JDriven Tech Radar Voorjaar 2022

Onze kijk op recente ontwikkelingen in het veld

Commit to know.

Develop to grow.

Share to show.

jdriven

Introductie

Onze ervaren specialisten werken dagelijks mee aan tal van softwareprojecten in Nederland en zijn betrokken in wereldwijde community's. Halfjaarlijks komen wij vanuit JDriven bij elkaar om te bespreken wat wij aan nieuwe trends en ontwikkelingen zien. Deze trends proberen wij te vangen in een technologieradar. Elke editie van de radar laat verschuivingen zien t.o.v. een vorige editie. Een verschuiving kan betekenen dat wij een technologie interessanter zien worden waar van toepassing, of juist minder geschikt ongeacht de toepassing. Indien een trend niet meer voorkomt in een latere editie, dan zijn er geen nieuwe ontwikkelingen en/of ervaringen die ons eerdere beeld zouden hebben bijgesteld. In dit document willen we toelichten welke verschuivingen we de afgelopen periode hebben waargenomen, om op basis daarvan weer richting te geven aan wat wij inzetten en aanraden.

Tech Radar

Het idee voor het opstellen van een tech radar komt voort vanuit Thoughtworks. Zij dragen al langer periodiek met een radar hun visie uit op nieuwe trends en ontwikkelingen. Bovendien raden zij iedereen aan een eigen radar op te stellen.

Bij JDriven onderschrijven we dat. Het opstellen van een Radar is een leerzame en waardevolle ervaring waarin onderling kennis wordt gedeeld en een technisch bewustzijn wordt gecreëerd. Wij geloven erin dat specialisten zelf in staat moeten zijn om het gereedschap voor hun werkzaamheden samen te stellen. Met het opstellen van een radar faciliteer je discussies over technologie, om als organisatie de juiste balans te vinden in wat voor risico's en voordelen innovatie kan opleveren. Wij kunnen u helpen dit op te starten binnen uw organisatie. Laat uw teams elkaar inspireren tot innovatie en gezamenlijk komen tot een set aan technologieën en technieken die de ontwikkeling in uw bedrijf versnellen.

Indeling

Een radar bestaat uit kwadranten en ringen, met daarbinnen blips om interessante technologieën en technieken aan te duiden.

De kwadranten verdelen de verschillende onderwerpen in categorieën.

- **Talen & frameworks** die je ondersteunen bij de ontwikkeling van software
- **Platformen** waarop je software kunt uitvoeren
- **Technieken** die je helpen betere software te maken
- **Tools** ter ondersteuning van je ontwikkel- en delivery-proces

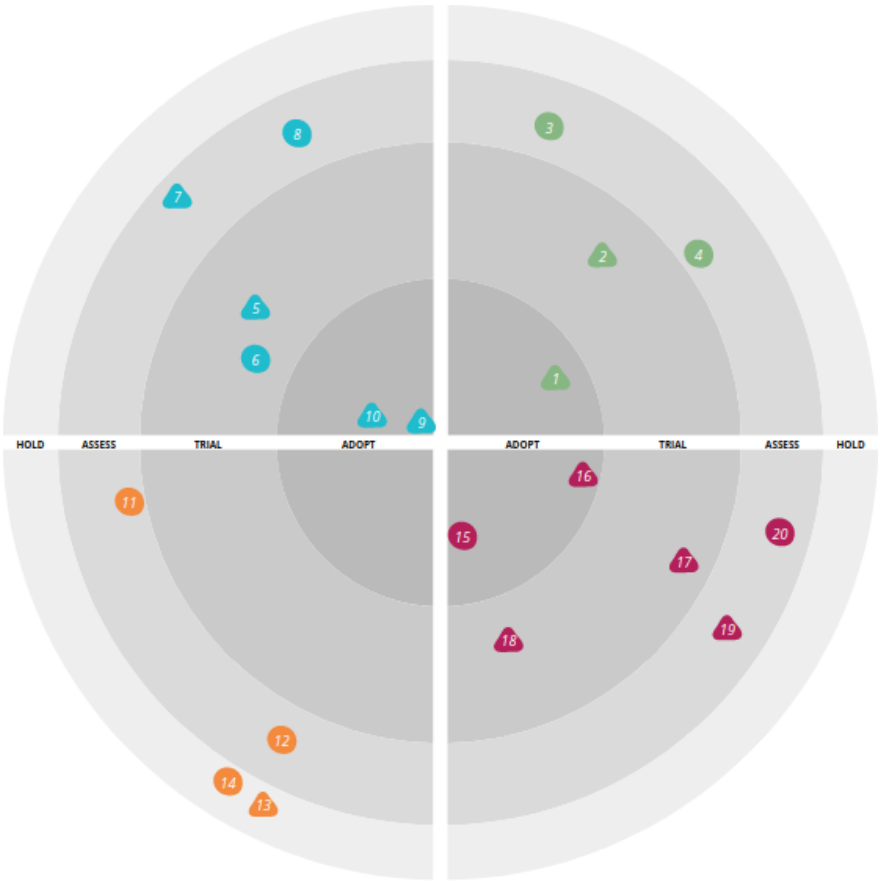
De ringen in elk kwadrant geven aan in welk stadium van adoptie wij denken dat die technologie zich bevindt.

