

# 3.1 Toetsen en beoordelen van activerende colleges

Koen Langendoen

## Samenvatting

Mijn competenties op het gebied van het toetsen en beoordelen van activerende colleges zijn gebaseerd op de BKO module 4 “Beoordeling en toetsing van leerresultaten” die ik gevolgd heb in 2003, en de twee colleges Compilerbouw die ik gedoceerd heb in 2002 en 2003. De cijfers voor dit vak worden vastgesteld aan de hand van de practicum resultaten en een afsluitend tentamen. In 2002 was dit een mondeling tentamen, in 2003 een schriftelijk (juist/onjuist vragen). De reden om over te stappen was het aantal deelnemende studenten: 23 in 2002 en 79 in 2003. Deze toename van studenten had ook consequenties voor de nakijkprocedure van het praktisch werk: de eerste keer deed ik het zelf, de tweede keer heb ik 2 student assistenten en 1 AIO ingeschakeld (en aangestuurd).

## 1 Afstemming

**Competentie:** De docent is in staat te toetsing af te stemmen op facultaire regelingen.

### 1.1 bewijsstukken

Het Compilerbouw (in4020<sup>1</sup>) vak maakt verplicht onderdeel uit van zowel de Masters Computer Science<sup>2</sup> als de Masters Computer Engineering<sup>3</sup>. Het vak is ingeschaald op een studielast van 4 studiepunten (160 uur). Verder worden er geen specifieke eisen gesteld aan de vorm en inhoud van het college. Wel zijn er doelen en eindtermen<sup>4</sup> van de complete Masters opleidingen geformuleerd, maar die blinken uit in vaagheid.

Omdat ik veel belang hecht aan het motto ‘al doende leert men’ heb ik gekozen voor de formule met een uitgebreid practicum (80 uur), waarin de studenten enkele deelaspecten in alle detail uitwerken, naast het meer globale college en bijbehorend mondeling/schriftelijk tentamen (samen 80 uur). Het eindcijfer wordt bepaald als het gewogen gemiddelde van het practicumcijfer (70%) en het tentamenresultaat (30%). De nadruk op het praktisch gedeelte wijkt enigszins af van wat gebruikelijk is; de meeste docenten die een practicum aanbieden verlangen slechts dat er voldoende’ gescoord wordt voor het (kleine) practicum en nemen het tentamencijfer als eindcijfer.

### 1.2 validatie

Uit de enquête resultaten en de (persoonlijke) gesprekken met diverse studenten (Appendix A, Validatie bronnen<sup>5</sup>) is gebleken dat er geen fundamentele kritiek is op de formule

<sup>1</sup>[http://blackboard.icto.tudelft.nl/bin/common/course.pl?course\\_id=\\_6011\\_1&frame=top](http://blackboard.icto.tudelft.nl/bin/common/course.pl?course_id=_6011_1&frame=top)

<sup>2</sup>[http://academics.its.tudelft.nl/nl/info2002/Master\\_TI.pdf](http://academics.its.tudelft.nl/nl/info2002/Master_TI.pdf)

<sup>3</sup>[http://academics.its.tudelft.nl/nl/info2002/Master\\_Et.pdf](http://academics.its.tudelft.nl/nl/info2002/Master_Et.pdf)

<sup>4</sup><http://www.pds.twi.tudelft.nl/~epema/OCI/curr97.ps>

<sup>5</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/1.1/Ontwerpen-van-activerende-colleges.pdf>

(zwaar practicum, licht tentamen), maar dat de studenten alleen moeite hadden met de grootte van de practicumopdrachten. Het aantal bestede uren overstijgt ruim de norm: gemiddeld 127 ipv. 80 uur. Vreemd genoeg was er geen verschil tussen informatica en elektro studenten terwijl de laatsten toch duidelijk minder goed beslagen ten ijs kwamen.

### 1.3 zelfreflectie

Het practicum wordt door veel studenten als te zwaar ervaren. Navraag leert dat dat voor een groot gedeelte komt omdat studenten niet gewend zijn om met grote (onbekende) stukken software om te gaan, en relatief weinig ervaring met de Unix tools (C compiler, debugger) hebben die gebruikt worden. Wat betreft de informatica studenten zullen deze lacunes in de toekomst verholpen worden omdat het programmeeronderwijs in de Bachelorsfase aangepast wordt. Wat betreft de elektro studenten is er een extra vak (1 studiepunten) C programmeren aan het curriculum toegevoegd, maar omdat dit niet genoeg diepgang biedt, is er voor gekozen om de practicumopdrachten aan te passen (verlichting + meer nadruk op code generatie).

