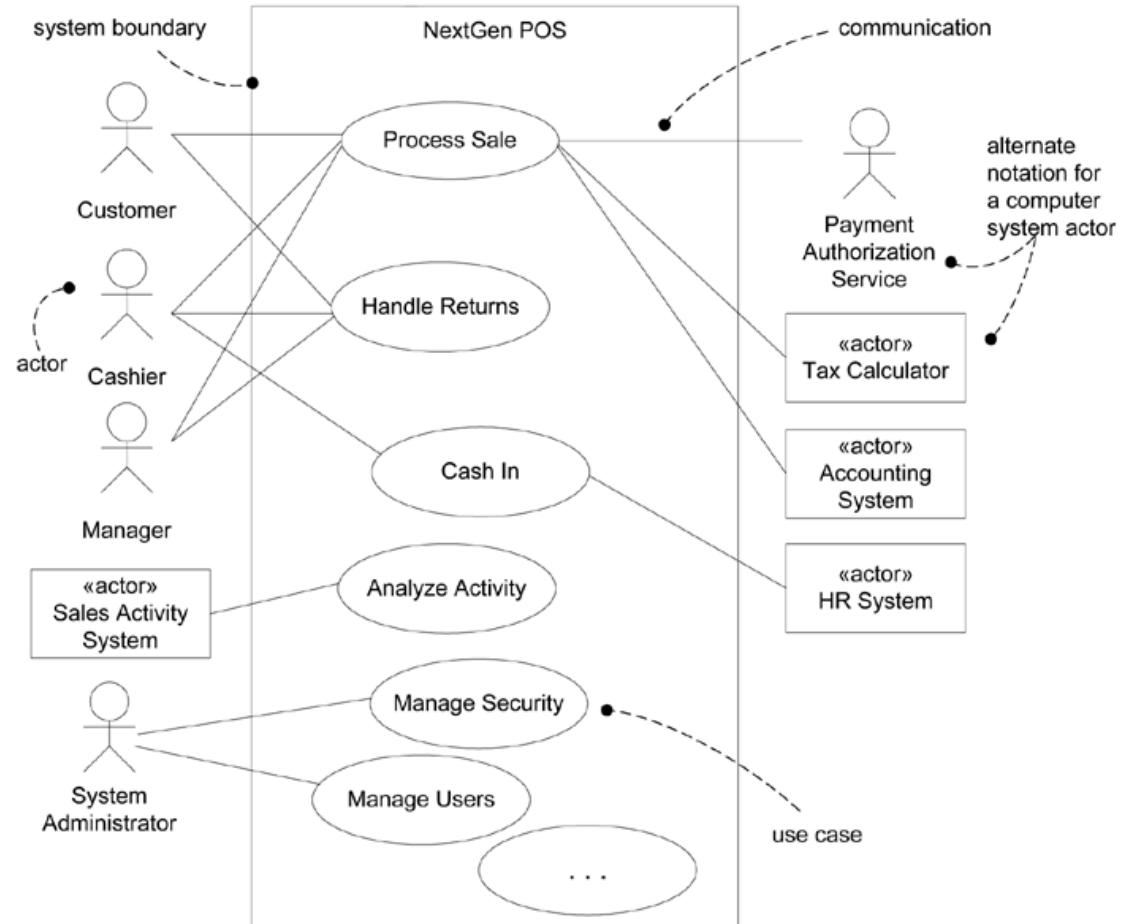
USE CASES

- **Special Requirements (Bijzondere Vereisten)**
 - **Niet-functionele vereisten** zoals prestaties, beveiliging en gebruiksvriendelijkheid.
 - **Data- en bedrijfsregels** die invloed hebben op de Use Case.
 - Voorbeeld: "De betaling moet **binnen 3 seconden** verwerkt worden."
- **Extension Points (Uitbreidbare punten)**
 - Specifieke momenten in de **event flow** waar **extra gedrag** kan worden toegevoegd.
 - Gebruik voor uitbreidingen ("extend") in complexe scenario's.
 - **Voorbeeld:**

"Tijdens het afrekenen kan een klant kiezen om een kortingscode toe te voegen (extension point: 'Korting toepassen')."
- **Additional Information (Aanvullende informatie)**
 - **Extra verduidelijkingen** die nodig zijn voor het correct begrijpen van de Use Case.
 - Kan gaan over **business context, uitzonderingen of specifieke beperkingen**.
 - **Voorbeeld:**

"Deze Use Case mag enkel worden uitgevoerd door een gebruiker met een admin-rol."

USE CASES



**VOORBEELD VAN EEN
USE CASE:**

**"BIBLIOTHEEKBOEK
LENEN"**



ALGEMENE INFORMATIE

- **Use Case Naam:** Bibliotheekboek Lenen
- **Doel:** Een geregistreerde gebruiker leent een boek uit de bibliotheek.
- **Actoren:**
 - **Primaire Actor:** Bibliotheekgebruiker
- **Systeemgrens:** Het bibliotheeksysteem beheert de boekuitlenen, inclusief controle op beschikbaarheid en leenlimieten.

PRECONDITIONS

- De gebruiker moet geregistreerd en ingelogd zijn.
- Het boek moet beschikbaar zijn voor uitlening.

BASIC FLOW (HAPPY PATH)

1. De gebruiker zoekt een boek op in het bibliotheeksysteem.
2. Het systeem toont de beschikbaarheidsstatus van het boek.
3. De gebruiker selecteert "*Boek lenen*".
4. Het systeem controleert of de gebruiker binnen zijn uitleenlimiet blijft.
5. Het systeem registreert de uitlening en stelt de uitleendatum en retourdatum vast.
6. Het systeem bevestigt de uitlening en toont de uitleendetails.
7. De gebruiker ontvangt een bevestigingsmail met de leendetails.

ALTERNATIVE FLOWS

- **A1. Boek niet beschikbaar:**
 - Het systeem geeft aan dat het boek is uitgeleend en biedt de optie om een reservering te plaatsen.
- **A2. Gebruiker heeft te veel openstaande leningen:**
 - Het systeem weigert de aanvraag en toont een melding over de uitleenlimiet.
- **A3. Gebruiker heeft openstaande boetes:**
 - Het systeem weigert de aanvraag en vraagt de gebruiker om de boete te betalen.

POSTCONDITIES

- De uitlening is geregistreerd en het boek wordt toegewezen aan de gebruiker.
- De beschikbaarheidsstatus van het boek is aangepast.
- De gebruiker heeft een bevestiging ontvangen.

SPECIAL REQUIREMENTS

- Een gebruiker kan maximaal **5 boeken tegelijk lenen.**
- De maximale uitleentermijn is **4 weken.**
- Boetes worden automatisch berekend bij te late inlevering.
- Het systeem moet real-time beschikbaarheid tonen.