Onderzoeksvoorstel

Onderzoek naar de compilatie en bind van een coolgen programma via een Azure DevOps pipeline binnen de mainframe omgeving van ArcelorMittal Gent met proof of concept.

Bachelorproef, 2023-2024

Dylan Vermeersch

E-mail: dylan.vermeersch@student.hogent.be
Co-promotor: D. Marichal (ArcelorMittal Gent, didier.marichal@arcelormittal.com)

Samenvatting

Dit onderzoek zal gaan over het uitwerken van een werkende Azure DevOps pipeline om coolgen programma's te compileren en binden met behulp van IBM Dependency Based Build (DBB) en zijn ingebouwd framework. Er werd gezocht naar een oplossing om de mainframe omgeving van ArcelorMittal Gent te moderniseren om zo aantrekkelijker te zijn voor afgestudeerden en om gebruik te maken van de nieuwere technologieën binnen het mainframe landschap. In de proof of concept is er een proefopstelling opgezet zodat de coolgen applicaties kunnen worden gecompileerd en gebind via een Azure pipeline. Op die manier hoeft de ontwikkelaar niet zelf de compilatie en/of bind te starten. Het verwachte resultaat is dat de pipeline een correcte compilatie en bind kan uitvoeren zonder dat de ontwikkelaar zelf iets moet uitvoeren op de mainframe omgeving en dat het versiebeheer volledig kan beheerd worden door Azure DevOps. Zo zal er een einde komen aan de vele stappen die nodig zijn om een compilatie en bind van een coolgen programma uit te voeren ook zal er voortaan in een Git ondersteunende IDE gewerkt kunnen worden.

Keuzerichting: Mainframe Expert
Sleutelwoorden: coolgen, DevOps, pipeline

Inhoudsopgave

1. Introductie

Mainframe is een van de oudste en meest gebruikte computertechnologieën ooit, maar na al die jaren dat er getwijfeld werd over het wel of niet afschaffen van de technologie. Is er een groot tekort aan pas afgestudeerden ontstaan, die verse krachten zouden het voortouw kunnen nemen in de vernieuwing van de mainframe. Volgens Broadcom (2024) is de mainframe meer dan ooit toegankelijk en compatibel in projecten die zich niet enkel op het mainframe platform situeren. Dit komt volgens hun door de mogelijkheid om de DevOps workflow toe te passen waardoor de flexibiliteit en automatisatie mogelijkheden toeneemt. Verder maakt het ook de samenwerking met teamleden, zowel binnen een mainframe team als er buiten, makkelijker en meer gestroomlijnd. Hierdoor is IBM Dependency Based Build een goede manier om een mainframe te moderniseren omdat deze tool van IBM ervoor zorgt dat je op een DevOps manier kan werken met een mainframe. Deze workflow bestaat uit een pipeline en een repository, in het geval van dit onderzoek zal de pipeline(s) verzorgd worden door Azure Pipelines en de repositories zullen aangeboden worden via Azure Repos. Beide services maken deel uit van het Azure DevOps pakket dat nog heel vaak zal vermeld worden doorheen het onderzoek en de proof of concept.

De leveranciers van mainframe software hebben het belang van modernisering in het landschap opgemerkt en zijn dus al enkele jaren volop tijd en geld aan het pompen in nieuwe gebruiksvriendelijke en meer hedendaagse tools. Die tools zouden het werken op zo'n omgeving vereenvoudigen voor zowel gebruikers, ontwikkelaars en administrators.

Dit onderzoek werd in samenspraak met het mainframe team van ArcelorMittal Gent opgestart om een efficiëntere manier te vinden om de compilatie en bind uit te voeren van een coolgen programma. Het onderzoek zal vooral impact hebben bij de mainframe system administrators en bij uitbreiding ook de ontwikkelaars van coolgen programma's in ArcelorMittal Gent.

Het doel van dit onderzoek is dan ook om een coolgen programma, gemaakt met een moderne Git ondersteunende Integrated Development Environment, kortweg IDE, zoals bijvoorbeeld Visual Studio Code. Volledig automatisch te kunnen compileren en binden op het moment dat er wijzigingen opgeslagen worden naar de Azure Repo, dit zal door een Azure Pipeline in gang gezet worden. Hierdoor wordt het bestaande proces dat een aanzienlijk aantal stappen extra heeft vereenvoudigd tot één stap, namelijk de applicatie opslaan en pushen naar een Azure Repo.

