**HOGESCHOOL ROTTERDAM / CMI**

**Software delivery**

**INFDEV016B**



|  |
| --- |
| Aantal studiepunten: 3  Modulebeheerder: Kevin van Ingen |

# Modulebeschrijving

|  |  |
| --- | --- |
| **Modulenaam:** | Software delivery |
| **Modulecode:** | **INFDEV016B** |
| **Aantal studiepunten en studiebelastinguren:** | Dit studieonderdeel levert 3 studiepunten op, hetgeen overeenkomt met een studielast van 84 uur.  De verdeling van deze 84 uur over de collegeweken is als volgt:  Begeleide colleges:  - gedurende 8 weken: 8 \* 150 minuten 20 uur  Onbegeleide uren per week:  - bestuderen theorie (8 \* 3 uur per week) 24 uur  - uitwerken van opdrachten (8 \* 5 uur per week) 40 uur  Totaal 84 uur |
| **Vereiste voorkennis:** | - |
| **Werkvorm:** | Werkcollege (gecombineerd theorie en practicum) |
| **Toetsing:** | Praktische opdrachten |
| **Leermiddelen:** | Continuous Delivery, Jez Humbble, David Farley, Addison Wesley, ISBN 978-0-321-60191-9  Daarnaast ook online lesmateriaal |
| **Draagt bij aan competentie:** |  |
| **Leerdoelen:** | 1. Je bent bekend met industriestandaard tools en technieken voor software deployment 2. Je kunt een geautomatiseerde software deployment ontwikkelen 3. Je kunt continuous integration in de praktijk brengen 4. Je kunt een softwareontwikkelingsomgeving voor een team inrichten 5. Je hebt een basisvaardigheid in serverbeheer en kan hiermee geautomatiseerd onderhoud aansturen |
| **Inhoud:** | Deze module gaat over geautomatiseerde oplevering van software. Het beslaat continuous integration, automated deployment, geautomatiseerd testen, build scripts en serverbeheer. |
| **Opmerkingen:** | Aanwezigheid is verplicht voor toetsing (demonstratie) van de opdrachten. |
| **Modulebeheerder:** | Kevin van Ingen |
| **Datum:** | 21 oktober 2015 |

# 1. Algemene omschrijving

## Inleiding

In deze module maak je je vaardig in het domein van software delivery. Software delivery beslaat het automatisch opleveren van software op geautomatiseerde wijze. Het gehele proces van software ontwikkeling in teams, het testen van software tot de deployment naar honderden productieservers. Belangrijke software kan op deze manier elke minuut van het jaar online blijven, maar toch meerdere malen per dag een update ontvangen. In een internationale wereld is de ouderwetse ‘nachtelijke upgrade’ echt iets uit het verleden. In deze module leer je de technieken om dit te verwezenlijken.

## Relatie met andere onderwijseenheden

**INFPRJ0156**

In deze onderwijsperiode werk je in project 56 aan een beroepsgerichte opdracht, waar je de kennis en vaardigheden uit deze module direct toepast. Dit project draait om professionele software ontwikkeling.

**INFANL01-6**

In deze module leer je (geautomatiseerd) testen.

**INFDEV01-4**

In deze module heb je contact gehad met software ontwikkelingsprincipes als gedistribueerd versiebeheer op code (m.b.v. GIT) en unit testing. Deze handelingen ga je automatiseren in dit vak.

## Leermiddelen

Verplicht:

* Continuous Delivery, Jez Humbble, David Farley, Addison Wesley, ISBN 978-0-321-60191-9

Facultatief:

* Jenkins, The Definitive Guide: [Link](http://it-ebooks.info/book/576/)
* Maven: Apache Maven 3 Cookbook (oa via [link](https://www.google.nl/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=0CEwQFjAB&url=http%3A%2F%2Fwww.zenisoft.cn%2Fcalibre%2Fget%2Fpdf%2F1032&ei=zM2IUeXbEcOR0AW9iIGgBw&usg=AFQjCNG0fjKlGTBTlr8KcBer6pmIK1afIA&sig2=LGGGZSXIkIbBe2Ivy7DPfw))
* 2014 Devops report: [Link](https://puppetlabs.com/2014-devops-report)

# Programma

Deze module wordt gegeven in werkcollegevorm. Dit combineert theorieles met practicum. Beide komen wekelijks aan bod. Je wordt geacht het boek te bestuderen vóór de les. In de opdrachten wordt hiernaar verwezen.

