**HOGESCHOOL ROTTERDAM / CMI**

**Project 56**

**INFPRJ0156**



|  |
| --- |
| Aantal studiepunten: 8  Modulebeheerder: Kevin van Ingen |

# Modulebeschrijving

|  |  |
| --- | --- |
| **Modulenaam:** | Project 56 – software engineering en architectuur |
| **Modulecode:** | **INFPRJ0156** |
| **Aantal studiepunten en studiebelastinguren:** | Dit studieonderdeel levert je acht studiepunten op, hetgeen overeenkomt met een studielast van 224 uren.  Begeleide colleges:  Les gedurende 18 weken: 4 \* 50 minuten 60 uur  Evaluatie week 9 & 19: 4 \* 50 minuten 6 uur  Onbegeleide uren per week:  Literatuurstudie 25 uur  Ontwikkeling applicatie (6 uur per week buiten lessen om) 110 uur  Documenteren applicatie 25 uur  Totaal 224 uur |
| **Vereiste voorkennis:** | Projectmatig werken uit het eerste jaar. |
| **Werkvorm:** | Projectonderwijs (groepswerk), werkcolleges skills |
| **Toetsing:** | Toetsing vindt plaats op basis van opgeleverd beroepsproduct en toetsing van projectproces. |
| **Leermiddelen:** | Algemene projecthandleiding, Hogeschool Rotterdam, 2012 |
| **Draagt bij aan competentie** |  |
| **Leerdoelen:** | Je beschikt over een assertieve houding in teamwork en weet duidelijk het verschil tussen assertief en subassertief gedrag in samenwerking.  Je kunt je softwareontwikkelproces ondersteunen door middel van een moderne softwareontwikkelstraat en maakt een projectmethode op maat voor het project.  Je kunt voor de code een kwaliteitsverbeteringsproces en bijhorende software inzetten (bv code peer review, tests, statische analyse).  Je kunt middels een schriftelijk software deployment plan over de uitrol van een complex systeem adviseren inclusief testplan voor integratie- en unittest.  Je adviseert over een geschikte software architectuur (met ontkoppelling in het ontwerp) voor software beschreven in een business case.  Je kunt onderhoudbare en uitbreidbare objectgeoriënteerde software realiseren bestaande uit meerdere programmeertalen en platformen.  Je kunt voor een complex (meerdere platformen) software product een software requirements specificatie opstellen met bijhorende niet-functionele eisen (ISO 25010).  Je kunt industriestandaard software-algoritmen toepassen. |
| **Inhoud:** | Je leert in groepsverband (proces) een projectopdracht voor een opdrachtgever te realiseren (product). Je werkt met meerdere software platformen en past industriestandaard technieken toe, zoals webservices, REST en design patterns. |
| **Opmerkingen:** | Aanwezigheid is verplicht (80% volgens OER). |
| **Modulebeheerder:** | Kevin van Ingen |
| **Datum:** | 10 augustus 2015 |

# 1. Algemene omschrijving

## Inleiding

Een groot software engineering project! Dat is iets iedereen een keer zelf moet meemaken om te zien hoe complex software projecten zijn. Een wirwar van softwarecomponenten die allemaal informatie uitwisselen over verschillende platformen, verschillende machines en met behulp van verschillende protocollen. Dat is waar echte software engineers dagelijks mee bezig zijn in de beroepspraktijk!

De volledige levenscyclus van software beslaat het analyseren, ontwerpen, bouwen, beheren van software. In dit project, maak je dit in het geheel mee. Elke fase heeft grote invloed op de volgende fase. Dit leer je door het gewoon te gaan doen!

De projectdocenten zijn daarin deze keer niet je opdrachtgever, maar je coach! De coach staat aan de zijlijn en volgt jouw voortgang om je op technisch gebied te kunnen begeleiden. Je legt je technische oplossingen wekelijks voor aan je docent en ontvang hier advies op.

De casus staat beschreven in het volgende hoofdstuk.

## Relatie met andere onderwijseenheden

De skills in een project zijn onlosmakelijk verbonden met het project zelf. Denk daarbij aan samenwerken, *leiding geven, overleggen, elkaar feedback geven en mondeling en schriftelijk communiceren. De* flankerende modules die je naast het project krijgt, hebben allemaal te maken met de kennis en vaardigheden die je in het project nodig hebt. Het onderwerp van het project is daarbij leidend. Bovendien trainen we je om een professionele beroepshouding te laten zien.

Hieronder wordt er aangegeven wat de relatie is van deze module met andere modulen. Hier kun je zien of je voldoende achtergrondkennis hebt of dat je je extra moet voorbereiden om deze module te kunnen volgen. Daarnaast zie je hoe analyse en project gebruik maken van deze module.