Tenslotte nog een meta reflectie. Binnen onze opleiding heeft een docent totale keuzevrijheid wat betreft de vorm en inhoud van “zijn” vakken. Dit model heeft als voordeel dat de individuele docent de zijns inziens optimale lesvorm kan kiezen, en veranderen, zonder tijdrovende bureaucratische overleggrondes. Het nadeel is echter dat iedere docent zijn eigen zaakjes regelt en er weinig of geen afstemming tussen de vakken onderling is. In mijn optiek zou enige sturing van “bovenaf” geen kwaad kunnen. Dat had de inpassing van het vernieuwde compilerbouwwak vast soepeler laten verlopen; nu ontbraken de eindtermen van het programmeeronderwijs van de Bachelorsfase, waardoor ik een te hoog niveau ingeschat had.

## 2 Leerdoelen

**Competentie:** De docent is in staat toetsen te ontwerpen die aansluiten bij de leerstof en het gewenste leerresultaat en kan daarvoor geschikte toetsvormen hanteren.

### 2.1 bewijsstukken

De toetsing van het compilerbouwwak bestaat uit twee delen: practicum + tentamen. Voor het practicum<sup>6</sup> is gekozen om twee fundamentele zaken uit te werken:

1. het verschil tussen top-down en bottom-up parsing, en
2. de implementatie van moderne object-oriented taalconstructies.

De studenten moeten in groepjes van twee een eenvoudige reference compiler aanpassen. Dit brengt de studenten ook in aanraking met standaard compilerbouw-tools zoals *lex* en *yacc*.

Voor het afsluitend tentamen had ik in eerste instantie gekozen voor een mondeling, maar door de enorme toeloop aan studenten (van 23 naar 79) heb ik de laatste keer een schriftelijk tentamen afgenomen. Omdat het practicum de diepte in gaat moet het schriftelijk tentamen de breedte afdekken. Daarom heb ik gekozen voor een juist/onjuist tentamen (60 vragen) want hiermee kun je alle facetten van het vak testen itt. een tentamen met open vragen waar slechts 5 of 6 onderwerpen aan de orde kunnen komen.

Om te zorgen dat de vragen goed de behandelde stof afdekken heb ik een toetsmatrijs (zie Tabel 1) opgesteld aan de hand van het college rooster<sup>7</sup>: elk college uur (28 in totaal) was goed voor circa 2 vragen.

<sup>6</sup><http://www.pds.twi.tudelft.nl/~koen/compilerbouw/2003/practicum.html>

<sup>7</sup><http://www.pds.twi.tudelft.nl/~koen/compilerbouw/2003/rooster.html>

Subject	Chapter	#questions
intro	1	2
lexical	2.1	5
parsing	2.2	9
attribute-grammars	3.1	4
manual-methods	3.2	4
interpretation	4.1	2
code-generation	4.2	8
assemblers-linkers	4.3	2
memory-management	5.1	2
garbage-collection	5.2	3
type-checking	6.1.2	3
object-types	6.2.9	3
routines	6.3 + 6.4.2	3
imperative-code-gen	6.4.1	2
functional	7	4
logic-programming	8	4
totaal	1 - 8	60

Tabel 1: Toetsmatrijs Compilerbouw.

## 2.2 validatie

De officiële evaluatie van het vak (samen met alle andere Masters vakken) heeft dit jaar (2003) plaatsgevonden voor het eind van het college zodat de studenten nog niet konden oordelen over de 2e practicum opgave en het schriftelijk tentamen. Uit mijn persoonlijke enquête<sup>8</sup> van vorig jaar (2002) blijkt dat studenten het practicum over het algemeen goed waarderen, zo vinden de meesten de opdrachten erg uitdagend.