- **Adopt** → Wij raden sterk aan deze technologie te gebruiken, waar van toepassing.
- **Trial** → Interessant om alvast ervaring mee op te doen (in een project dat het risico kan dragen).
- **Assess** → Goed om beter te begrijpen en toekomstige impact in te schatten, maar nog niet om toe te passen.
- **Hold** → Niet (meer) gebruiken.

In de volgende secties zullen we onze kijk op de recente ontwikkelingen per kwadrant toelichten.

Commit. Develop. Share.

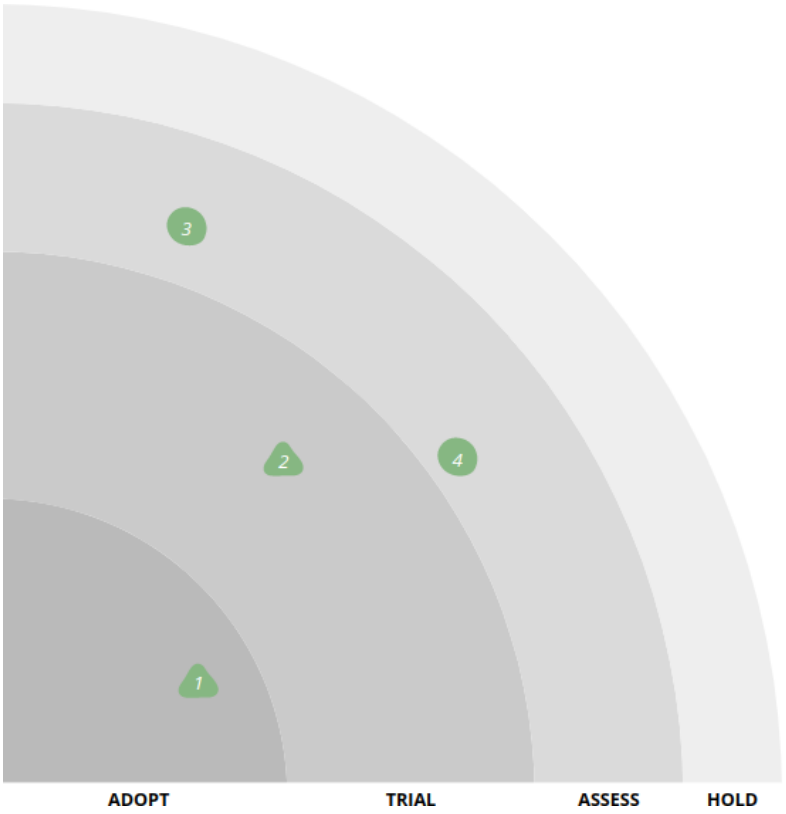
© 2022 JDriven



Tools	Talen & frameworks
ADOPT Renovate Spotless **TRIAL** Kustomize Sentry **ASSESS** GitHub Copilot OpenTelemetry **HOLD** -	**ADOPT** Kotlin Coroutines **TRIAL** Unity3D **ASSESS** jMolecules WASM serverless **HOLD** -
Platforms	**Technieken**
ADOPT - **TRIAL** - **ASSESS** TimescaleDB Yugabyte **HOLD** CircleCI Kalix	**ADOPT** Kanban Testing diamond **TRIAL** Low-code Mob Programming **ASSESS** Linked data Quantum Technology **HOLD** -

Commit. Develop. Share.

Talen & frameworks



ADOPT

1. Kotlin Coroutines

TRIAL

2. Unity3D

ASSESS

3. jMolecules
4. WASM serverless

Kotlin Coroutines

ADOPT

Kotlin Coroutines is een first-class Kotlin library voor het schrijven van leesbare en onderhoudbare asynchronous, non-blocking code. De library is ontwikkeld door JetBrains en bouwt verder op de low-level *suspending function* APIs in Kotlin.

Wanneer je een applicatie schrijft in Kotlin waarbij je gebruikmaakt van asynchronous, non-blocking code, dan is Coroutines de de facto standaard. Het is stabiel, wordt goed ondersteund door IntelliJ en maakt het schrijven van multi-threaded code een stuk eenvoudiger en beter onderhoudbaar.

