ISAAC

MACH

Hoe weet ik of het bij mijn bedrijf past?

7 vragen die je jezelf zou moeten stellen

Turning .complexity into profitability

Inhoud

MACH: hoe weet ik of het bij mijn bedrijf past?

7 vragen die je jezelf zou moeten stellen

Microservices, API-first, Cloud-native SaaS en Headless – kortweg MACH. Dit begrip staat ineens volop in de belangstelling. Hoewel de losse concepten al geruime tijd bestaan en bekend zijn, wordt er – door de in juni 2020 opgerichte MACH Alliance – nu een paraplubegrip op deze technische visie geplakt. Is 'MACH' het nieuwe 'Cloud'? Net zo invloedrijk als 'SaaS'? Of is dat toch nog te veel eer?

ISAAC CTO Friso Geerlings voorziet met MACH een complexe uitdaging, maar ook iets dat veel waarde biedt. Met name voor bedrijven die een unieke, omnichannel klantbeleving willen creëren of een complexe value chain goed laten samenkomen.

In dit whitepaper kom je te weten:

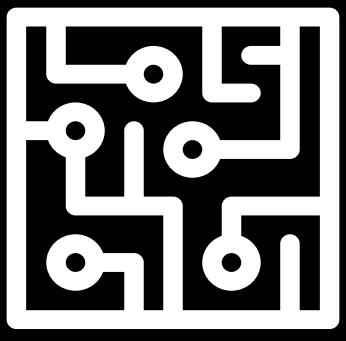
- Wat kenmerkend is aan een MACH-architectuur;
- Welke voordelen een MACH-architectuur heeft ten opzichte van een klassiek platform;
- Bij welke (business)doelen en cases MACH de moeite waard is om te overwegen;
- Op welke zeven punten je jouw organisatie klaar moet maken om met een MACH-architectuur aan de slag te gaan.

Over MACH: Microservices, API-first, Cloud-native SaaS en Headless

MACH is ontstaan als onderdeel van de ‘digitale transformatie’-vloedgolf. Het begon vooral als manier om grotere webapplicaties, mobiele kanalen en front-end-facing projecten te ontwerpen opgebouwd uit losse, onafhankelijke componenten van verschillende vendors. Nu dringt deze visie vanaf de buitenrand van bedrijven steeds dieper het hele IT-landschap binnen.

Componenten architectuur

Kenmerkend aan de MACH-architectuur zijn openheid en interoperabiliteit, geen monolieten maar applicaties die zijn opgebouwd in beheersbare componenten. Een MACH-architectuur wordt opgebouwd in (micro)services, die toegankelijk zijn via gestructureerde APIs.



MACH

M icroservices – In de context van MACH zijn microservices de componenten die de businessfunctionaliteit van het ecosysteem bevatten. De microservices doen ieder één specifiek ding, en doen dat goed. Ze zorgen, van nature, voor functionele decompositie. Dit kunnen services zijn die je als bedrijf zelf bouwt, omdat ze bijvoorbeeld nog niet in de markt bestaan, maar veruit de meeste services neem je af in de cloud. De benodigde services combineer je slim tot je eigen, unieke value chain.

A PIs – Ontsluiten van de services via heldere ontkoppelpunten, APIs, is cruciaal voor het 'composable' aspect van een MACH-architectuur. Een API is een façade en contract in één. Daarachter kan een service zich onafhankelijk van de rest ontwikkelen, terwijl gebruikers weten waar ze op kunnen vertrouwen.

C loud-native SaaS – MACH is pas echt schaalbaar als je het als service bij een vendor afneemt en zorgeloos draait en schaalt in de cloud. MACH is geboren in de cloud en direct beschikbaar zonder infrastructuurwerk.

H eadless – De front-end staat in MACH volledig los van de achterliggende businessfunctionaliteit, die is ingekapseld in microservices. Je kan userinterfaces variëren, testen, de volgorde en aanpak specifiek maken zonder je businessproces steeds te wijzigen. Unieke customer journeys, UX en snelle releases zijn immers kernwaarden van MACH.

