

BIJ12 architectuur en kaders en richtlijnen

6 december 2019
versie 0.5

Colofon

<i>Document informatie</i>	
<i>Titel</i>	BIJ12 architectuur en kaders en richtlijnen
<i>Auteur</i>	Roger Smits
<i>Versie</i>	1.0
<i>Status</i>	Definitief
<i>Datum</i>	31 maart 2020
<i>Bestandsnaam</i>	
<i>ISO Document</i>	
<i>(ISO) Proces</i>	

	<i>Naam</i>	<i>Rol</i>
Documenteigenaar	Roger Smits	Informatieadviseur / Programma manager
Proceseigenaar	Jolinda van der Endt	Directeur BIJ12
Procesverantwoordelijke	Karin Cox	Informatie manager

<i>Versiebeheer/wijzigingshistorie</i>				
<i>Versie</i>	<i>Status</i>	<i>Datum</i>	<i>Beschrijving</i>	<i>Auteur</i>
01	concept	15 juli 2019	Opzet en structuur document	Roger Smits
02	concept	25 juli 2019	Opmerkingen verwerkt	Roger Smits
03	concept	21 aug 2019	Nieuwe inspiratie	Roger Smits
04	concept	2 okt 2019	Reviews KC, JH, SV, IE verwerkt	Ivo Eggink
05	concept	6 dec 2019	Review BH verwerkt en SSS en CSS toegevoegd	Roger Smits
06	Def. concept	11 dec 2019	Voorwoord en aanleiding toegevoegd en redactieslag	Karin Cox
1.0	Definitief		Reactie MThee 23 maart verwerkt	Karin Cox

Inhoudsopgave

1	Algemeen	6
1.1	Aanleiding	6
1.2	Doel en scope van dit document	6
1.3	'Pas toe of leg uit'	6
1.4	IV-team voert regie op en coördineert ontwikkeling applicatielandschap	6
2	Architectuur	8
2.1	Multitier	8
2.2	Lagen	9
2.3	BIJ12 architectuur.....	10
2.4	Voordelen van de BIJ12 architectuur	11
3	Programmeertalen en frameworks	12
3.1	Wat is een programmeertaal?	12
3.2	Wat is een software framework?	12
3.3	Context.....	12
3.4	Het probleem.....	12
3.5	Keuze voor programmeertaal en frameworks.....	12
4	Basisfunctionaliteiten	16
4.1	Basiscomponent.....	16
4.2	Voorkeurs-software	18
5	BIJ12 architectuur plaat	21
5.1	BIJ12 architectuurplaat - schematisch	22
6	BIJ12 ontwikkelomgeving	23
6.1	Opzet	23

Voorwoord

BIJ12 werkt aan een professionaliseringsslag op het gebied van informatiemanagement. In 2018 is het informatiebeleid 2018-2021 opgesteld, met daarin afspraken over het waarom, wat en waarmee voor goed informatiebeheer, voor meer overzicht op het applicatielandschap en meer grip op de kosten van de informatievoorziening te krijgen.

Het informatiebeleid is begin 2020 geactualiseerd en opnieuw vastgesteld, onder de noemer: Informatiebeleid BIJ12 2020-2021 (versie 2, d.d. 31 maart 2020). Het informatiebeleid geeft antwoord op de vraag hoe BIJ12 sturing geeft aan haar informatiehuishouding. Een van de speerpunten van het informatiebeleid is 'werken onder architectuur': vaste afspraken over hoe informatievoorzieningen eruit zien. Wat dit inhoudt, is verder uitgewerkt in het voorliggende document: de BIJ12 architectuurstuursystemen en richtlijnen.

Waarom dit nodig is? Stel we doen niet aan architectuur en informatieplanning:

- Hoe weten we dan wat er de komende jaren geïnvesteerd moet worden in de vernieuwing van IT (software en hardware)?
- Als een systeem of platform vervangen moet worden, hoe weet ik dan wat er geraakt wordt? Waar ontstaat overlap en waar vallen gaten?
- Als we nieuwe diensten willen aanbieden aan provincies en/of ketenpartners, hoe weten we dan wat we daarvoor moeten doen/aanpassen?
- Als we gegevens willen/moeten uitwisselen met ketenpartners, waar zitten die dan? En hebben we die gegevens eigenlijk wel? En moeten we voor iedere ketenpartner iets unieks maken?
- Als we niet alles tegelijk kunnen, wat moeten we dan eerst doen? Wat daarna?

IV-team stelt de kaders

Het BIJ12-brede team Informatievoorziening (IV-team) geeft invulling aan het informatiebeleid en stelt de kaders en richtlijnen voor het werken onder architectuur. Met als doel: een efficiënter en overzichtelijk applicatielandschap (overzicht), helderheid over de afhankelijkheden, overlap en eventuele gaten tussen de verschillende werkprocessen, interne & externe informatiestromen, IT-voorzieningen en verleende diensten (inzicht in de 'samenhang der dingen') en regie voeren op verandering (bepalen wat eerst moet en wat later kan) om toe te groeien naar een toekomstbestendige, efficiënte en effectieve interne en externe informatievoorziening.

Dit document bevat de kaders voor de verdere inrichting en (door)ontwikkeling van de BIJ12-informatievoorziening. Het is een groeidocument dat wordt aangepast als daar aanleiding toe is.

Aansluiten bij architectuurprincipes overheid

Standaard wordt binnen de overheid gebruik gemaakt van de referentiearchitectuur NORA (Nederlandse Overheid Referentiearchitectuur). De kaders en principes in dit architectuurstuurstelsel zijn een aanvulling en/of verbijzondering van de NORA-principes voor BIJ12. Het toepassen van de architectuurstuursystemen en principes helpt de organisatie om de 'waarheen', 'wat', 'hoe' en 'waarmee' vragen te beantwoorden.

- **Waarheen:** Waar wil de organisatie over 3 tot 5 jaar staan?
- **Wat:** Welke bedrijfsprocessen zijn daarvoor nodig en welke bedrijfsobjecten zijn daarvoor relevant?
- **Hoe:** Hoe willen we onze medewerkers en middelen samen inzetten om de bedrijfsactiviteiten uit te voeren?
- **Waarmee:** Welke medewerkers hebben we hiervoor nodig?
- **Waarmee:** Welke (geautomatiseerde) voorzieningen hebben we hiervoor nodig?

