HO
GENT

Faculteit Bedrijf en Organisatie

Best practices om continuous integration en continuous delivery uit te rollen in een SAPUI5 webapplicatie op SAP Cloud Platform

Renaat Haleydt

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Harm De Weirdt
Co-promotor:
Pieter-Jan Deraedt

Instelling: Amista

Academiejaar: 2018-2019

Tweede examenperiode

Faculteit Bedrijf en Organisatie

Best practices om continuous integration en continuous delivery uit te rollen in een SAPUI5 webapplicatie op SAP Cloud Platform

Renaat Haleydt

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Harm De Weirdt
Co-promotor:
Pieter-Jan Deraedt

Instelling: Amista

Academiejaar: 2018-2019

Tweede examenperiode

Woord vooraf

Samenvatting

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

1. Inleiding

IT-bedrijven willen hun klanten sneller te hulp kunnen schieten, betere software leveren, maar worstelen nog vaak met deadlines die verstrijken en veel problemen tijdens de oplevering van software. Gelukkig is er doorheen de jaren al veel progressie gemaakt door principes zoals DevOps bijvoorbeeld toe te passen, een mindset om de samenwerking tussen alle afdelingen binnen een it-bedrijf vlotter te maken. Een Continuous Integration en Continuous Delivery (CI/CD) pipeline opzetten is een onderdeel van DevOps en zou de werking van een bedrijf ten goede komen. Een CI/CD pipeline zorgt voor de automatisatie van testen, builds en deployment en heeft als grote voordeel dat er sneller wijzigingen doorgevoerd kunnen worden.

Amista is een groeiend bedrijf binnen IT consultancy. Het is een dochterbedrijf van Boutique dat zich toespitst op SAP. Sinds hun vijfjarig bestaan werken er al 60 personen voor Amista. Ze hebben Infrabel, Alcopa, Danone, AG Real Estate en nog anderen tot hun cliënteel, zoals op hun site te lezen valt (**Amista2018**). Ze zijn in twee landen actief, Frankrijk en België, maar werken ook samen met mensen uit India. Amista heeft zich verdiept in de Sales, Marketing en Service Management takken van bedrijven. Ze helpen hun klanten ook op gebied van Innovation, Integration, HCM Services (implementatie van succesfactoren binnen de HR afdeling) en Digital Learning. De missie van Amista is om samen met hun klanten innovatieve en kwalitatieve oplossingen aan te bieden met behulp van het volledige gamma dat SAP te bieden heeft. Omdat Amista graag de wensen van hun klanten zo goed mogelijk probeert te vervullen, proberen ze zelf zo innovatief mogelijk te zijn. Ze spelen met het idee om een CI/CD pipeline op te zetten voor de software development.

Om een Continuous Integration en Continuous Delivery pipeline op te stellen zijn er vandaag veel tools beschikbaar. Deze thesis zal verschillende build schedulers vergelijken

op basis van snelheid, hoeveelheid beschikbare documentatie en configureerbaarheid met de huidige set-up die Amista gebruikt. Aan de hand van de 'winnende' build scheduler wordt er een handleiding beschreven om een pipeline op te zetten binnen een SAPUI5 webapplicatie dat verbonden is met SAP HANA en gehost wordt op SAP Cloud Platform.

1.1 Probleemstelling

Amista heeft enkele klanten waar ze SAPUI5 webapplicaties voor maken, deze wordt vaak gecombineerd met een SAP HANA database dat gebruik maakt van Node.js en wordt allemaal gehost op SAP Cloud Platform. Voor projecten wordt er momenteel een datum afgesproken voor oplevering. Hier komt vaak veel stress en problemen bij kijken. Om dit te vermijden kan een Continuous Integration en Continuous Delivery pipeline helpen. Ook is het niet makkelijk om met de huidige gang van zaken wijzigingen van de klanten door te voeren. Dankzij een CI/CD pipeline kan men sneller wijzigingen doorvoeren en krijgt men sneller feedback van de klant. Om een CI/CD pipeline op te zetten binnen de hierboven beschreven omgeving is er nog niet veel informatie terug te vinden. Het probleem ligt hem in de combinatie van specifieke tools die gebruikt worden. Om te weten welke build scheduler het beste bij deze omgeving zou passen moeten er specifieke zaken onderzocht worden. Op internet is niet veel informatie terug te vinden welke build scheduler het beste zou passen binnen deze omgeving, ook niet om een CI/CD pipeline op te zetten. Daarom wil Amista heel graag een gepersonaliseerde vergelijking en handleiding om een pipeline op te stellen.