Commit. Develop. Share.

Unity3D

TRIAL

Unity3D is een marktleider op het gebied van game engines. Unity is uitgebracht in 2005, initieel exclusief voor macOS, maar is al jaren ook beschikbaar op Linux en Windows. Over de jaren heen heeft Unity3D zich bewezen als een developer-vriendelijke game engine, met een eerlijk licentiemodel, en ontwikkeld volgens het "build once, deploy anywhere" principe. Nu wordt het gebruikt door miljoenen developers wereldwijd, die er jaarlijks duizenden games en applicaties mee ontwikkelen. Unity3D heeft zich afgelopen jaren ook naar andere sectoren vertakt, zoals de animatie-, auto- en de bouw- & constructiesectoren. Daarmee heeft Unity3D een complete ontwikkeloplossing voor interactieve 3D-toepassingen. Deze uitbreiding heeft voor veel nieuwe gebruikers en hernieuwde interesse in de engine gezorgd.

Het is ons opgevallen dat veel organisaties die met AR/VR of 3D-visualisaties van hun data experimenteren, voor Unity3D kiezen. De toepassing van Unity3D buiten de traditionele game-omgeving, en de toenemende noodzaak om 3D-data te visualiseren, heeft ons overgehaald om Unity3D in onze tech radar van dit voorjaar toe te voegen. Als je als organisatie aan het innoveren bent met 3D-data, of oplossingen zoekt voor om jouw AR/VR-concepten tot leven te brengen, dan is Unity3D zeker een goede keus.

jMolecules

ASSESS

jMolecules is een set libraries met twee duidelijke doelen. Ten eerste helpt het bij het uitdrukken van DDD-concepten. Wanneer je nadrukkelijk de Ubiquitous Language voor class names gebruikt, wil je deze niet vervuilen met de benaming van de DDD-bouwblokken. jMolecules heeft annotaties om toch te kunnen aangeven welk type bouwblok deze classes representeren. Als alternatief heeft jMolecules ook Java interfaces gebaseerd op John Sullivan's series "Advancing Enterprise DDD". Hiermee kun je relaties tussen DDD-concepten in een domeinmodel middels het type system van Java vastleggen.

Ten tweede kan je met jMolecules expliciet markeren hoe een stuk code (package, class) een concept in softwarearchitectuur vervult. Denk hierbij aan het kunnen aangeven waar een component hoort in een layered- of onion-architecture.

Door het vastleggen van concepten uit DDD of softwarearchitectuur in de code, biedt dit ontwikkelaars naast conventies voor softwaremodellering extra duidelijkheid over de intentie van een stuk code. Omdat concepten in de code vastliggen, is het mogelijk om met geautomatiseerde tooling (ArchUnit, jqassistant) te verifiëren en documenteren of een implementatie voldoet aan de regels en concepten van een gekozen architectuur.

Commit. Develop. Share.

WASM serverless

ASSESS

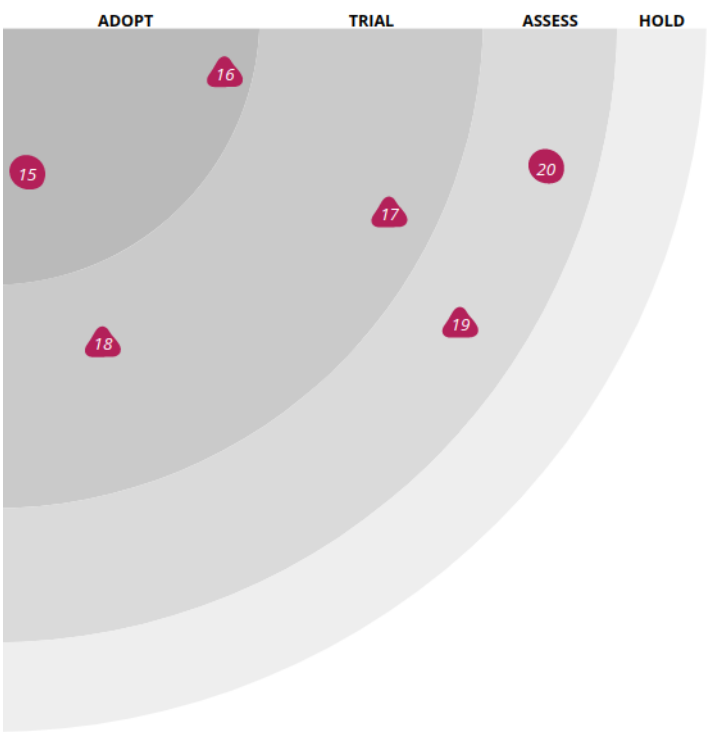
WebAssembly (Wasm) is een nieuwe taal die performance, security en portability biedt. Er zijn meerdere toepassingen voor Wasm. Een daarvan is Wasm voor serverless, omdat het zo snel is als native, geen cold-start problemen heeft en de developer een brede taalkeuze geeft. Zo kun je bijvoorbeeld Kotlin gebruiken om te compileren naar Wasm.