Het onderzoek zal als een succes worden beschouwd indien de proof of concept een positief resultaat geeft, dit wil zeggen dat er moet aangetoond worden dat het mogelijk is om een coolgen programma automatisch te laten compileren en binden door een Azure Pipeline met behulp van IBM DBB. Verder moet ook aangetoond worden dat er versiebeheer mogelijk is met behulp van Azure Repos om zo naar verschillende versies van programma's terug te kunnen keren. Indien beide vereisten worden ingelost kan er gesproken worden van een succesvol onderzoek en bijgevolg een succesvolle proof of concept.

2. State-of-the-art

Vroeger waren de industrieel gestandaardiseerde servers gestapeld tot aan het plafond in elk datacenter. In die tijd was de mainframe computer een dinosaurus die gedoemd was om uit te sterven. IBM zag toch nog een toekomst voor de mainframe, hierdoor bleven ze maar innoveren op het 'Z' platform. Ze werden ondersteund door de opkomst van hybride cloud modellen en vele bedrijfskritische applicaties die op het mainframe moesten blijven. (Moorhead, 2022)

Deze innovaties waren er niet geweest zonder de vastberadenheid van IBM om het platform in leven te houden en sterker nog het nog levendiger te maken dan het voordien was. Door met name de IBM Wazi Developer for Red Hat CodeReady Workspaces uit te rollen hebben ze een cloudnative ontwikkel ervaring voor het z/OS besturingssysteem gemaakt. Met Wazi kunnen ontwikkelaars hun Integrated Development Environment, kortweg IDE, naar keuze gebruiken om mainframe toepassingen te ontwikkelen op het Red Hat OpenShift Container Platform. Een ander gebied van innovatie voor IBM is het bieden van beveiliging voor commerciële cryptocurrency wallets. (Bloomberg, 2021)

Een van deze hedendaagse innovaties is IBM Dependency Based Build, IBM DBB werkt aan de hand van een aantal groovy scripts die de verschillende commando's voor bijvoorbeeld een compilatie van een programma op z/OS uitvoeren. Deze scripts zijn volledig aanpasbaar waardoor ze een heel hoge mate van personalisatie toelaten, dit is een grote troef vooral in de mainframe omgeving van bedrijven die vaak heel hard aangepast zijn aan de werkwijze of policy's van dat bedrijf. Volgens Porter (2019) is het leren werken met z/OS applicaties via groovy scripts en een moderne IDE zoals Visual Studio Code of IBM Developer for z/OS een goede manier om te leren omgaan met een mainframe zonder zelf op het mainframe zaken uit te voeren in het 3270 scherm. Dit zorgt ervoor dat vooral nieuwe ontwikkelaars die hun kans wagen in mainframe applicaties niet overweldigd worden door dat 3270 scherm en ze dus op een gekende/moderne manier kunnen kennismaken met een mainframe om dan eventueel later de stap te zetten naar werken met een green screen op hun eigen tempo. (Porter, 2019)

Om het uitrollen van applicaties met behulp van IBM DBB mogelijk te maken kan er gekeken worden naar een DevOps oplossing. Volgens Daniel Sokolowski (2021) verenigt DevOps software development en operations in cross-functional teams om zo de software delivery en operations te verbeteren. Hij zegt dat in een ideaal scenario de cross-functional DevOps teams hun services onafhankelijk uitrollen naar productie. In de praktijk is het vaak zo dat de correcte manier van het uitrollen van een service andere services nodig heeft en dus zo nood heeft aan coördinatie om zo op een correcte manier hun service te kunnen uitrollen naar productie. Vaak wordt dit probleem opgelost door teams die een communicatie binnenkrijgen en dan manueel de deployment uitvoeren alsook de benodigde coördinatie moeten bepalen. Dit zorgt er dus voor dat de teams niet meer onafhankelijk zijn van elkaar en dus niet het ideale scenario is wat dus niet ten goede komt van de software deployment en operations hun performantie. Een oplossing voor dit probleem is een geautomatiseerd systeem die de coördinatie en uitvoering voor zijn rekening kan nemen, dit is wat men verstaat onder een CI/CD pipeline. Dit is een constructie dat is bedoeld om de software development en operations terug onafhankelijk te maken van elkaar en zo de performantie ervan te verbeteren door een geautomatiseerde oplossing te bieden voor de coördinatie en uitvoering voor het uitrollen van een service. (Daniel Sokolowski, 2021)

Naast de huidige innovaties is het ook wel interessant om te kijken naar hoe het landschap er over pakweg 10 jaar zou kunnen uitzien. Volgens Christopher Tozzi (2022) zal de mainframe nog alom tegenwoordig zijn na die 10 jaar, ook zullen er nog meer innovaties zijn gemaakt om het integreren

met andere platformen zoals Windows en Linux of zelfs andere programmeertalen zoals Python, Javascript, C en vele anderen mogelijk te maken. Verder zullen de 'oude' technologieën van de COBOL programmeertaal blijven gemoderniseerd worden zodat het nog performanter wordt en nog makkelijker zal worden om COBOL-code te hergebruiken. De fysische voetafdruk van een mainframe zal ook alsmaar kleiner worden, nu is een moderne mainframe ongeveer even groot als een koelkast maar dat zou in de toekomst nog vele malen kleiner worden volgens Tozzi. (Tozzi, 2022)