Wellicht ten overvloede: dit document vervangt eerder opgestelde architectuurstuurstelsels voor delen van de BIJ12-organisatie. De eerdere documenten komen hiermee te vervallen.

1 Algemeen

1.1 Aanleiding

BIJ12 werkt aan een toekomstbestendige interne en externe informatievoorziening voor de provincies waarmee zij snel, (kosten)efficiënt en effectief kan inspelen op toekomstige ontwikkelingen zoals (het digitaal stelsel voor) de omgevingswet en zij steeds complexere informatievragen kan beantwoorden, die o.a. een gevolg zijn van de verdergaande digitalisering van de provinciale werkprocessen.

Kort gezegd: digitale data en informatie worden steeds belangrijker. BIJ12 is haar Informatievoorziening aan het optimaliseren om de juiste data en informatie op het juiste moment in het proces beschikbaar te kunnen stellen.

Makkelijker gezegd dan gedaan. Momenteel zijn namelijk nog niet alle applicaties en voorzieningen in lijn met elkaar. Het applicatielandschap is diffuus en lastig te beheeren. Er worden verschillende technologieën gebruikt voor eenzelfde soort functionaliteit en een deel van de technologie is verouderd en toe aan vervanging. Om ervoor te zorgen dat vernieuwingen op een eenduidige manier worden gerealiseerd, is het nodig om afspraken te maken over hoe het applicatielandschap (de IV) eruit ziet. Daarvoor dient dit document.

1.2 Doel en scope van dit document

Dit document beschrijft de architectuurkaders, richtlijnen en principes waarmee BIJ12 werkt aan het grip krijgen en houden op het aanschaffen, ontwikkelen/ bouwen, aanpassen en koppelen van IV/ IT-onderdelen voor de organisatie.

Het zijn algemeen toepasbare regels en richtlijnen om de uitvoering van de BIJ12-werkprocessen en de informatie en IV-ondersteuning die daarvoor nodig is, te faciliteren. Het team Informatievoorziening (IV-team) toetst op de naleving hiervan.

Het betreft een startdocument, een levend groeidocument dat wordt aangepast en/of uitgebreid als daar aanleiding voor is.

De kaders en richtlijnen voor de BIJ12 informatievoorziening worden gebruikt bij het formuleren en beoordelen van projectvoorstellen en voor systeemontwerpen. Daarnaast vormen ze een belangrijk instrument bij de uitvoering van projecten. Voor ieder project wordt daarom een Project Start Architectuur (PSA) opgesteld. Hierin wordt bij aanvang van een project aangegeven in hoeverre wordt voldaan aan de BIJ12-kaders en richtlijnen en ook welke aanvullende project-specifieke ontwerpkeuzen zijn gemaakt. Dit document biedt een praktische referentie voor projecten die te maken hebben met de interprovinciale informatievoorziening.

1.3 'Pas toe of leg uit'

BIJ12 hanteert het principe 'pas toe of leg uit' voor de in dit document opgenomen kaders en richtlijnen. Dit betekent dat als niet kan worden voldaan aan de kaders en richtlijnen, gemotiveerd moet worden waarom wordt afgeweken en wat de consequenties hiervan zijn op korte en lange termijn. Beoogde afwijkingen van de BIJ12-architectuur worden onderbouwd met argumenten voorgelegd aan het IV-team. De voorzitter van het IV-team besluit over de voorgestelde afwijking en legt verantwoording hierover af aan de directeur van BIJ12.

1.4 IV-team voert regie op en coördineert ontwikkeling applicatielandschap

Het IV-team concentreert zich op uniformiteit in ontwikkeltrajecten en borgt de informatiebeheerprocessen van BIJ12. Het IV-team checkt of aan de I-kaders en richtlijnen wordt voldaan en of de ontwikkelingen binnen BIJ12 uniform, volgens de architectuurafspraken en met de juiste prioriteit plaatsvinden. Het IV-team adviseert ook bij de overdracht van ontwikkeling van informatievoorzieningen naar beheer.

Het IV-team adviseert het MT en legt verantwoording af aan de directeur. De directeur is de eigenaar van het informatiebeleid en bijbehorende documenten, waaronder het architectuurdokument, het MT is gebruiker ervan. Het IV-team is kader stellend, maar niet beslissingsbevoegd; de directeur stelt het informatiebeleid vast.

Voor inhoudelijke keuzes over de informatievoorziening die gevolgen hebben voor de unit-begroting(en) of voor beleidsafspraken, stelt het IV-team een advies op voor het MT. De beslissingsbevoegdheid voor de inhoud ligt dan bij het MT. Wanneer geadviseerd wordt vanuit de toezichtstaken van het IV-team, heeft dit een dwingend karakter richting het MT.

2 Architectuur

Binnen BIJ12 worden diverse applicaties (door)ontwikkeld, aangeschaft en beheerd. Het is daarom belangrijk om een generieke architectuur op te stellen waaraan de voorzieningen moeten voldoen en/of waarnaar bestaande applicaties op termijn toe kunnen groeien.

De criteria / architectuurprincipes waaraan de architectuur moet voldoen, zijn:

- **Flexibiliteit;** *er wordt gewerkt met generieke koppelingen en veelgebruikte programmeertalen en software, waardoor wijzigingen eenvoudig en snel door te voeren zijn*
- **Uitwisselbaarheid:** *werken onder service-georiënteerde architectuur (SOA), conform open standaarden voor overheidsorganisaties;*
- **Schaalbaarheid;**
- **Herbruikbaarheid:** *hergebruik, voor standaard, voor maatwerk;*
- **Overdraagbaarheid:** *(applicatie) diensten en informatie kunnen verplaatst worden naar een andere leverancier intern of extern.*
- **Data vanuit de bron:** *data heeft één bron van waarheid / één oorsprong, de bron is eigenaar van de informatie en stelt deze ter beschikking*
- **Beheerbaarheid:** *applicaties zijn modulair opgebouwd met basiscomponenten. Dat zorgt voor een overzichtelijk en eenduidig applicatielandschap.*
- **Betrouwbaarheid en beschikbaarheid:** *data en informatie van een gekende kwaliteit (gevalideerd en inhoudelijk juist) en gegarandeerde uptime van applicaties conform afgesproken KPI's; weinig risico op uitval*
- **Informatieveiligheid en duurzame toegankelijkheid:** *gegevens worden opgeslagen en beheerd conform de beveiligingsprincipes BIJ12, gebaseerd op de Baseline Informatiebeveiliging Overheid (BIO) en ISO 27001/27002, de richtlijn archivering websites en de Wet Open Overheid*

In dit hoofdstuk wordt nader ingegaan op de architectuur. Het eerste gedeelte behandelt de fysieke scheiding (denk hierbij aan servers) van een applicatie; daarna wordt een applicatie onderverdeeld in conceptueel logische componenten. Tot slot zullen de fysieke en logische scheiding van een applicatie worden samengevoegd en wordt de applicatiearchitectuur visueel weergegeven.