In de les behandelt de docent theorie uit de bijhorende hoofstukken. Bij de weken waar een inleveropdracht vermeld staat, dien je deze bij het begin van de les voltooid te hebben. Na het theoriegedeelte van de les heb je kans om aan de volgende opdracht te werken en krijg je feedback van de docent over je opdrachten.

De opdrachten in de rechterkolom staan in de bijlagen beschreven (inclusief criteria).

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Literatuur** | **Lesinhoud** | **Inleveren / afvinken (zie bijlagen)** |
| 1 |  | Inleiding vak  Inleiding beheer  Begin practicum opdracht A |  |
| 2 |  | Inleiding beheer en infrastructuur  Basis principes SD | Inleveren opdracht A |
| 3 | H1, 2 | Casus Etsy.com  Build scripts (maven) |  |
| 4 | H3, 4 | Continuous integration  Versiebeheer van code | Inleveren opdracht B |
| 5 | H5, 6 | Deployment  Docker |  |
| 6 | H7 + 13 | Geautomatiseerd testen  Commit stage | Inleveren opdracht C |
| 7 | H8, 9 | Deployment  Configuratie management |  |
| 8 | H10,11  H12 + 14 | Data management  Continuous Delivery in projecten | Inleveren opdracht D |
| **Eind week 8: DIGITAAL inleveren alle opdrachten!!** | | | |

1. **Toetsing en beoordeling**

## 3.1 Procedure

De module wordt afgesloten met een practicumcijfer. Het practicumcijfer is een gewogen gemiddelde van vier practicumopgaven. Cesuur ligt op 5.5. De weging van de opdrachten is weergegeven in de onderstaande tabel:

|  |  |
| --- | --- |
| Opdracht A | 10 % |
| Opdracht B | 20 % |
| Opdracht C | 30 % |
| Opdracht D | 40 % |

***Practicum***

Tijdens het practicum werk je aan je vaardigheden als softwarespecialist. Je gaat zelf een omgeving opzetten voor het uitvoeren van softwaredeployments. Deze omgeving bouw je helemaal zelf op. Dit is onderverdeeld in verschillende deelopgaven. De opgaven en de criteria staan beschreven **in de bijlagen**.

Om je kennis en inzicht te testen kan de docent je vragen tijdens de les een aanvullende opdracht te maken. Het eindoordeel voor het vak is het gewogen gemiddelde van deze opdrachten.

Je maakt een werkdocument waar je per opdracht je code en ‘bewijs’ van de werkzaamheid verzameld (bv screenshots). Je maakt dus één document waar de vier opdrachten als hoofstuk onderdeel van zijn.

**Let op!** Elke week lever je je ‘werkdocument’ in op N@tschool bijgewerkt met de laatste opdracht. Deze worden op plagiaat gescanned.

## 3.2 Herkansing

Indien je een onvoldoende behaalt voor dit vak, kan je het practicum in week 10 van de volgende onderwijsperiode herkansen door de gehele set aan opdrachten bij je docent opnieuw aan te bieden. Je docent voorziet je in een extra aanwijzing voor de herkansingsopdracht. Deze herkansing voer je zelfstandig uit. Indien je project geen goede voedingsbodem vormt voor het vak dan wijst je docent een ander softwareproject aan om je devops-vaardigheden op los te laten.

## 3.3 Inzage

Inzage is conform de regels in de hogeschoolgids op het geroosterde inzagemoment. Geef tijdig aan de modulebeheerder door dat je inzage wenst. Dan kan je materiaal op tijd verzameld worden.

**Bijlage 1. Toetsmatrijs**

|  |  |  |
| --- | --- | --- |
|  | **Leerdoelen** | **Vraag** |
|  | Je kunt een geautomatiseerde software deployment ontwikkelen | Opdracht C |
|  | Je kunt continuous integration in de praktijk brengen | Opdracht B, C |
|  | Je kunt een softwareontwikkelingsomgeving voor een team inrichten | Opdracht A, B |
|  | Je bent bekend met industriestandaard tools en technieken voor software deployment | Opdracht D |
|  | Je hebt een basisvaardigheid in serverbeheer en kan hiermee geautomatiseerd onderhoud aansturen | Opdracht A |

# Practicumopdracht

## Opdracht A – Beheer

Start en voortgang tonen bij les 1

Inleveren bij aanvang van les 2.

