Delta-N
Connecting the Cloud

DevOps in de Cloud

DevOps met Azure en Azure DevOps

Delta-N BV | Laan van Waalhaven 450 | 2497 GR Den Haag | 085 – 487 52 00 | info@delta-n.nl | www.delta-n.nl

Delta-N
Connecting the Cloud

Contents

1 Introductie

DevOps is al een tijd een begrip. Er wordt veel gesproken over DevOps**.** En hoewel DevOps op zich los staat van Cloud computing, kan de Cloud wel gezien worden als een 'enabler' voor DevOps. Nu het gebruik van de Cloud meer en meer ingeburgerd raakt, zien we dat de discussie rondom, en zelfs de introductie van DevOps in een stroomversnelling komt. In deze whitepaper kijken we op welke manier Microsoft Azure en Azure DevOps ondersteunend kunnen zijn aan de introductie van DevOps in uw organisatie.

2 Snel inspelen op een veranderende markt

Eén van de belangrijke drijfveren om te starten met DevOps is de behoefte van bedrijven om sneller in te kunnen spelen op de veranderende markt. Hoe sneller nieuwe ideeën en/of producten aan de klant aangeboden kunnen worden, hoe beter! Met behulp van Agile processen hebben development afdelingen al een slag gemaakt in het regelmatig opleveren van kleine delen software. Dit in tegenstelling tot de watervalmethode die een aantal jaren geleden nog veelvuldig toegepast werd. Als we Agile goed toepassen betekent dit dat er vaker een release opgeleverd wordt vanuit Development. Het wil echter nog niet zeggen dat deze release ook snel en zonder problemen in productie geplaatst kan worden!

DevOps is aanvullend aan Agile. Het verbetert de samenwerking, communicatie en integratie tussen Development en Operations en draagt eraan bij dat opleveringen vanuit Development ook daadwerkelijk production ready zijn. Hierdoor kunnen opleveringen gestandaardiseerd, snel en veilig in productie genomen worden door Operations zodat deze zo snel mogelijk waarde toevoegen voor de klant.

2.1 Leerproces op basis van feedback

DevOps stimuleert een korte release cycle. Door de nauwe samenwerking tussen Development en Operations ontstaat een continue stroom aan feedback tussen beide afdelingen. Belangrijk hierbij is vooral het leerproces dat gevoed wordt door de feedback.

Een voorbeeld: Bij het toepassen van DevOps levert Development niet meer alleen een stuk software op, maar zorgt het ook voor (geautomatiseerde) configuratie instellingen van de test- en productieomgevingen. Hierdoor wordt het voor Operations veel eenvoudiger om de software ook daadwerkelijk in productie te zetten. Operations kan Development ondersteunen bij het zo optimaal mogelijk aanleveren van deze configuratie informatie. Operations op haar beurt, deelt actief monitoringinformatie over de performance en stabiliteit van de software met Development. Dit levert Development een beter inzicht in de kwaliteit van de opleveringen. Ze kunnen direct zien wat de impact van een bepaalde wijziging op productie is en kunnen, indien nodig, direct anticiperen.

Figuur 1: Continu leerproces op basis van feedback tussen Development, Operations en Product Owner.

De feedback vanuit de productieomgeving is niet alleen voor Development van belang maar heeft ook business value. De Product Owner (binnen een Agile project) maakt voor ieder iteratie die opgeleverd wordt een inschatting van de waarde die wordt toegevoegd. Op basis van de feedback vanuit Operations kan de Product Owner de eerdere inschatting valideren en indien nodig kan hij/zij een verfijning aanbrengen in de backlog.

Doordat het proces van 'in productie name' van opgeleverde software veelvuldig en gezamenlijk wordt uitgevoerd, zal dit steeds beter en sneller verlopen. Er is sprake van een continu leerproces dat uiteindelijk resulteert in een hogere beschikbaarheid van de dienst en een aanzienlijk kleinere kans op mislukkingen!

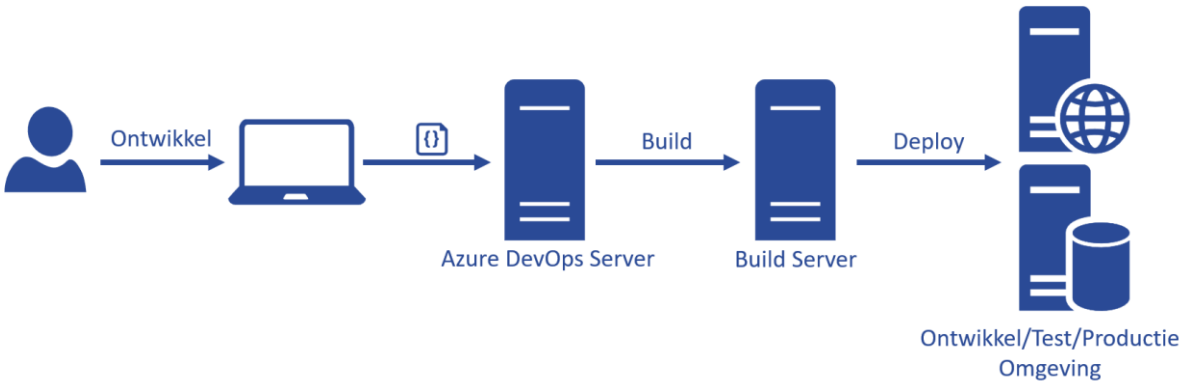
2.2 Cultuur, processen en tools

DevOps wordt enerzijds gezien als een 'social movement' waarbij een cultuur gecreëerd wordt waarbinnen de afdelingen Development en Operations samenwerken. Anderzijds is het een verzameling van processen en tools die deze samenwerking ondersteunt en optimaliseert. In dit document zullen we de focus leggen op de tools die Microsoft Azure biedt en op welke manier die ondersteunend kunnen zijn aan bepaalde processen. In de praktijk zal dit niet genoeg blijken. Er dient minimaal evenveel aandacht besteed te worden aan het creëren van de juiste cultuur binnen de organisatie!

DevOps stimuleert een snellere time to market. Het verhoogt de kwaliteit van de software en standaardiseert de in productie name daarvan. Dit resulteert in een effectieve organisatie die snel in staat is om een idee en/of kans om te zetten in waarde voor de klant. Het spreekt voor zich dat organisaties die DevOps onder de knie krijgen hiermee een aanzienlijk voordeel hebben t.o.v. hun concurrenten!

2.3 Softwareontwikkelingstraject

Voordat we ingaan op de manier waarop Microsoft Azure en Azure DevOps ondersteunend kunnen zijn aan DevOps, is het goed om eens te kijken naar een eenvoudige release cycle in een traditioneel softwareontwikkelingsproces. Onderstaand figuur geeft dit proces schematisch weer.