1.2 Onderzoeksvraag

- Wat zijn de voor- en nadelen van een CI/CD pipeline in het algemeen en specifiek voor Amista?
- Is het mogelijk om een Continuous Integration en Continuous Delivery pipeline te implementeren voor de ontwikkelingen van een SAPUI5 applicatie op SAP Cloud Platform?
- Welke tools moeten we gebruiken om een CI/CD pipeline op SAP Cloud Platform te implementeren als we vergelijken op snelheid, configureerbaarheid en hoeveelheid documentatie.
- Hoe kunnen we deze implementatie tot een succes brengen?

1.3 Onderzoeksdoelstelling

Deze thesis zou als hoofddoel een proof-of-concept zijn hoe een CI/CD pipeline te gebruiken in de dagelijkse werking van Amista. Maar deze thesis gaat ook op zoek naar de beste tools om de pipeline op te bouwen rekening houdend met snelheid, configureerbaarheid met de tools die Amista gebruikt en hoeveelheid documentatie er te vinden is.

1.4 Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:

In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.

In Hoofdstuk 4 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen.

De set-up van de omgeving wordt in detail besproken in Hoofdstuk 5.

Een vergelijking tussen de verschillende build schedulers kan men vinden in Hoofdstuk 6

In Hoofdstuk 7 wordt er een gepersonaliseerde handleiding gegeven om met de beste build scheduler een CI/CD pipeline te implementeren.

De waarom-vraag is minstens even belangrijk als de hoe-vraag. Concreet: wat zijn de voor- en nadelen van een CI/CD pipeline te integreren in het algemeen en specifiek voor Amista? Een antwoord op deze vragen kan je in Hoofdstuk 3.

In Hoofdstuk 8, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op de onderzoeksvragen. Daarbij wordt ook een aanzet gegeven voor toekomstig onderzoek binnen dit domein.