Met een MACH-landschap hoef je je nauwelijks nog druk te maken over upgrades, schaalbaarheid en flexibiliteit. Het landschap draait grotendeels in de cloud en door de architectuur met microservices en APIs ontstaat er veel flexibiliteit en beheersbaarheid. Services vervangen, verbeteren of verwijderen gaat relatief eenvoudig, zonder dat alles helemaal op de schop moet. En doordat de back-end en systeemintegraties losstaan van de front-end ('headless'), stapt MACH volledig weg van grote, verticaal geïntegreerde platformen waarin de userinterface gewoonlijk sterk verbonden is met de achterliggende business logica.

Kortom, de 'composable' MACH-architectuur biedt, naast de flexibiliteit, de mogelijkheid tot een veel hogere verandersnelheid. In tegenstelling tot een klassiek (on-premise) platform dat vooral om een vastomlijnde aanpak vraagt.

MACH is misschien hip, maar wanneer is het waardevol?

Een unieke customer journey is steeds belangrijker geworden. En om die snel en effectief te kunnen aanpassen op veranderende omstandigheden. Eén van de uitdagingen van veel bedrijven is een uniek product versus een unieke beleving. Neem bijvoorbeeld maaltijdbezorgdiensten. Los van het feit dat bijvoorbeeld Thuisbezorgd.nl en Deliveroo een iets verschillende doelgroep willen aanspreken, is hun product in essentie hetzelfde: maaltijden bezorgen. Zij zullen echt moeten concurreren op beleving.

Om dat voor elkaar te krijgen is er onder andere een maatwerk front-end nodig. Misschien ook wel meerdere, zoals een (native) app, conversational commerce en website. Die front-ends wil je koppelen met een reeks van services die op verschillende manieren gecombineerd en uitgebreid kunnen worden, uiteraard op basis van de unieke propositie van jouw bedrijf. Dat vraagt dus om een 'composable architecture', in plaats van een platform dat een tamelijk strak business format biedt.

Frustraties over een klassiek platform

Veel bedrijven hebben geprobeerd om unieke belevingen, maar ook complex georganiseerde value chains, in de klassieke platformen in te richten. Ze maken daarvoor gebruik van enorme 'suites' van klassieke platform vendors. Een combinatie van grote CMS-systemen of DXP's, omvangrijke commercepakketten en uitgebreide marketingpakketten zijn eerder de standaard dan een uitzondering. Daarin zit dan wel van alles geïntegreerd: search, analyse, ranking, content, workflow, etc., maar brengt ook op een aantal punten uitdagingen met zich mee:

- De pakketten zijn **duur**. De up-front licentiekosten zijn vaak flink. Je betaalt daarmee gelijk het hele pakket, terwijl je er in het begin misschien niet eens volledig gebruik van maakt.
- De maatwerk extensies, complexe en statische integraties of andere afhankelijkheden, een flinke klus als je dat moet **upgraden**. 'Does it feel like you are upgrading your systems all the time?' is hier misschien wel één van de belangrijkste vragen. En bovendien is dat ook kostbaar. 'We geven intussen meer geld uit om het platform in de lucht te houden dan aan nieuwe businessfunctionaliteiten', is inmiddels – helaas – een welbekende quote.
- **Point releases met gedoe** zijn echt een ding. Ook het steeds weer opnieuw upgraden naar de nieuwste versie van bijvoorbeeld Sitecore, Adobe Experience Manager, Magento of SAP Commerce is tijdrovend en brengt ook gewoon veel kosten met zich mee.
- En wat dacht je van de **matige innovatiekracht** van de klassieke, verticaal geïntegreerde platformen? Innovatie gaat vaak traag. Er is misschien een jaarlijkse feature-release, maar voor je 'die ene handige feature van Google' ook hebt, ben je soms jaren verder. Dat moet allemaal veel sneller kunnen.
- Maar ook de **moeizame configuratie** is een welbekende frustratie. Met name de noodzaak om 100% van de kennis te hebben van een platform, om uiteindelijk misschien slechts 15% van de functionaliteiten te gebruiken.