Figuur 2: Schematische weergave van softwareontwikkelingsproces.

Het Development team bestaat uit één of meerdere ontwikkelaars die verantwoordelijk zijn voor het opleveren van de gevraagde functionaliteit. Ontwikkelaars maken gebruik van een PC en/of laptop met daarop Visual Studio en gebruiken Azure DevOps Server (voorheen Team Foundation Server (TFS)) als Version Control Systeem. Veelal wordt Azure DevOps Server daarnaast ingezet als ondersteuning van de Agile werkwijze van het team. Er wordt gebruik gemaakt van geautomatiseerde builds vanuit waar ook de unittesten en de deployment naar de ontwikkel- en/of test servers geregeld wordt. Uiteindelijk zal er een (handmatige) overdracht plaatsvinden naar de afdeling Operations, die de oplevering in de productieomgeving plaatst.

Veel organisaties maken voor de ondersteuning van dit softwareontwikkelingsproces gebruik van resources in hun eigen datacenter. Ze hebben een eigen Azure DevOps-server en het development team beheert de eigen ontwikkel- en/of testservers. De productieomgeving valt veelal niet onder de verantwoordelijkheid van het development team en bevat daarom niet per definitie dezelfde configuratie als de ontwikkel- en/of testomgeving.

Opvallend is, dat er vaak geen directe feedbackloop is vanuit Operations terug naar Development. Een softwarerelease wordt door Development ontwikkeld, getest en vervolgens overgedragen naar Operations. Communicatie tussen beide afdelingen vindt vaak alleen plaats als er problemen ontstaan bij de in productie name en/of in productie zelf.

2.4 De combinatie van Azure, Azure DevOps en DevOps practices

Microsoft Azure, in combinatie met Azure DevOps (voorheen VSTS), biedt voor ieder van de genoemde onderdelen in het softwareontwikkelingsproces een aantal interessante alternatieven en mogelijkheden. Zo kan het development team, in plaats van Azure DevOps Server (TFS), gebruik maken van een door Microsoft beheerde Azure DevOps omgeving. Daarnaast kunnen Azure Virtual Machines gebruikt worden voor het snel (geautomatiseerd) 'in de lucht' brengen van ontwikkel- en of testomgevingen. Gaan we nog een stap verder dan maken we gebruik van de standaard bouwblokken van het Azure PaaS of SaaS platform en deployen we webapplicaties direct naar Azure Web Apps. In deze variant komt een aanzienlijk deel van de configuratie en het beheer van de (productie) omgeving te vervallen. Standaard features zoals roll forward en rollback zijn hierbij krachtige hulpmiddelen om een oplevering gecontroleerd van ontwikkel, naar test en uiteindelijk naar productie door te zetten.

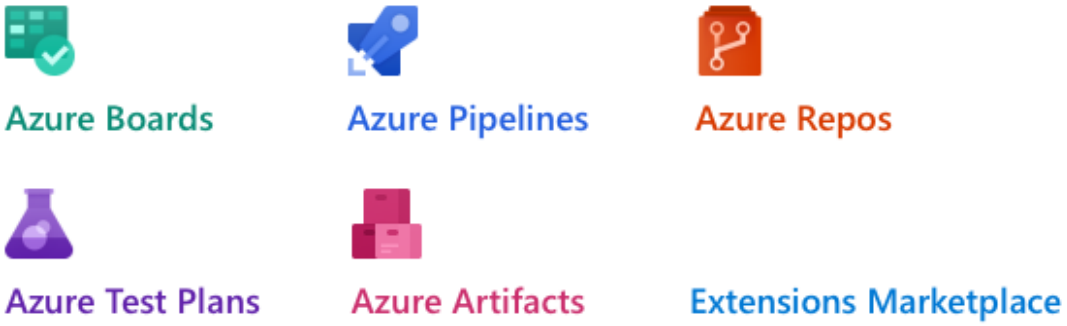
De in Azure DevOps en Azure ingebouwde monitoring- en managementfunctionaliteit biedt Development en Operations de mogelijkheid om applicaties continue te monitoren en daar waar nodig direct in te grijpen. Dit alles met het doel om de stroom van opleveringen vanuit Development snel, gecontroleerd en voorspelbaar naar productie te krijgen.

3 Azure DevOps

Veel organisaties maken al gebruik van Azure DevOps Server (of voorloper Team Foundation Server (TFS) voor o.a. version control en de ondersteuning van hun Agile werkproces. De modernere online as a service variant voor TFS is Azure DevOps. In dit hoofdstuk kijken we naar de mogelijkheden die Azure DevOps biedt. We focussen ons daarbij op de onderwerpen die een relatie hebben met DevOps.

3.1 Wat is Azure DevOps?

Azure DevOps is de onlineversie van Azure DevOps Server en voorloper Team Foundation Server (TFS). Ook Azure DevOps integreert naadloos met de Visual Studio IDE en Microsoft Azure.



Naast features die we al kennen uit TFS zoals: version control en work item management, biedt Azure DevOps een aantal extra mogelijkheden die het product zeer interessant maken voor DevOps. Denk hierbij bijvoorbeeld aan Build pipelines, Release pipelines, Load Testing en Continuous Deployment naar Azure. Hoewel een aantal van deze diensten ook beschikbaar zijn in TFS zullen we later zien dat het feit dat deze diensten online beschikbaar zijn een aantal voordelen heeft.

3.2 Waarom Azure DevOps?

Veel organisaties gebruiken al TFS (of nu Azure DevOps Server). In de basis zijn Azure DevOps Server (TFS) en Azure DevOps gelijk. Ze hebben dezelfde code base, en worden beide door hetzelfde team ontwikkeld. Waarom dan toch kiezen voor Azure DevOps?

3.2.1 Geen beheer, altijd up-to-date, meer flexibiliteit!

Eén van de meest voor de hand liggende voordelen van Azure DevOps ligt in het feit dat er geen beheer meer is over Azure DevOps. Microsoft stelt Azure DevOps als dienst beschikbaar en neemt daarmee de verantwoordelijkheid voor de correcte werking en beschikbaarheid van deze dienst. Aangezien we Azure DevOps afnemen als dienst en de software automatische geupdate wordt, weten we zeker dat we altijd gebruik maken van de meest up-to-date versie.
Dit in tegenstelling tot een eigen on-premises Azure DevOps Server-installatie, waarbij wijzelf verantwoordelijk zijn voor de onderliggende hardware, de back-up, het installeren van updates, en het uitvoeren van migraties naar nieuwe versies. Organisaties die nu al gebruik maken van TFS kennen het

vast wel. Op enig moment wordt gekozen voor een bepaalde versie van TFS die vervolgens binnen de organisatie uitgerold wordt. Vanaf dat moment wordt er vaak meerdere jaren gebruik gemaakt van die versie van TFS. Een eventuele upgrade van TFS in de periode daarna wordt voorafgegaan door vergaderingen, migratieplannen, budgetaanvragen en beslismomenten. Niet echt Agile!