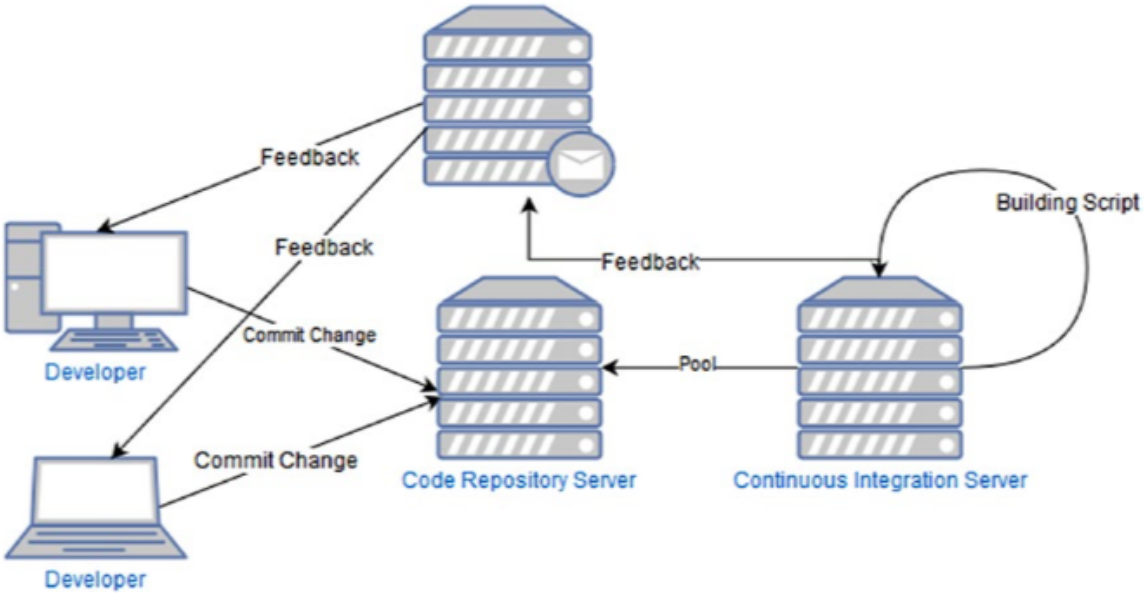
2. Stand van zaken

Bedrijven zijn constant op zoek naar betere en snellere resultaten. In het software development circuit is het vandaag soms nog lang wachten voor een wijziging effectief doorgevoerd wordt. Men levert nog vaak software op aan het einde van de sprint, wat soms voor problemen zorgt als men veel code tegelijk aflevert. Met een nieuwe software development methode - Continuous Integration en Continuous Delivery genaamd - wil men deze problemen zoveel mogelijk vermijden. Men spreekt vaak van een CI/CD pipeline als men het heeft over Continuous Integration en Continuous Delivery, maar er is nog een derde speler dat men kan invoeren: Continuous Deployment. Samen vormen zij de 3 musketiers om software projecten een grotere slaagkans te geven. Om een duidelijker beeld te scheppen van een CI/CD pipeline zal dit hoofdstuk eerst uitleg verschaffen over DevOps, Continuous Integration, Continuous Delivery en Continuous Deployment. Daarna wordt er kort in detail gegaan hoe het bedrijf Amista eruit ziet en met welke tools ze werken. Tot slot wordt er uitgelegd hoe de grote structuur van een CI/CD pipeline eruit ziet.

2.1 Continuous Integration, Continuous Delivery & Continuous Deployment

DevOps

DevOps is een samentrekking van development en operations en is een welbekend begrip binnen de informatica wereld. Het heeft als doel om de 'state of mind' binnen een bedrijf te veranderen zodat alle lagen/departementen vlotter samenwerken. Het is een praktische 'gids' dat bedrijven kunnen gebruiken om de communicatie tussen developers en systeembeheerders beter te maken. Deze twee verschillende lagen in een bedrijf willen namelijk



Figuur 2.1: Voorbeeld van een Continuous Integration set-up (**Riti2018**)

hetzelfde: zo snel mogelijk kwaliteitsvolle software opleveren. DevOps is gebaseerd op Agile development, maar gaat verder dan dat. Het gaat dieper in op automatisatie, integratie, samenwerking en communicatie. Continuous Integration, Delivery en Deployment zijn kenmerkend voor DevOps, omdat het mee inzet op snellere oplevering van kwaliteitsvolle software. (**Riti2018**)

Continuous Integration

Dit is een eerste stap in de pipeline waarbij de geschreven code wordt gecommit naar een 'repository management server'. Hierdoor wordt de 'source code' automatisch opnieuw opgebouwd en slaagt voor de testen die automatisch zijn opgestart. De focus bij deze stap ligt bij de teamleden (**Fowler2006**), van hen wordt verwacht dat ze -op een regelmatige basis- hun code pushen (integreren met de master applicatie). Een goede samenwerking tussen de verschillende leden van het development team is broodnodig om tot het gewenste eindresultaat te bekomen. Het doel van een Continuous Integration is om de integratie feilloos te laten verlopen wanneer men software ontwikkelt en geen functionaliteiten verliezen na een merge (**Riti2018**).
Bovenstaande uitleg is makkelijker te begrijpen aan de hand van een voorbeeld. Op Figuur 2.1 is een grafische voorstelling van dit voorbeeld terug te vinden. De developer commit code naar de repository die te vinden is op het source-control systeem. De Continuous Integration server krijgt het bericht dat er code is toegevoegd, haalt de laatst toegevoegde code op en laat de testen runnen. Wanneer alle testen slagen zal de CI server de code compilen en feedback bezorgen aan de developer. In dit voorbeeld maakt men gebruik van een externe mail server om die feedback te verzenden. Deze stappen gebeuren elke keer er code naar de repository gestuurd wordt.

Continuous Delivery

Eens het team met succes de Continuous Integration toepast kan men overschakelen naar de volgende stap: Continuous Delivery. Het is een manier dat ervoor zorgt dat de code die van de Continuous Integration stap komt gebuild wordt en voorbereidt wordt voor een release. Er is echter wel nog een menselijk hand nodig om de build van deze stap te deployen en voor de buitenwereld beschikbaar stellen (**Fowler2013**).

Continuous Deployment

De gelijkenis met Continuous Deployment is treffend, maar er is wel degelijk een verschil. Hier gaat men automatisch de veranderde code naar productie brengen. De veranderingen gaan door de volledige pipeline en eens ze slagen voor alle testen wordt - zonder menselijke interactie - de code naar productie gebracht (**Claps2015**). Dit wordt soms ook wel de 'train to production' genoemd, omdat elke code dat gepushed wordt naar de source-control automatisch tot bij de klant geraakt.