Momenteel bieden de grote cloud providers nog geen support voor Wasm, maar een Wasm VM (bijv. WasmEdge) kan gemakkelijk naast Node.js draaien in een lambda zonder performanceverlies. Daarmee win je dus wel vrijheid in taalkeuze, betere security en, als Wasm wel direct ondersteund wordt, ook performancewinst.

Wasm is nog volop in ontwikkeling, maar wat het mist aan volwassenheid wordt gecompenseerd door potentie. Vooral de potentie om Kotlin te kunnen schrijven voor serverless functions, met daarbij een performance- en securitywinst, maakt de techniek waardevol om verder te onderzoeken.

Commit. Develop. Share.

Technieken



ADOPT

15. Kanban
16. Testing diamond

TRIAL

17. Low-code
18. Mob Programming

ASSESS

19. Linked data
20. Quantum Technology

Kanban

ADOPT

In de software-industrie, is Scrum de meestgekozen methodiek om software te maken en te beheren. Het heeft zich bewezen als een betrouwbare manier van werken. Alleen soms is zelfs Scrum te langzaam voor projecten. De vele rituelen, strikte 2/3-wekelijkse sprints, en het toch minder open staan voor veranderingen tijdens de sprint zijn remmende factoren die sommige projecten zich niet kunnen veroorloven. Wat wij vaak zien is dat organisaties dan Scrum "forceren" terwijl er prima alternatieven zijn die beter passen bij zulke projecten.

Kanban is zo een alternatief. Het is inmiddels een vaak gekozen techniek voor project management. De methode visualiseert zowel het proces (workflow) als het daadwerkelijke werk dat het proces doorloopt. Het heeft tot doel om eventuele problemen snel te identificeren en op te lossen. Bij Kanban word niet zo strak vastgehouden aan 2 wekelijkse sprints, daarnaast zijn de rollen flexibeler in te vullen door meerdere teamleden en kan het team makkelijker omgaan met veranderingen. User stories worden opgepakt gebaseerd op prioriteit, en er is ruimte om belangrijke user stories snel op te pakken. Stakeholders hebben meer ruimte om dringende verzoeken snel opgepakt te zien worden.

Als een project een continue oplevering van functionaliteit moet waarborgen en/of kleine user stories (denk aan bug fixes, feature requests) binnenkomen die snel geïmplementeerd kunnen worden, dan is Kanban het overwegen waard.

Commit. Develop. Share.

Testing diamond

ADOPT

Om het testen van software naar een hoger niveau te tillen wordt vaak gekozen voor het groeperen van verschillende soorten testen. Een veelgebruikte onderverdeling is in de volgende drie categorieën: * UI- en end-to-end testen * Integratietesten * Unittesten

In een traditionele software-omgeving ligt de focus op het testen van de units; daarna volgen de integratietesten en als laatste de UI-testen. Vaak wordt dat weergeven in een piramidevorm. De belangrijkste testen vormen de voet van de piramide, de andere testen vullen de tweede laag en de top. Het gebruik van de 'testpiramide' is een goede manier om monolieten door te testen.

In het huidige software-landschap is het gebruik van de testpiramide echter niet toereikend, doordat er vaak gebruikgemaakt wordt van meerdere microservices. De eerdergenoemde groeperingen zijn echter nog steeds relevant, maar een goede werking van de onderlinge verhouding tussen de services is van groter belang dan de afzonderlijke units. Wanneer je dit uittekent, verandert de piramide naar een andere vorm: de diamant. Vanuit dat oogpunt zijn de integratietesten nu het hart van je test suites, de andere testgroepen zijn secundair. De 'testdiamant' is daarom een uitstekende methode om complexe multi-service architecturen testbaar te maken.

Low-code

TRIAL

Middels een low- of een no-code platform kunnen businessgebruikers applicaties ontwikkelen zonder te coderen. Het “programmeren” gaat met de hulp van user interfaces om processen en bronnen aan elkaar te modelleren. Bij een no-code platform wordt alles via de user interface gedaan. Bij een low-code platform is het wel mogelijk om code toe te voegen om het doel te bereiken.

