Ontwerp Softwaresystemen
Module 1.2 van de studies BIT en INF
— Werkdocument —

Luis Ferreira Pires
Marieke Huisman
Jan Kamphuis
Arend Rensink
Klaas Sikkel

22 mei 2013

Inhoudsopgave

1 Inleiding

Dit is het ontwerpdocument van Module 1.2 (Softwaresystemen) in het beoogde nieuwe BSc-curriculum van INF en BIT. Het beschrijft de opzet van de module, de gemaakte keuzes, en de beoogde invulling van de tien weken van de module, in termen van werkvormen , behandelde stof en ideeën voor opgaven en toetsen.

Aan de hand van dit document zal de module in meer detail ingevuld worden. De volgende stap is het opzetten van een handleiding waarin de weekindelingen en opgaven in meer detail uitgewerkt zullen worden.

Hieronder vatten we eerst de belangrijkste randvoorwaarden voor het ontwerp van de module op. In de volgende paragrafen gaan we op het eigenlijke ontwerp in.

1.1 Ontwerpcriteria

De ontwerpcriteria voor de module, inclusief leerdoelen, zijn in een eerder stadium opgesteld. Voor de volledigheid zijn de criteria als appendix aan dit document toegevoegd (Appendix A).

1.2 Inbedding in het curriculum

De hier beschreven module zal afgenomen worden door de studierichtingen INF en BIT. In deze opleidingen vormt dit Module 1.2.

De ervaring met het bestaande curriculum leert dat de inhoud voor BIT-studenten lastig is, terwijl de module voor INF-studenten tot de kern van de studie en vaak ook het interessegebied behoort; maar ook onder de laatsten zijn er elk jaar een aantal die veel minder affiniteit met het onderwerp blijken te hebben en er hard voor moeten werken.

1.2.1 Benodigde voorkennis

Bij het ontwerp van de module is er van uitgegaan dat in Module 1.1 al enige onderwerpen aan de orde komen die hier als voorkennis kunnen worden verondersteld:

Algoritmiek Kennismaking met imperatief algoritmisch denken en elementaire complexiteitscriteria, aan de hand van zoek- en sorteeralgoritmen op integer-arrays.

Recursie Kennismaking met recursiviteit (in het kader van functioneel programmeren).

Zowel BIT- als INF-studenten hebben van deze onderwerpen kennis genomen. Dit is vanwege het hierboven genoemde verschil in vaardigheden tussen de studenten van deze twee studies van groot belang.

1.2.2 Geleverde voorkennis

De module Softwaresystemen (1.2) levert directe voorkennis aan de module Data & Informatie (1.4). Ook andere modules kunnen er van uit gaan dat studenten na het volgen van Module 1.2 voldoende ontwerp- en programmeervaardigheden hebben.

1.3 Wiskundelijn

Op UT-niveau is vastgelegd dat elke eerstejaarsmodule naast het opleidingsspecifieke gedeelte ook een component wiskunde inhoudt, in het vervolg de *wiskundelijn* genoemd. Voor Module 1.2 is de omvang van de wiskundecomponent 3 EC, oftewel eenvijfde van de totaal beschikbare tijd.

De inhoud van de wiskundelijn bestaat uit:

Newton Functieleer, integreren.

De verroostering van de wiskundelijn is UT-breed afgesproken, in ieder geval voor wat totale omvang en tijdstippen van hoorcolleges en toetsing betreft. De indeling van de werkcollege- en zelfstudie-uren is in overleg met de opleiding bepaald. In dit document wordt slechts sporadisch ingegaan op de inhoud van de wiskundelijn.

2 Opzet en keuzes

2.1 Globale indeling

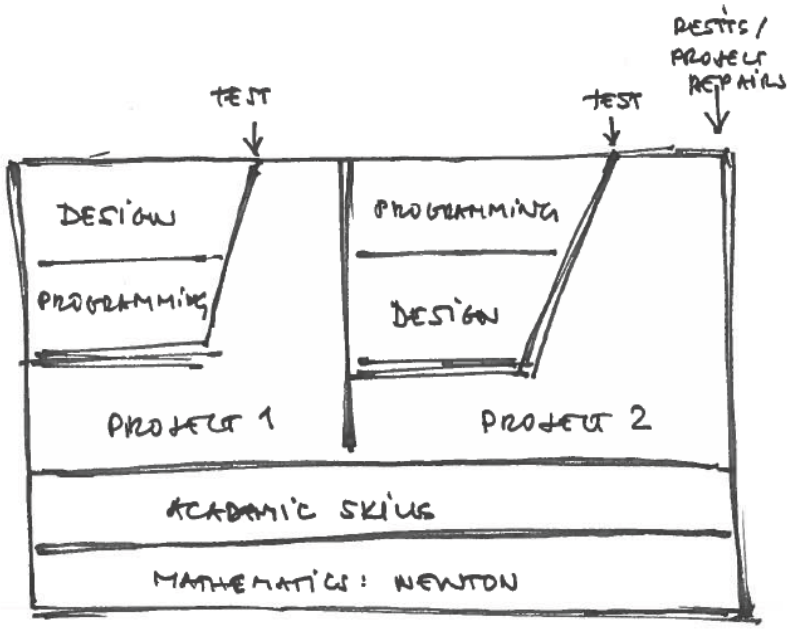
Naast de wiskundelijn bestaat de module uit twee (hoofd)leerlijnen: de ontwerp- en de programmeerlijn. Om een idee te geven: in vergelijking met het bestaande curriculum overlapt de ontwerplijn grotendeels met de vakken Informatiesystemen (IS) en Software Engineering-modellen (SEM) en de programmeerlijn met de vakken Programmeren 1 en 2. Inhoudelijk wordt er een sterk verband gelegd tussen ontwerp- en programmeerlijn.

Orthogonaal op de leerlijnen staan de zgn. *crosscutting concerns*; zie §2.4. Deze zijn grotendeels ingebed in de stof van de ontwerp- dan wel programmeerlijn; dat geldt echter minder voor het onderwerp "academische vaardigheden", dat om die reden in dit document wordt beschreven als vierde leerlijn.

In het onderstaande gebruiken we de volgende afkortingen voor de leerlijnen in Softwaresystemen:

W De wiskundelijn
O De ontwerplijn
P De programmeerlijn
A Academische vaardigheden (zie §2.4.4)

Figuur 1 geeft de houtskoolschets van de module, met daarin een verdeling van de O- en P-lijnen over het kwartiel, alsmede de W-lijn en de academische vaardigheden.



Figuur 1: Houtskoolschets van de module

2.2 Werkvormen

Uitgangspunt: De hele week is verroosterd, met 40 uur/week (geen 42!). Dat wil zeggen dat er voor elk uur een beoogde soort activiteit (werkvorm) in het rooster is opgenomen, in combinatie met de leerlijn.

Het werk wordt uitgevoerd in groepen van wisselende grootte. De gebruikte terminologie:

G∞ Alle studenten die de module volgen
G24 Traditionele werkcollegegroep; in de praktijk 20–30 man, soms minder
G*x* met $x = 2, 4, 8, \ldots$: een groep die in principe uit x studenten bestaat, gebruikt bij practicum- en project-vormen

In Tabel 1 worden de werkvormen opgesomd en uitgelegd.

Afk	Naam	Omschrijving	Groep
col	**Colstructie**	Een mengvorm van hoorcollege en werkcollege.	∞
diag	**Diagnostische toets**	Gebruikt om studenten de kans te geven hun eigen kunnen in te schatten. Deelname verplicht, geen effect op eindcijfer.	∞
ipr	**Instructiepracticum**	Een mengvorm van werkcollege en (strak geregisseerd) practicum: studenten doen kleine opdrachten, theoretisch en praktisch, die afgetekend moeten worden. Dit wordt gedaan in G2-verband, per G24 begeleid door een docent of studentassistent.	2
hc	**Hoorcollege**	College voor G∞ waarin de stof geheel of gedeeltelijk de revue passeert. De functie van het hc is hoofdzakelijk motiverend en structurerend; het wordt in het onderstaande daarom ook wel "motiverend hoorcollege" (mhc) genoemd.	∞
mhc	**Motiverend hc**	Zie hoorcollege	∞
pfb	**Peer feedback**	Studenten vertellen elkaar wat ze gedaan hebben en wat ze van andermans werk vinden. In principe zelfgestuurd.	6–8
pj	**Project**	Deels in college/practicumzaal met aanwezige SA. Het project wordt in G4- of G2-verband gedaan (voor O- resp. P-lijn), per G24 door een SA begeleid.	2 of 4
qa	**Vragenuur**	Gelegenheid voor studenten om vragen te stellen, voor docenten om nader toe te lichten.	∞
tts	**Toets**	Summatieve toets, deel uitmakend van het tentamen.	∞
wc	**Werkcollege**	Groepssessie waarin opgaven door de studenten individueel worden doorgewerkt met actieve aanwijzingen van de docent; af en toe wordt iets klassikaal uitgelegd.	24
zs	**Zelfstudie**	Vindt thuis plaats, of (in sommige gevallen) onder begeleiding in een collegezaal.	1

Tabel 1: Werkvormen

2.3 Materiaal

Als materiaal is gekozen voor twee boeken en een combinatie van tools. Daarnaast zal er een handleiding worden samengesteld.

Ontwerplijn: *Object Oriented Systems Analysis and Design using UML*, Edition 4, van Bennett, McRobb & Farmer [Bennett].