2.2 SAP

SAP is een Duitse onderneming opgericht in 1972 dat softwareoplossingen aanbiedt voor grote ondernemingen. Met meer dan 413.000 klanten verspreid over 180 landen mag SAP zich marktleider noemen op gebied van bedrijfssoftware. SAP heeft zicht gespecialiseerd in ERP, Enterprise Resource Planning software dat alle processen van het bedrijf automatiseert (**SAPERP2019**). Een ERP systeem beheert meerdere functies en bedrijfsprocessen van 1 bedrijf op basis van een centrale database. Het heeft als doel de gegevens van de organisatie optimaal te gebruiken in de gehele organisatie en een betere beheersing van de bedrijfsprocessen voorzien. De oplossingen dat SAP aanbiedt is vooral bedoeld voor de grotere bedrijven. Ze bieden software aan voor elke mogelijke industrie die er vandaag de dag bestaat, maar pakken vandaag uit met hun specialiteiten in de cloud business software. Hieronder worden kort de programma's beschreven die gebruikt worden in de voorbeeldapplicatie. Het geeft een grof beeld van de omgeving waar Amista graag een CI/CD pipeline wil voor integreren.

SAP Cloud Platform

SAP Cloud Platform is een platform as a service (PaaS), dat aangeboden wordt door SAP. het is een online platform dat - door hardware en software samen te brengen - applicaties overal toegankelijk maakt en samenbrengt tot 1 platform online (**SAPSE2018**). SAP Cloud Platform wordt zowel voor development als deployment gebruikt, maar reikt ook de hand aan verschillende technologieën: Internet of Things, big data, Artificiële Intelligentie enzovoort. Het is een platform dat zowel on-premise - waarbij software enkel lokaal op een computer beschikbaar is - als cloud technologieën samen kan brengen. Je kan er de technologieën ook uitbreiden en zelf ontwikkelen. Het haalt zijn kracht uit de perfecte integratie met andere SAP software die je ook nog eens kan uitbreiden.

SAPUI5

SAPUI5 is een framework dat uitgevonden is door SAP en bevat verschillende libraries die bovenop JavaScript gebouwd zijn. Men kan via het SAP Cloud Platform front-end applicaties maken en deployen die geschreven zijn in SAPUI5. Het is een framework dat bedoeld is om HTML5 applicaties te bouwen die bijna automatisch responsive zijn zonder veel bijkomende code toe te voegen. Het is bedoeld om dezelfde lay-out en hetzelfde gebruik voor de eindklant te garanderen. Het biedt aan de developers een resem aan UI controls aan zodat er een consistenter en beter UX design gehanteerd wordt. (**SAPSEa**)

SAP HANA

In-Memory Data Platform staat er er als titel op de site van SAP te lezen. Het is een platform dat gebruik maakt van het RAM geheugen van de computer, wat enorme snelheden met zich meebrengt, maar ook een enorm kostenplaatje. Dit even terzijde wordt SAP Hana aangeprezen als een platform om ingewikkelde, real-time analytische berekeningen uit te voeren op data. Het is een relationeel database management systeem (RDMBMS) dat geïntegreerd kan worden in SAP Cloud Platform, waarbij het mogelijk is om zowel on-premise als in de cloud te werken, of een combinatie van beiden.

3. Voor- en nadelen CI/CD pipeline

Dit gaat over de algemene voordelen van zo een pipeline, maar ook specifiek van Amista. Om een antwoord te krijgen op de algemene voordelen wordt er als basis teruggegrepen naar de literatuur. Er is namelijk al enorm veel geschreven over een CI/CD pipeline en wat de voor- en nadelen kunnen zijn. Bijkomstig is er een interview afgenomen met de expert omtrent DevOps, Patrick Debois. Het interview en de literatuur bieden samen de perfecte combinatie om de voordelen en nadelen te bespreken. Op de vraag wat de voor- en nadelen voor Amista zullen zijn kan enkel iemand van Amista op antwoorden natuurlijk. Er zijn vragen gesteld aan Pieter-Jan Deraedt om te kijken hoe Amista denkt te winnen bij het integreren van een CI/CD pipeline.

3.1 Voordelen

- Als men kleine stukjes code commit naar de centrale repository en de testen slagen niet, weet men dat het aan dit klein deeltje van de code ligt en kan men de fout veel sneller opsporen.
- In de meeste omgevingen met een CI/CD pipeline is het niet nodig om een QA team te hebben om de testen uit te voeren, wat de kosten doet dalen natuurlijk.

3.2 Nadelen

4. Methodologie

Deze thesis gaat op zoek naar de beste build scheduler voor de specifieke set-up die ze bij Amista hanteren: een SAPUI5 webapplicatie met een SAP HANA database draaiend via Node.js en alles gehost op SAP Cloud Platform.