Low-code en ook no-code platformen zitten in de lift en vormen een mogelijke oplossing voor de groeiende vraag naar ICT-oplossingen en het gebrek aan software engineers.

De voordelen van het gebruik van deze platformen zitten in tijd en beschikbaarheid van schaarse resources zoals programmeurs.

Het gebruik van deze platformen heeft ook nadelen. Zo zit je vast aan de betreffende vendor die het platform levert en is de functionaliteit die je kan realiseren beperkt door de grenzen van wat het platform kan leveren. Ook de integratie met externe systemen kan lastig zijn, zeker wanneer er niet over een standaard protocol gecommuniceerd wordt. Tevens moet je als bedrijf bewust zijn dat wanneer er eenmaal voor een platform gekozen is het lastig is om te migreren naar een andere oplossing.

Low-code en no-code platformen zijn geschikt om snel een MVP op te leveren. Belangrijk hierbij is dat de processen die geautomatiseerd worden relatief simpel zijn en weinig afhankelijkheden hebben. Wanneer je zelf aan de slag wilt met een low-code platform lees dan ook dit artikel van Thoughtworks.

Commit. Develop. Share.

Mob Programming

TRIAL

Mob Programming is een extremere vorm van pair programming waarbij het hele team tegelijk aan dezelfde feature werkt. Tijdens het mob programmen is er één driver achter het toetsenbord en de rest van het team geeft de driver instructies.

Hoewel dit op het eerste gezicht inefficiënt lijkt zijn wij van mening dat het ook enkele voordelen biedt: - Kennisdeling; Doordat het hele team ermee bezig is, is de kennis goed verdeeld - Training; Junioren krijgen veel ondersteuning op deze manier - Geen review meer nodig - Meerdere perspectieven

Hoewel het niet voor ieder team of probleem zal werken hebben wij er in de praktijk goede ervaringen mee.

Linked data

ASSESS

Linked Data is een techniek waarbij data aan elkaar gekoppeld en geïnterpreteerd kan worden zonder dat de data verplaatst hoeft te worden. Met Linked Data bouw je een gestructureerd netwerk op van je data, en dat kan dan gebruikt worden voor rijkere analyses, en om makkelijker je data, met context, te delen. In academische kringen gaat de term 'Linked Data' al jaren rond. Alleen is er de afgelopen tijd een toegenomen interesse voor de techniek, en is Linked Data het academische perspectief ontstegen. De technologie ontwikkelt zich razendsnel door. Dat komt doordat er allerlei standaarden zijn gedefinieerd om basiszaken te beschrijven. Zulke standaarden beschrijven bijvoorbeeld hoe je een adres kan beschrijven en interpreteren. Dankzij de inmiddels volwassen standaardisatie is Linked Data makkelijker toe te passen. Het Kadaster stelt bijvoorbeeld publieke Linked Data endpoints beschikbaar zodat je hun datasets aan elkaar kunt verbinden. Als je aan het onderzoeken bent hoe je je data kunt opslaan op een manier die makkelijk interpreteerbaar is voor machines, en die ook gekoppeld kan worden aan andere datasets, dan stellen wij voor om Linked Data uit te proberen.

Commit. Develop. Share.

Quantum Technology

ASSESS

Quantum technology houdt zich bezig met het natuurkundige fenomeen, dat sommige deeltjes zich in een zogenaamde superpositie kunnen bevinden. In een dergelijk geval is hun waarde niet alleen uit te drukken in een digitale 0 of 1, maar in waarschijnlijkheden dat ze die waardes hebben. Pas wanneer dit gemeten wordt, nemen ze een waarde aan. Computerbits die dit gedrag vertonen, noemen we quantum bits, of kortweg qubits. Bedrijven als QuTech houden zich bezig met het bouwen van de hardware met en rondom qubits. De kracht hiervan is dat problemen, die met klassieke computers exponentieel moeilijker worden om op te lossen met het introduceren van extra variabelen, met quantumcomputers slechts lineair moeilijker worden.

Terwijl de hardware zich nog ontwikkelt, kunnen mensen zich al voorbereiden op de komst hiervan. De logische circuits die geprogrammeerd moeten worden om dergelijke oplossingen met qubits uit te denken, vereisen een nieuw soort denkwijze. Er zijn programmeertalen zoals Strange en Qiskit waarmee dergelijke circuits al geschreven kunnen worden in o.a. Java en Python, en computers en SaaS diensten die simulatoren of echte quantumcomputers aanbieden om dergelijke programma's te draaien. Het is momenteel vooral zaak om problemen die met quantumcomputers beter (of uitsluitend) opgelost kunnen worden te leren identificeren. Denk aan het traveling salesman probleem.

