Introductie tot de cursus

Introductie tot de cursus

1 De cursus

De cursus draagt de Engelse naam Software evolution: dit omvat de activiteiten die volgen op de release van een softwaresysteem. Ook nadat een systeem is uitgebracht en in productie is genomen, zal de software moeten worden aangepast aan veranderende omstandigheden, om zo aan de wensen van de gebruikers te blijven voldoen. Hierbij kan worden gedacht aan het herstellen van fouten, het uitbreiden van de functionaliteit, het achterhalen van de structuur van een systeem en het verbeteren van de programmacode. In de cursus wordt ingegaan op de processen voor het onderhouden van een systeem (software maintenance), en de problemen die daarbij optreden. Verder worden er technieken aangeboden voor het analyseren en transformeren van programmacode. Met deze technieken wordt het makkelijker om bestaande systemen aan te passen. In twee practicumopdrachten gaat u zelf aan de slag met een voorbeeld van een techniek voor programma-analyse, namelijk het toepassen van softwaremetrieken om zo de kwaliteit en onderhoudbaarheid van het softwareproduct te bepalen. Hierbij wordt gebruikgemaakt van de domeinspecifieke programmeertaal Rascal, die speciaal voor dit type analyse en transformatie is ontworpen. Deze cursus is tot stand gekomen in samenwerking met de Universiteit van Amsterdam (UvA) en het Centrum Wiskunde & Informatica (CWI). De cursus wordt al vele jaren gegeven aan de UvA, waar hij deel uitmaakt van de eenjarige masteropleiding Software Engineering: de OU-cursus is gebaseerd op de cursus uit Amsterdam en volgt de structuur van die cursus. Het CWI ontwikkelt en onderhoudt de domeinspecifieke programmeertaal Rascal waarvan in deze cursus gebruik wordt gemaakt, en doet actief onderzoek op dit gebied. De webpagina van de taal is te vinden op rascal-mpl.org.

Plaats in de opleiding
De cursus Software evolution van de Open Universiteit bevindt zich op masterniveau. De cursus is een verplicht onderdeel van de masteropleiding Software Engineering en maakt deel uit van de gebonden keuze binnen de masteropleiding Computer Science. Om deze cursus te kunnen volgen heeft u minstens een wo-bacheloropleiding nodig, of heeft u zich dit kennisniveau eigengemaakt. Verondersteld wordt dat u over de volgende voorkennis beschikt:
– Vertrouwdheid met reguliere en contextvrije grammatica's, en met reguliere expressies. Deze kennis wordt bijvoorbeeld aangebracht in de OU-cursus Formele talen en automaten (of diens voorganger Talen en ontleders).

– Kennis van OO-programmeren, als bijvoorbeeld gegeven in de OU-cursussen Objectgeoriënteerd programmeren met Java 1 en 2 (maar een gedegen kennis van bijvoorbeeld C# is ook goed). In de practicumopdrachten zullen Javaprogramma's worden geanalyseerd: om dit te kunnen uitvoeren is het noodzakelijk om de structuur van een Javaprogramma te begrijpen.
– Een globaal overzicht van de software lifecycle. Bij voorkeur bent u in een praktijksituatie in aanraking geweest met de problemen die optreden bij het onderhouden en aanpassen van grootschalige softwaresystemen.
– Bekendheid met functioneel programmeren is niet strikt noodzakelijk, maar het maakt het uitvoeren van de opdrachten bij de cursus wel gemakkelijker.

2 Cursusmateriaal

Het schriftelijke materiaal bestaat uit deze studeerwijzer, een reader met artikelen en twee reference cards (zogenaamde 'cheat sheets') voor de programmeertaal Rascal.
Deze referentiekaarten zijn ook te vinden op de website van Rascal: rascal-mpl.org.

De studeerwijzer
In de studeerwijzer staat algemene informatie over de cursus, de opdrachten, de beoordeling en de begeleiding. Leest u dit zorgvuldig door voordat u de rest van de cursus bestudeert. De inhoud van de cursus is onderverdeeld in vijf blokken. Bij elk blok wordt een korte inleiding gegeven en staat aangegeven welk materiaal moet worden bestudeerd, zoals dia's en artikelen. Alle artikelen zijn opgenomen in de reader van deze cursus. Bij een aantal blokken staat ook extra materiaal vermeld dat niet standaard tot de cursus behoort maar als aanvulling op de stof kan worden geraadpleegd. In de literatuurlijst van ieder blok staat duidelijk gemarkeerd welke artikelen extra zijn. In de literatuurlijst staan ook verwijzingen naar tekstboeken die niet tot de verplichte stof behoren.

