OO-design

Course OOSE-OOAD   
2020-2021 Semester 2  
Robert Boudewijn  
www.robertboudewijn.nl

Inhoudsopgave

[1. GRASP 2](#_Toc66721148)

[1.1. Controller 2](#_Toc66721149)

[1.2. Creator 2](#_Toc66721150)

[1.3. Information Expert 3](#_Toc66721151)

[1.4. Low Coupling 3](#_Toc66721152)

[1.5. High Cohesion 3](#_Toc66721153)

[2. SOLID 3](#_Toc66721154)

[2.1. Single Responsibility Principle 3](#_Toc66721155)

[2.2. Open/Closed Principle 3](#_Toc66721156)

[2.3. Dependency Inversion Principle 3](#_Toc66721157)

[3. GoF Design Patterns 3](#_Toc66721158)

[3.1. Observer 4](#_Toc66721159)

[3.2. Strategy 5](#_Toc66721160)

[3.3. State 5](#_Toc66721161)

[3.4. Adapter 5](#_Toc66721162)

[3.5. Factory Method 5](#_Toc66721163)

[Bibliografie 6](#_Toc66721164)

# GRASP

GRASP staat voor ‘Generalized Responsibility Assignment Software Patterns/Principles’. Het bestaat uit 9 richtlijnen die kunnen worden gebruikt om in objectgeoriënteerde systemen verantwoordelijkheden toe te kennen aan klassen of objecten. Elke richtlijn behandelt een combinatie van een veelvoorkomend probleem en een algemeen toepasbare oplossing voor dat probleem met betrekking tot het ontwerp van softwaresystemen. (Wikipedia-bijdragers, 2016)

Voor de laatste letter van acroniem GRASP wordt zowel Patterns (patronen) als Principles (richtlijnen) gebruikt. Een pattern is een beschrijving van een probleem en een oplossing, dat kan worden toegepast in nieuwe contexten (M. Koolwaaij, 2020). Lees hierover meer in hoofdstuk 3.

De 9 richtlijnen zijn:

1. Controller
2. Creator
3. Information Expert
4. Low Coupling
5. High Cohesion
6. Polymorphism
7. Pure Fabrication
8. Indirection
9. Protected Variations

In OOSE OOAD word er gefocust op de eerste 5.

## Controller

De controller is verantwoordelijk voor het behandelen van de verzoeken van actoren. De controller is de tussenpersoon tussen uw gebruiker die op "Verzenden" klikt en uw back-end om dat te laten gebeuren (R. Fadatare, 2019).

**Voorbeeld van probleem:**  
Welke (niet-UI) class moet de system events afhandelen?

**Oplossing:**

Geef de responsibility aan een class die:

1. Het totale systeem representeert
   1. Façade controller
2. Of de use case representeert
   1. Use-case controller

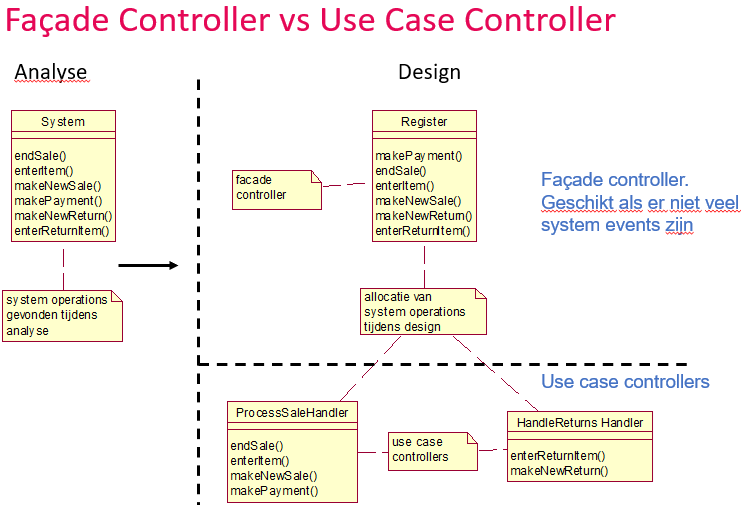
**Factoren die de keuze beïnvloeden:**

1. De mate van koppeling of cohesie

Figuur 1; Hoort het Syteem al deze events op te vangen of is dat dus een taak van een controller?