4.1 Onderzoeksdomein 1: Voorbeeldapplicatie dat Amista zal gebruiken

Hoe ziet de omgeving er uit waar een CI/CD pipeline geïntegreerd moet worden er uit? In de literatuurstudie is de uitleg over de tools terug te vinden, maar hier worden de onderliggende relaties tussen de tools besproken. Er zal een test omgeving opgezet worden die hier besproken zal worden.

4.2 Onderzoeksdomein 2: Vergelijken van de beschikbare tools

Wat zijn voor Amista nu de beste tools om een CI/CD pipeline te integreren in hun software development? Welke tools scoren het beste op vlak van snelheid, configureerbaarheid met SAP en de tools die Amista gebruikt en documentatie die te vinden is online.

5. Voorbeeldapplicatie

5.1 Inleiding

6. Vergelijking van de tools

6.1 Tools voor een CI/CD pipeline

Er bestaan verschillende repository management services die een source control systeem hebben. GitHub, GitLab en Bitbucket zijn enkele voorbeelden uit het rijtje. Build schedulers zorgen ervoor dat de procedures worden samengesteld en dat de builds worden getriggerd. Voorbeelden hiervan zijn: Jenkins, Travis CI, GitLab-CI en Bamboo. Sonatype Nexus en Archiva zijn dan weer voorbeelden van tools die gebruikt worden als artifact repository manager, deze houden bij wijze van spreken de code bij die klaar is om te deployen.

Source Control System

Dit systeem houdt alle veranderingen bij aan de code en zal de veranderingen ook beheren zodat ze niet overlappen. Het laat toe dat meerdere developers tegelijk aan - soms dezelfde - code werken. Een source control systeem houdt dan de veranderingen bij wat elke developer gedaan heeft. Best practice is er maar 1 versie van de software die stabiel is, meestal master genoemd. Een gangbare praktijk binnen dit systeem is het maken van branches. Hier wordt een kopie gemaakt van de stabiele master, waardoor men wat kan knoeien in de branch zonder de master te beschadigen. Als men overtuigd is van de kwaliteiten dat men op een branch gemaakt heeft kan men de branch 'mergen' in de master. Deze techniek wordt ook wel 'revision control' of 'version control' genoemd (**Skelton2014**) en (**Riti2018**). Voorbeelden van zo een source control system zijn Git, CVS (Concurrent Version System), Subversion en Mercurial.

Repository Management Services

Wordt ook wel Code Repository Server genoemd. Hier wordt de software van het source control system opgeslagen. Dit kan op een interne server opgeslagen worden, of op een externe die door bedrijven aangeboden wordt. Dit maakt het makkelijker voor developers om samen te werken en dezelfde bron te gebruiken en is nodig om een goede Continuous Integration aan te bieden.

Build Scheduler

Een Build Scheduler kan ook wel een Continuous Integration Server genoemd worden. Dit zorgt ervoor dat - telkens wanneer er code gecommit wordt - de pipeline wordt uitgevoerd. De taken van de build scheduler zijn:

- Code ophalen van de Repository Server en deze samenvoegen met de oude code
- De testen uitvoeren
- Het builden van de software
- Feedback geven aan de developer over voorgaande stappen

Deze taken kunnen ook door een script volbracht worden, maar het is belangrijk dat deze taken automatisch gebeuren telkens er code gecommit wordt naar de repository manager (**Riti2018**).

Artifact Repository Manager

Als laatste zijn de Artifact Repository Managers aan de beurt. Deze repository manager houdt alles wat nodig is om de applicatie te deployen bij zoals

- packaged application code
- application assets
- infrastructure code
- virtual machine images
- configuration data

Deze tool houdt alle geschiedenis bij van de bovenstaande files. Zoals eerder vermeld kan men vanaf hier de software (automatisch) deployen. Dit is de allerlaatste fase in de CI/CD pipeline (**Skelton2014**).

7. Handleiding van een CI/CD pipeline

Voor Amista zou het een ideaal scenario zijn als ze een mooie 'handleiding' over de best practices om een CI/CD pipeline op te starten met de winnende tools uit Onderzoeksdomein 2. Zo heeft Amista een 'gepersonaliseerde tutorial' hoe een pipeline op te zetten met de tools die het beste matchen met de noden van het bedrijf zelf.