Flexibiliteit met MACH

MACH lost veel hiervan op. Omdat je het landschap opbouwt in microservices, kun je heel specifiek kiezen wat je nodig hebt. Daardoor kun je ook gerichter investeren in kennis. Een MACH-landschap draait bovendien in de cloud, dus over schaalbaarheid, infrastructuur en eventuele piekbelastingvraagstukken hoef je je geen zorgen te maken. Alle benodigde services over de gehele value chain en customer journey kun je zelf kiezen en opbouwen met verschillende componenten, op een uniforme manier verbonden via APIs. Dat wordt met name mogelijk gemaakt door de mindset waarmee de SaaS MACH-producten zijn gebouwd. Bij een MACH-service is typisch heel goed nagedacht over de API, de primaire interface ervan, en is echt 'API-first' gedacht bij mogelijke interacties.

Meer voordelen dan nadelen?

Maar heeft MACH dan alleen maar voordelen? Nee, MACH heeft niet alleen maar voordelen en brengt ook nieuwe en andere uitdagingen met zich mee. Zie het beginnen met een MACH-architectuur als schetsen op een lege A4. Zo moet je onder meer in staat zijn om te bedenken hoe je een systeem opbouwt, welke services je kiest en waar je in het selectieproces op moet letten. Een klassiek platform biedt dan meer houvast, omdat er een duidelijke visie is op hoe zaken moeten werken.

Klassieke platformen bieden hulp en richting bij processen als: hoe een order wordt afgehandeld in een commerce-monoliet of hoe het CMS qua edit-flow werkt. In een DXP staat dat gewoon grotendeels vast. Variëren kost moeite, maar de default is 80% juist en bruikbaar.

Enerzijds geeft dat steun, maar het biedt dus soms wel wat minder vrijheid. MACH biedt in dat opzicht meer vrijheid. Als jouw customer journey weinig tot geen unieke aspecten nodig heeft, bijvoorbeeld omdat je product zelf al super uniek of niche is, dan blijft een klassiek platform overigens vaak de betere keuze. Er is dan simpelweg minder snel behoefte aan een unieke customer journey bij dat product.

Nieuwe uitdagingen met MACH

Meer complexiteit en integratiewerk

In tegenstelling tot een klassiek platform brengt MACH meer complexiteit. Je moet het totale landschap kunnen overzien, meer integratiewerk doen en het vraagt om meer eigen keuzes. De ontwikkeling van technische kennis is daarbij eigenlijk de grootste uitdaging om de overstap te kunnen maken naar MACH. Of je kiest voor een integratiepartner, zoals ISAAC, waarmee je die kennis en capabilities in huis haalt. Eigenlijk is dat één van de spannendste fases in de overstap.

Selectieproces vendors complex

Een andere, actuele uitdaging is de vendor selectie. Op dit moment is er nog geen

platform of community die hier goed bij kan helpen, zoals de Cloud Native Computing Foundation (CNCF) of, als je een beetje tussen de marketingsaus door kunt kijken, Gartner en Forrester dat aanbieden. Zij bieden houvast rond cloud infrastructuur en 'klassieke grote vendors', maar nog niet echt voor MACH-services. Eind juni werd er wel een eerste stap gezet met de lancering van de MACH Alliance.

Het gaat zeker een bijdrage leveren dat een industriegroep van technologiebedrijven een label heeft geplakt op de visie en architectuur met Microservices, API-first, Cloud Native SaaS en Headless. En dat zij de MACH-visie gaan uitdragen en verder laden. Maar daarmee zijn we er nog niet. Bij technische keuzes heb je behoefte aan 'stewardship'. Een écht onafhankelijk, breder overzicht van alle MACH-services met de mogelijkheden en beperkingen, waarin ook vendors zijn opgenomen die niet zijn aangesloten bij de MACH Alliance.

De MACH Alliance zou daarbij kunnen helpen door te doorgronden wat de positie is van services qua features en USP's. Maar ook door informatie over volwassenheid van de services te delen; hoe lang bestaat een service al, hoeveel klanten zitten erop en wat is de sfeer van tevredenheid? Bijvoorbeeld: welke branches zijn succesvol met dit MACH-CMS of deze MACH-Searchtool? Op dit moment is het de vraag of de MACH Alliance uiteindelijk ook een onafhankelijke adviesbaak wordt, maar het is wel iets waar bedrijven echt behoefte aan zullen hebben als ze concreet met MACH aan de slag willen.

Overstappen naar MACH: ja of nee?