1. Hoeveel verantwoordelijkheden krijgt de façade controller? Misschien zijn nog meer controllers nodig, en moet de façade controller delegeren

**Een Controller is géén user-interface object**,system events mogen daardoor niet worden afgehandeld door de presentatielaag (UI) (M. Koolwaaij, 2020).



Figuur 2; Een Façade controller is in dit geval een algemene super controller. Dit is handig wanneer je een snel een applicatie wilt opzetten. Echter wanneer het (te) groot (-er) wordt is het handig om de meerdere kleine controllers te maken.

## Creator

Creator is een GRASP-patroon dat helpt om te beslissen welke klasse verantwoordelijk moet zijn voor het maken van een nieuwe instantie van een klasse. Het maken van objecten is een belangrijk proces en het is handig om een principe te hebben om te beslissen wie een instantie van een klasse moet maken. (R. Fadatare, 2019)

**Voorbeeld van een probleem:**

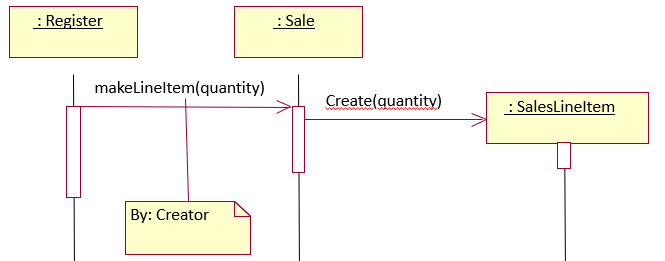
Welk object mag een instantie van een class A aanmaken?

**Oplossing:**

Geef die verantwoordelijkheid aan een class B waarvoor geldt:

1. B ‘bevat' A
2. B legt A vast
3. B gebruikt A het meest
4. B heeft de initialisatie-data voor A

(M. Koolwaaij, 2020)



Figuur 3; In dit geval maakt Sale gebruik van een SalesLineItem. Dus hebben we temaken met optie b. B legt A vast

## Information Expert

information-expert is een principe dat wordt gebruikt om te bepalen waar verantwoordelijkheden, zoals methoden, berekende velden, enzovoort, moeten worden gedelegeerd. (R. Fadatare, 2019)

**Voorbeeld van probleem:**

Ik moet responsibilities aan classes toewijzen welk algemeen principe kan ik hierbij gebruiken?

**Oplossing:**

Wijs een responsibility toe aan die class die de vereiste informatie heeft om aan die responsibility te voldoen.

**Voordeel van Expert:**

Information hiding( een principe dat stelt dat interne details van de implementatie van een klasse verborgen moeten blijven voor andere klassen) wordt dan in stand gehouden. (M. Koolwaaij, 2020)



Figuur 4; Hier is goed te zien dat Information Expert is toegepast. Omdat, een total behoort bij een Sale, een SubTotal bij een SalesLineItem en een Prijs bij een Product(Specification)