Het wegvallen van het beheer op de eigen Azure DevOps Server-omgeving (TFS) is niet het enige voordeel dat Azure DevOps te bieden heeft. Azure DevOps biedt ook extra flexibiliteit in de manier waarop we als organisatie werken. De wereld om ons heen staat niet stil en de manier waarop we softwareontwikkelingstrajecten uitvoeren veranderd nog zeer regelmatig. Dit wordt veroorzaakt door een veranderende markt, projectervaringen, de producten en/of frameworks die we gebruiken en de continue veranderende mogelijkheden van platformen, zoals Azure, waarop we onze software deployen.

Microsoft hanteert voor Azure DevOps een release cycle van drie weken. Dit betekent dat er iedere drie weken nieuwe features worden toegevoegd aan Azure DevOps die dan ook per direct beschikbaar zijn voor gebruikers.
Aangezien Microsoft zelf ook DevOps toepast, gebruikt Microsoft de verzamelde monitoringinformatie van Azure DevOps en de feedback van gebruikers actief bij het verfijnen van de backlog voor Azure DevOps. Hierdoor is Microsoft in staat om Azure DevOps naadloos aan te laten sluiten aan de sterk veranderende omgeving waarbinnen IT-organisaties opereren. Het feit dat Microsoft zelf ook grootgebruiker is van Azure DevOps versterkt dit effect alleen maar. Microsoft ondervindt zelf dagelijks waar de verbeterpunten liggen van Azure DevOps.

De meeste van de nieuwe mogelijkheden van Azure DevOps zullen uiteindelijk ook in Azure DevOps Server (voorheen TFS) beschikbaar komen. Echter, dit gebeurt via de updates die ongeveer om de drie maanden uitgeleverd worden. Daarnaast zit er vaak nog een aanzienlijke tijd tussen het beschikbaar komen van een nieuwe versie van Azure DevOps Server en het moment dat deze versie ook daadwerkelijk beschikbaar is binnen de organisatie. Azure DevOps biedt ons de flexibiliteit om direct gebruik te maken van de nieuwe mogelijkheden.

3.2.2 Beschikbaarheid versus toegankelijkheid

Bij het maken van een keuze tussen on-premises of Azure DevOps zijn we geneigd een beslissing te nemen op basis van de beschikbare mogelijkheden binnen één van beide de varianten. Een veel gehoorde opmerking is dat Azure DevOps geen mogelijkheid biedt om wijzigingen door te voeren in de proces templates, er geen datawarehouse voor rapportage is en dat de integratie met SharePoint ontbreekt. Op het eerste gezicht zijn dit valide redenen om niet te kiezen voor Azure DevOps.

Echter, als we ons realiseren dat we over drie weken de beschikking hebben over een ‘nieuwe’ versie van Azure DevOps, waarin ontbrekende functionaliteit misschien ineens wel beschikbaar is, dan kunnen we ons afvragen of we onze keuze tussen Azure DevOps Server en Azure DevOps moeten laten afhangen van beschikbaarheid van mogelijkheden. Ondertussen kun je het proces template wel aanpassen, is uitgebreide rapportage grotendeels aanwezig en zijn er uitgebreide integratiemogelijkheden met meerdere systemen.

Het is misschien veel verstandiger om een keuze te maken op basis van toegankelijkheid van mogelijkheden. Wat bedoelen we hiermee? Azure DevOps Server biedt de mogelijkheid om builds in

te richten en load tests uit te voeren. Echter, om dit in te regelen, hebben we wel een build server en test servers nodig. Deze zijn niet altijd per direct beschikbaar voor het Development team. Op dat moment lopen we het gevaar dat Development besluit om dan maar even geen build in te richten en de performance test uit te stellen tot latere iteratie. Dat is niet wat we willen, want hoe sneller we feedback krijgen over de gemaakte software, hoe sneller mogelijke problemen opgelost kunnen worden.

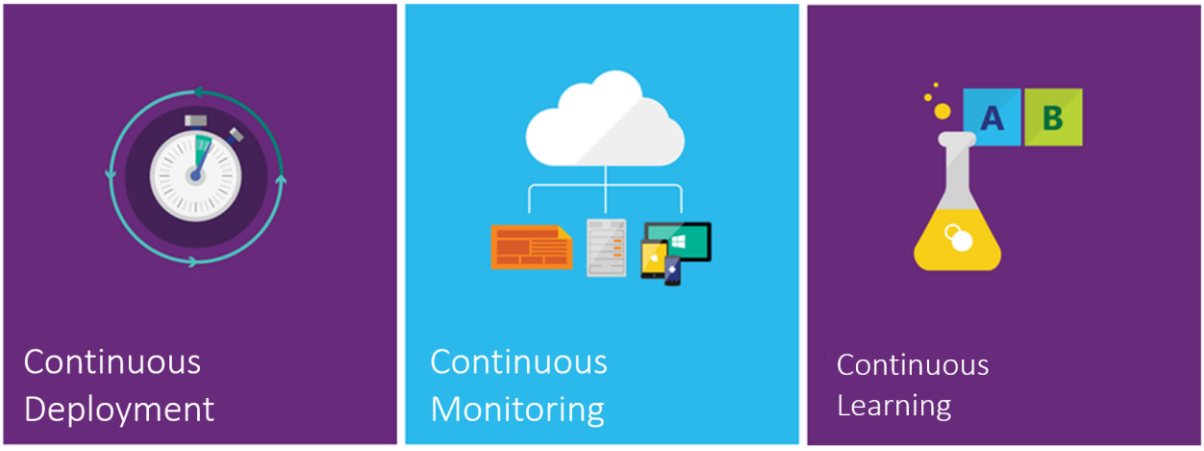
In Azure DevOps zijn de Build Pipeline en de Load Testing Service direct toegankelijk. Beide zijn met een paar muisklikken in te richten. Binnen enkele minuten maken we met behulp van Azure DevOps en Azure een load test voor onze applicatie die enkele miljoenen gebruikt simuleert!

Kortgezegd, toegankelijkheid stimuleert adoptie en maakt het voor Development en Operations een stuk eenvoudiger om zogenaamde best practices te integreren in hun werkwijze.