Dit boek is gekozen omdat het een goede balans kent tussen inzicht en praktische informatie, de laatste met name over het gebruik van UML. Hiermee is het niet nodig om een apart boek alleen voor UML te gebruiken. Bovendien is het boek redelijk nieuw (4de editie in 2010 uitgebracht) en als zodanig refereert het naar de laatste versie van UML en naar recente literatuur. Het boek heeft de vorm van een studieboek, met een inzichtelijke indeling en voldoende voorbeelden.

Voor software-metrieke in week 7 gaan we zelf een stuk schrijven op basis van boeken en informatie van het Internet. Daarnaast gaan we een Eclipse plugin gebruiken in het instructiepracticum, zoals bijvoorbeeld `http://metrics.sourceforge.net/`[1]. Dat betekent dat de terminologie in ons eigen stuk over software-metrieke op de gekozen plugin afgestemd moet worden.

Programmeerlijn: *An Introduction to Programming and Object-Oriented Design Using Java* van Niño & Hosch [Niño & Hosch], aangevuld met stukken uit CoreJava (deel I en II) van Horstmann & Cornell [Core Java 1, Core Java 2].

[1] Hoewel op deze site staat dat Eclipse 3.1 vereist is, werkt deze plugin prima met Eclipse 4.2

Het boek van Niño & Hosch wordt al jaren met tevredenheid gebruikt in Programmeren 1&2. Het gebruikt de "objects first"-strategie die wij ook voorstaan (en die nodig is om de band met de ontwerplijn goed te krijgen) en bevat de meeste onderwerpen die we willen behandelen.

Voor het schrijven van specificaties gebruiken we als aanvullend materiaal een eigen handleiding over JML. Dit wordt een gestripte versie van de JML handleiding die gebruikt wordt bij System Validation (pre- en postconditions, klasse-invarianten, specificatie-expressies), geschreven door Marieke Huisman. Dit wordt als appendix in de reader opgenomen.

Onderwerpen die niet in [Niño & Hosch] staan zijn: multithreading en netwerkprogrammeren en security. Hiervoor gebruiken we het volgende materiaal:

- Multithreading, hoofdstuk 14 van [Core Java 1], tot *BlockingQueues* (p. 819 - 877).
- Networking, hoofdstuk 3 van [Core Java 2], tot *Making URL Connections* (p. 185 - 210)
- Security, [Core Java 2], hoofdstuk 9 volledig (p. 803 - 892).

De bedoeling is dat het niet nodig is dat studenten [Core Java 1, Core Java 2] aanschaffen, in plaats daarvan willen we hoofdstukken uit het boek in de handleiding zetten dan wel een elektronische versie van het boek op Blackboard zetten. In hoeverre dit copyright-technisch kan wordt nog uitgezocht.

Tooling: We voorzien het gebruik van de volgende tools.

- Eclipse in combinatie met Visual Paradigm (zie `http://www.visual-paradigm.com`)
- OpenJML voor het typechecken van specificaties (zie `http://jmlspecs.sourceforge.net/`)
- voor test coverage: EMMA, stand-alone en Eclipse plugin (zie `http://emma.sourceforge.net/`, `http://www.eclemma.org/`)
- aanbieden van mogelijkheid om verschillende tools te gebruiken die feedback geven op programmeerstijl etc. (o.a. CheckStyle).

Eclipse wordt momenteel ook al gebruikt voor het de programmeervakken. Voor Visual Paradigm is gekozen omdat het naar verwachting zowel voor de ontwerp- als voor de programmeerlijn bruikbaar is; daarnaast is het met een gratis onderwijslicentie beschikbaar met de gewenste functionaliteit (alle diagrammen die nodig zijn voor de ontwerp- en programmeerlijn, en "reverse engineering") en wordt actief onderhouden. Visual Paradigm wordt dit jaar beschikbaar gesteld aan de Programmeren 2 studenten als alternatief voor de verouderde Borland tools.

OpenJML lijkt de meest geschikte tool voor JML. Alle onderdelen van JML die we gebruiken worden door OpenJML ondersteunt. Geïnteresseerde studenten kunnen ook de mogelijkheden uitproberen om specificaties te valideren (bijv. door runtime checking). Informatie hoe dit gegaan kan worden moet ergens beschikbaar zijn (bijv. in context van een bonusopgave).

EMMA is een tool dat op eenvoudige wijze meet welke regels Java-code tijdens het draaien van een programma worden uitgevoerd, en hier dan een net rapportje in HTML-formaat van kan ophoesten. Daaruit valt bijvoorbeeld line coverage direct af te leiden. Het tool bestaat ook als Eclipse-plugin. De bedoeling is dat studenten dit gebruiken om te meten en te laten zien in hoeverre hun testcases de hele code afdekken.

Voor de programmeer-feedback tools bestaan een aantal systemen die de studenten automatisch feedback kunnen geven op hun programmeerstijl, commentaar etc. Deze systemen moeten meestal wel gefinetuned worden voor de eisen die wij stellen. Het plan is om een studentassistent aan te stellen die zich hier in gaat verdiepen, en om de studenten vervolgens een geschikte omgeving aan te bieden. Gebruik hiervan zal niet verplicht zijn.

Handleiding: Bevat alle opgaven voor de ontwerp- en programmeerlijn, per week ingedeeld.

De handleiding beschrijft van week tot week wat er van de studenten verwacht wordt. Zowel de opgaven die elke week gemaakt worden als de randvoorwaarden voor het project staan hierin vermeld. Voor de docenten is er een exemplaar waarin ook de uitwerkingen staan. *De handleiding zal in het Engels worden uitgevoerd.*

Een (groot) deel van de uitwerkingen zal achteraf via Blackboard aan de studenten ter beschikking worden gesteld.

Overig: Ook bij de W-lijn en de academische vaardigheden (zie §2.4.4) hoort materiaal; de keuze hiervan is elders al gemaakt. Voor Academische Vaardigheden wordt gebruik gemaakt van *Skill Sheets* (Van Tulder, Pearson 2012) [Skill Sheets].

2.4 Crosscutting concerns

In elke module van de opleiding INF dient aandacht te worden geschonken aan een aantal "crosscutting concerns." Hieronder wordt puntsgewijs aangegeven welke onderwerpen in deze module aan de orde komen.

2.4.1 System development

Software Management

- UML diagrammen: bijv. klassediagrammen, activiteitendiagrammen (processen), use case diagrammen (requirements)
- Software life cycles
- Gebruik Visual Paradigm

Requirements Engineering

- Requirements elicitation d.m.v. interviews
- Requirements specification d.m.v. UML-diagrammen

Testen

- Unit testen
- Maken testplan
- Test coverage meten m.b.v. plugin

Specificatie en Verificatie

- Pre- en postconditiespecificaties, klasseinvarianten en loopinvarianten, gebruik JML syntax/tools
- Enige intuitie over het redeneren met loopinvarianten om postcondities aan te tonen
- Bonusoefening: spelen met JML validatietools

2.4.2 Concurrency

- Basisprincipes concurrent programming in Java (threads, start, join, wait, notify, locks dmv synchronized)
- Gedistribueerd programmeren adhv Socket API
- Graphische User Interfaces en het belang van threads
- Geoefend tijdens practicum en toegepast in eindproject programmeerlijn.

2.4.3 Security

Dit onderwerp wordt ingevuld door studenten te laten kennismaken met de Java-ondersteuning op dit punt. Daarnaast is er een tweetal motiverende hoorcolleges. Het gaat bijvoorbeeld om de volgende onderwerpen:

- Access control mechanismen & policies
- Public Key Infrastructure, digital signatures, code signing

In het eindproject van de P-lijn komen security aspecten naar voren, zoals bijvoorbeeld authenticatie mechanismes en toegangscontrole tot resources in remote systemen.

2.4.4 Academische vaardigheden

Het thema van de module is "Zelfwerkzaamheid en Planning." De stof komt overeen met [Skill Sheets, §B3–4,B8–10].

Leerdoelen Aan het einde van de module kent de student de basisprincipes van timemanagement op het gebied van prioriteren, plannen en bewaken van de uitvoering. De student heeft kennisgemaakt met verschillende vormen van self assessment, in ieder geval op het gebied van persoonlijkheidskenmerken en uitstelgedrag. De student is in staat te reflecteren op de effectiviteit van zijn eigen gedrag op prioriteren, plannen en uitvoeren en kan hierbij op basis van de aangedragen principes concrete verbeterpunten formuleren.

Om dit in te vullen zijn de volgende contactmomenten voorzien:

- Week 1: Uitleg leerstijlen, invullen eigen schema & bespreken hiervan
- Week 2: Colstructie over persoonlijkheidskenmerken en uitstelgedrag; bijhouden eigen tijdsbesteding
- Week 3: Peer feedback op tijdsbesteding; opstellen planning projectwerk
- Week 5: Peer feedback met reflectie op projectplanning en uitvoering; analyse groepsparticipatie
- Week 6: Projectmanagement; aanwijzingen voor reflectie in eindopdracht

In het verslag van de eindopdracht is een verplichte component waarin gereflecteerd dient te worden op de planning en de uitvoering.

Voor meer detail zie de invulling van de betreffende weken.

2.5 Projecten

De O-lijn behelst een project dat wordt afgerond in week 4, en de P-lijn eindigt met een groter project. Hieronder wordt beschreven hoe deze projecten er uit zien.