7.1 Continuous Delivery principles

7.2 CI/CD pipeline op SAP Cloud Platform

SAP Cloud Platform biedt de mogelijkheid om verschillende omgevingen op te stellen waarin je kan werken als developer. Het vergt enige vereisten om te voldoen aan de regels van Continuous Integration (**Kramer2018**):

- Hou alles goed bij via een version control systeem
- Automatiseer de build
- Zorg ervoor dat tijdens de build er Unit testen lopen
- Het team moet op regelmatige basis commits uitvoeren
- Elke verandering moet gebuild worden
- Als er errors tevoorschijn komen tijdens de build moeten die opgelost worden
- De build moet uitgetest worden op een kopie van de productieomgeving
- Automatiseer de deployment

Eens deze regels toegepast zijn kunnen we spreken van een CI implementatie. Vaak wordt CI in combinatie gebracht met Continuous Delivery. Om dit in een vloeiende lijn te laten

gaan spreekt men van een CI/CD pipeline.

7.3 CI/CD pipeline volgens SAP

SAP is een Duitse onderneming dat softwareoplossingen aanbiedt voor grote ondernemingen en heeft zicht gespecialiseerd in het maken van ERP pakketten. Dat is software dat alle processen van het bedrijf opneemt (**SAPERP2019**). Een programmeur schrijft nieuwe code voor een verandering die de klant wil uitvoeren. Idealiter zou dit - voor het mergen naar de masterapplication - eens door een voter build moeten gaan, waar automatische test aanwezig zijn die kijken of de code geen problemen zou geven als je die zou mergen met de master. Een laaste stap voor de code naar de master gemerged wordt, is het toepassen van code reviews door collega developers (het 4-ogen principe). Na het samenvoegen wordt automatisch de CI-build geactiveerd. De code gaat door de automatische tests. Eens de testen slagen worden de wijzigingen geïntegreerd op de master.

Dan komt de Continuous Delivery fase, waarbij de code nog eens door een test systeem gaat. Deze fase gebeurt allemaal automatisch, maar er kunnen ook manueel testen uitgevoerd worden. Eens de code door deze fase raakt is ze klaar om te deployen. Bij Continuous Deployment worden de wijzigingen dus automatisch naar buiten gebracht (**Kramer2018**).

8. Conclusie

A. Onderzoeksvoorstel

Het onderwerp van deze bachelorproef is gebaseerd op een onderzoeksvoorstel dat vooraf werd beoordeeld door de promotor. Dat voorstel is opgenomen in deze bijlage.

A.1 Introductie

Betere, redundante projecten opleveren, dit is waar ieder bedrijf naar streeft. Amista is een consultancy bedrijf dat op zoek gaat, samen met hun klanten, naar oplossingen binnen de wereld van SAP. Bij Amista weten ze heel goed waar de noden van hun klanten liggen en zouden hier dan ook graag op inspelen. Klanten willen namelijk de gewenste veranderingen onmiddellijk te zien krijgen, de wachttijd op een oplevering die ze willen doorvoeren moet minimaal blijven. Meestal werkt een heel team aan de oplevering van een project, dit leidt tot verschillende versies van de code. Men weet niet precies op welk punt de code 'echt werkt'. Continuous Integration en Continuous Delivery kan een hulpmiddel zijn om bij elke push werkende software te hebben. Er wordt namelijk gecontroleerd of de code voldoet aan criteria om het als 'werkende' te beschouwen (de code wordt groen verklaard).

Vandaag de dag verwachten klanten dat het programma blijft werken eens een oplevering doorgevoerd wordt (dit wordt vaak 0 downtime genoemd in het vakjargon). Een manier om deze nood op te vangen is het implementeren van Continuous Deployment.

Deze studie biedt een weg aan bedrijven die denken om een CI/CD pipeline te integreren op SAP Cloud Platform om te slagen in hun opdracht. Het gaat specifiek over SAPUI5 webapplicaties die gebruik maken van SAP HANA, NodeJS en HTML5. De best practices om zo een systeem uit te rollen worden besproken, alsook wat de voor- en nadelen zijn van

bepaalde frameworks/tools.

Onderzoeksvragen:

- Wat zijn de voor- en nadelen van een CI/CD pipeline te integreren in het algemeen en specifiek voor Amista?
- Is het mogelijk om op een eenvoudige manier een Continuous Integration en Continuous Delivery pipeline te implementeren voor de ontwikkelingen van een SAPUI5 applicatie op SAP Cloud Platform?
- Hoe kunnen we deze implementatie tot een succes brengen?
- Welke tools moeten we gebruiken om een CI/CD pipeline op SAP Cloud Platform te implementeren als we vergelijken op snelheid, configureerbaarheid, documentatie en kostprijs?