## 2.3 zelfreflectie

Mijn ervaringen met twee jaar compilerbouw doceren hebben me gestrekt in de overtuiging dat de combinatie van practicum (in de diepte) en mondeling/schriftelijk tentamen (in de breedte) de studenten een goede (praktische) kennis laten maken met het compilerbouw vak. Echter, zowel het practicum als het afsluitend schriftelijk tentamen zullen nog verfijnd moeten worden. Het practicum moet beter afgestemd worden op enerzijds de elektrotechniek studenten en anderzijds de informatica student zonder affiniteit voor techniek. De moeilijkheidsgraad van het schriftelijk tentamen moet beter aansluiten bij het niveau van de studenten (nu scoorde slechts 42% een voldoende voor het 1<sup>e</sup> schriftelijk tentamen).

## 3 Formulering

**Competentie:** De docent is in staat representatieve, ondubbelzinnige toetsvragen te maken.

### 3.1 bewijsstukken

Met behulp van de toetsmatrijs (Tabel 1), een zelfgemaakte item-bank van ca. 170 vragen en een random true/false generator heb ik een 60-tal vragen geselecteerd voor het 1<sup>e</sup> tentamen op 27/6/03<sup>9</sup>. Het hertentamen staat gepland voor 29/8/03.

<sup>8</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/in4020/enquete-resultaten.pdf>

<sup>9</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/in4020/2003-06-27.pdf>

Om de kwaliteit van de vragen te beoordelen heb ik de drie assistenten van het Compilerbouw practicum de 60 vragen laten maken. Naar aanleiding van hun (foutieve) antwoorden en commentaar zijn er enkele vragen vervangen en diverse formuleringen aangepast.

### 3.2 validatie

In het totaal hebben 67 studenten meegedaan aan het schriftelijk tentamen. Analyse mbv. het verwerkingspakket Sonate liet zien dat er 17 vragen waren waar de studenten “slecht” gescoord hadden (Quality 1 of 2 op een schaal van 5). Nadere bestudering van deze vragen door mezelf, de assistenten, en studenten (via antwoorden op Blackboard) liet zien dat ik bij 2 vragen zelf het verkeerde antwoord had gegeven, dat er 3 als ‘instinker’ beschouwd moesten worden, en dat er nog 6 te onduidelijk/complex geformuleerd waren. Daarop heb ik besloten bij deze 11 vragen beide antwoorden (juist/onjuist) goed te rekenen. De uiteindelijke betrouwbaarheidsanalyse<sup>10</sup> van Sonate laat zien dat de KR-20 score 0,67 is (was initieel 0,57). Ondanks dat dit onder de door Sonate geadviseerde waarde van 0,80 is, blijken de cijfers<sup>11</sup> erg netjes verdeeld te zijn met een gemiddelde van 5 en 42% voldoende. De cijfers zijn berekend door 30 punten af te trekken (raadkans) en vervolgens te delen door 3.

### 3.3 zelfreflectie

Het is me tegengevallen hoe moeilijk het nog is om goede juist/onjuist vragen te formuleren. Echter, zelfs na correctie van 11 van de 60 vragen waren de resultaten nog erg mager (42% voldoende). Na ruggespraak met diverse studenten heb ik twee belangrijke zaken geconstateerd:

1. het was het allerlaatste tentamen voor de zomervakantie, dus een aantal studenten is in tijdnood gekomen en heeft niet alle stof bestudeerd, en
2. studenten wisten niet goed wat te verwachten omdat dit het eerste juist/onjuist tentamen ooit was, dus er kon minder gericht gestudeerd worden.

Beide effecten hebben een negatieve invloed gehad op de uitslag van het tentamen. Echter doordat het tentamen cijfer slechts voor 30% meetelt in het eindcijfer, hebben de meesten het vak wel gehaald.

Verder is het me opgevallen dat studenten “goedgelovig” zijn en meestal het antwoord juist aankruisen als ze twee bekende termen zien, zelfs als die in een verkeerd verband staan. Dit komt tot uitdrukking in de probleemvragen die Sonate signaleert: van de 17 verdachte vragen hadden er 13 met onjuist beantwoord moeten worden. Omdat Sonate geen rekening houdt met de scheve raadkans van de studenten (voorkeur voor juist) zijn de analyse resultaten onbetrouwbaar. Dus de Sonate statistieken moeten (helaas) met een korreltje zout genomen worden.