2.5.1 O-project: Requirements model en requirements analysis

Bij de ontwerpopdracht werken studenten in groepen van 4 aan een requirements model en een requirements analysis model te maken van een niet-triviaal systeem. We volgen daarbij het boek van Bennett. Een requirements model omvat o.a. een requirements list, een glossary, en een use case model. Om de requirements op te stellen is het nodig met verschillende belanghebbenden te gaan praten. Een requirements analysis model voegt daar o.a. communication diagrams en class diagrams aan toe. Daarmee komen alle onderdelen die tot nu toe in de ontwerplijn behandeld en geoefend zijn aan bod in het ontwerpproject. De casus voor 2013/14 is FoodCo Ltd (casestudy B in Bennett et al.), een gefingeerd agrarisch bedrijf dat voorheen zijn producten alleen leverde aan een grote supermarkteken (als huismerk), maar nu vanwege de afgenomen marges een productielijn voor een eigen merk op wil zetten. Mocht deze casus minder goed bevallen, dan kan in volgende jaren een andere worden uitgewerkt.

Opdracht Maak een requirements model, een requirements analysis model en een analyse van interacties en toestanden voor [nader te omschrijven delen van] het nieuwe informatiesysteem voor FoodCo. Een beschrijving op hoofdlijnen is gegeven in hoofdstuk B1. Teksten van aanvullende interviews worden ter beschikking gesteld. Heeft u nog verdere vragen, dan kunt u daarvoor terecht bij [namen + rollen]. Houd er rekening mee dat deze mensen druk bezet zijn en maak tijdig een afspraak. De gevraagde modellen komen overeen met de modellen in de hoofdstukken A2, A3 en A4 in het boek.

Organisatie Voor het project staan 7,5 dagdelen ter beschikking: telkens één middag in week 1 t/m 3, in week 4 nog 4,5 dagdelen. Bij de voor het project verroosterde blokken zijn studentassistenten telkens een deel van de tijd aanwezig voor begeleiding.

Beoordeling De gevraagde analyse omvat een verslag met daarbij de volgende soorten modellen

- Requirements List
- Use Case Model
- Glossary
- Initial System Architecture
- Communication Diagrams
- Class Diagram
- Sequence Diagrams
- State Machines

Het verslag met de modellen wordt beoordeeld op

- Volledigheid van de modellen (zijn alle belangrijke eisen opgenomen)
- Syntactische en semantische correctheid (zijn de modelleringstechnieken juist gebruikt, zijn de modellen consistent en is de analyse correct)
- Leesbaarheid en overzichtelijkheid van het verslag

2.5.2 P-project: Spelletje bouwen

De studenten ontwerpen en programmeren een gedistribueerd multiplayer-game. Dit wordt gedaan in paren. Het spel wordt van tevoren door de docent(en) geselecteerd, doorgaans op basis van een bestaand (bord)spel; van deze keuze kan niet worden afgeweken. In het spel komen alle aspecten van het ontwerpen en ook alle behandelde aspecten van de taal Java aan de orde.

Opdracht Elke uitwerking bestaat uit een client en een server. Per G24 wordt er een protocol afgesproken volgens welke de clients van de ene uitwerking met de servers van de andere uitwerking moeten kunnen samenwerken. De afspraken over het protocol worden door één van de studenten bijgehouden en bij onduidelijkheden bijgewerkt; deze student krijgt hier een bonus voor.

Elke client moet ook een "automatische" spelmodus bieden, waarin de computer dus de zetkeuze doet.

Er moet een grafische user interface gemaakt worden die informatie geeft over de toestand van het spelletje.

De servers moeten communiceren met een highscore server die de beste scores bijhoudt. Deze highscore server wordt door studenten-assistenten geïmplementeerd en werkt met authenticatiemechanismes om de authenticiteit van de server te controleren. Als bonus kan er geprobeerd worden om de beveiliging van highscore server te breken om zo de positie van een gebruiker in de highscore server te verbeteren.

Organisatie Het project begint in week 5; in weken 5–7 wordt er een paar uur per week voor ingeruimd, vooral in klassikale besprekingen en discussies over de eisen en het te implementeren protocol. Ook wordt als onderdeel van het practicum gevraagd om te beginnen aan de implementatie van onderdelen van de eindopdracht.

Vanaf week 8 is er meer tijd beschikbaar voor het project; in weken 9 en de eerste helft van 10 wordt voltijds aan het project gewerkt.

Van de studenten wordt verwacht dat ze in het begin een planning maken; in de loop van het project wordt gecontroleerd dat ze zich daar ook aan houden. Deze controle (en de meeste begeleiding) gebeurt door studentassistenten.

Beoordeling De afronding bestaat uit een toernooi, waarin de programma's tegen elkaar spelen. De winnaar van het toernooi krijgt een bonus.

De uitwerkingen worden beoordeeld op een aantal aspecten, waaronder in ieder geval:

- Verslaglegging, o.a. documentatie van de definitie van eisen en het ontwerp
- Specificatie en (code-)documentatie
- Testplan en uitvoering daarvan
- Correcte werking van het programma (o.a. gedurende het toernooi)
- Grafische vormgeving
- Beveiliging van de communicatie met de highscore server
- Software-metrieke analyse met gebruik van tools zoals `http://metrics.sourceforge.net/`

Het cijfer wordt bepaald door een basiscijfer, dat bereikt wordt wanneer aan een aantal minimumeisen voldaan is; daarboven kunnen bonuspunten verdiend worden als er extra functionaliteit geïmplementeerd is.

2.6 Resources

2.6.1 Menskracht

De menskracht nodig voor deze module (zonder rekening te houden met de wiskundelijn) bestaat uit 2 docenten voor de ontwerplijn en 2 docenten voor de programmeerlijn. De module kent een modulecoördinator (liefst een

SUMMatieve cijfers (SUMM): 1 t/m 10

	Cijfer	Weging	Minimum	
Wiskunde		1	4,5	Herkansingen geregeld volgens systeem Wi
Ontwerptoets		1	4,5	Tentamenmoment week 4; herkansing week 10
Programmeertoets		1	4,5	Tentamenmoment week 9; herkansing week 10
SUMM-gem	0,0			**Gewogen gemiddelde van SUMM-cijfers**
SUMM-eind	**0,0**	3	6,0	**Maximaal 5,0 als één van SUMM-cijfers < minimum**

PERFormancecijfers (PERF): 5 t/m 10, een 5 staat voor onvoldoende inspanning

	Cijfer	Weging		
Ontwerpproject		1		Opleveren eind week 4
Programmeerproject		1		Opleveren week 10
PERF-gem	0,0			**Gewogen gemiddelde van PERF-cijfers**
PERF-eind	**0,0**	**2**		**Maximaal 5,0 als één van PERF-cijfers < 6,0**

TOTaalcijfer

TOT-gem	0,0	**Gewogen gemiddelde van SUMM en PERF**
TOT-eind	**0,0**	**Maximaal 5,0 als SUMM < minimum of PERF < 6**

Figuur 2: Toetsschema Module 1.2

van deze 4 docenten) die als contact persoon voor de module fungeert. Deze docenten zijn in principe ook verantwoordelijk voor de activiteiten gerelateerd aan de academische vaardigheden, maar worden in deze activiteiten eventueel bijgestaan door een onderwijskundige.

Deze docenten bereiden alle activiteiten in de module voor (hoorcolleges, colstructies, practica, projecten en toetsen). Ze geven de hoorcolleges en colstructies, en zorgen voor aansprekende sprekers voor de motiverende hoorcolleges. Practica en projecten worden zo veel mogelijk door studenten-assistenten en/of promovendi begeleid. De docenten kijken de toetsen na, mogelijk geholpen door studenten-assistenten en/of promovendi.

2.6.2 Zalen

Deze module vereist twee type zalen:

1. Collegezalen voor rond 100 studenten voor de hoorcolleges en toetsen.
2. Zalen voor colstructie, practica en projecten waar studenten in groepen van 24 kunnen werken. De voorkeur gaat uit naar een multifunctionele zaal waarin het makkelijk en snel omgeschakeld kan worden tussen een opstelling met groepen van 24 studenten naar een klassikale opstelling met de hele groep van 100 studenten.

Door de laptop-plicht in beide opleidingen (INF en BIT) zijn practicumzalen uitgerust met computers niet meer nodig. Aan de andere kant moeten colstructie-, practica- en projectzalen van genoeg stopcontacten en wificapaciteit worden voorzien.

2.7 Toetsing

De toetsing verloopt volgens het bij INF opleidingsbreed afgesproken stramien, waarbij het tentamencijfer een gewogen gemiddelde is van twee soorten deelcijfers:

Summatieve cijfers (SUMM). Deze worden (in de module Softwaresystemen) verkregen op basis van schriftelijk afgenomen theoretische toetsen. Hierin wordt vooral kennis en begrip van deelonderwerpen getest. Elk individueel SUMM-cijfer moet tenminste 4,5 bedragen; het gemiddelde van de SUMM-cijfers moet bovendien tenminste 6,0 bedragen (beide vóór afronding). Er kan dus tot op zekere hoogte tussen de SUMM-cijfers gecompenseerd worden.

Performancecijfers (PERF). Deze worden (in Softwaresystemen) verkregen op basis van praktische toetsen, namelijk de twee projecten. Bij PERF-cijfers wordt uitgegaan van het principe dat er een duidelijk gedefinieerd minimum-niveau bestaat dat, wanneer het behaald wordt, een 6,0 garandeert; hogere cijfers zijn het gevolg van extra prestaties bovenop dit minimum. Een onvoldoende kan niet gecompenseerd worden, maar vereist een aanvulling.