3.3 Via Continuous Deployment en Continuous Monitoring naar Continuous Learning

Met behulp van DevOps streven we naar een zo kort mogelijke time-to-market van een oplevering van onze software. Dit alles met behoud van een hoge kwaliteit en voorspelbaarheid.
In het streven naar een organisatie waarin we DevOps optimaal ingeregeld hebben, zullen we een aantal aspecten onder de knie moeten krijgen. Microsoft onderkent een aantal stadia in de route naar DevOps.

Allereerst zullen we Continuous Delivery moeten implementeren, vervolgens regelen we Continuous Monitoring in om uiteindelijk de vruchten te plukken van Continuous Learning. Klinkt goed, maar wat betekent dit allemaal precies en op welke manier kan Azure DevOps en Azure ons hierbij ondersteunen?



3.3.1 Continuous Deployment

Als we Agile goed toepassen, levert Development een continue stroom op aan opleveringen die ieder een stuk toegevoegde waarde opleveren voor de klant. Continuous Deployment houdt zich bezig met het geautomatiseerd opleveren van deze bits in productie. We hebben hierbij een aantal uitdagingen.

Allereerst dienen we beschikking te hebben over ontwikkel- en testomgevingen. Vaak vereisen deze omgevingen diverse instellingen, tools en frameworks. Allemaal verschillende configuraties die gedurende het softwareontwikkelingstraject ook nog eens voortdurend wijzigen. Het niet eenvoudig om deze verschillende omgevingen onderling consistent te houden. Veel van de problemen die optreden tijdens het testen of bij de in productie name worden veroorzaakt door afwijkingen in configuraties of omgevingen. Wie kent niet de uitspraak: 'It works on my machine!?!'.

Daarnaast hebben we een release pipeline nodig. Afhankelijk van het product dat we opleveren kan deze variëren van een eenvoudig proces waarbij we een build, na een korte controle in een testomgeving, direct door zetten naar productie tot een ingewikkelde workflow met verschillende acties, controle-slagen en approval momenten. Hoe gaan Azure DevOps en Azure ons hierbij ondersteunen?

Een voorbeeld: Visual Studio beschikt over een zogenaamd 'Azure Resource Group Project' template. Dit project stelt ons in staat om onze deployment omgeving en alle bijbehorende configuratie vast te leggen als broncode. Een wizard inspecteert onze Visual Studio Solution en genereert op basis daarvan een beschrijving van de omgeving waarop de applicatie uitgerold kan worden. In deze beschrijving wordt bijvoorbeeld opgenomen op hoeveel virtual machines de applicatie moet worden uitgerold, welke versie van het .NET Framework beschikbaar moet zijn en op welke manier de applicatie in IIS geconfigureerd moet worden. Daarnaast wordt er onderscheid gemaakt tussen de verschillende omgevingen zoals een ontwikkel-, een test- en een productie omgeving. Dit alles wordt opgeslagen in een leesbaar JSON-formaat. We kunnen dus zelf heel eenvoudig aanpassingen doorvoeren. Het spreekt voor zich dat dit alles op te nemen is in source control waardoor we versiebeheer hebben over onze omgevingsconfiguratie.

Op basis van de hierboven genoemde beschrijvingen van de verschillende omgevingen, kan Development vanuit Visual Studio een deploy doen naar één of meerdere omgevingen. Hierbij wordt de omgeving automatisch compleet ingericht zoals het beschreven is, inclusief de benodigde virtual machines, (Azure) websites, etc. We introduceren hiermee herhaalbaarheid in het deployment proces en elimineren de problemen die ontstaan door inconsistenties tussen verschillende omgevingen.

Uiteraard is het in de praktijk niet altijd gewenst dat Development, vanuit Visual Studio, een oplevering direct doorzet naar productie. Het is dan ook goed te weten dat Azure DevOps ook Release pipelines biedt. Met behulp van Release pipelines krijgen we meer controle op de release. We kunnen bijvoorbeeld een workflow definiëren waarbij we bepaalde approval momenten introduceren. Hiermee kunnen we ervoor zorgen dat een release niet naar de volgende omgeving doorgezet kan worden zonder dat er goedkeuring is van een bepaald persoon. De beschrijvingen van onze deployment omgevingen (output van Azure Resource Group Project) kan ook als input dienen voor Release pipelines.

3.3.2 Continous Monitoring en Continuous Learning

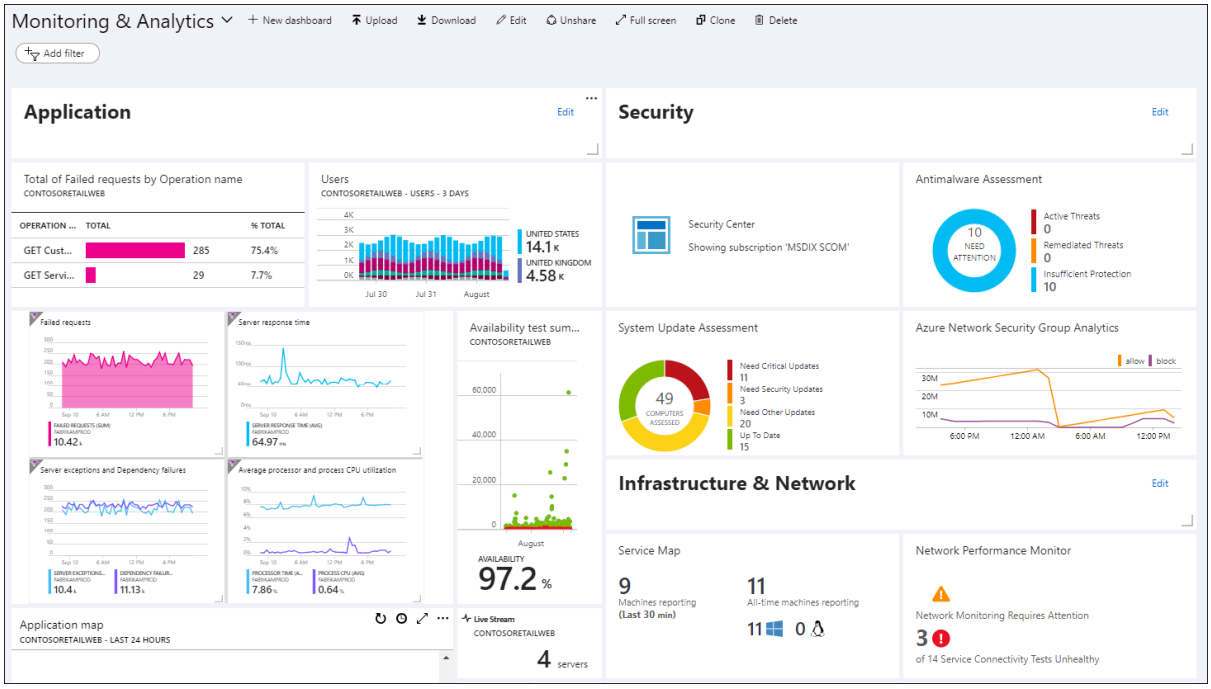
Laten we er nu vanuit gaan dat we Continuous Deployment op orde hebben en dat we onze software regelmatig en op een gecontroleerde wijze naar productie kunnen krijgen. Maar dan? Zijn we dan klaar? In veel organisaties lijkt het daar wel op! Zolang de gebruikers niet aan de telefoon hangen is Operations tevreden en zal Development een volgende sprint starten waarin ze de volgende items van