De meeste applicaties binnen BIJ12 zijn client-server applicaties (de gebruiker interacteert via een webbrowser met de applicatie die op een server 'draait'). De client-server structuur is daarom als uitgangspunt genomen in de uitwerking van de architectuur.

2.1 Multitier

2.1.1 Wat is multitier?

In softwareontwikkeling is de multitier architectuur een client-server architectuur waarin presentatie, applicatieverwerking en datamanagement functies fysiek zijn gescheiden. Het idee hierachter is om functionaliteit te ontkoppelen en hergebruik te stimuleren. Met een multitier architectuur kunnen nieuwe technologieën en andere componenten geïntegreerd worden zonder dat de hele applicatie opnieuw hoeft worden gebouwd. Dit maakt het beheer en de schaalbaarheid gemakkelijker. In het kader van informatiebeveiliging wordt gevoelige informatie veilig opgeslagen in de desbetreffende tier en is deze niet rechtstreeks verbonden met het internet.

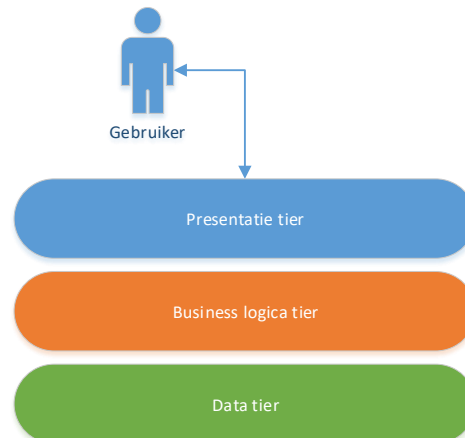
2.1.2 3-Tier architectuur

De 3-tier architectuur wordt mondiaal veel gebruikt en is min of meer de standaard om client-server applicaties op te zetten. De belangrijkste voordelen zijn: onafhankelijkheid, flexibiliteit, schaalbaarheid, herbruikbaarheid, beheerbaarheid en beveiliging.

De 3-tiers architectuur bestaat uit de volgende fysiek gescheiden tiers:

- *Presentatie tier*: bevat gebruiker georiënteerde functionaliteit en is verantwoordelijk voor het managen van de gebruikersinteractie met het systeem;
- *Business logica tier*: implementeert de kernfunctionaliteit van het systeem en omvat de relevante business logica;
- *Data tier*: omvat de data toegang en data opslag van het systeem.

In het onderstaande is de 3-tier architectuur schematisch weergegeven:



Bijna alle applicaties binnen BIJ12 hebben een client-server opzet en daardoor is het een logische keuze voor BIJ12 om applicaties volgens deze architectuur op te bouwen.

2.2 Lagen

In het voorgaande is de fysieke scheiding, de tiers, van een applicatie besproken. Deze scheiding vormt de fysieke basis waarop de verschillende softwarecomponenten van een applicatie 'draaien'. Door 'lagen' te benoemen, kan de opbouw van een applicatie verder worden uitgewerkt. Belangrijke criteria bij de uitwerking zijn wederom: flexibiliteit, schaalbaarheid, herbruikbaarheid en beheerbaarheid.

2.2.1 Wat is een laag?

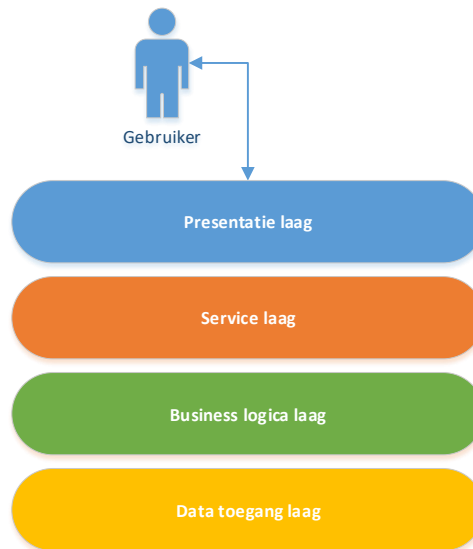
Een laag is een logische groepering van softwarecomponenten waaruit een applicatie bestaat. Dit helpt te differentiëren tussen de verschillende taken die de verschillende componenten uitvoeren. Het splitsen van een applicatie in afzonderlijke lagen die verschillende rollen en functionaliteiten hebben, maximaliseert de beheerbaarheid van de code. Daarnaast helpt de heldere splitsing om te bepalen waar beslissingen over technologie of ontwerp beslissingen moeten worden gemaakt.

2.2.2 De lagen

Zoals eerdergenoemd, zijn de meeste applicaties binnen BIJ12 client-server applicaties. Het onderscheid in tiers beschrijft alleen de fysieke scheiding, maar niet hoe de verschillende softwarecomponenten met elkaar samen dienen te werken. De lagen die 'standaard' worden gedefinieerd zijn: presentatie-, business- en data laag. Deze lagen communiceren rechtstreeks met elkaar en zijn daardoor verweven.

Om de lagen te 'ontkoppelen' en ondersteuning te bieden aan een servicegerichte architectuur (afgekort SGA) benoemt BIJ12 zogenoemde 'service lagen'. Deze 'service lagen' worden gepositioneerd tussen de presentatie-, business- en data laag. De 'ontkoppeling' heeft een positief effect op de flexibiliteit, herbruikbaarheid en beheerbaarheid.