## Low Coupling

## High Cohesion

# SOLID

## Single Responsibility Principle

## Open/Closed Principle

## Dependency Inversion Principle

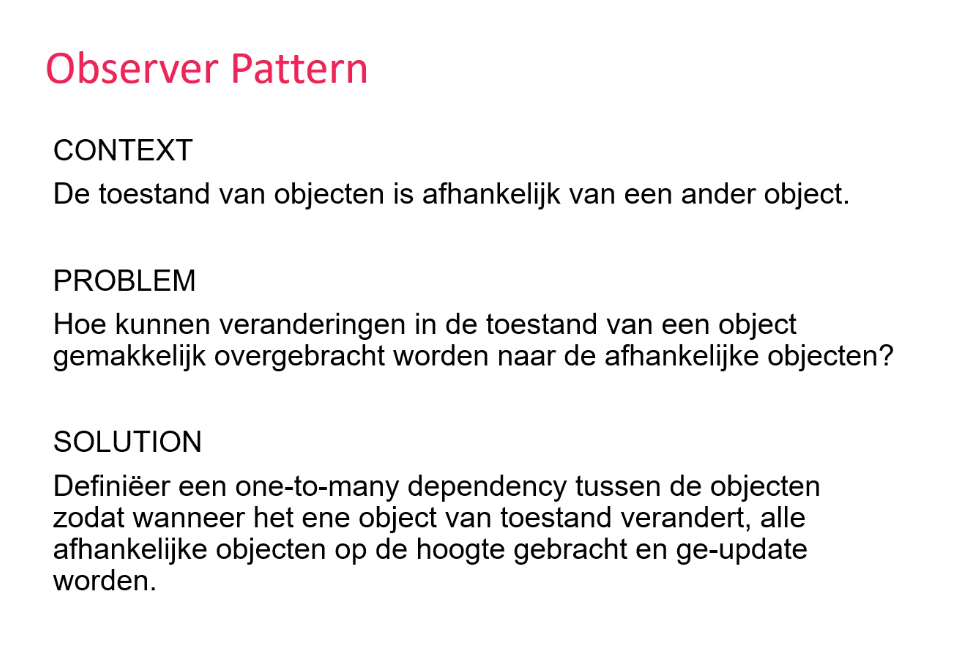
# GoF Design Patterns

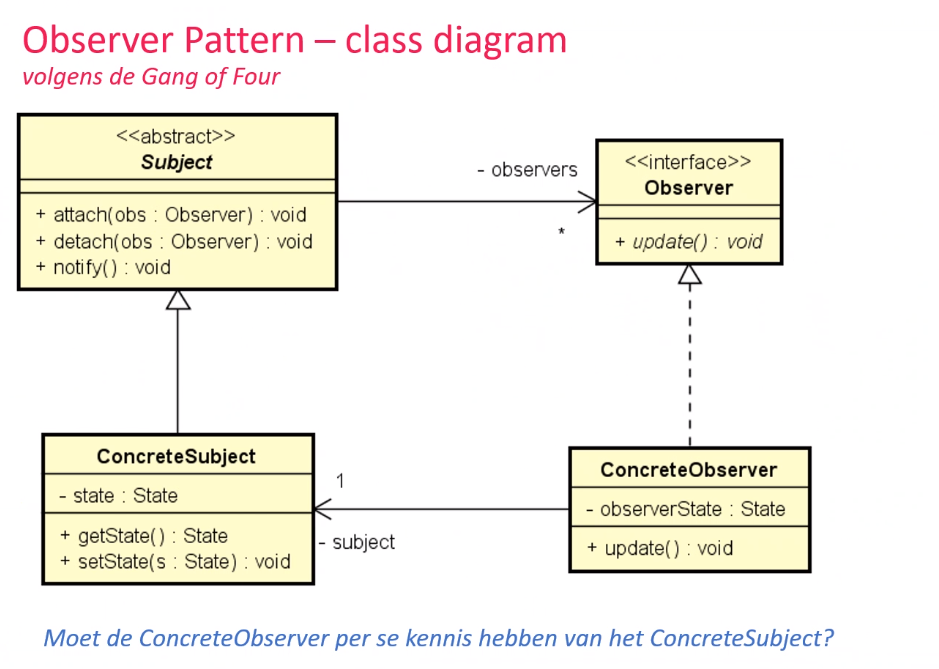
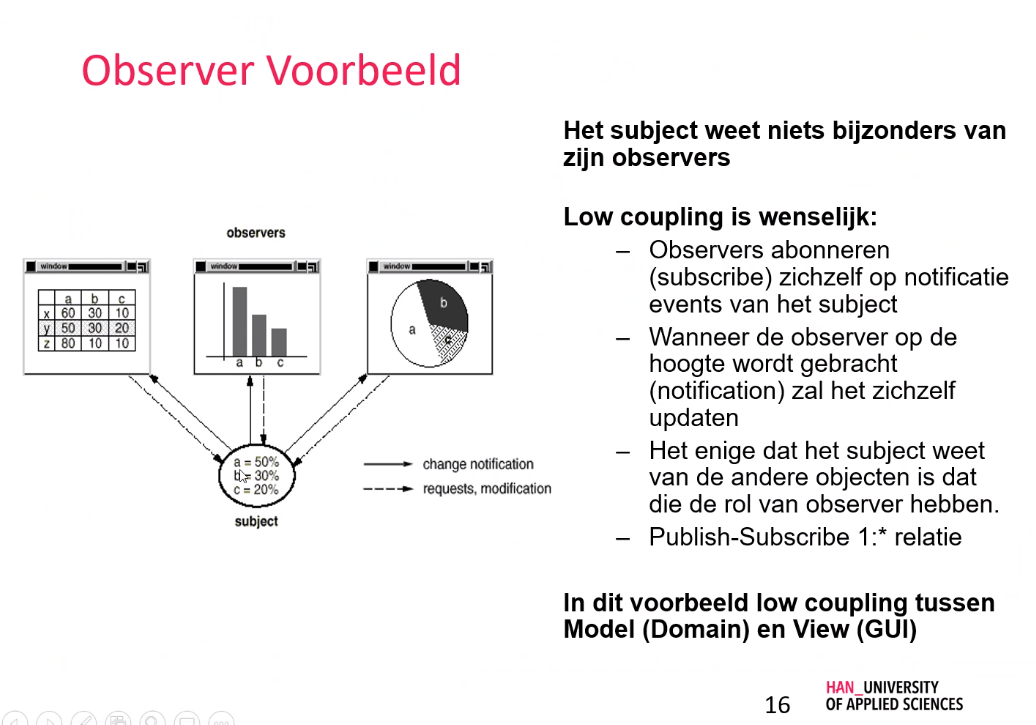
Een pattern is een beschrijving van een probleem en een oplossing, dat kan worden toegepast in nieuwe contexten (M. Koolwaaij, 2020). Patterns hebben een naam, probleem, oplossing en voorbeelden en/of discussies (M. Koolwaaij, 2020). Hierover meer in hoofdstuk 3.