de backlog oppakken. Beide afdelingen hebben geen enkel idee of de gebruikers tevreden zijn met de oplevering en of de nieuwe features überhaupt wel gebruikt worden.
Stel, we hebben de beschikking over uitgebreide monitoring informatie die Operations antwoord geeft op relevante vragen zoals: is de applicatie beschikbaar, treden er onverwachte exceptions op en wat is de performance van de applicatie op dit moment? Met behulp van dergelijke informatie kan Operations proactief inspelen op eventuele productieproblemen. In samenwerking met Development kunnen de productieproblemen wellicht zelfs nog voordat gebruikers aan de telefoon hangen worden opgelost.

Het wordt nog interessanter als we ook informatie hebben over de manier waarop gebruikers de applicatie gebruiken. We kunnen dan namelijk de vraag beantwoorden: zijn gebruikers succesvol met de applicatie? Het wordt voor de Business direct inzichtelijk welke onderdelen wel of niet gebruikt worden, hoe ze gebruikt worden en of de aanpassingen die zijn doorgevoerd in de oplevering het gewenste effect hebben.

3.3.3 Application Insights

Dit is nou precies waar **Application Insights** om de hoek komt kijken. Application Insights is een dienst van Azure die onderdeel is van Azure Monitor. Deze 'out of the box' service verzamelt de hierboven genoemde informatie (en meer!) en stelt deze beschikbaar. Het is toepasbaar voor applicaties die gehost worden in Azure maar ook voor on-premises applicaties. Het feit dat de informatie vanuit Application Insights volledig geïntegreerd is in de Azure portal en gecombineerd kan worden met andere monitordata, maakt het geheel extra krachtig. De portal biedt een totale DevOps view voor de applicaties waarvoor we verantwoordelijk zijn. Het geeft ons inzicht in de status van het Azure platform, het toont een inschatting van de gemaakte kosten en biedt een schat aan informatie voor Operations, Development en uiteindelijk ook de Business.



Als we als organisatie in staat zijn om ons Continuous Deployment proces op orde te krijgen dan kunnen we snel en gecontroleerd opleveren naar productie. Combineren we dat met de Continuous Monitoring mogelijkheden van Application Insights en Azure Monitor dan optimaliseren we daarmee niet alleen onze eigen IT-organisatie, maar bieden we ook toegevoegde waarde voor de Business. Nieuwe ideeën kunnen snel beschikbaar worden gesteld aan de klant. Op basis van het gedrag van de klant kan de Business toetsen in hoeverre het idee wel of niet levensvatbaar is en indien nodig kan de Business direct haar strategie aanpassen. Kortom we vormen een zelflerende organisatie die proactief kan inspelen op een markt die continue in beweging is!

4 Continuous Deployment

Continuous Delivery wordt vaak gezien als de eerste stap op weg naar DevOps. Maar om misverstanden te voorkomen, Continuous Delivery betekent niet dat iedere aanpassing in de source code ook direct en geautomatiseerd in productie terecht komt. Het gaat er bij Continuous Delivery om dat de software gedurende de levenscyclus altijd deploybaar is. Continuous Deployment is de techniek om geautomatiseerd software naar productie uit te rollen.

4.1 Continuous Deployment, stap voor stap

Continuous Deployment richten we niet in één dag in voor onze organisatie. Voordat we dat bereikt hebben zullen we een aantal stappen moeten doorlopen. Sommige van deze stappen zijn al behoorlijk ingeburgerd binnen ontwikkelteams, andere komen nog wat minder frequent voor.

4.1.1 Continuous Integration

Eén van de eerste zaken die we op orde moeten brengen is het build proces. Continuous Integration (CI), de geautomatiseerde build, levert als output de artifacts op die uiteindelijk naar de verschillende omgevingen gedeployed zullen worden. Team Foundation Server biedt al lange tijd de mogelijkheid om een geautomatiseerde build in te richten en ieder zichzelf respecterend ontwikkelteam maakt hier ongetwijfeld gebruik van. Tegenwoordig kunnen we ook gebruik maken van de hosted build feature van Azure DevOps. In dit geval hebben we geen eigen resources meer nodig voor het inrichten van bijvoorbeeld een build server. Hierdoor wordt een CI build eenvoudig om in te richten!

4.1.2 Automatisch testen

De volgende stap is het inrichten van het geautomatiseerd testen. In eerste instantie leggen we de focus op unit tests die de correcte werking van losse componenten aantonen. Vervolgens kunnen we de geautomatiseerde test uitbreiden met bijvoorbeeld ingewikkeldere integratie scenario's. Het maken van goede unit tests is niet altijd even eenvoudig en verreist enige oefening. Op bestaande projecten, waarvoor nog geen unit testen beschikbaar zijn, kan het starten met geautomatiseerd testen een tijdrovend werk zijn. In dat geval kan de Smart Unit Test-feature van Visual Studio uitkomst bieden! Hiermee kunnen we een complete set van unit tests genereren.

4.1.3 Deployment omgeving

De automatische build en geautomatiseerd testen helpen ons bij het valideren en/of garanderen van de kwaliteit van de opleveringen die we doen. De volgende stap is de provisioning van de omgevingen waarop we een deployment kunnen uitvoeren.
In eerste instantie richten we ons hierbij op het standaardiseren van de infrastructuur en de configuratie van de verschillende omgevingen. Juist hier kunnen we gebruik maken van de kracht van Microsoft Azure. Binnen een paar minuten hebben we aantal Virtual Machines beschikbaar waarop we onze software kunnen deployen. Met behulp van eigen templates en Powershell Desired State Configuration is het vervolgens mogelijk om het aanmaken en configureren van deze omgevingen volledig te automatiseren.