Het antwoord op de vraag of je (snel) moet overstappen naar MACH is genuanceerd. Het is van behoorlijk wat factoren afhankelijk of het interessant is om met MACH aan de slag te gaan. Als je een 'green field' hebt en gaat starten met een nieuw landschap, al dan niet voor een nieuwe doelgroep of markt, is het altijd verstandig om MACH te overwegen. Vanwege flexibiliteit en schaalbaarheid is het natuurlijk interessant. En omdat je alleen betaalt voor het gebruik van de services, kan dat ook kostenaantrekkelijk zijn.

Heb je nu al een platform, bijvoorbeeld Magento, Hybris of Drupal en wil je een unieke beleving neerzetten of een complexe value chain goed in je platform laten samenkomen, dan is het natuurlijk weer een ander verhaal. Het is dan altijd een goed idee een business case te onderzoeken over de middellange termijn, drie jaar is meestal een goed uitgangspunt, voor re-platforming versus upgradekosten. Het is dan wel weer belangrijk om vooraf de business- en digitale strategie helder in kaart te brengen. Op basis daarvan kun je namelijk pas goed bepalen welke services in de nieuwe MACH-architectuur moeten zitten.

Ook de klassieke platform vendors zijn inmiddels naar MACH-services aan het bewegen. De vraag of je zou moeten overstappen wordt daarmee nog een stapje genuanceerder. Geerlings denkt dat slechts enkele klassieke platformspelers zich voldoende kunnen heruitvinden. Het gaat soms ook stap voor stap. Kijk maar naar 'Drupal Headless', de APIs van Magento 2 of een PIM als Informatica. Er zullen ook best wat platformspelers zijn die de boot missen, omdat ze te lang comfortabel hun bestaande licentiemodellen in stelling blijven brengen en ook maar beperkt innoveren. Ze houden dan uiteraard wel klanten over, naar verwachting de klanten die zelf ook langzaam meebewegen, maar de grote golf van re-platforming onder druk van accelererende digitale transformatie gaan deze spelers echt missen. Kortom, kijk uit dat je niet op het verkeerde platform wedt: analyseer daarom de MACH- en cloudplannen van je huidige vendor om te bepalen welke keuze uiteindelijk het beste bij jouw bedrijf past.

MACH-architectuur: hoe bereid je jouw bedrijf er-op voor?

Wanneer duidelijk is dat je met MACH aan de slag wil gaan, of dat op z'n minst wil onderzoeken, zou iedere organisatie zichzelf en aantal vragen moeten stellen. MACH is nu eenmaal qua organisatiebehoefte echt heel anders dan een klassiek (on-premise of IaaS) platform. Dit hoofdstuk richt zich dan ook iets meer op de tactisch-operationele kant van MACH. ISAAC CTO Friso Geerlings vertelt op welke zeven punten je jouw organisatie moet klaarmaken om met een MACH-architectuur aan de slag te gaan.

#1 Is mijn organisatie (enigszins) Agile?

De 'composable' MACH-architectuur biedt veel flexibiliteit en de mogelijkheid tot een veel hogere verandersnelheid. Het Agile-gedachtegoed sluit daar het beste op aan. Dat hoeft op zich niet te betekenen dat het hele bedrijf op voorhand Agile is ingericht, maar wel op z'n minst de afdeling of business unit waar je met MACH gaat beginnen.

Dat proces begint vervolgens met het aanwijzen van een goede, liefst ervaren Product Owner. En die, in het geval van MACH, ook goed begrijpt wat er technisch nodig is. Rond de Product Owner vorm je een zelfsturend team dat met name in staat is om de technische release pipeline optimaal te beheersen.

De technische release pipeline is zo belangrijk, omdat je vanaf het begin wil profiteren van de hoge verandersnelheid. De eerste teamdoelen zijn daarom het releaseproces en Continuous Deployment (CD) onder controle hebben. Wanneer dat nog niet helemaal lekker loopt, dan is het een goed idee nog vaker te releasen. 'If it hurts, do it more often', is niet voor niets een bekende uitspraak rond releaseprocessen en CD. Agile is een randvoorwaarde om dit proces goed ingericht te krijgen.