Een aparte eigenschap van qubits in superpositie is dat ze met elkaar verstrengeld kunnen worden. Wanneer één van de twee dan wordt uitgelezen, ongeacht de afstand tussen de qubits, zal de waarde van de ander automatisch ook direct bepaald zijn. Deze eigenschap heeft impact op de security van nu; met name op het vlak van cryptografie. Dit besef heeft o.a. geleid tot de totstandkoming van veilige quantumnetwerken. Ook hier is de hardware nog in ontwikkeling, en dienen componenten van het OSI-model verder uitgewerkt te worden. Te zijner tijd zullen architecturen een plek kunnen gaan geven aan dergelijke netwerken.

Commit. Develop. Share.

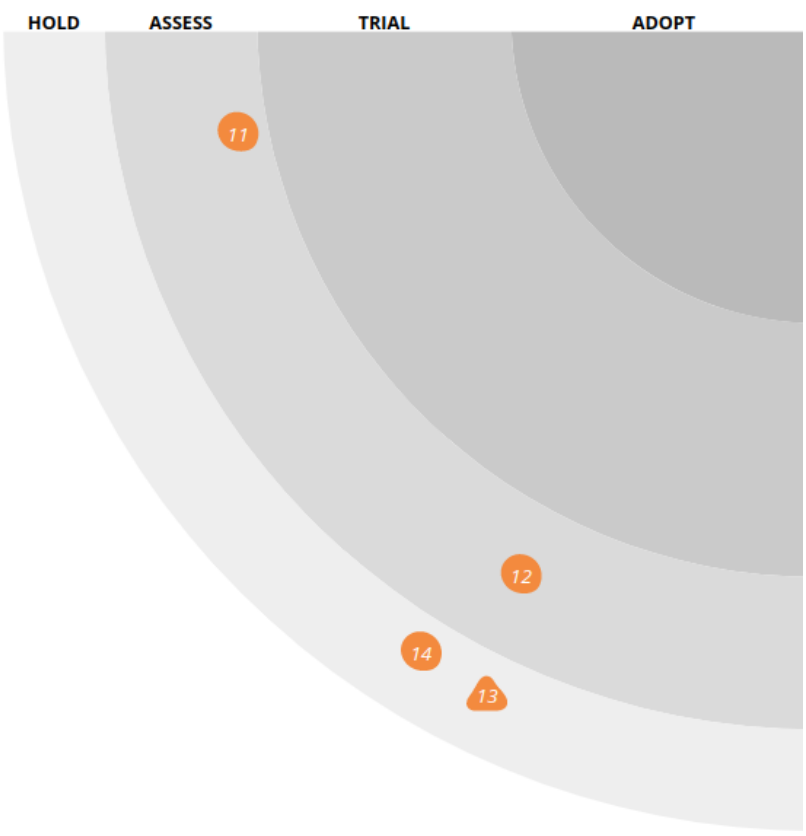
Platforms

ASSESS

11. TimescaleDB
12. Yugabyte

HOLD

13. CircleCI
14. Kalix



TimescaleDB

ASSESS

TimescaleDB is een interessante nieuwe speler op het gebied van timeseries databases. Concreet is het een open-source extensie op PostgreSQL die timeseries capabilities toevoegt aan de database. Een belangrijk aspect daarbij is de belofte dat timeseries data 94% minder diskopslag nodig heeft door het toepassen van op timeseries data toegespitste compressie.

De API van de database blijft op basis van SQL met volledig behoud van alle normale functionaliteit en compatibiliteit met normaal gebruik van PostgreSQL. Daarmee biedt het een unieke all-in-one oplossing voor software met zowel een relationele als timeseries database-behoefte.

De database heeft zijn eigen licentie voor de geavanceerde features (TSL), waarbij eigenlijk alles mag behalve het als eigen SaaS-dienst (DBaaS) doorverkopen. Niet verwonderlijk aangezien het bedrijf achter de database recentelijk 40 miljoen heeft opgehaald in een nieuwe investeringsronde ten behoeve van het opzetten en uitbouwen van hun eigen SaaS-dienst. Deze SaaS-dienst is 5 oktober 2021 live gegaan in drie AWS regions met o.a. auto-scaling van disk en point-in-time recovery. Daarmee biedt het wat je tegenwoordig van een DBaaS mag verwachten. Ook zelf deployen van TimescaleDB is simpel zat doordat verschillende partijen, waaronder TimescaleDB zelf, open-source OCI containers aanbieden met daarin de TimescaleDB extensie.

Wij zien TimescaleDB als een goede optie om te verkennen voor teams die reeds werken met PostgreSQL en een groeiende behoefte hebben aan het opslaan van timeseries data.