Een andere mogelijkheid die we hebben om dit doel te bereiken is het Azure Resource Group Project. We hebben al eerder gezien dat dit ons een manier biedt om de omgeving en bijbehorende

configuratie vast te leggen als source code. Onze infrastructuur en bijbehorende configuratie wordt daarmee beheersbaar. Het feit dat de configuratie wordt vastgelegd in een leesbaar JSON-formaat biedt ons nog een ander voordeel. Development en Operations kunnen de output van het Azure Resource Group Project gebruiken om afstemming te bereiken over de benodigde infrastructuur en configuratie. Feedback tussen beide afdelingen!

Het is belangrijk dat we de provisioning van onze omgevingen zoveel als mogelijk automatiseren. We streven naar een voorspelbaar en herhaalbaar proces zodat we iedere oplevering op een 'schone' omgeving kunnen uitvoeren. Hierdoor minimaliseren we de kans dat er allerlei handmatige en niet gedocumenteerde acties worden uitgevoerd op een development en/of test omgeving 'om de boel aan de praat te krijgen'. Uiteraard met het risico dat de oplevering vervolgens niet werkt in productie.

4.1.4 Release pipeline

Als laatste stap hebben we een mechanisme nodig dat de opgeleverde software (output van de CI build) op een gecontroleerde wijze door de verschillende omgevingen uitrolt. Team Foundation Server biedt hier al enige tijd Release Management voor. In Azure DevOps is dit beschikbaar als Release pipelines.

Met behulp van Release pipelines kunnen we exact specificeren welk pad een oplevering vanuit de Development omgeving naar Productie omgeving moet doorlopen. We kunnen daarbij ook aangeven welke goedkeuringen en geautomatiseerde gates er noodzakelijk zijn voordat een oplevering doorgezet kan worden naar een volgende omgeving. Daarnaast hebben we volledig inzicht in alle stappen die uitgevoerd zijn tijdens de deployment, wie er goedkeuring heeft gegeven en welke bugs zijn er opgelost in de release. Het eerdergenoemde Azure Resource Group Project kan als input dienen voor het configureren van de omgevingen die binnen Release pipelines gebruikt worden. Kortgezegd, we kunnen dus een geavanceerde release pipeline opzetten!

4.2 Continuous Deployment met Azure Websites en Azure DevOps

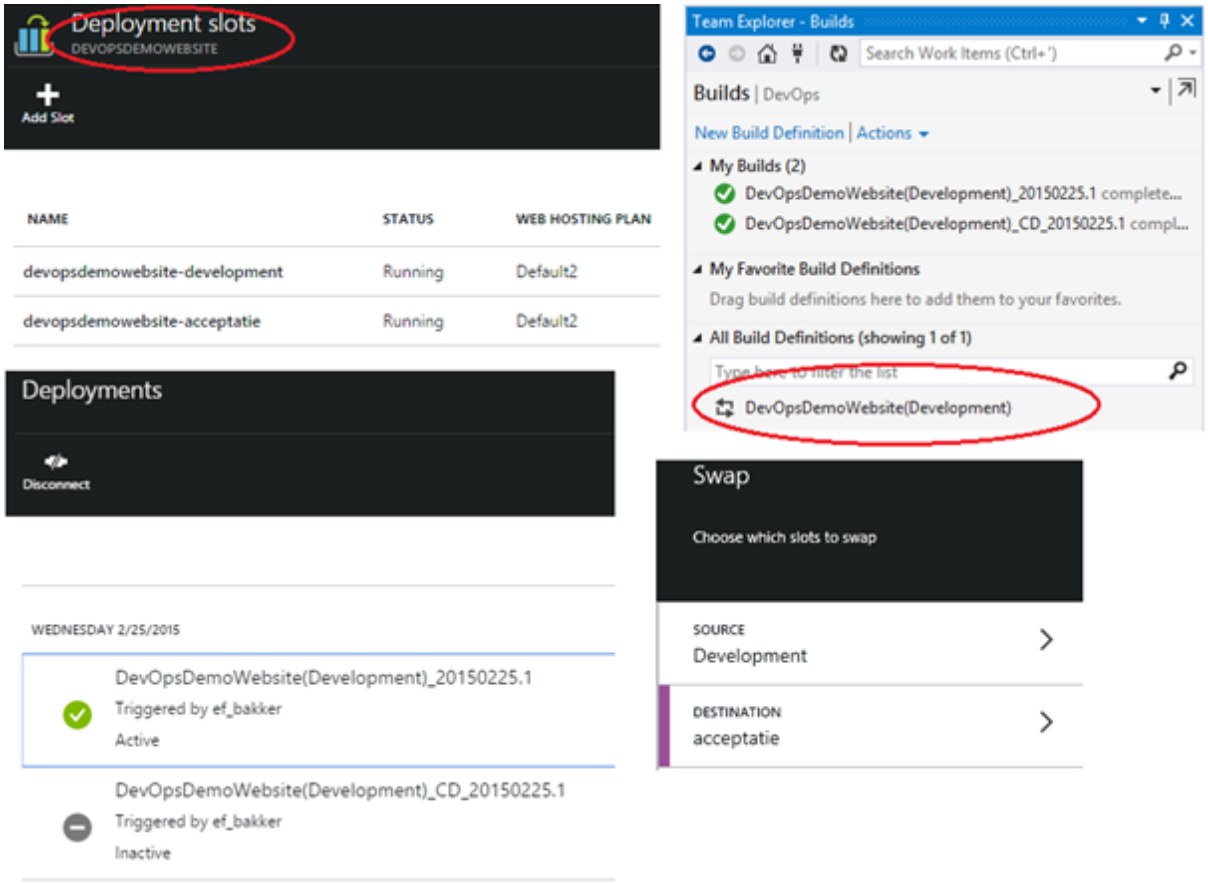
We weten nu dat we met behulp van Powershell, Desired State Configuration, Azure Resource Group Projecten en Azure VM's in staat zijn om geautomatiseerd een deployment omgeving op te zetten, die we vervolgens kunnen gebruiken binnen een geavanceerde release pipeline. Maar wat als we gebruik maken van een Azure SaaS-product zoals Azure Web Apps? In dat geval hebben we geen (eigen) virtual machines waar we onze deployment omgevingen op moeten configureren. Kunnen we dan geen Continuous Delivery doen? Gelukkig biedt Azure Web Apps i.c.m. Azure DevOps ook een aantal interessante mogelijkheden!