Een andere reden waarom het belangrijk is om Agile te organiseren, zijn de dynamische, cyclische processen. Bij het opzetten van MACH gaat het continu over de verhouding tussen services kiezen en het ontwerp van de complete architectuur. Dat hebben wij bij ISAAC ook duidelijk gemerkt in een project waarin we een MACH-architectuur ontworpen. Zo kozen we in eerste instantie voor Contentful als headless CMS. Gedurende het proces kwamen we erachter dat we beter Storyblok konden inzetten. Het bleek namelijk dat Storyblok, op dat moment, betere previewmogelijkheden had, wat één van de requirements was. Zo'n voorbeeld benadrukt ook echt het belang van het voorbereidingsproces.

#2 Heeft mijn front-end team voldoende programmeer skills?

MACH services bieden APIs aan die worden ontsloten in één of meerdere front-end, de headless component van de architectuur. De front-ender met een vormgevings- of multimediadesign-achtergrond zal daar niet zo goed mee uit de voeten kunnen. Dat is een belangrijk verschil met een klassiek platform waar het werk zich meer toespitst op visualisatie. Voor een front-end developer die met MACH-services aan de slag gaat, is het veel belangrijker om technische basisvaardigheden en software development kwaliteiten te bezitten. De front-end wordt boven op een set van MACH-services gebouwd en verbonden via APIs. Kennis van frameworks als Vue, React en brede ervaring met JavaScript zijn dan noodzakelijk.

Besef dat een MACH-landschap geheel naar eigen inzicht is samen te stellen. De front-ends kunnen uniek zijn en integratie is maatwerk. Het is een ecosysteem dat je optuigt vanuit een eigen, unieke businessbehoefte. En aangezien het succes van producten en diensten staat met eindgebruikers, zijn UX en User Testing daarom nog belangrijker dan anders. Je wil namelijk wel slagen met een unieke beleving én optimale conversie. Tijdig concepten en prototypen toetsen onder eindgebruikers en hun inzichten gebruiken om de ontwerpen weer te verbeteren, is het devies.

MACH Front-end frameworks voor meer ontwikkelsnelheid

Dankzij de 'H' van Headless geeft MACH je de totale vrijheid bij de keuze van front-ends. Vue.js of React, een flitsende native app of zelfs voice of machine-to-machine communicatie, het kan allemaal op dezelfde manier met de businesslogica interacteren en via alle kanalen een consistente customer journey realiseren.

Die vrijheid kent alleen wel een prijs. Front-end developers in een MACH-ecosysteem staan voor een stevige bouwopdracht en hebben daarvoor up-to-date kennis nodig. Gelukkig zijn er steeds meer MACH-ready front-end frameworks beschikbaar die een deel van het (kostbare) werk voor je doen. Denk aan tools als Vue Storefront of het Falcon PWA Storefront van Diety, die ook bij ISAAC in Eindhoven worden gebruikt.

In de praktijk zien we bij dit soort MACH front-end frameworks de releases flink kunnen versnellen. Soms scheelt het wel honderden ontwikkeluren. De extra afhankelijkheid die je creëert is wel een nadeel, want juist in de front-end wil je snel aanpassingen kunnen doen. Wees daarom, ook bij de front-end, kritisch tijdens de vendorselectie. Zorg dat je goed begrijpt wat je koopt, wat de roadmap van het product is, wat de exstensibility is (makkelijke dingen moeten makkelijk zijn, maar moeilijke dingen moeten wel mogelijk zijn) en hoe je toch zo duidelijk mogelijk een grens trekt tussen wat je zelf bouwt en de frameworks waarmee het ontwikkelteam mee aan de slag gaat.

#3 Hebben we de juiste integratie skills aan boord?

De ontwikkeling van technische kennis is voor bedrijven verreweg de grootste uitdaging om de overstap naar een MACH-architectuur te kunnen maken. Afhankelijk van de omvang van het landschap groeit de technische complexiteit. Wanneer het MACH-landschap nog klein is en bestaat uit twee tot drie services, dan zijn services in eerste instantie nog wel point-to-point te koppelen. Zodra het landschap groter

wordt, ontstaat vaak al gauw de behoefte om een integratie- en ontsluitingslaag, zoals een API Gateway, te introduceren. Deze laag wordt ingezet om dataoverdracht en (API) logica te regelen. En dat wordt helemaal belangrijk wanneer ook externe services van partners aan jouw MACH-landschap worden gekoppeld.