Materiaal op Studienet
Veel materiaal van de cursus is beschikbaar via Studienet, de elektronische leeromgeving van de Open Universiteit. Deze is te bereiken via studienet.ou.nl. Hier vindt u onder andere:
– verwijzingen naar de artikelen die staan vermeld in deze studeerwijzer
– de dia's die horen bij de hoorcolleges zoals gegeven aan de UvA
– mogelijke aanvullingen en errata op de stof
– informatie over de benodigde software
– een FAQ voor het gebruik van Rascal
– een algemene discussiegroep voor vragen over de cursus
– een inwerkopdracht met oefeningen over Rascal
– twee practicumopdrachten
– de video-opname van een gastpresentatie over software-productkwaliteit
– contactgegevens van de studiebegeleiders.

3 Het bestuderen van de cursus

De studielast bedraagt ongeveer 100 uur. Globaal staan er 40 uur voor de twee opdrachten (elk 20 uur) en 60 uur voor het bestuderen van de stof (inclusief het installeren van de software en de voorbereiding op het mondelinge tentamen). Bij aanvang van de cursus installeert u de Rascal-software: deze is beschikbaar als een Eclipse-plug-in. Volg de instructies die te vinden zijn op de Rascal-webpagina onder 'GettingStarted'. Documentatie van de taal is beschikbaar via de Rascal-tutor: deze kan worden opgestart via het Rascal-menu in Eclipse. Het opstarten van de tutor kan enige tijd duren. In de tutoromgeving zijn ook enkele kleine opgaven opgenomen om bekend te raken met de programmeertaal. De documentatie is ook te raadplegen via de Rascal-website (als manual). In de Eclipse-omgeving kan worden gezocht naar updates van de software. Op de OU cursussite staat aanvullende informatie over Rascal in de vorm van een FAQ.

Leerdoelen
Na het bestuderen van de cursus wordt verwacht dat u in staat bent om:
– een beargumenteerde keuze te maken uit verschillende technieken voor het analyseren van een programma in een bepaalde situatie
– de kwaliteit en structuur van een bestaand softwaresysteem te analyseren door het extraheren van feiten, en hier conclusies uit te trekken
– een afweging te maken tussen de voor- en nadelen van softwaremetrieken bij het bepalen van de productkwaliteit
– softwaremetrieken toe te passen op een bestaand systeem
– te beschrijven welke problemen er optreden bij het onderhouden en het uitbreiden van software, bekeken vanuit zowel het bedrijfskundige als het softwaretechnologische perspectief
– de oorzaken van software-evolutie te herkennen en de uit software-evolutie voortvloeiende problemen te analyseren en op te lossen
– uit te leggen hoe een systeem meer flexibel gemaakt kan worden door het toepassen van programmatransformaties, of door middel van reverse engineering
– gegevens over software-artefacten te visualiseren, en deze visualisatie te evalueren aan de hand van Tufte's grafische ontwerpprincipes, en Shneiderman's interactieprincipes.

Opbouw van de cursus
De cursus is opgebouwd uit vijf blokken: bij de blokken II en IV hoort een practicumopdracht. Merk op dat de totale studielast van de cursus niet gelijkmatig is verdeeld over de vijf blokken.

Blok	*Studielast*	*Opdracht*
I Introduction to software evolution	2	
II EASY meta-programming with Rascal	28	20 uur (opdracht 1)
III Topics in software evolution	14	
IV Towards visual software analytics	12	20 uur (opdracht 2)
V Mining software repositories	4	

4 Opdrachten

De twee practicumopdrachten moeten beide in Rascal worden uitgevoerd. Over deze programmeertaal is het volgende geschreven:

Rascal is an experimental DSL (Domain Specific Language) for analyzing and transforming source code. Rascal programs are written to transform programs from one version of a language to another, to generate code from a domain specific language, to collect metrics about programs, to reverse engineer the architecture of legacy software, to do impact analysis of bug fixes, to implement refactorings, etc. Our goal is that software engineers can create and adapt their own high quality meta tools instead of having to rely on the plethora of different off-theshelf and open-source tools with unknown quality. By providing high level language constructs for common meta programming tasks they should be able to construct any kind of meta programming tool with considerably less implementation overhead. Examples of such constructs are parsing, AST construction, complex pattern matching, computing relations between source code artefacts and manipulating these relations, and visualization.