Commit. Develop. Share.

Yugabyte

ASSESS

YugabyteDB is een horizontaal schaalbare cloud-native database die qua performance beter scoort dan bijvoorbeeld Cassandra. En dat terwijl YugabyteDB wel altijd consistentie garandeert. Vanaf versie 2.0 wordt echter niet alleen YugabyteDB's eigen op Cassandra geïnspireerde YCQL ondersteund maar ook een volledig PostgreSQL-compatible SQL API. Daarmee is het gelijk ook een van de enige horizontaal schaalbare SQL-databases. Anders dan de directe concurrent CockroachDB biedt YugabyteDB wel ondersteuning voor de volledige PostgreSQL query API met zaken als triggers en stored procedures.

De database, inclusief alle enterprise features, is volledig open-source. Dit vanwege de strategie om geld te verdienen met het leveren van DBaaS-diensten (SaaS) en niet met de database zelf. Op dit moment bieden ze zowel de optie tot het managen van een eigen DBaaS-service op eigen infrastructuur als een volledig gemanagede cloud-variant. Ook biedt Yugabyte zelf open-source helm charts en een operator voor deployment op Kubernetes. Dit samen met de horizontale schaalbaarheid maakt dat het ook echt als high available cluster op Kubernetes kan worden ingezet. Anders dan bijvoorbeeld PostgreSQL waarbij high availability met externe tooling moet worden ingeregeld en de schaalbaarheid beperkt blijft.

Wij zien YugabyteDB als goede optie voor een schaalbare database. Zowel self-managed direct op Kubernetes als vanuit de DBaaS-opties vanuit Yugabyte.

CircleCI

HOLD

CircleCI is een Continuous Integration en Continuous Deployment (CI/CD) cloud platform waarmee automatisch software gecompileerd, getest en gedeployed kan worden. CircleCI ondersteunt zeer complexe build pipelines op meerdere besturingssystemen (Linux, macOS en Windows) en hardware platforms (o.a. x86_64 en ARM).

Via integraties met onder andere GitHub, GitLab en Bitbucket kan automatisch bij elke commit een nieuwe build worden gestart. Builds draaien binnen Docker containers en pipelines worden geconfigureerd met een YAML file binnen de Git repository van een project.

Hoewel CircleCI een prima tool is, biedt het voor de meeste projecten weinig tot geen toegevoegde waarde boven tools als GitHub Actions, GitLab CI en Bitbucket Pipelines, terwijl er wel extra kosten aan verbonden zijn. Pas als een pipeline van een project baat heeft bij een hoge mate van parallellisatie of wanneer builds op verschillende (hardware) platforms nodig zijn, raden wij aan om naar CircleCI te kijken.

Kalix

HOLD

Kalix is een 4GL cloud framework om APIs te maken met eenvoudige database-integratie. Hiermee kun je eenvoudig technieken zoals Event Sourcing, CQRS en CRDTs gebruiken. Het gebruikt gangbare technieken zoals Protobuf, Sbt en Docker. Het werkt met verschillende talen: Scala, Java, JavaScript, TypeScript. Let op! De naam is gewijzigd. Vroeger heette het Akka Serverless. Kalix is inderdaad een veel betere naam, want het heeft niks met Akka te maken.

Commit. Develop. Share.

Tools

ADOPT

9. Renovate
10. Spotless

TRIAL

5. Kustomize
6. Sentry

ASSESS

7. GitHub Copilot
8. OpenTelemetry

Renovate

ADOPT

Renovate is een open source tool voor het automatiseren van dependency updates, vergelijkbaar met Dependabot welke we al in het voorjaar van 2020 behandelden. Bovenop wat we al gezien hebben van Dependabot, biedt Renovate ons een eenvoudige flow voor het oppikken van nieuwe projecten, met minimale configuratie. Maar waar Renovate zich in onderscheidt, is de officiële ondersteuning voor meerdere platformen zoals Azure, BitBucket & GitLab. Dit maakt het een aantrekkelijke optie voor iedereen die geen GitHub gebruikt, nadat Dependabot door GitHub is opgekocht en geïntegreerd.

Wij willen Renovate in het bijzonder uitlichten voor iedereen die geen GitHub gebruikt, om dezelfde voordelen te genieten op andere platforms. Gegeven wat we het afgelopen jaar hebben gezien met kwetsbaarheden met hoge gevoeligheid is er genoeg reden om bij te blijven, en automatisering helpt ons daarbij.

Commit. Develop. Share.

Spotless

ADOPT

Spotless is een tool voor het controleren (en toepassen) van code formatting. Waar spotless zich in onderscheidt, is het feit dat het een build tool plugin is dat meerdere talen ondersteunt. Hierdoor is het makkelijk te integreren in projecten en pipelines.