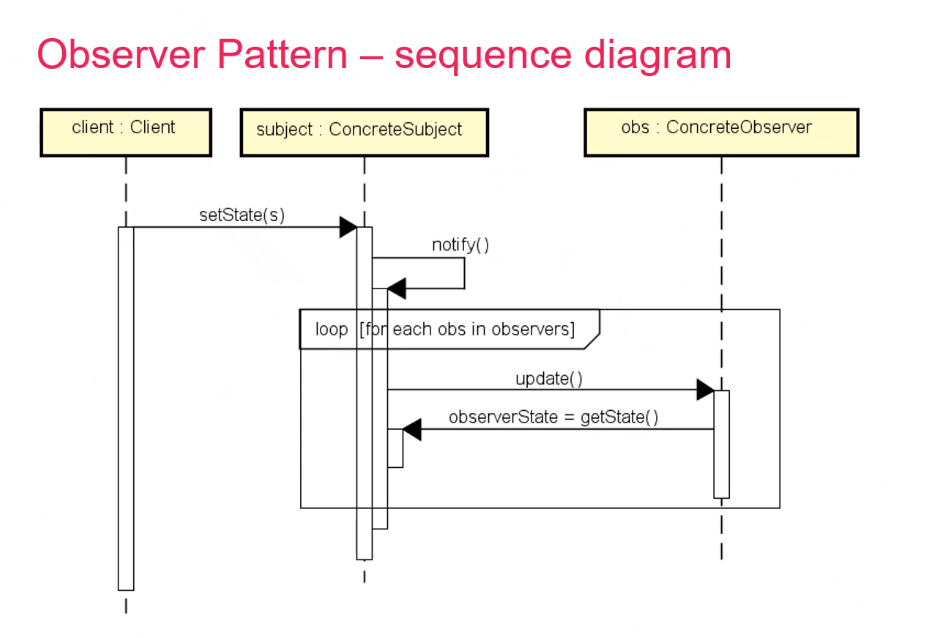
Een voorbeeld is:

**Context:**  Een man wil zijn auto in de garage zetten.  
**Problem:**  De garage is heel smal.  
**Solution:** https://www.youtube.com/watch?v=GfQlWCufkRo&ab\_channel=R0FLMAOWPIMP