* Voorbereiding op deze module
  + DEV; je hebt in dev01-2 en dev01-4 basisvaardigheden voor het programmeren van software opgedaan.
  + ANL01-2: je bent in staat om een specificatie van een softwaresysteem te modeleren.
* Parallel aan deze module en toegepast in het project:
  + INFANL05: in deze module wordt kennis over technieken en methoden in de software engineering aangeleerd.
  + INFDEV05: In deze module leer je nieuwe opslagtechnieken en dataverwerking.
  + INFANL06: In deze module wordt je het testen van software geleerd. Het opstellen van een testplan tot het uitvoeren van geautomatiseerd testen.
  + INFDEV06-1: In deze module wordt er dieper ingegaan op de datastructuren onderliggend aan je software, de performance van deze datastructuren en de situaties waarin je deze gebruikt.
  + INFDEV06-2: In deze module leer je hoe het software op geautomatiseerde wijze van ontwikkelomgeving gemigreerd wordt naar test en productieomgeving.

## Leermiddelen

Er is veel literatuur met best-practices. Hieronder staat een korte opsomming van wat we gebruiken dit kwartaal. Met name de blogs kun je in de eerste weken gebruiken om in te lezen.

* Algemene projecthandleiding, Hogeschool Rotterdam, 2012 (op N@tschool)
* Literatuur van de flankerende vakken.

Facultatief

* Domain Driven Design, Eric Evans, 2003  
  <http://www-public.int-evry.fr/~gibson/Teaching/CSC7322/ReadingMaterial/Evans03.pdf>
* The Clean Coder A Code of Conduct for Professional Programmers, Robert C. Martin, 2011, ISBN-13: 978-0137081073.
* Clean Code: A Handbook of Agile Software Craftsmanship, Robert C. Martin, 2008, ISBN-13: 978-0132350884.   
  <http://vitoex.googlecode.com/svn/trunk/Read/Clean%20Code.pdf>
* The Pragmatic Programmer: From Journeyman to Master, Andrew Hunt, 1999, 978-0201616224.

Weblinks voor onderwerpen die voor dit project belangrijk zijn:

* Full stack Developers: <http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer/>

## Structuur van het semester

Hieronder kan je zien hoe de vakken samenhangen. Al de flankerende vakken spelen een rol in de levenscyclus van een complex software-engineeringproject.

# Programma en inhoud

## Casus

Automatisering heeft parels aan techniek voortgebracht, maar ook gedrochten aan onnodige complexe applicaties! Het verschil zit vaak in het kiezen van de juiste architectuur! Grote bedrijven hebben tientallen tot honderden softwaresystemen, waarvan een groot gedeelte ook nog informatie met elkaar uitwisselt. Veel web-gebaseerde systemen bieden een gebruiksvriendelijke front-end op een onwijs complexe back-end.

In dit project gaan jullie kennismaken met diverse ‘archetypes’ in de software-industrie. Webservices, data-parsing, client-server architectuur, worden gecombineerd met complexe berekeningen.

**Wordt een software engineer!**

Gedurende dit half jaar richten we ons rigoureus op een solide basis software-engineeringvaardigheden. Dit gaan we dagelijks trainen. Een belangrijk onderdeel wat je dit project ontwikkeld is de bijhorende houding. Een software engineer herkent problemen en kan zelfstandig zich nieuwe technieken eigen maken en deze inzetten als onderdeel van een groter systeem.

## Opdracht

We gaan aan slag om een softwaresysteem om de data van CityGis te schrijven. CityGis krijgt elke seconde enorme hoeveelheden data binnen en verwerkt dit in een omgeving waarmee de hulpdiensten zijn aan te sturen.

Meer inhoudelijke informatie over eisen aan het systeem vind je in het business visiedocument.

**De techniek(en)**

Je gaat dit project meerdere softwareplatformen laten samenwerken. Het systeem wat je gaat bouwen bestaat uit vier onderdelen: een crawler voor producten en prijzen, een zoekmachine, een pc-builder en een rapportagesysteem. Je moet bij het ontwikkelen van deze systeemonderdelen minimaal twee platformen (Java (of JVM gebaseerd zoals Scala en Groovy), C#/.Net, Python, Ruby) kiezen uit de lijst. Daarnaast kies je een single-page client-side techniek zoals AngularJS, ReactJS of BackboneJS om de front-end te realiseren. De lijst is samengesteld door docenten en vertegenwoordigen populaire talen en platformen die passen binnen het object-georienteerd paradigma. Daarnaast zijn het talen/platformen die waardevol zijn om te beheersen voor je op stage gaat. Verder gaan we onze [Full-stack-skills](http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer/) trainen dit half jaar!

**Let op!** Het verkennen van de (on)mogelijkheden van de talen is een belangrijke stap. Er worden geen begeleidende lessen aangeboden per platform. Je projectdocent wijst je alleen (bij problemen) in de juiste richting. Het is belangrijk dat je zelf de oplossingen leert te verkennen.