Bron: devnology.nl/nl/bijeenkomsten/details/19

Meer informatie over de installatie van de benodigde software is te vinden op de cursussite. De opdrachten worden individueel gemaakt; op verzoek en na goedkeuring van de examinator mag er worden samengewerkt in een tweetal. De volledige opdrachtomschrijving van beide opdrachten is te vinden op de cursussite, samen met al het aanvullende materiaal. Hier vindt u ook de eisen en criteria waarop wordt gelet bij de beoordeling van de opdrachten. Voor de beoordeling en de bijkomende mondelinge toelichting wordt verwezen naar het kopje 'tentaminering'. Hieronder volgt tot slot nog een korte impressie van beide opdrachten. In de eerste practicumopdracht wordt u gevraagd om een aantal softwaremetrieken uit te werken, en deze te toetsen op bestaande systemen geschreven in Java van verschillende omvang. De gebruikte metrieken zijn afkomstig uit een kwaliteitsmodel en geven inzicht in de structuur en kwaliteit van een systeem. Uit de gevonden resultaten moeten vervolgens conclusies worden getrokken over de onderhoudbaarheid van de software en de risicogebieden. De eerste opdracht maakt deel uit van Blok II. In de tweede opdracht wordt één onderwerp op het gebied van programma-analyse verder uitgediept, namelijk het visualiseren van kwantitatieve gegevens over software-artefacten. Het doel van deze opdracht is om met een grafische voorstelling de uitkomsten van de metrieken uit de eerste opdracht inzichtelijk te maken. Hierbij wordt theorie aangereikt om visualisaties te evalueren, zoals Tufte's grafische ontwerpprincipes en Shneiderman's interactieprincipes. De tweede opdracht maakt deel uit van Blok IV.

5 Tentaminering

Om de cursus af te ronden moet aan de onderstaande verplichtingen worden voldaan. Tussen haakjes staat de weging van de onderdelen vermeld:
– *Eerste opdracht (50%)*: de uitwerking van de onderhoudbaarheidsanalyse wordt opgestuurd naar de docent en wordt (op afspraak) mondeling toegelicht en besproken. Deze opdracht wordt beoordeeld met een cijfer.
– *Tweede opdracht (50%)*: de uitwerking van de visualisatie van de metrieken wordt opgestuurd naar de docent en wordt mondeling toegelicht. Deze opdracht wordt beoordeeld met een cijfer.
– *Mondeling tentamen*: er wordt getoetst of er voldoende kennis is opgedaan van de verplichte stof. Dit onderdeel wordt met een voldoende of onvoldoende beoordeeld.

Alle onderdelen moeten met een voldoende (≥ 6) zijn afgerond. Het eindcijfer van de cursus is het afgeronde gemiddelde van de twee opdrachten, onder de voorwaarde dat ook voor het mondeling tentamen een voldoende is gehaald. Het mondelinge tentamen vindt in principe plaats aansluitend op de toelichting bij de tweede opdracht. Er zijn dus twee contactmomenten. Als richtlijn wordt een halfuur per onderdeel aangehouden, oftewel anderhalf uur in totaal. Opdrachten die door een tweetal zijn uitgewerkt mogen door beide teamleden in één sessie worden toegelicht. Het afsluitende mondeling tentamen is wel in alle gevallen individueel en bestrijkt alle blokken. Studenten moeten ervoor zorgen dat de uitwerkingen van de opdrachten minstens 24 uur voor de afspraak in het bezit zijn van de examinator. Afspraken worden in onderling overleg gemaakt met de examinator van de cursus en kunnen zowel face-to-face als op afstand plaatsvinden.

6 Begeleiding

Bij de cursus hoort een begeleidingscyclus die eenmaal per jaar wordt aangeboden. De begeleiding bestaat uit één face-to-face-startbijeenkomst en zes virtuele bijeenkomsten die worden gehouden met behulp van Elluminate. Zo mogelijk zal tijdens een van de bijeenkomsten een gastspreker ervaringen uit de praktijk komen presenteren, ter voorbereiding op de practicumopdrachten. Vragen over de inhoud of organisatie van de cursus kunnen per e-mail worden gesteld aan de examinator, of in de discussiegroep van Studienet. De contactgegevens van de betrokken docenten zijn te vinden op de cursussite.