In de onderstaande figuur wordt de door BIJ12 gedefinieerde lagenstructuur weergegeven:

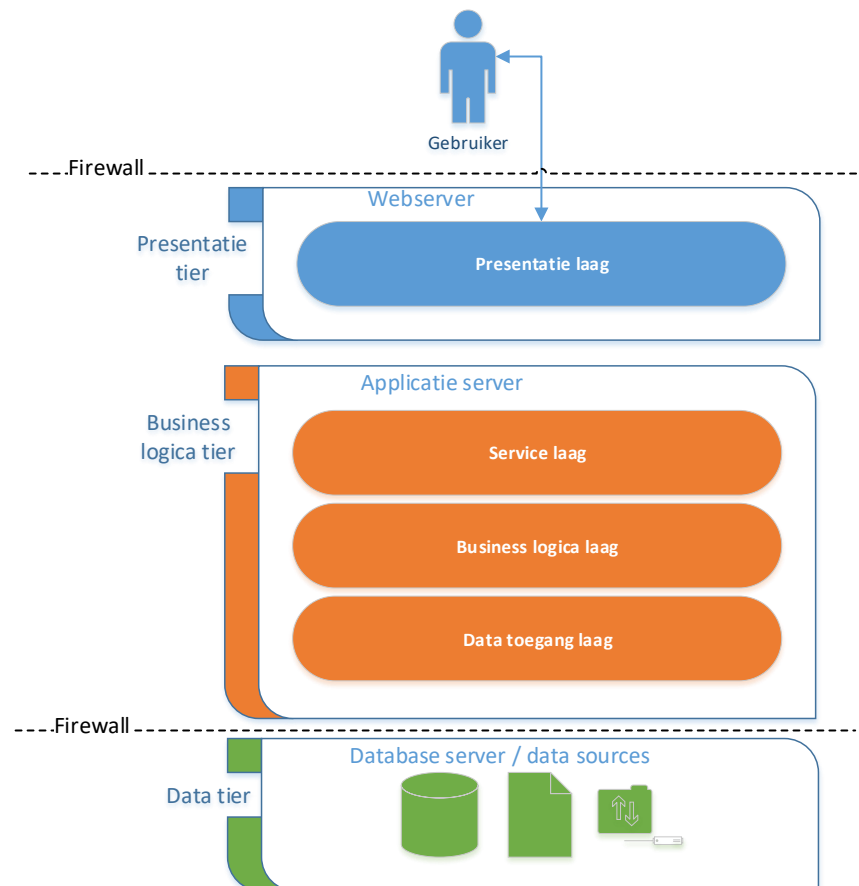


2.3

BIJ12 architectuur

De BIJ12 architectuur voor (client-server) applicaties is een combinatie van fysieke scheiding ('tiers') en functionele softwarematige scheiding ('lagen'). Daarnaast is het belangrijk dat de data door een extra firewall wordt afgeschermd en in de zogenaamde 'secure zone' wordt opgeslagen.

De BIJ12 architectuur is in het onderstaande figuur weergegeven:



2.4 Voordelen van de BIJ12 architectuur

De belangrijkste voordelen om een applicatie volgens architectuur op te bouwen, zijn:

- 1) Beheerbaarheid stack: het biedt de mogelijkheid de technologie stack in één tier te updaten met minimale impact op de andere tiers.
- 2) Ontwikkelaars kunnen ingezet worden op hun expertise. De meeste ontwikkelaars zijn geen 'full stack' ontwikkelaars, maar hebben zich gespecialiseerd in een bepaald gedeelte.
- 3) Schaalbaarheid: doordat de tiers zijn gesplitst, kan iedere tier onafhankelijk van de andere tiers worden geschaald. De load van iedere tier kan afzonderlijk worden verdeeld. Daarnaast biedt het de mogelijkheid om bijvoorbeeld de website in een publieke cloud neer te zetten en de overige tiers onsite te hosten.
- 4) Betrouwbaarheid en beschikbaarheid: omdat de tiers op verschillende servers staan, is er minder risico op uitval.
- 5) Beheerbaarheid code: de code is makkelijker te beheren omdat deze per laag is gesplitst.
- 6) Herbruikbaarheid. Omdat de applicatie is opgedeeld in verschillende onafhankelijke onderdelen met 'service lagen', kan gemakkelijk een onderdeel voor andere projecten worden hergebruikt.
- 7) Informatieveiligheid: (gevoelige) data wordt opgeslagen in een 'secure zone' en is niet rechtstreeks verbonden met het internet.

Nog te benoemen: Voordelen ten gunste van 'eindgebruikers', zoals mogelijkheden van data-integratie.

3 Programmeertalen en frameworks

In het vorige hoofdstuk is de structuur van een applicatie beschreven. Dit hoofdstuk gaat in op de programmeertalen en frameworks waarmee de lagen van de structuur worden gerealiseerd.

3.1 Wat is een programmeertaal?

Een programmeertaal is een formele taal waarin de opdrachten die een computer moet uitvoeren, worden geschreven. Deze talen hebben een andere syntaxis en grammatica dan natuurlijke talen. Deze laatste zijn te complex en ambigu om als programmeertaal te fungeren. Code die in een programmeertaal geschreven is, dient maar op één manier te kunnen worden 'begrepen' door de computer.¹

3.2 Wat is een software framework?

Een software framework is een raamwerk waarin software voorziet in generieke functionaliteit die selectief kan worden aangepast door additionele code te schrijven en daardoor applicatie specifieke software oplevert. Software frameworks kunnen ondersteunende programma's, compilers, code bibliotheken, tool sets en 'application programming interfaces' (API's) bevatten die alle verschillende componenten samenbrengen om de ontwikkeling van een project of systeem mogelijk te maken.

3.3 Context

Binnen BIJ12 zijn tot nu toe verschillende programmeertalen en frameworks gebruikt om functioneel soortgelijke applicaties te bouwen. Onder andere zijn de volgende programmeertalen gebruikt:

- Java
- .NET
- Python

Een programmeertaal maakt gebruik van frameworks om functionaliteit sneller te kunnen ontwikkelen. Het aantal gebruikte verschillende frameworks is vele malen groter dan het aantal programmeertalen.