Dat vraagt kennis van gangbare technieken zoals REST, APIs, JSON, JWT en OpenID Connect, maar bij wat meer ingewikkelde set-ups misschien ook kennis van message queueing, AWS of Azure Serverless, Webhooks en GraphQL. Ook de API Gateway is een voorbeeld. Uitdenken en ontwikkelen van een API strategy, inclusief API design, API versioning en bijbehorende developer experience, zijn typische integratieskills die bij een groot MACH-landschap belangrijk zijn om aan boord te hebben. MACH-services zijn nou eenmaal niet zo plug & play: ze zijn bedoeld om door developers gebruikt te worden. Er zit meer ontwikkelwerk aan en wat minder configuratie zoals bij een klassiek platform, waar development teams trouwens vaak wel enthousiast van worden.

GraphQL is “Query for APIs”: een vraagtaal zoals SQL dat is voor databases. Hoewel GraphQL nog betrekkelijk nieuw is, maakt het een stormachtige ontwikkeling door in API-land. Met name als belangrijke aanvulling op REST.

Front-ends worden nog flexibeler en losgekoppeld van de achterliggende services door de vrijheid die GraphQL ontwikkelaars biedt. Ook in reporting- en event gebaseerde contexten is het krachtig. Steeds meer MACH-vendors hebben ook een GraphQL-interface op hun services beschikbaar. Zeker iets om op te letten bij de selectie van de componenten van je ecosysteem.

Bedrijven die met een complex project als MACH aan de slag gaan, sluiten vaak digitale partners aan, zoals ISAAC, om kennis naar binnen te halen. Daarmee geven ze het

project een goede kickstart, beschikken ze makkelijker over kennis van complexe, specialistische aspecten en wordt de basisontwikkeling en integratiewerk vaak aan de partner uitbesteed. Als de basis eenmaal goed staat, worden – na uitgebreide kennisoverdracht – de partner resources vaak weer afgeschaald om zelf verder te gaan.

De belangrijkste uitdaging bij MACH is het integreren. Hoewel de services zelf, met hun krachtige en zorgvuldig voor jou ontworpen datamodellen en logica, goede functionaliteiten bieden, kun je met de integratie echt de mist in gaan. Voor die cruciale aspecten moet je gewoon de juiste kennis aan boord hebben, in-house of via een digitale partner.

#4 Security en inloggen, hoe richt ik dat in?

In tegenstelling tot een klassiek platform dat een geïntegreerde, centrale login heeft, moet je dat logischerwijs voor een MACH-architectuur zelf inrichten. Soms blijft een MACH-landschap beperkt tot één, twee of misschien drie gekoppelde services. Als dat zo is, zijn security en inloggen vaak afdoende geregeld binnen de afzonderlijke services. Dubbel inlog- en rechtenbeheerwerk is dan nog te overzien. Als het landschap groter wordt of gebruik gaat maken van een integratielaag, wordt het een ander verhaal. Rollen, toegangsrechten en credentials zijn dan over verschillende microservices verspreid en security is ook niet zomaar end-to-end geregeld.

Daarmee begeef je je op het terrein van Identity en Access Management (I&AM) en Single Sign-On (SSO). Bij een groeiend MACH-landschap, is het een goed idee om I&AM en SSO uit de services te nemen en apart onder te brengen in een Identity en Access Management platform zoals AWS Cognito. Daarmee kun je zowel SSO als role based access control goed inrichten en de werking ervan aantonen. Zeker als MACH-services worden ingezet in een certified- of regulated markt landschap, bijvoorbeeldISO 27001, PCI-DSS of in het zorgdomein, biedt deze set-up veel voordelen rond auditability en identity centralisatie.

#5 Wat moet ik weten over de kenmerken van een goede MACH test-set?

De MACH-architectuur is heel testbaar en geschikt voor geautomatiseerde teststraten, die ook zo belangrijk zijn om vaak te kunnen releasen. Omdat MACH-services via APIs worden ontsloten, kun je per service aan de slag met testen. Het totale landschap is daardoor in veel mindere mate een black box dan bij een klassiek platform.