2.7.1 Deeltoetsen en weging

Het eindcijfer zal bepaald worden als een ongewogen gemiddelde van de volgende vijf cijfers, met de hierboven genoemde randvoorwaarden voor SUMM- en PERF-cijfers (zie ook Figuur 2):

- Wiskunde (individueel; SUMM)
- Project O-lijn (in G4; PERF)
- Schriftelijke toets O-lijn (individueel; SUMM). Hierbij komen wellicht ook wat onderwerpen van de P-lijn aan de orde.
- Project P-lijn (in G2; PERF)
- Schriftelijke toets P-lijn (individueel; SUMM). Hierbij komt ook het theoretische deel van de O-lijn in week 5–8 aan de orde.

Appendix B laat de koppeling zien tussen leerdoelen van de module en deze deeltoetsen. Hiernaast wordt een combinatie van kleine diagnostische toetsen en peer feedback nagestreefd; zie §2.2.

2.7.2 Herkansingen

De herkansingen vinden plaats in week 10 voor de W-lijn, maar ook voor de O- en de P-lijn. Een student mag de toets O-lijn *óf* de toets P-lijn herkansen, niet allebei. Aanvullingen op het O-project en het P-project kunnen tot en met het eind van week 10 ingeleverd worden.

3 Weekindeling

3.1 Week 1

Uur	Ma	Di	Wo	Do	Vr
1	**hc D∞**	**O col D∞**	**O ipr S24**	W zs N	**P hc D∞**
2	**A col D∞**	**O col D∞**	**O ipr S24**	W zs N	**P ipr S24**
3	**W hc D∞**	O zs N	**P hc D∞**	**W wc D24**	**P ipr S24**
4	**W hc D∞**	O zs N	**P ipr S24**	**W wc D24**	**P ipr S24**
6	**O hc D∞**	**W zs S24**	**O pj4 S24**	**O col D∞**	**P qa D∞**
7	**O ipr S24**	**W zs S24**	**O pj4 S24**	**O col D∞**	**P ipr S24**
8	O zs N	**W wc D24**	O pj4 N	O zs N	**P ipr S24**
9	O zs N	**W wc D24**	O pj4 N	O zs N	**P ipr S24**

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

Het eerste college op maandagochtend is bestemd voor het uitleggen van de structuur en de werkwijze in de module.

3.1.1 Academische Vaardigheden

Voor de colstructie op Ma 2 is het volgende gepland:

- Toelichting leerdoelen en werkvormen
- Informatie: Introductie belang en basisbeginselen timemanagement
- Invullen oefening tijdbudget
- Instructie opdracht week 2: tijdschrijven (weekschema planning / uitvoering / resultaat)

3.1.2 Ontwerpen: Informatiesysteem-ontwerp en -eisen, activiteiten- en use case diagrammen

O-0: Introductie – Uitdagingen voor informatiesysteemontwerp *Doel*: Kennis van (delen van) [Bennett]: Wat zijn informatiesystemen? Waarom is het moeilijk om ze te ontwerpen? Hoe kan je die moelijkheden met succes het hoofd bieden? Wat is object-oriëntatie?

Ma 6 Hoorcollege met motiverend voorbeeld
Ma 7 Installatie en uitproberen van Eclipse en Visual Paradigm
Ma 6–8 Zelfstudie: aan te wijzen delen uit [Bennett, H1–4] lezen en vragen daarover beantwoorden.

O-1: Activiteitendiagrammen *Doel*: Activiteiten kunnen achterhalen uit een tekstuele beschrijving en kunnen vastleggen in een activiteitendiagram,

Di 1–2 Colstructie
Di 3–4 Zelfstudie: [Bennett, H5] plus extra materiaal (uitgebreider syntax voor aktiviteitendiagrammen, o.a. fork en rejoin)
Wo 1–2 Instructiepracticum: maak een aktviteitendiagram aan de hand van een casusbeschrijving

Project Begin van het project dat uiteindelijk in week 4 wordt afgerond. Opgave voor vandaag: De casus bestuderen en activiteitendiagrammen opstellen voor een paar essentiële processen. Het opstellen van activiteitendiagrammen dient hier vooral om er achter te komen of uit de beschrijving precies duidelijk is hoe de processen in elkaar zitten.

O-2 Requirements capture en use cases

Doel: Requirements kunnen achterhalen uit teksten en d.m.v. interviews / Requirements kunnen omzetten in een use case diagram / Een Requirements list op kunnen stellen.

Do 6–7 Colstructie
Do 8–9 Zelfstudie: [Bennett, H6,A2]. Afspraak maken voor een interview, waarvan de resultaten verwerkt moeten worden in het instructiepracticum
Wk 2, ma 6–7 Instructiepracticum

Merk op: interviews kunnen worden afgenomen met de projectgroep (G4). Dat betekent dat er genoeg personen beschikbaar en gebriefd moeten zijn om 25 interviews te kunnen geven.

Materiaal [Bennett, H.1–6]

3.1.3 Programmeren: Klassen + objecten, variabelen in al hun verschijningsvormen

Motiverend hoorcollege

- Verschil compilatie/bytecode/interpretatie; verschil met Python in M1.1
- Hoe worden klassendiagrammen (behandeld in de O-lijn) vertaald naar Java-klassen?
- Verschil ER-diagrammen en klassendiagrammen uitleggen (voor INF studenten die hebben ER-diagrammen in Module 1.1 al gezien)
- Verschil queries/commands

Instruerend practicum

- Hello World-opgave, gebruikmakend van plain texteditor + command line-aanroepen
- Opgaven uit boek (zie huidig P1)
- Modulo-berekening + three-way-lamp (zie huidig P1)
- Implementatie `Gast` uit hotelvoorbeeld

Toelichting weekindeling

Wo 3+4 Eerste mhc, gevolgd door Hello World
Vr 1–4 Nog een mhc, daarna werken aan ipr
Vr 6–9 Vragenuur voor de hele jaargang, daarna verder met ipr; afgetekend aan het eind van de dag

Materiaal [Niño & Hosch, H.0–4]. Nota bene: H4 overlapt met de stof uit M1.1

3.2 Week 2

Uur	Ma			Di			Wo			Do			Vr		
1	**W**	**diag**	**D∞**	**O**	**hc**	**D∞**	**O**	**ipr**	**S24**	W	zs	N	**P**	**hc**	**D∞**
2	**W**	**diag**	**D∞**	**O**	**hc**	**D∞**	**O**	**ipr**	**S24**	W	zs	N	**P**	**ipr**	**S24**
3	**W**	**hc**	**D∞**	O	zs	N	**P**	**hc**	**D∞**	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
4	**W**	**hc**	**D∞**	O	zs	N	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
6	**O**	**ipr**	**S24**	**W**	**zs**	**S24**	**O**	**pj4**	**S24**	**O**	**hc**	**D∞**	**P**	**diag**	**D∞**
7	**O**	**ipr**	**S24**	**W**	**zs**	**S24**	**O**	**pj4**	**S24**	**O**	**hc**	**D∞**	**P**	**ipr**	**S24**
8	**A**	**col**	**D∞**	**W**	**wc**	**D24**	O	pj4	N	O	zs	N	**P**	**ipr**	**S24**
9	**A**	**col**	**D∞**	**W**	**wc**	**D24**	O	pj4	N	O	zs	N	**P**	**ipr**	**S24**

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

3.2.1 Academische Vaardigheden

Gedurende deze hele week moeten studenten hun tijdsbesteding bijhouden.
Invulling colstructie Di 6–7:

- Informatie: Persoonlijkheidskenmerken: bv Big Five / MBTI / kernkwaliteiten (§B3, B4)
- Invullen self assesment bv Big Five
- Informatie: invloed persoonlijkheidskenmerken op :
  - leerstijl
  - selfmanagement
  - teamwork
- Informatie: Timemanagement en effectief gedrag, Eisenhowermatrix (belangrijk /urgent)(§B8)
- Informatie: Uitstelgedrag (“the procrastination equation”)(§B9)
- Self assessment uitstelgedrag

Matriaal: [Skill Sheets, §B3,4,8,9]

3.2.2 Ontwerpen: Requirements analyse, klassendiagrammen

Ma 6–7 Instructiepracticum van onderwerp O-2.

O-3 Requirements analyse (1): Analysis class diagram
Doel: Collaboration diagrams en eenvoudige klassendiagrammen (zonder generalisatie) afleiden uit use case diagrams

Di 1–2 Colstructie
Di 3–4 Zelfstudie: [Bennett, H7.1–7.5, 7.7, A3]
Wo 1–2 Instructiepracticum: maak bij een gegeven use case diagram de bijbehorende collaboration diagrams en een analysis class diagram

Project Studenten werken verder aan het project. Wat ze precies doen is niet voorgeschreven, maar deze middag is het mogelijk (op afspraak) om een interview te houden met een deskundige van het project.

O-4 Requirements analyse (2): Volledig class diagram
Doel: Volledige klassendiagrammen op kunnen stellen

Do 6–7 Colstructie
Do 8–9 Zelfstudie: Bennett H8
Ma wk 3, 6–7 Instructiepracticum: Stel een uitgebreid klassendiagram op aan de hand van een specificatie

Materiaal [Bennett, H7, 8, A3]

3.2.3 Programmeren: Specificaties (JML), testen (testplan, testraamwerk)

Motiverend hoorcollege

- Programming by Contract, gebruikmakend van JML-syntax
- Testen: V-module, unit-testen, regressie, automatisering, test suite, test coverage

Instruerend practicum

- Specificatie three-way-lamp, `Rectangle` in compileerbaar JML; run-time assertion checking
- Three-way-lamp ook mbv Java `enums`
- Testen: testplan maken voor `Gast`, `Kluisje` & tests schrijven en uitvoeren; coverage vaststellen
- TUI voor hotelsysteem

Diagnostische toets Meerkeuze over specificatie en testen.