## Programma en weekschema

Je werkt in een groep van **vier** personen.

|  |  |  |
| --- | --- | --- |
| **Week** | **Fase** | **Lesinhoud** |
| 1 | **Geen les ivm startweek eerstejaars** | |
| 2 | Analyse | Kick-off  Klassikale inleiding + technieken kiezen + casus doorspreken |
| 3 | Analyse | Groep toont analyse van situatie middels eigen functionele eisen |
| 4 | Ontwerp | Docent coacht discussie gedachten over ontwerp van de software |
| 5 | Ontwerp | Groep presenteerd technische prototype/spike van de technieken |
| 6 | Ontwerp | Groep presenteert ontwerp + feedback op ontwerp van docent |
| 7 | Implementatie | Docent controleert voortgang |
| 8 | Implementatie | Docent controleert voortgang |
| 9 + 10 | **Groep werkt zelfstandig aan implementatie** | |
| 11 | Implementatie | Presentatie deelimplementatie |
| 12 | Implementatie | Docent controleert voortgang |
| 13 | Implementatie | Scholing: elke groep schoolt de klas in ‘nieuwe’ technieken/toepassingen |
| 14 | Implementatie | Peer review op code |
| 15 | Implementatie | Docent controleert voortgang |
| 16 | Implementatie | Peer review op code |
| 17 | Beheer | Docent controleert voortgang |
| 18 | Beheer | Groep presenteerd test/deploymentstraat |
| 19 | Demonstratie / eindassessment | |
| 20 | Eindevaluatie | |

In de onderstaande tabel staan de op te leveren mijlpaalproducten per week. Voor alle (tussen)producten geldt: lever ze uiterlijk op vrijdag om 12.00 uur in de betreffende week in. Mijlpaalproducten met een \* tellen mee voor je eindcijfer.

|  |  |  |
| --- | --- | --- |
| **Week** | **Mijlpaalproduct** | **Doel** |
| 2 | Team aanmelden | Go / no go |
| 4 | Visie document | Go / no go |
| 9 | Software Requirments Specification \*  Teamevaluatie (bijlage 2) \* | Assessment |
| 18 | Software (code + uitvoerbare versie) \*  Software release document \*  Nieuwe teamevaluatie (bijlage 2) \* | Assessment |
| 19 | Demonstratie / eindassessment \* | Assessment |

\* Telt mee voor eindcijfer

1. **Toetsing en beoordeling**

|  |
| --- |
| Let op! In onze onderwijsfilosofie heb je bij flankerende vakken theorie en praktijk dat waar mogelijk of toepasbaar geïntegreerd is met het project. Maar dat betekend zeker niet dat je elk document bij elke vak **identiek** kan/mag inleveren! Ga per vak na wat de beoordelingscriteria zijn. |

## Voorwaarden

In het project word je beoordeeld op een combinatie van groepsprestatie en individuele inzet. Je moet voldoen aan de aanwezigheidsplicht en de (tussen)producten op tijd hebben ingeleverd.

|  |  |  |
| --- | --- | --- |
| **Onderdeel** | **Beoordelaar** | **Doel** |
| Tussentijdse beoordeling  - Softwarerequirementspecificatie  - Teamevaluatie 1 (bijlage 2)  - Code van prototype/spike | Projectdocent | Becijferen |
| Eind  - Teamevaluatie 2 (bijlage 2)  - Alle software (data, code + uitvoerbare versie)  - Software release document  - Testrapport | Projectdocent | Becijferen |
| Aanwezigheid | Projectdocent | Afvinken (moet voldoende zijn) |

## Eindcijfer

Het eindcijfer wordt vastgesteld op basis van een individueel gedeelte en een groepcijfer. De weging van de onderdelen staan in de onderstaande tabel. De relatie tussen leerdoelen en toetsing in bijlage 1.

|  |  |  |
| --- | --- | --- |
| **Onderdeel** | **Beoordelaar** | **Weging** |
| Tussentijds assessment (bijlage A) | Projectdocent | Cijfer (20 %) |
| Software release document + testrapport (bijlage B) | Projectdocent | Cijfer (20 %) |
| Alle software - data, code & implementatie (bijlage C) | Projectdocent | Cijfer (40 %) |
| Individueel gedeelte (bijlage D) (o.b.v. teamevaluaties en participatie in lessen) | Projectdocent | Cijfer (20 %) |

## Herkansing

Heb je een onvoldoende gehaald voor INFPRJ0156 dan kom je in aanmerking voor een herkansing.

*Herkansing*

Bij onvoldoende aanwezigheid en/of inspanning volgens het ‘onderwijs examen regelement’ kan deze module volgend jaar herkanst worden. In andere gevallen kunnen alle toetsonderdelen uit paragraaf 3.2 opnieuw worden aangeboden aan de projectdocent in week 10 van het volgende kwartaal.

Herkansingen worden uitgevoerd zonder begeleiding en zonder ondersteunend onderwijs. Alleen eindproduct(en) worden beoordeeld. Herkansingsopdrachten worden (volgens examenreglement) individueel verstrekt en kunnen afwijken van de originele casus.