3.4 Het probleem

De diversiteit van de gebruikte programmeertalen en frameworks en de nog grotere diversiteit door de combinatie van beiden, zorgt voor een aantal problemen. De belangrijkste problemen zijn:

- Beheerbaarheid en benodigde kennis / expertise voor changes: het vraagt kennis van veel verschillende technologieën om de applicatie te kunnen behouden en aan te passen wanneer nodig. Deze kennis is meestal verdeeld over meerdere personen.
- Ontwikkeling: afhankelijk van de beschikbare kennis binnen het ontwikkelteam, wordt gekozen voor een programmeertaal en bijbehorend framework.
- Herbruikbaarheid is minimaal vanwege gebrek aan een standaard
- Beveiliging: bij eventuele beveiligingsissues ontbreekt het inzicht en overzicht van de te nemen actie per applicatie.

Om de hierboven genoemde problematiek het hoofd te bieden, is het wenselijk om een generieke keuze te maken in het aanbod van programmeertalen en frameworks.

3.5 Keuze voor programmeertaal en frameworks

De keuze voor een programmeertaal met bijbehorend framework is een lastige. Er bestaat niet zoiets als een de 'beste programmeertaal'. Iedere taal heeft voor- en nadelen. Een multinational heeft meestal een zeer groot en divers ICT-landschap en benut de voordelen van die verschillende programmeertalen bieden. BIJ12 heeft een beperkt ICT-landschap, beperkte capaciteit en financiële middelen. Om deze reden dient een keuze gemaakt te worden voor een programmeertaal en framework.

¹ <https://nl.wikipedia.org/wiki/Programmeertaal>

Hieronder zijn criteria benoemd waarop de shortlist van programmeertalen zijn beoordeeld.

De keuze voor een framework is meestal een logisch gevolg van de gekozen programmeertaal. Daarom wordt in het onderstaande eerst ingegaan op de programmeertaal.

3.5.1

Programmeertaal

De shortlist van programmeertalen is tot stand gekomen door al gebruikte talen te vergelijken met de resultaten van een enquête uit 2019 van stackoverflow.com (72.525 respondenten). In het onderstaande figuur zijn de belangrijkste programmeertalen weergegeven:



Most Popular Technologies

Programming, Scripting, and Markup Languages



Op basis van het bovenstaande was het eenvoudig om de shortlist samen te stellen. De shortlist bestaat uit:

- C# (.Net)
- Java
- Python

In de onderstaande tabel zijn de programmeertalen op de shortlist gescoord op een zestal criteria:

	C# (.Net)	Java	Python
toekomstvast (+/- komende 5 jaar)	+++	+++	+++
ondersteuning van de 3-tier architectuur en bijbehorende lagen	+++	+++	+++
eenvoudig Service Oriented Architecture (SOA) kunnen implementeren	+++	+++	+++
open source	+++	+++	+++
grote community	+++	+++	+++
beschikbaarheid ontwikkelaars (eventueel zelf in dienst te nemen)	+	+	++

Uit de bovenstaande tabel blijkt dat de iedere keuze voldoet. Om deze reden is gekeken naar trends op het gebied van populariteit onder professionele ontwikkelaars. Uit deze trend blijkt dat Python de laatste jaren steeds populairder wordt en op dit moment de snelst groeiende programmeertaal is. Het is waarschijnlijk dat de beschikbaarheid van Python ontwikkelaars in de toekomst minimaal gelijk is aan Java of C# ontwikkelaars. Daarnaast is Python sneller te leren dan Java en C# en wordt het gebruikt in Esri software en FME-software. Verder wordt Python gebruikt in de applicatie NDVH en binnen de NDFF (eigen programmeurs in dienst). Op basis van het bovenstaande kiest BIJ12 voor de programmeertaal Python voor applicatieontwikkeling.

Achtergrond Python

De programmeertaal Python is open-source en wordt ieder jaar populairder. Python is ontwikkeld door Guido van Rossum en de eerste release is begin 1991 uitgebracht. De ontwerpfilosofie van Python benadrukt code-leesbaarheid. De opbouw van de taal en de object-georiënteerde (OO) benadering helpen de programmeur om heldere, logische code te schrijven voor kleine en grote projecten. Python wordt beheerd en doorontwikkeld door de Python Software Foundation.

3.5.2

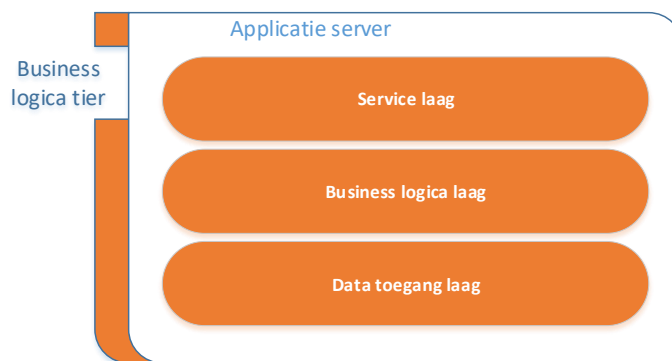
Frameworks

Door te kiezen voor Python is de keuze voor een bijpassend framework een relatief eenvoudige. Python zal worden gebruikt om de business logica tier te maken. Dit wordt ook wel de back-end of server-kant genoemd. Het meest populaire en meest geschikte framework in combinatie met Python is Django.

Achtergrond Django

Het Django framework is een open-source framework dat is geschreven in Python. De kracht van Django ligt vooral in de business logica tier. Het framework wordt onderhouden door de Django Software Foundation.

In het onderstaande figuur zijn de lagen weergegeven die met de combinatie Python en Django worden gerealiseerd in de 'business logica tier'.



Voor de oplettende lezer is duidelijk dat hiermee nog geen keuze is gemaakt voor de presentatie tier en presentatie laag. Dit is helemaal correct. De kracht van Python en Django ligt voornamelijk op het back-end gedeelte en niet zozeer op de front-end, de presentatie. Django kan gebruikt worden voor de presentatie, de user interface, als de webpagina geen fancy functionaliteit nodig heeft. Dit is sneller te bouwen en beperkt daardoor de kosten. Deze oplossing wordt ook wel Server Side Scripting (SSS https://nl.wikipedia.org/wiki/Server-side_scripting) genoemd. Het geniet de voorkeur om voor de presentatie een framework te gebruiken dat hiervoor specifiek is ontwikkeld, maar soms is dat overkill en verhoogt het de kosten onnodig.