Toelichting weekindeling

Wo 3+4 mhc over specificatie, gevolgd door eerste opgaven
Vr 1–4 mhc over testen, gevolgd door ipr
Vr 6–9 Disgnostische toets, eventueel gevolgd door vragen; daarna ipr afmaken en aftekenen

Materiaal [Niño & Hosch, H5–8]

3.3 Week 3

Uur	Ma	Di	Wo	Do	Vr
1	**O diag D∞**	**O col D∞**	**O ipr S24**	W zs N	**P hc D∞**
2	**W diag D∞**	**O col D∞**	**O ipr S24**	W zs N	**P ipr S24**
3	**W hc D∞**	O zs N	**P hc D∞**	**W wc D24**	**P ipr S24**
4	**W hc D∞**	O zs N	**P ipr S24**	**W wc D24**	**P ipr S24**
6	**O ipr S24**	**W zs S24**	**O pj4 S24**	**O col D∞**	W zs N
7	**O ipr S24**	**W zs S24**	**O pj4 S24**	**O col D∞**	W zs N
8	**A pfb D24**	**W wc D24**	O pj4 N	O zs N	**W tts D∞**
9	**A pfb D24**	**W wc D24**	O pj4 N	O zs N	**W tts D∞**

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

3.3.1 Academische Vaardigheden

Invulling van de peer feedbacksessie (Di 6–7, groepen van 4 personen):

- reflectie op tijdschrijven
- reflectie op tijdbudget (verschil theorie – praktijk –gewenste situatie)
- aandachtspunten voor opstellen planning projectwerk

3.3.2 Ontwerpen: Objectinteracties; collaboration, sequence en toestandsdiagrammen

Ma 6–7 Instructiepracticum van onderwerp O-4.

O-5 Requirements analyse (2): Objectinteracties

Doel: Interacties tussen objecten kunnen herkennen en weergeven in daarvoor geschikte diagrammen. Collaboration diagrams kwamen al aan de orde in O-3, daar komen nu sequence diagrams bij.

Di 1–2 Colstructie
Di 3–4 Zelfstudie: [Bennett, H9]
Wo 1–2 Instructiepracticum: maak bij een gegeven casus bijbehorende collaboration diagrams en sequence diagrams

Project Studenten werken verder aan het ontwerpproject.

O-6 Toestandsdiagrammen

Doel: Toestanden kunnen herkennen en benoemen, een toestandsdiagram kunnen opstellen.

Do 6–7 Colstructie
Do 8–9 Zelfstudie: [Bennett, H11, A4]
Ma wk 4, 6–7 Instructiepracticum: Maak een verfijnd ontwerp zoals in hoofdstuk A4, inclusief toestandsdiagrammen

do 6/7 hoorcollege

Materiaal [Bennett, H9, 11, A4]

3.3.3 Programmeren: Abstractie en overerving

Motiverend hoorcollege

- Abstractie en overerving; link met O-lijn (onder de noemer “generalisatie” ook in deze week behandeld)
- Overriding van methoden; effecten voor specificatie, i.h.b. contracten
- Sequnce-diagrammen, verband met O-lijn (deze week behandeld)

Instruerend practicum

- Wachtwoorden & Controlleurs (zie huidige P1)
- Hotel met verschillende typen kamers

Toelichting weekindeling

Wo 3+4 mhc over abstractie en overerving; eerste opgaven
Vr 1–4 mhc over overriding, aan de slag met ipr. Merk op dat er maar een halve dag beschikbaar is; dat is niet veel tijd!

Materiaal: [Niño & Hosch, H9–11]

3.4 Week 4

Uur	Ma			Di			Wo			Do			Vr		
1	O	pj4	N	O	pj4	N	O	pj4	N	O	zs	N	**P**	**hc**	**D∞**
2	O	pj4	N	O	pj4	N	O	pj4	N	O	zs	N	**P**	**ipr**	**S24**
3	O	pj4	N	**O**	**pj4**	**S24**	O	pj4	N	O	zs	N	**P**	**ipr**	**S24**
4	O	pj4	N	**O**	**pj4**	**S24**	O	pj4	N	O	zs	N	**P**	**ipr**	**S24**
6	**O**	**ipr**	**S24**	**W**	**hc**	**D12**	**O**	**pj4**	**S24**	**O**	**tts**	**D∞**	**P**	**ipr**	**S24**
7	**O**	**ipr**	**S24**	**W**	**hc**	**D12**	**O**	**pj4**	**S24**	**O**	**tts**	**D∞**	**P**	**ipr**	**S24**
8	O	pj4	N	**W**	**hc**	**D12**	O	pj4	N	**O**	**tts**	**D∞**	**P**	**diag**	**D∞**
9	O	pj4	N	**W**	**pfb**	**D12**	O	pj4	N	**O**	**tts**	**D∞**	**P**	**diag**	**D∞**

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)
Vet gedrukt = contactuur	

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

3.4.1 Wiskunde: Leibniz en Newton in de Informatica

De wiskundetijd op dinsdag 6–9 is bedoeld om de link tussen het wiskundeonderwijs van module 2 (en evt. ook module 1) met informatica te maken.

Inhoudelijke overwegingen

- In het programmeerpracticum komt een opdracht om zelf een aantal wiskundige functies te implementeren: het gebruik van informatica voor de wiskunde.
- In dit dagdeel willen we laten zien dat de wiskunde uit deze modules ook op veel plekken binnen de informatica gebruikt wordt.
- Om de details te begrijpen, hebben de studenten nog niet voldoende kennis (dit zal vaak pas op Masterniveau in de opleiding voorkomen), maar deze voorbeelden kunnen wel als motivatie dienen.

Opzet

- Tijdens de uren 6 - 7 vinden er ‘speed dates’ plaats. De studenten praten in groepjes van 8-10 studenten met iemand die vertelt hoe de wiskunde van Newton en Leibniz in zijn onderzoek gebruikt wordt. Elk groepje heeft 3 gesprekken van 30 minuten (13:45 - 14:15, 14:20 - 14:50, 14:55 - 15:25).
- Mogelijke deelnemers vanuit de opleiding:
  - Paul Havinga/PS: bereik van wireless sensoren
  - CAES, hardware analyse
  - Mariëlle Stoelinga (FMT), probabilistische analyse
  - Bodo Manthey (DMPP), geavanceerde complexiteitsanalyse
  - DACS, netwerkanalyse, performance analyse
  - logistics, MB
  - HMI, graphics
  - Jan Broenink, control theory
- Tijdens het 8e uur wordt dor studenten in paren (G2) een presentatie van 5 minuten in elkaar gezet over de volgende vragen:
  - Welke toepassingen was je je van te voren van bewust?
  - Over welke toepassingen heb je vandaag geleerd?
  - Wat sprak je het meeste aan?
  - Wat verandert dit aan je kijk op de W-lijn?
- Tijdens het 9e uur presenteren de G2-groepjes hun bevindingen aan elkaar, in groepen van plm. 12 (zodat er 6 presentaties te verwachten zijn)

3.4.2 Ontwerpen: Toets

Ma 6–7 Instructiepracticum van onderwerp O-6

Project In het project wordt een ontwerpdocument opgeleverd voor FoodCo Ltd, analoog aan de hoofdstukken A2, A3 en A4 van [Bennett]. Het document bestaat uit een aantal diagrammen en een tekst waarin de samenhang van de diagrammen en, waar nodig, ontwerpkeuzes worden toegelicht. De precieze eindtermen worden nog nader omschreven als de opdracht wordt uitgewerkt.

Voor het project zijn nog 4,5 dagdelen beschikbaar, het moet uiterlijk woensdagmiddag 18:00 worden ingeleverd.

Ma 1–2 Individuele planning maken voor de rest van het project, in overleg met groep (G4)
Ma 5–6 Tijdens instructiepracticum terugkoppeling op ingeleverde planning
Ma 1–4, 6–7, Di 1–4, Wo 1–4, 6–9 Verder werken aan project.
Di 3–4, Wo 5–6 Begeleiders zijn aanwezig voor vragen en feedback

Toets (individuele schriftelijke toets)

De toets heeft de vorm van een schriftelijk tentamen. Bij een gegeven casus dienen (individueel) ontwerpdiagrammen gemaakt te worden.

Do 1–4 Zelfstudie, voorbereiding op de toets
Do 6–9 Toets

3.4.3 Programmeren: Interfaces en abstracte klassen; arrays

Deze week is er alleen op vrijdag programmeren

Inhoudelijke overwegingen

- Arrays hebben ze gezien in M1.1; het gaat hier om verversing
- Dit hoeft alleen in ipr
- Link met W-lijn: definieer Functie interface, met verschillende implementaties, o.a. polynomen. Definieer operaties zoals integreren en afleiden op symbolische wijze, o.a. implementatie van kettingregel met gebruik van late binding. Zelfde toepassing kan in week 5 ook gebruikt worden om verschillende representaties te illustreren.

Motiverend hoorcollege

- Waarom interfaces? (Geen multiple inheritance)
- Interfaces vs. abstracte klassen: wat wanneer?