3. Methodologie

De methodologie bestaat uit een aantal fases. In de eerste fase van dit onderzoek wordt er een uitgebreide literatuurstudie gemaakt door middel van literatuuronderzoek. Deze literatuurstudie zal gebruikt worden als voornaamste gegevensbron voor het verdere verloop van het onderzoek. De literatuurstudie heeft vier grote onderwerpen namelijk de mainframe, IBM Dependency Based Build, Git workflows en Azure DevOps. In het hoofdstuk omtrent de mainframe zal er gekeken worden wat een mainframe is, waarvoor het gebruikt wordt en wat de geschiedenis ervan is.

In het stuk over IBM Dependency Based Build zal er beschreven worden wat IBM DBB is, wat het doet en hoe het werkt. Verder zal er ook te vinden zijn wat een aantal van de valkuilen kunnen zijn waarmee rekening gehouden moet worden.

Het voorlaatste deel van de literatuurstudie zal een aantal Git workflows toelichten en onderzoeken, hierin zal er gezocht worden naar een workflow die zowel flexibel, veilig en robuust is. Allemaal eigenschappen die synoniem staan aan een mainframe dus dat moet ook getoond worden dat dat nog altijd mogelijk is met een Git workflow.

In het laatste deel van de literatuurstudie zal er beschreven worden wat Azure DevOps is, wat het kan doen en waarvoor het in het onderzoek zal gebruikt worden. Een aantal onderwerpen die aan bod zullen komen in dit hoofstuk zullen betrekking hebben op onderdelen van de Azure DevOps suite meerbepaald Azure Repos en Azure Pipelines. Waarvoor die zaken in het onderzoek gebruikt kunnen worden en wat voor meerwaarde dat geeft aan het onderzoek.

Deze eerste fase die geschat wordt op anderhalve week heeft als resultaat een diepgaande en volledige literatuurstudie die de rode draad zal zijn wat betreft informatie in het onderzoek. De literatuurstudie moet antwoord kunnen bieden op de volgende vraag. Hoe kan er een DevOps workflow gerealiseerd worden op een mainframe met behulp van IBM Dependency Based Build?

Het doel van de volgende fase is om IBM Dependency Based Build volledig te configureren op de omgeving van ArcelorMittal Gent, zo zal er onder andere een aantal .properties bestanden moeten aangemaakt en aangevuld worden en indien nodig zullen er zelf properties moeten toegevoegd worden aan deze bestanden. Door deze properties aan te passen zal IBM DBB de volledige toegang hebben tot de nodige datasets en bestanden op zowel z/OS als op Unix System Services (USS) om een goede werking van de scripts horende bij de software te kunnen garanderen. Dit zal gebeuren aan de hand van een vooraf bepaald applicatie die gebruikt zal worden als case study om zo de goede properties en configuraties te vinden. Dit vergt wat opzoekwerk en wordt geschat op 1 week om de volledige configuratie van IBM DBB te vervolledigen.

Na de configuratie van IBM DBB kan er overgegaan worden naar de grootste fase van het onderzoek namelijk het uitzetten en realiseren van de proof of concept. Deze fase is een heel belangrijke omwille van het feit dat hier de resultaten worden bekomen om een conclusie te trekken op het einde van dit onderzoek. De opdracht bestaat erin om de volledige ontwikkeling van een vooraf gekozen coolgen applicatie van het mainframe te halen als case study, dat wil zeggen dat de source aangemaakt zal worden in een Git ondersteunende IDE zoals bijvoorbeeld Visual Studio Code of IBM Developer for Z (IDz) en vanuit diezelfde IDE een pipeline gestart wordt om het programma dan te compileren, binden en te plaatsen in de juiste load libraries op z/OS. Dit zal onder meer gebeuren via een push vanuit de IDE naar de Azure Repo die dan het aangepast programma opslaat en daarna een pipeline start om het aangepaste programma ook te compileren en te binden en met als doel het plaatsen van de bekomen load module na de bind stap in de juiste load library te plaatsen. Verder moet er ook bij elke succesvolle aanpassing van het programma ook een nieuwe versie ontstaan zo zal een programma CGTST met initiële versie 1.0 na een succesvolle aanpassing bijvoorbeeld de versie 1.1 krijgen toegewezen. Zodat er op een duidelijke en overzichtelijke manier kan bijgehouden worden wat de huidige versie is van een programma, de vorige versies en eventueel de mogelijkheid om naar een vorige versie terug te gaan. Na deze fase die zes en een halve week zal duren zal er genoeg informatie zijn om een goede en volledige conclusie te trekken op de vraag of het zin heeft om volledig over te schakelen naar een DevOps geba-

seerde manier van werken binnen de mainframe omgeving van ArcelorMittal Gent.