**Doel**: Je maakt jezelf vaardig in beheer van servers. Deze is nodig om volgende opdrachten te maken. Maak een script waarmee jij je software server-omgeving netjes aanmaakt.

In project 5/6 ben je ermee bezig een software te ontwerpen en implementeren. Je implementatie is afhankelijk of gepasseerd op een aantal software pakketten en servers. In opdracht A maak je een inventarisatie van de benodigde software en servers (minimaal 2 software/servers), zodat deze daarna via een shell script geïnstalleerd en geconfigureerd worden. Gebruik Linux als OS. In vele gevallen wordt software op een Linux system met een root account geïnstalleerd. Dat is niet veilig!! Alle software moet beperkte rechten hebben op een server. Je beoogde server infrastructuur moet - zover mogelijk - versie onafhankelijk en gestandaardiseerd zijn voor alle ontwikkelaars in je projectteam. Je shell script zorgt voor een vereenvoudig installatie en configuratie voor alle ontwikkelaars.

Licht je script toe in een Word-verslag en lever dit in via Natschool in de inlevermapje voor opdracht A. Je mag screenshots maken ter verduidelijking.

**Toelichting bij matig.**

Je software komt dadelijk online op een server te draaien. Daar moet je voorbereidingen voor treffen. Je maakt user-sandboxes. Bijvoorbeeld voor een python webserver met mongodb, moet je in ieder geval een user aanmaken die uiteindelijk het python run-commando gaat uitvoeren én een aparte user voor de databaseserver. Maak eerst de user aan en installeer dan pas alles. Wissel van account met su. Met ls l kan je zien welke mappen bij welke users/groepen horen. Download software zoals de python runtime via apt-get commando’s. Maak via symbolic links snelkoppelingen in je home directory naar plaatsen waar je frequent instellingen moet wijzigen.

**Toelichting bij voldoende.**

Maak een overzicht naar alle belangrijke directories (vergeet system, db, python logs niet) zodat deze makkelijk op 1 plaats staan doormiddel van symbolic links. Zorg ervoor dat je je installatie kunt testen dmv bv een hello-world applicatie. Je kunt je volledige huidige set-up (inclusief configuratie) in een archiefbestand plaatsen. Je server werkt na een herstart weer, zorg er dus voor dat je server weer opstart na een reboot.

**Toelichting bij goed.**

Veel software wordt geinstalleerd met hun versienummer zoals /bin/python**2.7**/. Dat is voor jouw software onhandig. Je bent afhankelijk van configuratie en gevoelig voor updates. Dmv symbolic links kan je bv een /bin/python-current/ maken die verwijst naar de echte omgeving. Op deze manier overleeft je software systeemupdates en zijn ze in het geval van errors makkelijk te herstellen. Je kan je script bijstellen adv inputargumenten bv –archivename backup waardoor hij bij 11 je filename veranderd adv je argument. Je laat dmv bv een cronjob elke nacht een script lopen om je logs in te pakken en oude logs te verwijderen.

**Toelichting bij uitstekend.**

Er kan altijd iets mis gaat. Rechten ontbreken, users of mappen bestaan reeds. Dmv checks en condities kan jouw script daar rekening mee houden. Kijk ook op: <http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO-6.html>

Zie hieronder het beoordelingsschema.

Beoordelingsschema en criteria staan in de onderstaande tabel. Let op, deze zijn opbouwend. Een hoog cijfer kan alleen worden behaald als ook de lagere critieria zijn behaald. Je moet het bewijs paraat hebben in de les als de docent hierom vraagt (bij aanvang van de les gereed). Bereid dit dus voor! Het resultaat moet verwerkt zijn in 1 shell-script.