Diagnostische toets Pubquiz overerving

Toelichting weekindeling

Vr 1–4 mhc interfaces; daarna ipr uitloop overerving & interfaces
Vr 6–7 Aftekenen ipr
Vr 8–9 Pubquiz-diagnostische toets over overerving, naadloos overgaand in borrel

Materiaal [Niño & Hosch, H9–11] (zie vorige week)

3.5 Week 5

Uur	Ma			Di			Wo			Do			Vr		
1	W	zs	N	**P**	**hc**	**D∞**	**O**	**col**	**D∞**	W	zs	N	**P**	**qa**	**D∞**
2	W	zs	N	**P**	**ipr**	**S24**	**O**	**col**	**D∞**	W	zs	N	**P**	**ipr**	**S24**
3	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
4	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
6	**P**	**hc**	**D∞**	**W**	**zs**	**S24**	P	zs	N	**P**	**pj24**	**D24**	**P**	**pfb**	**S24**
7	**P**	**ipr**	**S24**	**W**	**zs**	**S24**	P	zs	N	**P**	**pj24**	**D24**	P	ipr	N
8	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	**A**	**pfb**	**D24**	P	ipr	N
9	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	**A**	**pfb**	**D24**	P	ipr	N

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. x

3.5.1 Academische Vaardigheden

Invulling van de peer feedbacksessie (Do 8–9, groepen van 4 personen):

- Reflectie op planning – uitvoering tijdens O-project
- Feedback op participatie in groep evt dmv kernkwaliteitenspel
- Opstellen planning eindproject

3.5.2 Ontwerpen: Software-architectuur en -ontwerp

O-7 Software-architectuur / Software-ontwerp

Doel: De koppeling kunnen maken tussen hoogniveau informatiesysteem ontwerp en het ontwerp van de softwaresystemen die het hoogniveau ontwerp implementeert

Wo 1–2 Colstructie

Do 8–9 Zelfstudie: aan de hand van [Bennett, H13]

3.5.3 Programmeren: Lijsten, genericiteit, collections (arrays)

Inhoudelijke overwegingen

- Wiskundige interpretatie van sets vs. lijsten
- Verschillende implementaties van zowel sets als lijsten
- Aandacht voor contract van collection-methoden
- Complexiteit van div. operaties
- Overall klasse- en interface-hiërarchie van Java
- Maps?? Opnieuw de wiskundige interpretatie, misschien relaties vs. functies?
- Arrays zijn ook maps, predicaten zijn ook verzamelingen, sets zijn ook maps
- Recursieve datastructuren

Motiverend college

- Boodschap: om efficiëntie te behalen is zorgvuldige keus van datastructuren nodig. Externe spreker niet nodig, we hebben voldoende ervaring in huis
- Ook mhc over lusinvarianten

Instructiepracticum

- Iets doen met performance-verschillen
- Omgaan met groot bestand, bijvoorbeeld realistische data: probeer en meet verschillende implementatie
- Maps vooral oefenen, niet teveel in college
- Leren manipuleren, aan de andere kant ook iets doen dat nuttig is
- Schrijf zelf generieke klasse
- Lusinvarianten oefenen
- Recursie over linked list, terugverwijzen naar M1.1

Practicum

- Arrays als variatie op lijsten
- Leaderboard-achtige functionaliteit met tekstuele interface (voorbereidend op eindopdracht)
- Testplan opstellen + uitvoeren

Project In het eerste projectblok kunnen klassikaal de eisen voor het spel ontwikkeld worden, onder begeleiding van een docent. Iedereen dient hiervan voor het koppel waarin hij gaat werken aantekeningen te maken. Let wel: het gaat hier nog *niet* om het protocol, dat komt pas in week 7 ter sprake.

Aspecten die in de definitie van eisen aan de orde kunnen komen:

- Distributie over client/server
- Verantwoordelijkheden client/server: wie houdt geldigheid van een zet bij?
- GUI
- Aanmelden: gebruikersnaam, wachtwoord
- Ranking

Toetsing In de eindtoets kan een vraag gesteld worden over welke datastructuur voor een gegeven probleem geschikt is, en waarom

Materiaal [Niño & Hosch, H12–14,19–20] behalve dat we de Java-collections willen gebruiken

Toelichting weekindeling:

Ma 6–9 Motiverend hoorcollege voor de stof van deze week, gevolgd door opdrachten in practicumvorm.
Di 1–4 mhc over luinvarianten; daarna doorwerken aan de opgaven. Opgaven moeten afgetekend worden; bij voorkeur nu, anders later in de week.
Wo 6–9 Eventueel ipr-opgaven afmaken; i.c.m. bestuderen stof uit boek.
Do 6–7 Requirements voor eindopdracht bestuderen en (klassikaal) bespreken.
Vr 1–9 Vragenuur dan wel uitleg practicum; daarna uitvoering practicum. Afgesloten door sesie met demonstreren en bespreken van uitwerkingen. Practicum moet afgetekend worden.

3.6 Week 6

Uur	Ma			Di			Wo			Do			Vr		
1	**W**	**diag**	**D∞**	**P**	**hc**	**D∞**	**O**	**col**	**D∞**	W	zs	N	**P**	**qa**	**D∞**
2	**W**	**diag**	**D∞**	**P**	**ipr**	**S24**	**O**	**col**	**D∞**	W	zs	N	**P**	**ipr**	**S24**
3	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
4	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
6	**P**	**hc**	**D∞**	**W**	**zs**	**S24**	P	zs	N	P	pj2	N	**P**	**pfb**	**S24**
7	**P**	**ipr**	**S24**	**W**	**zs**	**S24**	P	zs	N	P	pj2	N	P	ipr	N
8	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	**A**	**col**	**D∞**	P	ipr	N
9	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	**A**	**col**	**D∞**	P	ipr	N

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

3.6.1 Academische Vaardigheden

Invulling colstructie Do 8–9:

- Informatie: verruimen planningshorizon → week → semester (§B10, §B11)
- Illustratie "Big rocks first"
- Informatie: Ingrediënten voor effectief projectmanagement: "getting things done": volledige takenlijst, "next action" definiëren, eigenaarschap, etc (nader in te vullen op basis van observaties knelpunten ontwerpproject in peerfeedbacksessie)
- Aanwijzingen voor reflectiegedeelte eindverslag

Materiaal: [Skill Sheets, §B10–11]

3.6.2 Ontwerpen: Design patterns

O-8 Design patterns

Doel: Bekende patronen voor het opstellen van software oplossingen kunnen toepassen

Wo 1–2 Colstructie

Do 8–9 Zelfstudie: aan de hand van [Bennett, H15]

3.6.3 Programmeren: Exceptions, I/O, GUI

Motiverend hoorcollege

- Technisch: GUI (MVC) en exceptions in Java (exceptions kort!)
- Externe college (HMI? Vanessa?): gebruikersaspecten

Instruerend practicum

- I/O-opgave: lezen en schrijven van kaarten? Verkiezingsuitslagen? Verschillende formaten, bewustwording voor- en nadelen
- Exceptions: afvangen systeemexcepties (I/O); eigen exceptie-klasse schrijven en gebruiken. Bijvoorbeeld TUI voor stemmachine met excepties voor invoerfouten gebruiker?
- Testen van exceptions
- Stemmachine met GUI, gebruik makend van MVC

Practicum

- GUI voor boter-kaas-eieren
- Bord ontwerpen voor eindproject

Toelichting weekindeling

Ma 6–9 mhc over GUI + exceptions in Java, daarna oefenen in ipr
Di 1–4 mhc over gebruikersaspecten GUI, daarna verder met ipr
Wo 6–9 Evt. ipr afmaken, zs over lusinvarianten
Do 6–7 Ontwerp bord voor eindproject
Vr 1–5 Boter-kaas-eieren en eigen bord eindproject
Vr 6–9 pfb met wederzijds demonstreren eigen bord; daarna evt. aan doorwerken (zonder SA-begeleiding)

Materiaal [Niño & Hosch, H15–18]

3.7 Week 7

Uur	Ma			Di			Wo			Do			Vr		
1	**W**	**diag**	**D∞**	**P**	**hc**	**D∞**	**O**	**col**	**D∞**	W	zs	N	**P**	**qa**	**D∞**
2	**W**	**diag**	**D∞**	**P**	**ipr**	**S24**	**O**	**col**	**D∞**	W	zs	N	**P**	**ipr**	**S24**
3	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
4	**W**	**hc**	**D∞**	**P**	**ipr**	**S24**	O	zs	N	**W**	**wc**	**D24**	**P**	**ipr**	**S24**
6	**P**	**hc**	**D∞**	**W**	**zs**	**S24**	P	zs	N	**P**	**diag**	**D24**	W	zs	N
7	**P**	**ipr**	**S24**	**W**	**zs**	**S24**	P	zs	N	**P**	**pj24**	**D24**	W	zs	N
8	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	**P**	**pj24**	**D24**	**W**	**tts**	**D∞**
9	**P**	**ipr**	**S24**	**W**	**wc**	**D24**	P	zs	N	P	pj2	N	**W**	**tts**	**D∞**

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. *x*

3.7.1 Ontwerpen: Softwaremetrieken

O-9 Softwaremetrieken

Doel: De belangrijkste metrieken voor software complexiteit kunnen benoemen en bestaande software kunnen evalueren aan de hand van deze metrieken

Wo 1–2 Colstructie

Do 8–9 Zelfstudie: aan de hand van extra materiaal (niet gedekt door [Bennett])

3.7.2 Programmeren: Parallel + Netwerkprogrammeren

Motiverend hoorcollege

- Principes van threads: start/join, synchronised, locks (Java 1.5)
- Network programming: weinig over vertellen, alleen refereren aan bekende concepten (URL, IP-adres, poorten), de rest is API. (Evt. extern college?)