Als de presentatie in de vorm van de webpagina veel interactie met de gebruiker heeft en bijvoorbeeld grafieken, grids of draaitabellen nodig heeft, dan is dit te realiseren met het framework Angular. Dit framework is gemaakt door Google. Zie voor meer informatie <https://angular.io/>

Deze oplossing wordt ook wel Client Side Scripting (CSS https://en.wikipedia.org/wiki/Dynamic_web_page#Client-side_scripting) genoemd.

Om de kosten en doorlooptijd te beperken, kan een commerciële bibliotheek van user interface componenten worden gebruikt met het Angular framework. Deze bibliotheek is gemaakt door Sencha <https://www.sencha.com/products/extangular/>. Dit bedrijf is al jaren leidend in het veld van deze professionele bibliotheken. Om een beter beeld te krijgen van de bovenstaande tekst kun je het volgende filmpje bekijken: <https://youtu.be/edx3AIitArU>

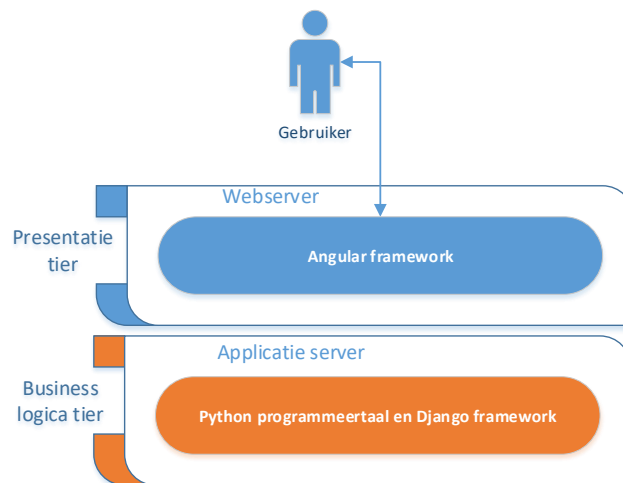
Achtergrond Angular

Het Angular framework is een open-source webapplicatie framework dat wordt geleid door het Angular Team van Google en door de community van individuen en bedrijven.

3.5.3

Gekozen programmeertaal en frameworks

In het onderstaande figuur is weergegeven waarvoor Python, Django en Angular binnen de architectuur worden gebruikt.



4 Basisfunctionaliteiten

Alle BIJ12 applicaties hebben een aantal functionaliteiten gemeen. Denk hierbij bijvoorbeeld aan de opslag van data in een database of het beheren van gebruikersaccounts. In dit hoofdstuk worden deze basisfunctionaliteiten opgedeeld in 2 categorieën: basiscomponenten en voorkeurs-software.

4.1 Basiscomponent

Een 'basiscomponent' is een applicatie die al is ingericht en voor bepaalde functionaliteit kan/moet worden hergebruikt door meerdere applicaties. Voor terugkomende functionaliteit kiest BIJ12 daarom voor één basiscomponent. Dit verhoogt de ontwikkelsnelheid omdat deze functionaliteit niet hoeft te worden gebouwd en het heeft een positief effect op de 'total cost of ownership' (TCO). Daarnaast kunnen ontwikkelingen worden hergebruikt, zorgt het voor kennisopbouw binnen de organisatie en uiteindelijk voor lagere kosten per applicatie omdat de kosten kunnen worden verdeeld.

Binnen BIJ12 zijn de onderstaande basisfunctionaliteiten benoemd en beschikbaar voor gebruik:

- Extraheren, Transformeren, Laden (ETL)
- Business Intelligence en Rapportage
- Integratie
- Online formulieren
- Toegangsbeheer

In de onderstaande subparagrafen worden de basisfunctionaliteiten verder uitgewerkt.

4.1.1 *Extraheren, Transformeren, Laden (ETL)*

In de kern geeft de afkorting precies weer welke functionaliteit dit basiscomponent biedt: het ophalen van gegevens, het transformeren van gegevens en het wegschrijven van gegevens. De laatste jaren is dit domein flink doorontwikkeld en biedt het steeds meer functionaliteit. Ook de komende jaren staan nog interessante ontwikkelingen op het programma.

Voor meer info over ETL zie: [https://nl.wikipedia.org/wiki/Extraction, Transformation and Load](https://nl.wikipedia.org/wiki/Extraction,_Transformation_and_Load) (wel enigszins gedateerde informatie).

Binnen BIJ12 wordt ETL vooral gebruikt voor het valideren van (gis)data (datakwaliteit), samenvoegen van (gis)data en het samenstellen van specifieke datamarts voor het reporting Data Warehouse (DWH).

Leverancier: Safe Software (<https://www.safe.com>)

Producten: FME-Server en 3 engines

Soort licentie: Proprietary

Omgevingen: OTAP op infrastructuur technische leverancier

Opmerking: Op dit moment is het niet mogelijk om toegangsbeheer van FME-server te koppelen via LDAP. Gebruikersaccounts dienen in FME-server zelf te worden beheerd.

4.1.2 *Business Intelligence en Rapportage*

Deze functionaliteit omvat het omzetten van gegevens naar informatie. Deze informatie kan op diverse manieren aan gebruikers beschikbaar worden gesteld: rapporten, dashboards en stories. Daarnaast is het mogelijk om gebruikers de mogelijkheid te bieden om zelf analyses op de gegevens uit te voeren. Ook hier heeft dit domein de laatste jaren grote sprongen gemaakt.

Voor meer info over business intelligence zie: [https://nl.wikipedia.org/wiki/Business intelligence](https://nl.wikipedia.org/wiki/Business_intelligence)

Binnen BIJ12 wordt dit basiscomponent voornamelijk ingezet om gegevens uit een aantal applicaties op een Excel-achtige manier online beschikbaar te stellen voor geautoriseerde gebruikers. Wellicht dat in de toekomst de analytische mogelijkheden gebruikt gaan worden. Dit basiscomponent is gekoppeld aan het reporting Data Warehouse en aan de LDAP.

Leverancier: Yellowfin (<https://www.yellowfinbi.com/>)
 Producten: Yellowfin enterprise licentie
 Soort licentie: Proprietary
 Omgevingen: P op ATOS-infrastructuur. In verband met de hoge kosten van een licentie is ervoor gekozen om alleen een P-omgeving op te zetten. In de applicatie is het OTAP-principe wel vorm gegeven.