Om deze reden zien wij Spotless als een goede oplossing om discussies over code style & formatting uit het team weg te nemen, zeker met de opkomst van meer diverse teams en ontwikkelomgevingen.

Kustomize

TRIAL

Kustomize is een template engine voor Kubernetes configuraties. Het maakt het mogelijk om een eenduidig declaratief systeem te gebruiken om de volledige configuratie te specificeren voor meerdere deployment targets, zoals *test*, *staging* en *productie*. Kustomize doet dit door middel van een systeem van lagen van templates, de overlays. Elke overlay heeft de mogelijkheid de details van de laag erboven in te vullen of aan te passen. Het gebruik van Kustomize verenigt dit templatesysteem voor de gehele applicatiecontext, en wordt zowel voor in-house tools als voor off-the-shelf applicaties gebruikt. Bovendien is Kustomize volledig geïntegreerd in de standaard Kubernetes `kubectl` tool sinds versie 1.14. Dit maakt het gemakkelijk om Kustomize toe te passen in bestaande systemen voor continuous deployment. Door zijn aard als templatesysteem komt de syntax van Kustomize in hoge mate overeen met de reguliere Kubernetes configuratiebestanden.

Hoewel de templating syntax voor Kustomize af en toe gecompliceerd kan voorkomen, maakt de gelaagde architectuur het eenvoudig om een reguliere Kubernetes configuratie stap-voor-stap aan te passen aan Kustomize. Dit zorgt er ook voor dat het eenvoudiger is om overzicht te houden over de verschillen in configuratie van de deployments. Omdat de reguliere Kubernetes configuratie (met wat kleine aanpassingen) ook een correcte Kustomize configuratie is, zitten er weinig tot geen nadelen aan het uitproberen van Kustomize.

Sentry

TRIAL

Sentry is een cross-platform applicatiemonitoring-tool met een focus op foutrapportages. Er zijn SDK's voor vrijwel elk platform, of het nou serverless, backend, frontend of mobiele applicaties betreft. Allemaal sturen ze optredende fouten voorzien van relevante context naar een Sentry backend-server. Daarbij heeft Sentry intelligente ontdubbeling, en heeft het verscheidene alerting-instellingen en integraties. De backend-server is ofwel hosted af te nemen, zelf te hosten, of geïntegreerd met GitLab te gebruiken.

In onze ervaring is dit een onmisbare tool gebleken om onverwachte fouten aan het licht te brengen en te volgen, ongeacht de technologiestack. Daarnaast is de prijsstelling vriendelijker ten opzichte van bredere monitoring-oplossingen als DataDog, en kun je er eenvoudig op kleine schaal mee beginnen.

Commit. Develop. Share.

GitHub Copilot

ASSESS

GitHub Copilot is een code generation assistent en plugin aangedreven door Codex, een nieuw AI-systeem van OpenAI. Door de AI getraind op code in publieke GitHub repositories is GitHub Copilot beter in het genereren van context-gerelateerde code dan andere code assistents. Hierdoor kan het code of tekst genereren in elke context, zij het documentatie, commentaar, functies of code.

Github Copilot stuurt een deel van de data in de huidige file naar de server, waar deze door automatische processen wordt geanalyseerd en door de AI wordt gehaald voor suggesties. Alle stappen worden opgeslagen voor toekomstige voorspellingen. Aangeraden wordt om voorzichtig om te gaan met tool zoals deze.

Voorlopig is er geen license en worden gebruikers toegelaten op basis van een wachtlijst.

OpenTelemetry

ASSESS

OpenTelemetry beoogt een eenvoudige, universele, vendor-neutrale, loosely coupled oplossing te zijn voor logging, metrics & tracing. Gesteund door de Cloud Native Compute Foundation en de grote spelers in de observability-wereld, heeft dit goede kans op termijn onze projecten te raken. Er zijn tools, API's en SDK's beschikbaar in verschillende talen voor het instrumenteren van applicaties, verzamelen van meetdata en om deze metrics, logs en traces te exporteren. Daarmee wordt het mogelijk de performance en het gedrag van applicaties te analyseren in een backend naar keuze.

Wij zien vooral de mogelijkheden om middels deze taal- en vendor-onafhankelijke standaard eindelijk één oplossing te hebben die herhaaldelijk is toe te passen, ongeacht de lokale situatie. Voor Java is er al een indrukwekkende set aan integraties met bestaande frameworks, waaronder een Java agent om daar snel mee te kunnen starten.

Commit. Develop. Share.

Commit.
Develop.
Share.