4.2.1 Deployment Slots

Azure Web Apps biedt ons een out of the box, een volledig beheerde omgeving voor het hosten van bijvoorbeeld onze ASP.NET MVC-website. Eén van de mooie features van Azure Websites is dat we hier zogenaamde Deployment Slots kunnen definiëren. Een Deployment Slot kunnen we omschrijven als een kopie van onze website in een eigen Azure Website omgeving. We kunnen voor onze website bijvoorbeeld een development en een acceptatie slot definiëren. Dit betekent dat er naast de

productieversie van onze website nog twee extra omgevingen beschikbaar zijn die we elk via een eigen URL kunnen benaderen.

Als we vervolgens onze source code in Azure DevOps beheren, dan kunnen we in de Azure Portal een koppeling leggen tussen een Deployment Slot en onze Visual Studio Solution in version control repository. In het plaatje hieronder zien we dat er voor de website *DevopsDemoWebsite* twee extra Deployment Slots gedefinieerd zijn. Het development slot is gekoppeld aan version control. Op het moment dat we deze koppeling maken, wordt er automatisch een build definition aangemaakt in onze Azure DevOps omgeving. Deze zorgt ervoor dat vanaf dat moment iedere build output in het development slot van onze website terecht komt. De Azure Portal toont voor ieder van de Deployment Slots een overzicht van alle builds die daar terecht zijn gekomen. De laatste build wordt automatisch de actieve deployment, maar met één druk op de knop kunnen we binnen het Deployment Slot terugschakelen naar een eerdere build.



4.2.2 Deployment Slot Swap

Met behulp van de Swap functionaliteit in de Azure Portal, kunnen we een bepaalde oplevering naar een ander Deployment Slot doorzetten. Op deze manier kunnen we bijvoorbeeld heel eenvoudig een oplevering uit het development Deployment Slot doorzetten naar het acceptatie Deployment Slot. Met behulp van deze functionaliteit kunnen we dus ook voor Azure Web Apps een eigen release pipeline

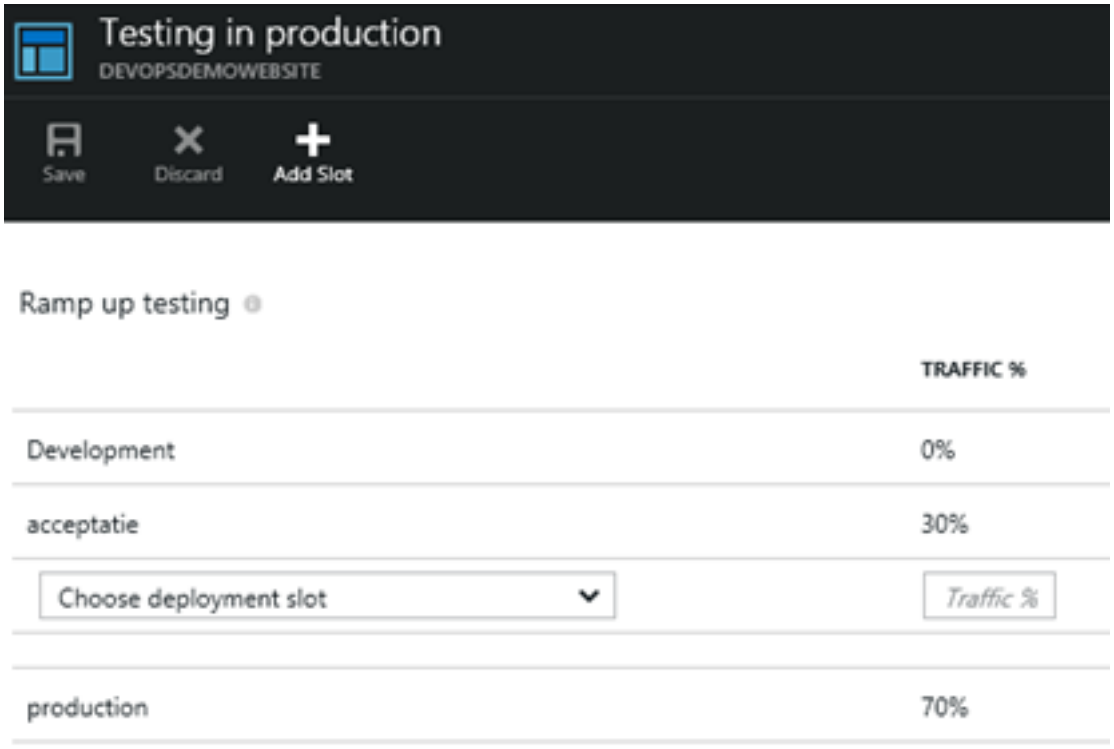
opzetten. In vergelijking met Release biedt het minder mogelijkheden voor geavanceerde workflows binnen de release pipeline. Vaak hebben we dat, in eerste instantie, ook helemaal niet nodig en biedt bovenstaande een voldoende krachtig middel om op een verantwoorde manier om te gaan met onze releases!

4.2.3 Azure Websites Production Testing

Eén van de belangrijke pijlers van DevOps is het snel reageren op feedback vanuit Development, Operations of gebruikers. Deze feedback kunnen we echter niet altijd krijgen uit een ontwikkel- of testomgeving.

Stel, we hebben een wijziging doorgevoerd die de performance van de website moet verbeteren. Met behulp van de hierboven genoemde Deployment Slots kunnen we deze wijziging testen in de ontwikkel en/of test omgeving. Als het goed is, geeft dit een realistisch beeld van de effectiviteit van de aanpassing. Echter, is het in zo'n geval niet waardevol als we de wijziging ook kunnen testen in een productie omgeving? Een ander voorbeeld. We hebben een aanzienlijke userinterface aanpassing doorgevoerd in onze applicatie. Voordat we deze wijziging in productie doorvoeren, willen we meten of de wijziging het gewenste effect heeft op eindgebruikers. Hoe doen we dit?

Voor beide scenario's biedt Azure Websites Production Testing uitkomst! Met deze feature zijn we instaat om een gedeelte van het verkeer op onze productiewebsite af te laten handelen door bijvoorbeeld de versie die in het acceptatie slot draait. Dit betekent dat een deel van de gebruikers, zonder dat zij het weten, gebruik maken van de nieuwe versie van de software. Zoals we in het figuur hieronder zien, kunnen we zelf bepalen welk percentage van het verkeer op de productiewebsite door een ander deployment slot afgehandeld dient te worden. Op basis van feedback en monitoring informatie, krijgen we met behulp van production testing een nog beter beeld van de kwaliteit en effectiviteit van een bepaalde aanpassing en kunnen we een beargumenteerde keuze maken om een swap actie wel of niet uit te voeren!