De volgende fase van dit onderzoek is om de resultaten die doorheen dit onderzoek zijn bekomen te analyseren. Deze analyse zal in samenspraak gebeuren met ArcelorMittal en een aantal ontwikkelaars om ook te polsen naar hun mening over het onderzoek en de resultaten ervan. De fase waarin deze analyse gebeurt zal 1 week in beslag nemen.

Na de resultatenanalyse volgt de fase waarin er conclusies worden getrokken, dit zal afhankelijk zijn van hoeveel vereisten er daadwerkelijk zijn bereikt en aan welke eventueel niet werd voldaan. Vereisten die zeker aan bod zullen komen in deze fase zijn als volgt.

- Kan het programma gecompileerd worden?
- Kan het programma gebind worden?
- Is de compilatie volgens de juiste compilatie parameters gebeurd?
- Is de bind volgens de juiste bind parameters gebeurd?
- Wordt het resultaat van de compilatie/bind op de juiste plaats en manier opgeslagen?
- Is het proces automatisch of moet er nog ergens manueel ingegrepen worden?

Voor deze fase zal er een focusgroep worden gemaakt om de vereisten en het al dan niet behalen ervan te bespreken en meningen uit te wisselen met mensen binnen de mainframe omgeving van ArcelorMittal Gent. Hiervoor wordt opnieuw 1 week gerekend.

De laatste fase is om de scriptie af te werken, zo zal er in die week gekeken worden om alle puntjes op de i te zetten en alles nog eens goed na te kijken alvorens in te dienen. Deze laatste fase zal niet langer dan 1 week duren en zal ook het einde van dit onderzoek inluiden.

4. Verwacht resultaat, conclusie

Aan de hand van de vereisten die gesteld werden in samenspraak met ArcelorMittal Gent zal er gekeken worden of die ook daadwerkelijk worden ingelost. Die vereisten zijn als volgt :

- Het coolgen programma moet zonder fouten gecompileerd worden.
- Het programma moet succesvol gebind worden en in de juiste load libraries opgeslagen worden.
- Het programma moet uitvoerbaar zijn na compilatie en bind.
- Er moet een automatisch versiebeheer systeem zijn gebaseerd op Git versiebeheer.
- Alle taken die hierboven zijn opgesomd moeten automatisch kunnen gebeuren.

De verwachting is dat het onderzoek zal aantonen dat de bovenvernoemde vereisten ingelost kunnen worden, dit zal als resultaat geven dat er een eenvoudige en overzichtelijke pipeline zal werken die een coolgen programma compileerd, bind en aan versiebeheer doet. Hierdoor zal er geconcludeerd kunnen worden dat het wel degelijk mogelijk is om ontwikkelaars te laten werken buiten een mainframe omgeving aan een coolgen applicatie en zo op die manier een release management systeem te gebruiken dat gebasseerd is op IBM DBB met behulp van Azure DevOps en Git.

Referenties

Bloomberg, J. (2021, april 9). *How the mainframe became a surprising platform for innovation*. Verkregen december 10, 2023, van https://siliconangle.com/2021/04/09/mainframe-became-surprising-platform-innovation/

Broadcom. (2024, februari 12). *Modernize Mainframe Development Processes and Tooling* (Broadcom, Red.). https://mainframe.broadcom.com/devops

Daniel Sokolowski, G. S., Pascal Weisenburger. (2021). Automating Serverless Deployments for DevOps Organizations. *Automating Serverless Deployments for DevOps Organizations*, 57–69. https://doi.org/10.1145/3468264.3468575

Moorhead, P. (2022, april 5). *IBM Z16 – The Mainframe Is Dead, Long Live The Mainframe*. Verkregen december 10, 2023, van https://www.forbes.com/sites/patrickmoorhead/2022/04/05/ibm-z16--the-mainframe-is-dead-long-live-the-mainframe/?sh=60b91a16ee9d

Porter, K. (2019, juni 7). *z/OS development with IBM Dependency Based Build and IDz*. Verkregen december 10, 2023, van https://kalebporter.medium.com/modern-z-os-development-with-ibm-dependency-based-build-9cf945d72bfb

Tozzi, C. (2022, november 17). *Where Will Mainframes be in Ten Years?* Verkregen december 10, 2023, van https://www.precisely.com/blog/mainframe/mainframe-industry-ten-years