A.2 State-of-the-art

SAP Cloud Platform

SAP Cloud Platform is een Platform-as-a-service (PaaS), een platform dat - door hardware en software samen te brengen - applicaties overal toegankelijk maakt en samenbrengt tot 1 platform (**SAPSE2018**). SAP Cloud Platform is een development en deployment platform dat ook de hand reikt aan verschillende technologieën: Internet of Things, big data, Artificial Intelligence enzovoort. Het is een platform dat zowel on-premise als cloud technologieën samen kan brengen, die je kan uitbreiden en zelf kan ontwikkelen. Het haalt zijn kracht vooral uit de perfecte integratie van andere SAP software die je ook nog eens kan uitbreiden.

Continuous Integration Dit is een manier om software te maken waar de focus ligt bij de teamleden (**Fowler2006**). Zij worden verwacht hun geschreven code op regelmatige basis te integreren met de master applicatie. Op die manier gaat de code door een molen die een geautomatiseerde build zal uitvoeren en kijkt of de code slaagt voor Unit tests. Een best practice is om te testen in een kopie van de productieomgeving, zo heb je weinig risico op ongelukken. Deze techniek van implementeren brengt bepaalde voordelen met zich mee: er zijn minder problemen om de code te integreren op de master applicatie waardoor men sneller kan voldoen aan de eisen van de klant.

Continuous Delivery De code wordt getest de testen slagen wordt de code

afgeleverd in een formaat dat klaar is om te deployen. Bij deze stap is er een menselijke keuze nodig om de software naar de klant of gebruiker te brengen (**Fowler2013**).

Continuous Deployment
Dit is een manier om software uit te sturen en als klaar te beschouwen. Het is een mogelijkheid om alle soorten wijzigingen - inclusief nieuwe functies, configuratiewijzigingen, bugfixes en experimenten - op een veilige, snelle en automatische manier bij de klant of gebruiker te brengen. Deze manier van deployen is makkelijker om te voldoen aan de wijzigingen die moeten doorgevoerd worden. Er komt geen grote release bij te pas waar iedereen bang afwacht of de update stand houdt. Door telkens kleine veranderingen op een veilige manier door te voeren zorgt men ervoor dat de applicatie veel minder kans heeft op falen (**Claps2015**). Het is aan te raden dat wanneer je Continuous Deployment wil implementeren in je project, je eerst Continuous Integration en Continuous Delivery op punt moet zetten. Ook is het belangrijk om een goede flow van versie control te hebben binnen het team en je moet gebruik maken van een automated deploy script die je aan de build hangt .

CI/CD integreren op SAP Cloud Platform
SAP Cloud Platform biedt de mogelijkheid om verschillende omgevingen op te stellen waarin je kan werken als developer. Het vergt enige vereisten om te voldoen aan de regels van Continuous Integration (**Kramer2018**):

- Hou alles goed bij via een version control systeem
- Automatiseer de build
- Zorg ervoor dat tijdens de build er Unit testen lopen
- Het team moet op regelmatige basis commits uitvoeren
- Elke verandering moet gebuild worden
- Als er errors tevoorschijn komen tijdens de build moeten die opgelost worden
- De build moet uitgetest worden op een kopie van de productieomgeving
- Automatiseer de deployment

Eens deze regels toegepast zijn kunnen we spreken van een CI implementatie. Vaak wordt CI in combinatie gebracht met Continuous Delivery. Om dit in een vloeiende lijn te laten gaan spreekt men van een CI/CD pipeline.

CI/CD pipeline volgens SAP
SAP is een Duitse onderneming dat softwareoplossingen aanbiedt voor grote ondernemingen en heeft zicht gespecialiseerd in het maken van ERP pakketten. Dat is software dat alle processen van het bedrijf opneemt (**SAPERP2019**). Een programmeur schrijft nieuwe code voor een verandering die de klant wil uitvoeren. Idealiter zou dit - voor het mergen naar de masterapplication - eens door een voter build moeten gaan, waar automatische test aanwezig zijn die kijken of de code geen problemen zou geven als je die zou mergen met de master. Een laaste stap voor de code naar de master gemerged wordt, is het toepassen van code reviews door collega developers (het 4-ogen principe). Na het samenvoegen wordt automatisch de CI-build geactiveerd. De code gaat door de automatische tests. Eens de testen slagen worden de wijzigingen geïntegreerd op de master.

Dan komt de Continuous Delivery fase, waarbij de code nog eens door een test systeem gaat. Deze fase gebeurt allemaal automatisch, maar er kunnen ook manueel testen uitgevoerd worden. Eens de code door deze fase raakt is ze klaar om te deployen. Bij Continuous Deployment worden de wijzigingen dus automatisch naar buiten gebracht (**Kramer2018**).