Instruerend practicum

- Veel oefenen met threads; opgaven op basis van het huidige P2
- Client-server-architectuur en GUI-opgaven ook uit P2
- Speed-up van parallelle processen?
- Implementatie klein netwerkprotocol in voorbereiding op eindproject

Toelichting weekindeling

Ma 6–9 mhc over threads, daarna oefenen in ipr
Di 1–4 mhc kort over network programming, daarna locks; verder met ipr
Wo 6–9 Evt. ipr afmaken, zs over threads
Do 6 Diagnostische toets parallellisme d.m.v. vragenlijst
Do 6–9 Ontwerp protocol eindproject (klassikaal)
Vr 1–4 Implementatie protocol eindproject

Materiaal: [Core Java 2, H3]

3.8 Week 8

Uur	Ma	Di	Wo	Do	Vr
1	P pj2 N	**P hc D∞**	**O col D∞**	P pj2 N	**P qa D∞**
2	P pj2 N	**P ipr S24**	**O col D∞**	P pj2 N	**P ipr S24**
3	P pj2 N	**P ipr S24**	O zs N	P pj2 N	**P ipr S24**
4	P pj2 N	**P ipr S24**	O zs N	P pj2 N	**P ipr S24**
6	**P hc D∞**	**P ipr S24**	P zs N	**P diag D24**	**P pfb S24**
7	**P ipr S24**	**P ipr S24**	P zs N	**P pj24 D24**	P ipr N
8	**P ipr S24**	**P ipr S24**	P zs N	**P pj24 D24**	P ipr N
9	**P ipr S24**	**P ipr S24**	P zs N	P pj2 N	P ipr N

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)
Vet gedrukt = contactuur	

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. x

3.8.1 Ontwerpen: Constraints (OCL)

O-10 'Constraints'

Doel: Technieken voor het beschrijven van operaties en constraints in UML modellen kunnen benoemen en toepassen

Het gaat hier vooral om OCL (principes en sommige veel gebruikt constructies). We hebben niet de illusies dat de studenten in staat zullen zijn om OCL in detail te leren.

Wo 1–2 Colstructie; hier wordt de relatie met JML besproken

Wo 3–4 Zelfstudie aan de hand van [Bennett, H10]

In het project kan het gebruik van OCL constraints bonuspunten opleveren.

3.8.2 Programmeren: Security

Motiverend hoorcollege

- Authenticatie, Public Key Infrastructure, Digital signatures
- Java security framework en code signing

Instructiepracticum/zelfstudie

- Oefeningen met applets en code signing on de toegang to kritische resources te beperken
- Competitie: groepen proberen om via applets in te breken in de infrastructuur van de andere groepen
- Practicumomgeving moet uitgebreid worden met een web infrastructuur (web server en bijhorende gereedschappen)
- Java tutorial alleen voor zelfstudie

Materiaal: Java-tutorial (`http://docs.oracle.com/javase/tutorial/security`); [Core Java 2, Hoofdstuk 9].

Toelichting weekindeling

Ma 1–4 Project: Planning, eerste ontwerp
Ma 6–9 mhc over access control; ipr over applets & jars
Di 1–4 mhc over public keys & digital dignatures; ipr over signing code and granting permissions
Di 6–9 ipr over generating & verifying signatures

Wo 6–9 Zelfstudie: Java tour `http://docs.oracle.com/javase/tutorial/security`
Do 1–4 Project: Testplan, verder met ontwerp
Do 6 Diagnostische toets security (vorm nog nader te bepalen; evt. vragenlijst)
Do 7–9 Ontwerp (klassikaal) een PKI om bij het project te gebruiken
Vr 1–4 ipr afmaken & aftekenen; PKI implementeren
Vr 6–9 pfb over PKI; daarna onbegeleid verder met ipr/pj

3.9 Week 9

Uur	Ma	Di	Wo	Do	Vr
1	**P tts D∞**	P pj2 N	P pj2 N	P pj2 N	P pj2 N
2	**P tts D∞**	P pj2 N	P pj2 N	P pj2 N	P pj2 N
3	**P tts D∞**	P pj2 N	P pj2 N	**P pj2 D24**	P pj2 N
4	**P tts D∞**	P pj2 N	P pj2 N	**P pj2 D24**	P pj2 N
6	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**
7	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**	**P pj2 S24**
8	P pj2 N	P pj2 N	P pj2 N	P pj2 N	P pj2 N
9	P pj2 N	P pj2 N	P pj2 N	P pj2 N	P pj2 N

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. x

3.9.1 Programmeren: Eindproject

Overwegingen

- Begin van de week programmeertoets.
- Elke dag aan het begin van de middag 2 uur begeleiding door SA's. Hierbij moet ook de planning in de gaten worden gehouden.
- Donderdag 3-4 is er een "impress me" sessie. Hier zijn de (hoofd)docenten en student-assistenten van het vak aanwezig en de studenten moeten laten zien wat ze allemaal al kunnen, en hoever ze zijn met de eindopdracht. Dit is bedoeld om enige druk op de ketel te houden en om contact af te dwingen.
- Vrijdag eind van de middag moet er een versie van het verslag ingeleverd worden voor peer feedback (zie Wk 10, Ma 6–7). Elke groep stuurt het conceptverslag naar een andere groep volgens een gegeven schema, en krijgt feedback van deze andere groep. De peer feedback wordt door de studenten-assistenten in week 10 afgetekend
- Verder zelfstandig werken aan eindproject.
- Structurering eindproject moet nog nader uitgewerkt worden.
- Bonussen kunen verdiend worden met:
  - Protocolafspraken
  - Winnen tournooi
  - Beveiliging van tegenstanders breken en daardoor vals spelen
  - Chatfunctie
  - Challengefunctie

3.10 Week 10

Uur	Ma	Di	Wo	Do	Vr
1	**W tts D∞**	P pj2 N	**P pj2 S24**	P zs N	**O pj4 S24**
2	**W tts D∞**	P pj2 N	**P pj2 S24**	P zs N	**P pj2 S24**
3	**W tts D∞**	P pj2 N	**P pj2 S24**	O zs N	**O pj4 S24**
4	**W tts D∞**	P pj2 N	**P pj2 S24**	O zs N	**P pj2 S24**
6	**P pj2 S24**	**P pj2 S24**	**P tts D24**	**P tts D∞**	O pj4 N
7	**P pj2 S24**	**P pj2 S24**	**P tts D24**	**P tts D∞**	P pj2 N
8	P pj2 N	P pj2 N	**P tts D24**	**O tts D∞**	O pj4 N
9	P pj2 N	P pj2 N	**P tts D24**	**O tts D∞**	P pj2 N

Afk	Werkvorm
col	Colstructie
diag	Diagnostische toets
hc	Hoorcollege
ipr	Instructiepracticum
pfb	Peer feedback
pj	Poject
qa	Vragenuur
tts	Toets
wc	Werkcollege
zs	Zelfstudie

Afk	Leerlijn
A	Ac. Vaardigheden
O	Ontwerpen
P	Programmeren
W	Wiskunde

Afk	Begeleiding
D	Docent
S	Studentassisgent
N	Geen (zelfstandig)

Vet gedrukt = contactuur

Afk	Groepsgrootte
∞	Hele jaargang
x	Groep van plm. x

3.10.1 Academische Vaardigheden

In het eindverslag van het project dient expliciet gereflecteerd te worden op

- Teamwork, refererend aan hetgeen in Module 1.1 aan de orde is geweest (tenminste voor INF-studenten!);
- Planning eindopdracht (in Week 5 opgesteld) en relatie met daadwerkelijke uitvoering.

3.10.2 Programmeren: Eindproject en tournooi

- Op maandag 6 - 7 moet het leveren van peer feedback afgetekend worden (zie week 9).
- Maandag en dinsdag aan het begin van de middag 2 uur begeleiding door SA's. Hierbij moet ook de planning in de gaten worden gehouden.
- Op woensdag, de dag van het tournooi is de hele dag begeleiding aanwezig.
- Vrijdagochtend is er begeleiding aanwezig voor die studenten die aanvullingen op het O- of P-project moeten inleveren.

3.10.3 Toetsen

In deze week zijn een aantal toetsmomenten ingebouwd:

- Op maandagmiddag de W-herkansing;
- Op woensdagmiddag het P-tournooi;
- Op donderdagmiddag herkansingen voor de O- en P-toets. Elke student mag meedoen aan slechts één van deze herkansingen.
- Op vrijdagmiddag ruimte voor het inleveren van aanvullingen op het O- dan wel P-project.

Wie geen herkansingen nodig heeft, heeft donderdag en vrijdag vrij.

Referenties

[Bennett] S. Bennett, S. McRobb, and R. Farmer. *Object Oriented Systems Analysis and Design using UML.* McGraw-Hill, 4th edition, 2010.

[Core Java 1] C. S. Horstmann and G. Cornell. *Core Java*, volume I: Fundamentals. Prentice Hall, 9th edition, 2012.

[Core Java 2] C. S. Horstmann and G. Cornell. *Core Java*, volume 2: Advanced Features. Prentice Hall, 9th edition, 2012.

[Niño & Hosch] J. Nino and F. A. Hosch. *An Introduction to Programming and Object-Oriented Design Using Java*. Wiley, 3rd edition, 2008.