|  |  |
| --- | --- |
| **Beoordeling** | **Cijfer** |
| Onvoldoende   1. Niet voldaan aan onderstaande eisen 2. Te laat ingeleverd of in slechte staat (voorblad, spelling, grammatica, stijl) 3. Het script wordt niet (geheel) uitgevoerd door errors | 1 |
| Matig   1. Je hebt een script gemaakt dat diverse mappen en files aanmaakt. 2. Je hebt voor elk stukje server-software [[1]](#footnote-1)een nieuwe user aangemaakt. 3. Je hebt software (automatisch bv via apt-get silent) geïnstalleerd via je script. 4. De nieuwe user wordt eigenaar van de mappen. 5. Je hebt een symbolische link tussen twee locaties aangemaakt. | 4 |
| Voldoende   1. Je maakt je logs makkelijk beschikbaar op één plek via symbolic links. 2. Je hebt voorbeeldsoftware (zoals een webpagina) toegevoegd om te kunnen bevestigen dat de omgeving werkt. 3. Je kan de directory inpakken (archief maken). 4. Je server kan een herstart aan en komt daarna weer netjes live. | 6 |
| Goed   1. Je hebt versieonafhankelijkheid (bv met Java) bereikt via algemene symbolic links naar specifieke directories. 2. Het script werkt verschillend aan de hand van input-argumenten. 3. Je laat je scripts periodiek een check uitvoeren om ‘clean up’ te doen (bv oude logs). | 8 |
| Uitstekend   1. Het script kan omgaan met foutmeldingen (bv als een locatie niet bestaat). | 10 |

**Tip!** Leer jezelf Linux via:

* Video:
  + [http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=PracticalUnix](http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=PracticalUnix" \t "_blank)
* Shell scripts
  + <http://web.stanford.edu/~fringer/teaching/operating_systems_03/handouts/lecture4.pdf>
* Cheatsheet:
  + <http://www.cheat-sheets.org/saved-copy/ubunturef.pdf>

## Opdracht B – Software compositie - build & run (bv: Jenkins, Maven, Gradle, Ant)

Voortgang tonen bij les 3

Inleveren en beoordeling bij aanvang van les 4.

**Doel**: Je gaat aan de slag met het geautomatiseerd bouwen van software. Je gaat verder met het opzetten van je CD omgeving. Je richt het stuk van uitchecken tot packagen in via concrete stappen.

Beoordelingsschema en criteria staan in onderstaande tabel. Let op, deze zijn opbouwend. Een hoog cijfer kan alleen worden behaald als ook de lagere critieria zijn behaald. Je moet het bewijs paraat hebben in de les als de docent hierom vraagt. Bereid dit dus voor!

|  |  |
| --- | --- |
| **Beoordeling** | **Cijfer** |
| Onvoldoende   1. Niet voldaan aan onderstaande eisen. 2. Te laat ingeleverd. 3. Errors in het script of bouwproces. | 1 |
| Matig   1. Je hebt een shell script voor het uitchecken van software. 2. Je software dependencies worden automatisch bij elke build binnen gehaald. 3. Je kan de bouwstappen los uitvoeren. | 4 |
| Voldoende   1. Je hebt een repetitief bouwproces ingericht. 2. Je software voert zinnige software testen (voorkeur Unit Tests) uit. 3. Je hebt verschillende bouwstappen met tussenproducten. 4. Je hebt bovenstaande voor je één platform van INFPRJ56 opgezet. | 6 |
| Goed   1. Je voert code kwaliteitschecks uit (checkstyle, findbugs, pmd). 2. Je hebt configuraties parameters toegevoegd om verschillende builds mogelijk te maken (voeg bv een database locatie toe). | 8 |
| Uitstekend   1. Je gebruikt sonar om je code kwaliteit te waarboren. 2. Je doet bovenstaande voor meerdere platformen. | 10 |

**Integratie-omgevingen voorbeelden:**

* Jenkins / Hudson
* TeamCity [link](http://confluence.jetbrains.com/display/TCD7/Installing+and+Configuring+the+TeamCity+Server)
* Team Foundation Server
* Bamboo

**Bouwomgevingen voorbeelden:**

* Maven
* Gradle
* Ant

**Configuratie management voorbeelden:**

* Ansible
* Chef
* Puppet
* Salt

## Opdracht C – Release documentatie

Voortgang tonen bij les 5 Inleveren aan het eind van week 6.

**Doel**: Je gaat aan de slag met release strategie. Dit is voornamelijk in hoofdstuk 10 behandeld. Je beschrijft de strategie voor de release van de software van je project. Qua performance mag je je erop richten dat 1000 mensen simultaan het dashboard willen raadplegen. We streven naar een volwassen en veilige omgeving zonder downtime voor software-updates.