## Observer





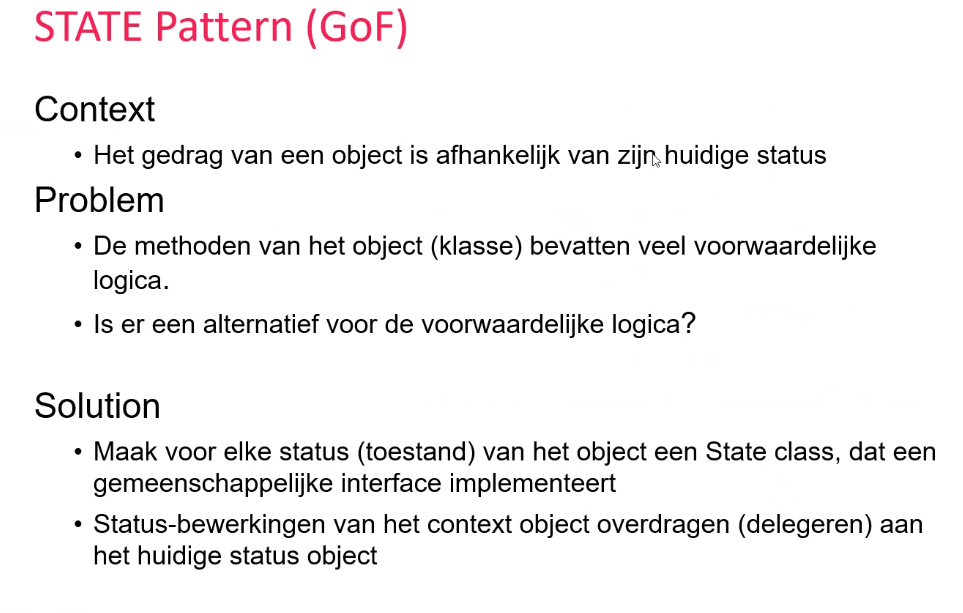
## Strategy





## State

Een state pattern heeft te maken met een state machine. Deze hebben we gehad in de propedeuse met EPD.



## Adapter

## Factory Method

# Bibliografie

M. Koolwaaij. (2020, 08 26). Thema 8 OO Design principles.

R. Fadatare. (2019, 06 15). *Controller GRASP Pattern*. Opgehaald van sourcecodeexamples: https://www.sourcecodeexamples.net/2018/06/controller-grasp-pattern.html

R. Fadatare. (2019, 06 16). *Creator GRASP Pattern*. Opgehaald van sourcecodeexamples: https://www.sourcecodeexamples.net/2018/06/creator-grasp-pattern.html

R. Fadatare. (2019, 06 16). *Information Expert GRASP Pattern*. Opgehaald van sourcecodeexamples: https://www.sourcecodeexamples.net/2018/06/information-expert-grasp-pattern.html#:~:text=Information%20expert%20(also%20expert%20or,computed%20fields%2C%20and%20so%20on.

Wikipedia-bijdragers. (2016, 01 08). *GRASP*. Opgehaald van Wikipedia: https://nl.wikipedia.org/wiki/GRASP#:%7E:text=GRASP%20is%20een%20Engels%20acroniem,kennen%20aan%20klassen%20of%20objecten.

# Figuren

[Figuur 1; Hoort het Syteem al deze events op te vangen of is dat dus een taak van een controller? 2](file:///C:\Users\Rober\Documents\1.0%20ICA\Leerjaar%202\blok%203\AIM-OOSE--\OOAD\Samenvatting%20OO%20design%20.docx#_Toc66724049)

[Figuur 2; Een Façade controller is in dit geval een algemene super controller. Dit is handig wanneer je een snel een applicatie wilt opzetten. Echter wanneer het (te) groot (-er) wordt is het handig om de meerdere kleine controllers te maken. 3](#_Toc66724050)

[Figuur 3; In dit geval maakt Sale gebruik van een SalesLineItem. Dus hebben we temaken met optie b. B legt A vast 4](#_Toc66724051)

[Figuur 4; Hier is goed te zien dat Information Expert is toegepast. Omdat, een total behoort bij een Sale, een SubTotal bij een SalesLineItem en een Prijs bij een Product(Specification) 4](#_Toc66724052)