Het is gebruikelijk om een aparte Quality Assurance (QA) omgeving op te tuigen en daarin volop in te testen en te proberen. Daarin wil je in ieder geval een goede, geautomatiseerde test-set opzetten die vertrouwen geeft. De MACH-architectuur stelt je daartoe in staat, doordat alles via APIs is aan te sturen en dus te automatiseren. Release often' is een strategie die perfect bij MACH past, maar wel alleen als de test-set op orde is.

Ook het op orde hebben van test data provisioning is belangrijk, maar lang niet vanzelfsprekend. De beschikbaarheid van goede, realistische testdata en die data goed in de (test) MACH-service krijgen, zijn vaak wel een ding. Geautomatiseerd kunnen inladen van testdata zou daarom een criteria moeten zijn bij de vendor-keuze. Wellicht zou je op termijn ook kunnen investeren in automatisch anonimiseren van data uit je productieomgeving voor testdoeleinden.

#6 Hoe houd ik de kosten onder controle?

Iedere MACH-service wordt afgerekend via een maandelijkse fee. Dat klinkt in eerste instantie heel overzichtelijk en misschien ook wel aantrekkelijk. Bij de implementatie van een klassiek platform betaal je flinke up-front licentiekosten. Met MACH-servicesheb je de vrijheid om 'fit-for-use' te kiezen en kosten zouden daardoor moeten meegroeien met de groei van de organisatie.

En toch kun je daar ook kanttekeningen bij plaatsen. Voordat je iets kiest, is het

belangrijk om goed te begrijpen hoe het pricingmodel van een service in elkaar zit. Wanneer je de ambitie hebt om bijvoorbeeld te schalen naar grote aantallen 'low value' gebruikers, dan kunnen services met pricingmodellen die per eindgebruiker afrekenen ineens behoorlijk in de papieren gaan lopen. In zo'n geval wil je misschien liever betalen voor daadwerkelijk gebruik, bijvoorbeeld het aantal API calls of bundels van API calls. Daarbij, als het MACH-ecosysteem groeit, ontstaat ook de noodzaak voor Single Sign-On. En dan komt vaak de enterprise versie van de MACH-services om de hoek kijken om dat te ondersteunen: SSO zit typisch niet in de goedkope instap-tiers. En die enterprise versies zijn gewoon best prijzig.

Dat betekent dus dat een MACH-landschap in veel gevallen niet goedkoper is dan een klassiek platform. Uiteraard is klein beginnen een voordeel. Met enkele services of weinig gebruikers, waardoor de opstartkosten voor een overstap wel een stuk aantrekkelijker zijn dan bij een monolithisch platform. Het is dan ook aan te raden om de business case voor een MACH-landschap te bouwen rond andere aspecten dan alleen het vermeende kostenvoordeel. Onderzoek het vanuit invalshoeken als business agility, de mogelijkheid om een unieke beleving te creëren of complexe value chains beter onder te brengen. Op die punten biedt MACH echt toegevoegde waarde, als dat voor jouw business relevant is.

#7 Hoe monitor ik business en techniek in een MACH-architectuur?

Een MACH-architectuur wil je eigenlijk op twee manieren monitoren en daar twee cockpits voor inrichten. Een business operationele cockpit waarmee bijvoorbeeld omzet en revenue worden doorgemeten. En een technisch operationele cockpit waarmee performance van de systemen wordt doorgemeten. Door je totale performance gedetailleerd inzichtelijk te maken, kun je eventuele problemen snel signaleren en lokaliseren. Denk hierbij bijvoorbeeld aan een combinatie van Application Performance Monitoring (APM), Analytics en BI. En uiteraard kies je ook hiervoor tools die helemaal MACH-ready zijn.

Tot slot

Het antwoord op de vraag of je (snel) moet overstappen naar MACH is genuanceerd en het opzetten van een MACH-architectuur behelst veel meer dan het puur in gebruik nemen van een online SaaS-platform.