## 4 Evaluatie

<b>Competentie:</b> De docent is in staat zijn onderwijs te (laten) evalueren en bij te stellen op basis van de verschillende evaluatiegegevens.
--------------------------------------------------------------------------------------------------------------------------------------------------

<sup>10</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/in4020/2003-06-27-sonate.pdf>

<sup>11</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/in4020/2003-06-27-cijfers.pdf>

## 4.1 bewijsstukken

Ten behoeve van de evaluatie van het compilerbouw vak als onderdeel van de informatica Masters heb ik een evaluatie<sup>12</sup> gemaakt waarom het practicum zo moeizaam verliep en hoe dat verbeterd kan worden.

## 4.2 validatie

## 4.3 zelfreflectie

Het is nu de tweede keer dat ik het practicum moet bijstellen omdat het teveel werk is voor de studenten. De eerste verfijning was ingecalculeerd, de tweede is veroorzaakt door de veranderde studentenpopulatie (verplicht vs. keuze vak). Helaas betekent dit wel dat het niveau omlaag gaat. Gelukkig wordt het programmeeronderwijs in de Bachelorsfase nu verbeterd en hoop ik in het vervolg daarvan de vruchten te kunnen plukken.

# 5 Analyse

**Competentie:** De docent is in staat te analyseren waarom een individuele student niet slaagt voor een toets en op basis daarvan de student te adviseren.

## 5.1 bewijsstukken

Er zijn nog geen individuele studenten geweest die om raad zijn komen vragen naar aanleiding van het schriftelijk tentamen Compilerbouw. Een globale analyse van de tentamenvragen leerde dat er niet één specifiek onderdeel is waar het gros van de studenten fout geantwoord heeft.

Bij het nakijken van de practicumopdrachten zijn er diverse herkansingen uitgedeeld als bleek dat de ingeleverde oplossing onder de maat was. Elke ingeleverde compiler wordt getest mbv. een reeks testprogramma's. Bij een herkansing wordt altijd of een cruciaal testprogramma meegeleverd of een reeks aanwijzingen gegeven wat er fout gaat. Dit blijkt informatief genoeg voor de studenten om hun compiler flink te verbeteren.

## 5.2 validatie

## 5.3 zelfreflectie

# 6 ICT

**Competentie:** De docent is in staat om nieuwe ontwikkelingen en technologieën (zoals het gebruik van ICT) op een zinvolle manier te gebruiken bij het toetsen en evalueren van onderwijs.

## 6.1 bewijsstukken

- Bij het nakijken van de juist/onjuist tentamens maak ik gebruik van het Sonate systeem.
- Tijdens het vervaardigen van het eerste tentamen heb ik eigenhandig een elektronische item-bank aangelegd. Ik heb diverse scripts geschreven om automatisch uit een lijstje vraagnummers een tentamen met/zonder antwoorden te genereren.

<sup>12</sup><http://rama.pds.twi.tudelft.nl/~koen/bko/in4020/evaluatie>

- De blackboard site van het vak (in4020<sup>13</sup>) bevat een elektronisch archief met oude tentamens (nu dus 1 oefen tentamen, en 1 ‘echt’ tentamen).
- Het praktisch werk dient per e-mail ingeleverd te worden, en de cijfers/beoordelingen worden ook weer per e-mail teruggestuurd.

## 6.2 validatie

## 6.3 zelfreflectie

Als ‘echte’ informaticus hou ik de ICT zaken graag zelf in de hand. Vandaar dat ik gekozen heb om bijvoorbeeld m’n eigen item-bank op te zetten. Ik verwacht dat de investeringen (in tijd) van het ontwikkelen vd nodige scripts (software) er in de loop vd jaren wel uit komt.

Het Sonate systeem werkt redelijk soepel, alleen heb ik iets minder vertrouwen gekregen in de statistieken die geproduceerd worden omdat die geen rekening houden met ‘goedgelovige’ studenten die bij twijfel geneigd zijn *juist* te antwoorden (zie sectie 3.3).

---

<sup>13</sup>[http://blackboard.icto.tudelft.nl/bin/common/course.pl?course\\_id=\\_6011\\_1&frame=top](http://blackboard.icto.tudelft.nl/bin/common/course.pl?course_id=_6011_1&frame=top)