Natuurlijk omvat Continuous Deployment veel meer dan de hierboven beschreven onderwerpen. We hebben gezien dat de combinatie van Azure DevOps, Release pipelines en Microsoft Azure krachtige mogelijkheden biedt voor Continuous Deployment. Het biedt een mooie start en het is zeker de moeite waard om daar eens mee te experimenteren! Een volgende stap op weg naar DevOps is Continuous Monitoring.

5 Continuous Learning

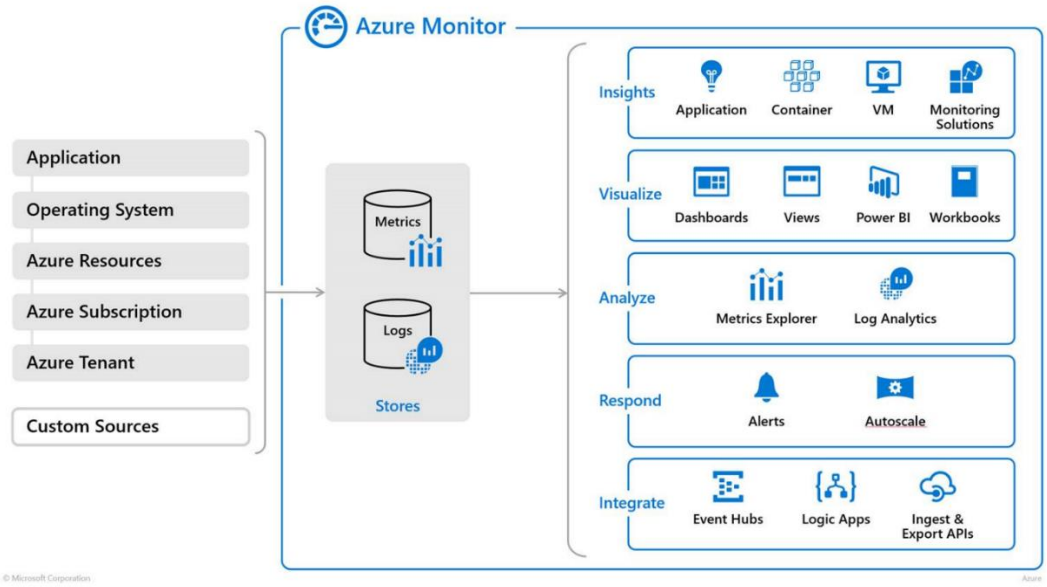
Continuous Learning sluit de loop van DevOps. Om te leren moet je meten en dat is een heel belangrijk onderdeel van DevOps. Meten wat de gebruikers doen, meten hoe het met de applicatie en de omgeving gaat, en meten hoeveel fouten gedetecteerd zijn.

5.1 Continuous Monitoring

Het doel van Monitoring is om, op basis van data uit de applicatie en de infrastructuur, beslissingen te kunnen nemen over verder ontwikkeling van de business. Het is dus niet alleen voor het alarmeren als er iets mis gaat. De data moet zo rijk zijn dat de product owner inzicht heeft aan welke volgende functionaliteit gewerkt moet worden.

5.2 Monitoring met behulp van Azure Monitor

Azure Monitor is een volledig set aan tooling om alle onderdelen van een applicatie te monitoren. Azure Monitor bestaat uit drie onderdelen: Bronnen waaruit de data verzameld wordt, data stores waar alle data opgeslagen wordt en tools om de data te benaderen



Figuur 3: In het boven getoonde diagram is te zien dat er niet alleen data verzameld kan worden vanuit applicaties.

De toolset is opgedeeld in specifieke verantwoordelijkheden.

- **Insights** is de centrale plek om inzicht te krijgen in Applicaties, containers, VM's en ander monitoring oplossingen.
- **Visualize** maakt het mogelijk om dashboarden in Azure of Power BI te creëren en de data te filteren voor de juiste analyse
- **Analyze** om dieper in te gaan op de data en tot de kern te komen van het probleem of vraagstuk

- **Respond** kan gebruikt worden om te alarmeren en om bijvoorbeeld een up of out scale te doen
- **Integrate** maakt het mogelijk om met andere systemen informatie te delen en te automatiseren

Om te starten met monitoring zal je er eerst voor moeten zorgen dat deze data beschikbaar is. Applications Insights, zoals eerder aangegeven onderdeel van Microsoft Azure Monitor, is hier een hele goede oplossing voor. Het is voor ontwikkelaars eenvoudig toe te voegen aan een applicatie en is direct beschikbaar tijdens het ontwikkelen. De ontwikkelaar kan dus gelijk data verzamelen over het gedrag van de applicatie. Zonder veel code wordt kan je al inzicht krijgen in de applicatie. Om specifieke data beschikbaar te stellen aan Azure Monitor kan er gebruikt gemaakt worden van de Application Insights SDK. Zelfs tijdens het debuggen van de code zal deze eigen data al verschijnen in Application Insights.

Azure Monitor is dus een laagdrempelige manier om inzicht te krijgen in de verschillende onderdelen van het applicatielandschap. Met de verschillende tooling in Azure Monitor kan de business de juiste beslissingen te nemen en succesvol te zijn en blijven en klanten binden.

Deze whitepaper

Deze whitepaper is geschreven door Edward Bakker. Al 12 jaar op rij is de prestigieuze titel Microsoft Most Valuable Professional (MVP) aan hem toegekend. Hij werkt als Microsoft Cloud Consultant voor Delta-N en is gespecialiseerd in Microsoft Azure, Azure DevOps en software architecture.

De whitepaper is in juni 2019 geactualiseerd door Mark van den Berg, Senior DevOps Consultant.

Over Delta-N

Delta-N realiseert al sinds 1999 flexibele en doordachte IT-oplossingen voor klanten op basis van Microsoft-technologie. Wij zijn gespecialiseerd in Cloud Technologie en helpen organisaties meer te bereiken door op een verantwoorde manier de stap naar de Cloud te maken.

Al ruim 10 jaar hebben wij een speciale unit die zich volledig richt op DevOps. We zijn Microsoft Gold Partner voor o.a. DevOps en Application Development en zijn tevens certified DORA (DevOps Research and Assessment) partner.

We werken volgens een vaste aanpak en maken hierbij gebruik van Agile methodieken, DevOps practices en Microsoft tools zoals Azure DevOps. Onze specialisten beschikken dan ook over ruime kennis en ervaring op het gebied van DevOps.

Meer weten over DevOps? Kijk op www.delta-n.nl/DevOps

Contact informatie

Delta-N
Laan van Waalhaven 450
2497 GR Den Haag
085 – 487 52 00

Info@delta-n.nl
www.delta-n.nl

Volg ons op Social Media en blijf op de hoogte!