We vatten graag de belangrijkste inzichten in deze whitepaper voor je samen:

- Het is van behoorlijk wat factoren afhankelijk of het voor jouw bedrijf interessant is om met MACH aan de slag te gaan. Wat is de uitgangspositie: een 'green field' of een bestaand platform, wat zijn businessdoelen en hoe zit het met een unieke beleving versus hoe uniek je product of dienst is? Neem je digitale strategie in alle gevallen als startpunt.
- Een MACH-landschap is in veel gevallen niet goedkoper dan een klassiek platform. Het is aan te raden om de business case voor een MACH-landschap te bouwen rond andere aspecten dan alleen het vermeende kostenvoordeel. Onderzoek het vanuit invalshoeken als business agility, de mogelijkheid om een unieke beleving te creëren of complexe value chains beter onder te brengen. Op die punten biedt MACH echt toegevoegde waarde, als dat voor jouw business relevant is.
- Eenmaal besloten om aan de slag te gaan met MACH? Dan is het veruit het belangrijkste om ervaren integratie- en front-end specialisten aan boord te hebben, of als partner in de arm te nemen. Agility is ook een voorwaarde, maar iets dat je vaak wel ingericht krijgt of al hebt. De technische kant van MACH is behoorlijk complex, in ieder geval als het landschap een beetje omvang krijgt en de term ecosysteem verdient.
- Begin met een klein experiment met twee of drie services. Als je gaat opschalen, maak dan heel bewust je volgende keuze. Dat klinkt voordehand liggend, maar is niet per se eenvoudig. Hoewel MACH je per component eenvoudig keuzes laat heroverwegen, is het uiteindelijk moeilijker om het 'cement' van het landschap te vervangen. Test tijdig, monitor wijs, beveilig en integreer met expertise.

Tot slot, in het begin van de whitepaper stelden we de vraag: 'Is MACH het nieuwe cloud of net zo invloedrijk als SaaS?' Het is fijn dat er nu een term is die helpt om onderscheid te maken tussen services die maar voor een deel MACH zijn en services die volwaardig aan alle aspecten voldoen. Of het net zo invloedrijk gaat worden, zal de tijd uitwijzen.

Over de expert

Friso Geerlings – Chief Technology Officer bij ISAAC

Met een brede technische achtergrond richt Friso het vizier van ISAAC op effectieve, veelal open source, internettechnologie om de meest complexe vraagstukken voor klanten te voorzien van een effectieve digitale strategie en oplossing. De hoofdfocus van Friso ligt op customer experience, webtechnologie, APIs, data-driven decisions, transacties, highperformance systemen en integratie. De brug tussen developers, gebruikers en systemen bouwen staat voor hem al jaren centraal.

Over Digital Agency ISAAC

Als een digitale transformatie een complexe uitdaging is, dan zien wij volop kansen. Wij helpen bedrijven met digitale strategie, design, development en integratie, voor meer online succes.

Bij ISAAC beschikken we over drie kerneigenschappen. **Strategische diepgang** zie je aan de manier waarop we met digitale vraagstukken omgaan. We vragen door om vraagstukken en doelen écht inzichtelijk te maken. En we beschikken niet alleen over veel technische kennis. Dankzij onze jarenlange ervaring hebben we ook veel kennis van digital commerce, integratie en financial services. **Daadkracht** vertaalt zich in onze 'getting things done'-mentaliteit. We zijn gefocust op leveren. En wat we opleveren, werkt. Meteen. Onze 130 specialisten in Eindhoven leveren uitsluitend **op maat**. Voor onze projectteams is het business-as-usual om een aanvulling te zijn op in-house IT-teams en nieuwe oplossingen naadloos te integreren met bestaande systemen en processen.

Onze visie, manier van denken en werken, vertaalt zich al jaren in klinkende resultaten in het hart van de digitale kanalen van onze klanten. We combineren technische expertise, ervaring en de eigenschappen van ons team. Dat maakt ons als Digital Agency een strategisch digitaal partner. Zo kunnen we software development versnellen, risico's eruit halen en weten we ook de meest complexe vraagstukken succesvol te transformeren. We love **.com**plexity!

Een greep uit ons portfolio

ingenico

EDCO

Diebold Nixdorf

Signify

Atradius Collections

DMG

payvision

van Dale

Qander. CONSUMER FINANCE

CM.

DAF

BNP PARIBAS

+31 40 2908979
welkom@isaac.nl
www.isaac.nl

Marconilaan 16
5621AA Eindhoven