Tools die gebruikt kunnen worden om een CI/CD pipeline te implementeren
Er bestaan verschillende source code repositories waar je de versies van je code kan beheren. GitHub, Git, GitLab, Bitbucket zijn voorbeelden van zo een tools. Build schedulers zorgen ervoor dat de procedures worden samengesteld en dat de builds worden getriggerd. Voorbeelden hiervan zijn: Jenkins, Travis CI, GitLab-CI en Bamboo. Sonatype Nexus en Archiva zijn voorbeelden van tools die gebruikt worden als repository manager, deze houden bij wijze van spreken de code bij die klaar is om te deployen.

A.3 Methodologie

Eerst worden de voor- en nadelen van het integreren van een CI/CD pipeline in een SAPUI5 applicatie uitgeschreven. Deze studie zal de voor- en nadelen van de verschillende tools onderzoeken op vlak van snelheid, configureerbaarheid met SAP en de tools die Amista gebruikt, documentatie die te vinden is online en de kostprijs. Er wordt een voorbeeldapplicatie

gemaakt die zal helpen bij het uitschrijven van de best practices om een CI/CD pipeline uit te werken aan de hand van de tools die gekozen werden. De voorbeeldapplicatie wordt een omgeving waar het mogelijk is om kleine deeltjes code te wijzigen en die dan testen of ze klaar zijn om te deployen.

A.4 Verwachte resultaten

Uit onderzoek zal blijken dat de nadelen niet zullen opwegen tegen de vele voordelen die een CI/CD pipeline te bieden heeft. Amista maakt reeds gebruik van Git als versiebeheersysteem, dit zal ongetwijfeld doorwegen op de keuze. Ook het feit dat Git open source en dus helemaal gratis te gebruiken is heeft zijn voordelen. GitHub, GitLab en Bitbucket zijn allemaal een online repository management systeem om Git projecten te beheren. GitLab is open source, maar er bestaat een formule waar een onderneming moet betalen voor de diensten en server beheer. Als een onderneming gebruik wil maken van GitHub en Bitbucket zal ze geld op tafel moeten leggen. Wanneer we kijken naar de samenwerking met build schedulers, zien we dat de online repository management systemen hun eigen tool hebben. Zo heeft GitLab een eigen gemaakte Continuous Integration tool, GitLab CI genaamd. Deze tool is gratis te gebruiken tot 2000 minuten per maand (**GitLabPricing2019**). Wanneer er gebruik gemaakt wordt van Bitbucket zal hoogst waarschijnlijk Bamboo gebruiken, deze twee tools komen van hetzelfde bedrijf en werken bijgevolg naadloos samen. Wanneer men GitHub gebruikt is het mogelijk om Travis CI te implementeren. Jenkins heeft dan weer plug-ins ter beschikking waarbij de samenwerking met de bovenstaande online repository management systemen verzekerd is.
Als we kijken naar de build schedulers zien we dat Jenkins een grote community heeft, het brengt geen kosten met zich mee (want het is open source) en biedt vele plug-ins aan om combinatie met andere tools makkelijk te laten verlopen. Sonatype Nexus biedt de mogelijkheid aan om te kiezen tussen de open source versie of de professionele versie die minstens $10/maand kost (**SonatypeNexusPricing2019**). Het verschil zit hem in de support die Sonatype biedt (**OBrien2010**). Apache Archiva is ook open source, maar daar is minder informatie over te vinden. De community is niet zo groot als bij Nexus.
In dit onderzoek wordt er ook een koppeling van de gevonden theorie aan

een klein praktijkvoorbeeld gemaakt. Dit aan de hand van best practices om een Continuous Integration/Continuous Delivery pipeline op te zetten en een soort gids om die best practices toe te passen op de omgeving waar Amista mee werkt.

A.5 Verwachte conclusies

Als onderneming heeft het zeker voordelen om een CI/CD pipeline te integreren. Gaande van onmiddellijke feedback op geschreven code van developers, betere implementatie, 0 downtime. Het grootste nadeel zal de tijd van implementatie zijn, maar ook de kostprijs. De integratie en onderhoud van zo een infrastructuur brengt heel wat aanpassingen met zich mee die geld kosten. Er moeten ook veel test geschreven worden om een goede pipeline te bekomen. Het zal mogelijk zijn dat bedrijven, mits een goede handleiding van best practices bij de hand, een succesvolle CI/CD pipeline kunnen integreren. Git, Jenkins en Nexus zullen als winnaars uit de bus komen om zo de pipeline te maken. Rekening houdend met de kosten en de configureerbaarheid met de tools die Amista gebruikt en beschikbare documentatie.