4.1.3 *Integratie*

Deze functionaliteit omvat het integreren van applicaties door middel van een integratieplatform: de Enterprise Service Bus (ESB). Een ESB is de meest recente generatie integratie architectuur binnen het IT-landschap. In dit model communiceren verschillende applicaties niet rechtstreeks met elkaar, maar via de ESB en worden de taken van al die systemen onderverdeeld in losse modules (endpoints). De ESB kan worden gezien als een soort stekkerdoos waarop applicaties met verschillende soorten "stekkers" aansluiten.

Voor meer info zie: https://nl.wikipedia.org/wiki/Enterprise_service_bus

Binnen BIJ12 wordt dit basiscomponent gebruikt door een aantal applicaties voor onder andere het ontsluiten van Basisregistraties, doorgegeven van gegevens (geen bulk data) en het ophalen en doorleveren van ingevulde online formulieren. Buiten BIJ12 wordt ons basiscomponent door een aantal provincies gebruikt om diverse provinciale systemen te integreren. In de toekomst kan de ESB ook worden gebruikt om te koppelen met bijvoorbeeld de DSO. Veel stekkers (zogenaamde koppelvlakken) zijn al ontwikkeld en kunnen worden hergebruikt voor diverse doeleinden.

Leverancier: iTrajectum (<https://itrajectum.nl/>) en Microsoft (<https://www.microsoft.com/nl-nl/cloud-platform/biztalk>)
 Producten: Easy bus en Biztalk. Easy bus is een schil om Biztalk heen. Deze schil zorgt voor veel extra functionaliteit en verkort de ontwikkeltijd van 'stekkers'.
 Soort licentie: Proprietary
 Omgevingen: OTAP in Azure cloud in Nederland. De productieomgeving is dubbel uitgevoerd om bedrijfskritische processen te kunnen blijven ondersteunen.

Opmerking: het functioneel-, applicatie- en technisch beheer is uitbesteed bij iTrajectum. Deze keuze is gemaakt vanwege de aard van een ESB-platform. Voor BIJ12 kan het platform als PAAS (platform as a service) worden gezien.

4.1.4 *Online formulieren*

Deze functionaliteit biedt het ontwerpen en beschikbaar stellen van online formulieren door middel van voorgedefinieerde functionaliteiten of zelf te realiseren functionaliteiten. Het platform kent veel mogelijkheden, maar heeft zeker ook een aantal beperkingen. In de praktijk wordt niet altijd rekening gehouden met de beperkingen van het platform, waardoor verderop in de keten problemen kunnen ontstaan. De belangrijkste beperkingen zijn:

- E-formulier is bedoeld voor éénmalige invoer. Dit betekent dat het niet mogelijk is gegevens aan te passen nadat het formulier is verzonden.
- E-formulier is bedoeld voor anonieme gebruikers. Een koppeling met het toegangsbeheer van BIJ12 is niet mogelijk.
- Het voor-invullen van het formulier met informatie die al bekend is in de systemen bij BIJ12 is nauwelijks te realiseren. Gebruikers dienen hierdoor vaak al bekende informatie opnieuw in te vullen. Hiervoor is een workaround, maar dit vergt een andere inrichting van de formulieren en gebruikersbeheer binnen het platform.

Binnen BIJ12 wordt dit basiscomponent door alle units gebruikt. De formulieren zijn zeer divers en gaan qua functionaliteit van simpel tot zeer complex. Een aantal formulieren wordt door de ESB opgehaald op een plek gezet waar FME-server de inhoud verder verwerkt.

Leverancier: FormDesk (<https://nl.formdesk.com>)
Producten: FormDesk account en alle extra modules
Soort licentie: Abonnement
Omgevingen: SAAS op een eigen server in een 'groen' datacentrum in Nederland.

Opmerking: diverse provincies bewegen weg van FormDesk. Onderzocht moet worden of er een alternatief beschikbaar is.

4.1.5

Toegangsbeheer

Deze functionaliteit zorgt voor het beheer van de toegang van gebruikers tot informatiesystemen.

Binnen BIJ12 is ervoor gekozen om het toegangsbeheer van applicaties op gebruikersnaam te baseren. Dit zorgt ervoor dat een gebruiker met dezelfde gebruikersnaam rechten kan krijgen voor meerdere applicaties. Dit heeft als nadeel dat het toegangsbeheer niet kan worden gedecentraliseerd. De beheerder beheert namelijk alle gebruikersaccounts en niet specifiek voor één applicatie. Het basiscomponent toegangsbeheer wordt gebruikt voor diverse applicaties, verspreid over meerdere units. Echter nog lang niet alle applicaties zijn aangesloten op dit basiscomponent.

Om de gebruikers de mogelijkheid te bieden hun persoonlijke gegevens aan te laten passen en hun wachtwoord te kunnen laten resetten, is het noodzakelijk dat het toegangsbeheer via internet te benaderen is. Dit is gerealiseerd door een applicatie (Fusion Directory) 'bovenop' het toegangsbeheer. Fusion Directory is gepositioneerd in de 'secure zone'.

Leverancier: OpenLDAP (<https://www.openldap.org/>) en FusionDirectory (<https://www.fusiondirectory.org/>)
Producten: OpenLDAP en FusionDirectory
Soort licentie: OpenLDAP Public License (gratis) en onderhoudslicentie GOLD
Omgevingen: OTAP op ATOS infrastructuur

Opmerking: op dit moment is het toegangsbeheer gedecentraliseerd. Dit is geen wenselijke situatie.

Opmerking: authenticatie en autorisatie heeft de laatste jaren veel aandacht gehad en hierdoor zijn nieuwe opties beschikbaar gekomen die het onderzoeken waard zijn.

4.2

Voorkeurs-software

Voorkeurssoftware is de software die moet worden gebruikt om bepaalde functionaliteit te realiseren. Het verschil met een basiscomponent is dat voorkeurssoftware specifiek voor een applicatie wordt ingericht (niet herbruikbaar) en daardoor meestal een nieuwe installatie behelst. Het functionele beheer van een dergelijke installatie valt onder het beheer van de desbetreffende applicatie. De reden om functionaliteiten te koppelen aan voorkeurssoftware is om geleidelijk te groeien naar een generiek ict-landschap. Dit heeft een positief effect de 'total cost of ownership' (TCO).

