

Ontwerp van Informatiesystemen

Prof. Verelst



uickprinter
Koningstraat 13
2000 Antwerpen
www.quickprinter.be

Online samenvattingen kopen via

www.quickprintershop.be

Like us on Facebook!



www.facebook.com/quickprintershop

Ontwerp van Informatiesystemen

Table of Contents

Hoofdstuk 1: Wat is systeemontwerp?	3
Inleiding	3
Informatiesystemen	3
Organisaties en IS.....	3
Software engineering	4
Watervalmodel	5
Analyse	5
Ontwerp	5
Implementatie.....	5
Testen	6
Onderhoud	6
Nuancering	6
Methodologieën.....	7
Gestructureerd vs object-georiënteerd ontwerp	7
Hoofdstuk 2: Variabelen en Datastructuren	8
Constanten	8
Variabelen	8
Declaratie	8
Initialisatie	9
Toekenning.....	9
Basistypes	9
Integer	9
Real.....	10
Char en string	10
Boolean	10
Datastructuren	10
Arrays	10
Records.....	11
Hoofdstuk 3: Basisstructuren	11
Sequentie	11
Iteratie	11
For...next.....	12
Repeat... until	12
While.....	12
Selectie	12
If... then	12
Switch	13
Hoofdstuk 4: Procedures	13
Wat zijn procedures	13
Functie.....	13

Globale en lokale variabele	13
Parameters	14
Recursiviteit.....	14
Voordelen van procedures.....	14
Hoofdstuk 6: Softwarekwaliteit	15
Kwaliteit	15
Kwaliteitseisen	15
Afweging tussen kwaliteitseisen	16
Verband met onderhoud	16
Verband met architectuur.....	17
Hoofdstuk 7: Modules	17
Modules	17
Definitie	17
Communicatie	18
Kenmerken	18
Ontwerpen met modules: Voordelen	18
Bepalen van modules	19
IVO-architectuur.....	19
Koppeling	19
Cohesie.....	21
Hoofdstuk 9: Inleiding tot Object-oriëntatie	22
Inleiding.....	22
Abstract Data Type (ADT).....	22
Object-Oriëntatie.....	22
Objecten en Klassen	22
Attributen en methoden	23
Voordelen van OO	23
Inkapseling	23
Parameters in Java	24
UML.....	24
Objecten identificeren	24
Modulariteit en object-oriëntatie	24
Hoofdstuk 10: Relaties tussen klassen	26
Associatie	26
Generalisatie	26
Aggregatie	27
Hoofdstuk 11: Eigenschappen van Object-oriëntatie	27
Overerving.....	27
Polymorfisme	28
Statische Attributen en Methoden.....	29
Hoofdstuk 12: Patronen	29
Ontwerppatronen.....	29
Voordeel.....	30
Voorbeelden.....	30

- Subklasse erven methode over maar moeten ze wel schrijven
- Abstracte klasse kan concrete methode bevatten → niet andersom
- Interfaces
 - Een pure abstracte klasse
 - Enkel abstracte methode
 - Fungeert als template
 - Definieert welke methoden subklasse moeten bezitten
 - Overerving van meerdere interfaces kan wel
- Overloading
 - 2^{de} voorbeeld van polymorfisme
 - Geld voor methode binnen dezelfde klasse
 - Methode met dezelfde naam, maar verschillende parameterlijst
 - Bv constructen met verschillende parameters (eens naam niet meegeven want niet nodig)
 - Courant gebruikt bij constructor
 - De signature moet verschillen
 - Parameters moet verschillen
 - Type van parameters moet verschillen
 - Handig → let op cohesie!

Statische Attributen en Methoden

- Attributen en methode van klasse zijn mogelijk
 - Bv hoeveel objecten per klasse
 - Methode/attribuut aanroepen op de klasse → geen object voor nodig
 - Statisch attribuut → lijkt op globale variabele pascal
 - Maar enkel bruikbaar in de klasse
- Waarom dan Public static void main(String[] args)
 - Public → moet mainprogramma kunnen opstarten
 - Static → moet nog geen object hebben
 - Void → geen returnwaarde

Hoofdstuk 12: Patronen

Ontwerppatronen

- Oplossing voor veelvoorkomend ontwerpprobleem
- Kwaliteit empirisch bewezen
- Eigenschappen
 - Naam

- Probleem
- Oplossing
- Gevolgen
- Basiswerk
 - Gamma → 23 oplossingen
- Nieuwe patronen voortdurend voorgesteld
- Kennis van patronen is een essentieel onderdeel van de kennis van de programmeur

Voordeel

- Onderhoudbaarheid
- Hergebruikbaarheid
- Correctheid
 - Is al veel gebruikt en is dus correct
- Programmeertaalafhankelijk
- Communicatie

Voorbeelden

- Creational: aanmaak van objecten
- Structural: objecten samenvoegen
- Behavioral: communicatie en gedrag van objecten

Singleton

- Probleem: 1 en slechts 1 object van een klasse aanmaken
 - Vb. Drie pc willen iets afprinten te gelijk → printer drukt pagina's willekeurig af
- Oplossing → singleton
 - 1^{ste} aanroep: creëert object
 - 2^{de} aanroep: krijgt 1^{ste} object terug want er mag maar 1 object zijn
 - vb. Maar 1 printer

Facade

- Probleem: onoverzichtelijke verbanden tussen klassen
 - Vb. Software voor bankautomaat
 - Alle controles moeten uitgevoerd worden
- Oplossing
 - Facadeklasse: Extra klasse
 - Roept alle controles aan
 - Als er extra controles bijkomen → geen verandering in main
 - Enkel extra aanroep in facadeklasse

Observer

- Objecten die moeten reageren bij statuswijziging aan een ander object
- Oplossing

- Niet elke seconde vragen of er een verandering is
- Wel programma een notify laten sturen als data veranderd
- Koppeling