[Skill Sheets] R. Van Tulder. *Skill Sheets*. Pearson, 2012.

A Ontwerpcriteria Module 1.2 (Softwaresystemen)

Versie 1.1, dinsdag 4 september 2012

In deze module zetten studenten hun eerste stappen in het ontwerpen, implementeren en testen van softwaresystemen, en in het zelfstandig uitvoeren van projecten. Voor het ontwerpen leren ze gebruik te maken van modellen uit Software Engineering, in het bijzonder de ontwerpmodellen uit het UML-palet: klassen-, activiteiten- en toestandsdiagrammen, en maken ze kennis met verschillende software-ontwikkelprocessen (waterval, iteratief, agile). Voor het programmeren leren ze kernbegrippen van programmastructuur, algoritmiek (inclusief recursie en multithreading) en objectoriëntatie aan de hand van de programmeertaal Java, met aandacht voor correctheid in de vorm van (informele) pre- en postcondities. Bovendien wordt er aandacht besteed aan de Java-bibliotheken voor security en GUI-programmeren. Voor het testen leren ze te onderscheiden op welke niveaus getest kan worden (bijzonder unit- en systeemtests), de principes van een testplan en een paar eenvoudige testtechnieken. Daarnaast is er aandacht voor elementaire projectvaardigheden (planmatig werken, versiebeheer, projectmanagement).

A.1 Voorkennis

Geen

A.2 Leerdoelen

Na succesvol afronden van deze module kan de student

1. Een specificatie van een bestaand of nieuw te ontwerpen softwaresysteem opstellen met modellen uit de UML (klassen-, activiteiten- en toestandsdiagrammen), gebruik makend van daarvoor geëigend software-gereedschap
2. Zulke modellen interpreteren, uitleggen wat het verband is tussen modellen onderling en tussen modellen en werkende software, en wat het nut is van het opstellen van modellen naast het programmeren van de software
3. De relaties analyseren tussen ontwerpmodellen onderling en tussen ontwerpmodellen aan de ene kant en werkende code aan de andere kant
4. De principes, voordelen en nadelen van verschillende software-ontwikkelprocessen uitleggen
5. Kernbegrippen uit het imperatieve programmeren uitleggen en toepassen, zoals variabelen, datatypen, gestructureerde programmastatements, recursiviteit, lijsten, arrays, methoden en parameters
6. Kernbegrippen uit de objectoriëntatie uitleggen en toepassen, zoals object, klasse, waarde, type, objectreferentie, interface, specialisatie/overerving, compositie
7. Basisconcepten van GUI-programmeren uitleggen en toepassen, met gebruikmaking van het principe van Model/View/Controller
8. Eenvoudige multi-threaded programma's schrijven, en de werking en problemen (race-condities) van threads uitleggen
9. Basisconcepten van security in Java uitleggen en toepassen.
10. Software van enige complexiteit (tot ongeveer tien klassen) in Java schrijven, met toepassing van bovenstaande concepten en algoritmen voor zoeken en sorteren
11. Software van deze omvang documenteren, met gebruikmaking van informele pre- en postcondities en klasseninvarianten, en de correctheid van zelf geschreven software informeel beargumenteren
12. Uitleggen hoe zulke software getest kan worden, en zelf een testplan opstellen en uitvoeren

A.3 Verdere criteria voor het ontwerp

Een belangrijk punt van zorg is het (in het verleden duidelijk geconstateerde) verschil in instapniveau van de studenten. Bij het ontwerp van de module moet hiermee rekening gehouden worden, mogelijk door gevorderde studenten in te schakelen om de minder gevorderden bij te spijkeren, of door opdrachten in verschillende moeilijkheidsgraden aan te bieden.

Voor een aansprekende module is het verder van belang dat er goede, naar moderne maatstaven bruikbare tool-ondersteuning beschikbaar is — bij voorkeur een geïntegreerde toolomgeving die zoveel mogelijk modelleer- en programmeer-aspecten ondersteunt.

A.4 Veronderstellingen over het vervolg van de "leerlijn programmeren"

Zoals uit bovenstaand overzicht blijkt, beschikt de student na het volgen van deze module weliswaar over een gezonde basis aan programmeervaardigheden, maar zeker nog niet voldoende om de studie mee door te komen. Dat kan ook onmogelijk verwacht worden binnen een enkel kwartiel. Daarom is het van essentieel belang dat in de latere module "Data en Informatie" nog een fundamentele hoeveelheid tijd wordt ingeruimd voor verdieping van de hier aangeboden inhoud.

A.5 Vergelijking met het huidige INF-curriculum

De materie van deze module komt min of meer overeen met de inhoud van de huidige vakken

- Programmeren 1 (192135000, blok 1/1A)
- Een deel van Programmeren 2 (192135050, blok 1/2A), i.h.b. recursie, een kennismaking met multi-threading, en GUI-programmeren
- Informatiesystemen (192120100, blok 1/2A)
- Software Engineering Modellen (192135100, blok 2/1B).

Gegeven dat de module zelf een omvang van 12 EC zal kennen terwijl bovengenoemde vakken in totaal 15 EC tellen maar ook een duidelijke overlap vertonen (zie huidige vakomschrijving op Osiris!) lijkt dit een goed haalbare zaak.

A.6 Vergelijking met het ACM-curriculum

In termen van het ACM/IEEE Computer Science Curriculum (`http://ai.stanford.edu/users/sahami/CS2013`) overlapt de materie van deze module met de inhoud van de volgende kernonderwerpen ("core subjects"):

- HC/Programming Interactive Systems (elective)
- PD/Parallelism Fundamentals (2 uur)
- PL/Object Oriented Programming (grootste deel van 10 uur)
- SDF/Algorithms and Design (deel van 11 uur)
- SDF/Fundamental Programming Concepts (10 uur)
- SDF/Fundamental Data Structures (deel van 12 uur)
- SDF/ObjectOriented (8 uur)
- SDF/Development Methods (9 uur)
- SE/Software Processes (deel van 3 uur)
- SE/Software Project Management (3 uur)
- SE/Tools and Environments (3 uur)
- SE/Software Design (deel van 8 uur)
- SE/Software Construction (2 uur)
- SE/Software Verification Validation (deel van 3 uur)

De genoemde uren zijn contacturen volgens inschatting van bovengenoemd document; "deel van" betekent dat de inhoud van het kernonderwerp maar deels overlapt met de voorgestelde module. Een hele grove benadering levert een totaal van plm. 50 contactuur. Hierbij kan worden vermeld dat het hele ACM/IEEE-kerncurriculum 280 uur beslaat, waarschijnlijk de beste analogie met onze 6 basismodules. Aangezien $6 \times 50 = 300$ lijkt dit in ieder geval in dezelfde ordegrootte.

B Toetsschema

De module kent de volgende deeltoetsen (zie ook §2.7):

O-toets Ontwerptoets: schriftelijke toets over kennis binnen de O-lijn
O-project Ontwerpproject: opdracht betreffende vaardigheden binnen de O-lijn
P-toets Programmeertoets: schriftelijke toets over kennis binnen de P-lijn
P-project Programmeerproject: opdracht betreffende vaardigheden binnen de P-lijn
W-toets Wiskundetoets: schriftelijke toets over de W-lijn

Figuur 2 (Pag. 11) laat de weging van de deeltoetsen zien. De volgende tabel geeft de koppeling tussen leerdoelen en deeltoetsen:

	Leerdoel	**Beoordeling**
1	Een specificatie van een bestaand of nieuw te ontwerpen softwaresysteem opstellen met modellen uit de UML (klassen-, activiteiten- en toestandsdiagrammen), gebruik makend van daarvoor geëigend software-gereedschap	O-project O-toets
2	Zulke modellen interpreteren, uitleggen wat het verband is tussen modellen onderling en tussen modellen en werkende software, en wat het nut is van het opstellen van modellen naast het programmeren van de software	O-project O-toets
3	De relaties analyseren tussen ontwerpmodellen onderling en tussen ontwerpmodellen aan de ene kant en werkende code aan de andere kant	O-toets P-project
4	De principes, voordelen en nadelen van verschillende software-ontwikkelprocessen uitleggen	O-toets P-project
5	Kernbegrippen uit het imperatieve programmeren uitleggen en toepassen, zoals variabelen, datatypen, gestructureerde programmastatements, recursiviteit, lijsten, arrays, methoden en parameters	P-project P-toets
6	Kernbegrippen uit de objectoriëntatie uitleggen en toepassen, zoals object, klasse, waarde, type, objectreferentie, interface, specialisatie/overerving, compositie	P-project P-toets
7	Basisconcepten van GUI-programmeren uitleggen en toepassen, met gebruikmaking van het principe van Model/View/Controller	P-project
8	Eenvoudige multi-threaded programma's schrijven, en de werking en problemen (race-condities) van threads uitleggen	P-project
9	Basisconcepten van security in Java uitleggen en toepassen.	P-project P-toets
10	Software van enige complexiteit (tot ongeveer tien klassen) in Java schrijven, met toepassing van bovenstaande concepten en algoritmen voor zoeken en sorteren	P-project
11	Software van deze omvang documenteren, met gebruikmaking van informele pre- en postcondities en klasseninvarianten, en de correctheid van zelf geschreven software informeel beargumenteren	P-project P-toets
12	Uitleggen hoe zulke software getest kan worden, en zelf een testplan opstellen en uitvoeren	P-project
	Leerdoelen wiskinde	*W-toets*