Beoordelingsschema en criteria staan in onderstaande tabel. Let op, deze zijn opbouwend. Een hoog cijfer kan alleen worden behaald als ook de lagere critieria zijn behaald.

Schema is gebaseerd op bladzijde 250-252 van het boek. Je hoeft het plan alleen te beschrijven (dus ga niet alle scripts maken). Deze opdracht mag je maken in duo’s. Vanuit de helderheid van het boek hebben we het expres niet vertaald.

**Let op!** Geef aan het begin van het verslag aan wie (van het duo) verantwoordelijk is voor welk onderdeel.

|  |  |
| --- | --- |
| **Beoordeling** | **Cijfer** |
| Onvoldoende   1. Niet voldaan aan onderstaande eisen 2. Te laat ingeleverd of in slechte staat (voorblad, spelling, grammatica, stijl) | 1 |
| Matig als er een beschrijving is van:   1. Stakeholders 2. Asset and configuration management strategy. 3. Description of the technology used for deployment. 4. Plan for implementing the deployment pipeline. 5. Een release plan | 4 |
| Voldoende als de kwaliteit van de beschrijving is voldoende als het aannemelijk is dat twee releases van dezelfde versie op gelijke wijze in productie komen. Er is daarnaast een beschrijving is van:   1. Enumeration of the environments available for acceptance, capacity, integration, and user acceptance testing, and the process by which builds will be moved through these environments. 2. How upgrades to the production environment will be handled, including data migration | 6 |
| Goed als de kwaliteit van de beschrijving voldoende is en als het aannemelijk is dat een nieuwe (aangepaste) release van software een uniform traject/process doorloopt. Er is daarnaast een beschrijving is van:   1. Description of the processes to be followed for deployment into testing and production environments, such as change requests to be opened and approvals that need to be granted. 2. A discussion of the method by which the application’s deploy-time and runtime configuration will be managed, and how this relates to the automated deployment process. | 8 |
| Uitstekend als de kwaliteit van de beschrijving een beheerst process voor uitrol, maar ook problemen beschrijft. Daarnaast is er een beschrijving van:   1. Requirements for monitoring the application, including any APIs or services the application should use to notify the operations team of its state. | 10 |

**Opdracht D – Test, release & deployment**

Voortgang tonen bij les 7

Inleveren en beoordeling bij aanvang van les 8.

**Doel**: Je gaat een geautomatisseerde release pipeline opstellen. Dit bevat de test, release en deployment van een artefact.

De opdracht is gebaseerd op hoofdstuk 8,9,10 van het boek.

Beoordelingsschema en criteria staan in onderstaande tabel. Let op, deze zijn opbouwend. Een hoog cijfer kan alleen worden behaald als ook de lagere critieria zijn behaald. Je moet het geheel demonstreren in de les als de docent hierom vraagt. Bereid dit dus voor!

|  |  |
| --- | --- |
| **Beoordeling** | **Cijfer** |
| Onvoldoende   1. Niet voldaan aan onderstaande eisen 2. Te laat ingeleverd of in slechte staat (voorblad, spelling, grammatica, stijl) 3. Errors in de pipeline | 1 |
| Matig   1. Er is een release pipeline met stappen build, test, release, deployment. | 4 |
| Voldoende   1. Er zijn een smoke-tests voor de verschillende omgevingen 2. Er wordt documentatie/rapportage gegenereerd (bv: javadoc, test rapport, code analyse, etc) 3. Er wordt on-the-fly een test of serveromgeving opgezet mbv Docker (of vergelijkbaar). 4. Een nieuwe software versie (Git push) triggert een nieuwe build. | 6 |
| Goed   1. Je gebruikt servermanagement software (zoals onderstaand) om je omgevingen in te richten. 2. Je hebt meerdere VM’s voor je software ingericht die met elkaar communiceren. | 8 |
| Uitstekend   1. Er zijn testen voor meerdere fasen (develop, test, UAC). Bijvoorbeeld ook een front-end test voor een website. | 10 |

**Configuratie management voorbeelden:**

* Ansible
* Chef
* Puppet
* Salt

**Infrastructure management voorbeelden**

* Docker
* Vagrant

1. Software waar een internetverbinding naartoe gaat (zoals een web-server). [↑](#footnote-ref-1)