Binnen BIJ12 zijn de onderstaande functionaliteiten benoemd die met voorkeurssoftware dienen te worden gerealiseerd:

- Dataopslag
- Content management
- Geografische services
- Kaartviewer

- Webserver
- Applicatie server
- Operating System (OS)

In de onderstaande sub-paragrafen wordt de voorkeurssoftware verder uitgewerkt.

4.2.1 *Dataopslag*

Deze functionaliteit zorgt voor het opslaan van applicatie-specifieke data.
Voor meer info zie: <https://nl.wikipedia.org/wiki/Database>

Binnen BIJ12 wordt op dit moment diverse software gebruikt om deze functionaliteit te realiseren.

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: The PostgreSQL Global Development Group
(<https://www.postgresql.org/>)

Producten: PostgreSQL en PostGIS extentie voor geografische data

Soort licentie: PostgreSQL is uitgegeven onder the PostgreSQL Licentie, een liberaal Open Source licentie, vergelijkbaar met BSD or MIT licenties (gratis)

Omgevingen: volgens OTAP principe

4.2.2 *Content management*

Deze functionaliteit maakt het mogelijk informatie op een eenvoudige manier te publiceren op het internet via een softwaretoepassing.

Voor meer info zie: <https://nl.wikipedia.org/wiki/Contentmanagementsysteem>

Binnen BIJ12 wordt op dit moment diverse software gebruikt om deze functionaliteit te realiseren.

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: Drupal organisation (<https://www.drupal.org>)

Producten: Drupal

Soort licentie: GNU General Public License, versie 2 or hoger

Omgevingen: volgens OTAP principe

Opmerking: Drupal is een content managementsysteem en dient ook als zodanig te worden gebruikt. Dit betekent dat functionaliteit met standaard Drupal functionaliteit moet worden gemaakt. Maatwerk in Drupal is niet wenselijk vanwege de hoge kosten en inspanning die het met zich meebrengt bij upgrades en security patches. Als standaard Drupal niet de gewenste functionaliteit biedt, dan is het in veel gevallen een CMS niet het juiste platform om deze functionaliteit te realiseren.

4.2.3 *Geografische services*

Deze functionaliteit maakt het mogelijk geografische informatie via services te publiceren naar het internet. De services kunnen openbare of niet-openbare informatie bevatten en worden al dan niet beveiligd met een gebruikersnaam en wachtwoord.

Het spreekt voor zich dat de services met een https-verbinding worden gepubliceerd.

De meeste services die BIJ12 publiceert zijn een Web Map Service (WMS) of een Web Feature Service (WFS)

Voor meer info zie:

- https://nl.wikipedia.org/wiki/Web_Feature_Service
- https://en.wikipedia.org/wiki/Web_Feature_Service
- https://en.wikipedia.org/wiki/Open_Geospatial_Consortium
- https://en.wikipedia.org/wiki/Web_Map_Service
- https://nl.wikipedia.org/wiki/Web_Map_Service

Binnen BIJ12 wordt op dit moment diverse software gebruikt om deze functionaliteit te realiseren.

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: Open Source Geospatial Foundation (<http://geoserver.org/>)

Producten: GeoServer

Soort licentie: GeoServer is gratis software en open source
Omgevingen: volgens OTAP principe

4.2.4 *Kaartviewer*

Deze functionaliteit maakt het mogelijk geografische informatie te visualiseren en te publiceren op het internet.

Momenteel wordt Flamingo gebruikt om deze functionaliteit te realiseren. In 2020 zal Flamingo worden vervangen door GeoWeb, daarom is Flamingo hier niet verder uitgewerkt. In tegenstelling tot Flamingo heeft GeoWeb geen applicatie-specifieke installatie nodig, maar wordt de functionaliteit vanuit een generieke omgeving beschikbaar gesteld. GeoWeb zal daarom ook als basiscomponent worden benoemd.

Geoweb is een webviewer waarmee eenvoudig grote hoeveelheden geografische en administratieve gegevens kunnen worden aangeboden. Dit maakt het eenvoudig om informatie te delen, te analyseren en te presenteren.

GeoWeb is een softwareproduct dat data aan geografische kaarten toevoegt. Met dit pakket kunnen we:

1. De omgeving analyseren en erover rapporteren
2. Ruimtelijke informatie slim gebruiken in ons dagelijkse werk
3. Data combineren, op elkaar afstemmen en delen.

Leverancier: Sweco

Producten: Geoweb

Soort licentie: GeoWeb enterprise en GeoWeb catalogus

4.2.5 *Webserver*

Deze functionaliteit maakt het mogelijk om verzoeken vanuit de browser van de gebruiker af te handelen.

Voor meer info zie: https://en.wikipedia.org/wiki/Web_server

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: The Apache Software Foundation. (<https://httpd.apache.org/>)

Producten: Apache HTTP Server

Soort licentie: Apache License 2.0 (gratis)

Omgevingen: volgens OTAP principe

4.2.6 *Applicatie server*

Deze functionaliteit maakt het mogelijk om de gemaakte applicatie te 'draaien' in een server omgeving.

Voor meer info zie: https://en.wikipedia.org/wiki/Application_server

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: The Apache Software Foundation. (<http://tomcat.apache.org/>)

Producten: Apache Tomcat

Soort licentie: Apache License 2.0 (gratis)

Omgevingen: volgens OTAP principe

4.2.7 *Operating System*

Deze functionaliteit maakt het mogelijk dat de hardware en software wordt gemanaged en zorgt voor generieke functionaliteiten.

Voor meer info zie: https://en.wikipedia.org/wiki/Operating_system

BIJ12 heeft gekozen voor de onderstaande voorkeurssoftware:

Leverancier: Red Hat, Inc. (<https://www.redhat.com/en>) of Microsoft (<https://www.microsoft.com/nl-nl/cloud-platform/windows-server>)

Producten: Red Hat Enterprise Linux Server of Microsoft Server

Soort licentie: Proprietary

Omgevingen: volgens OTAP principe

5 BIJ12 architectuur plaat

In de voorgaande hoofdstukken zijn diverse technische onderdelen besproken. Hoe komt dit allemaal samen?

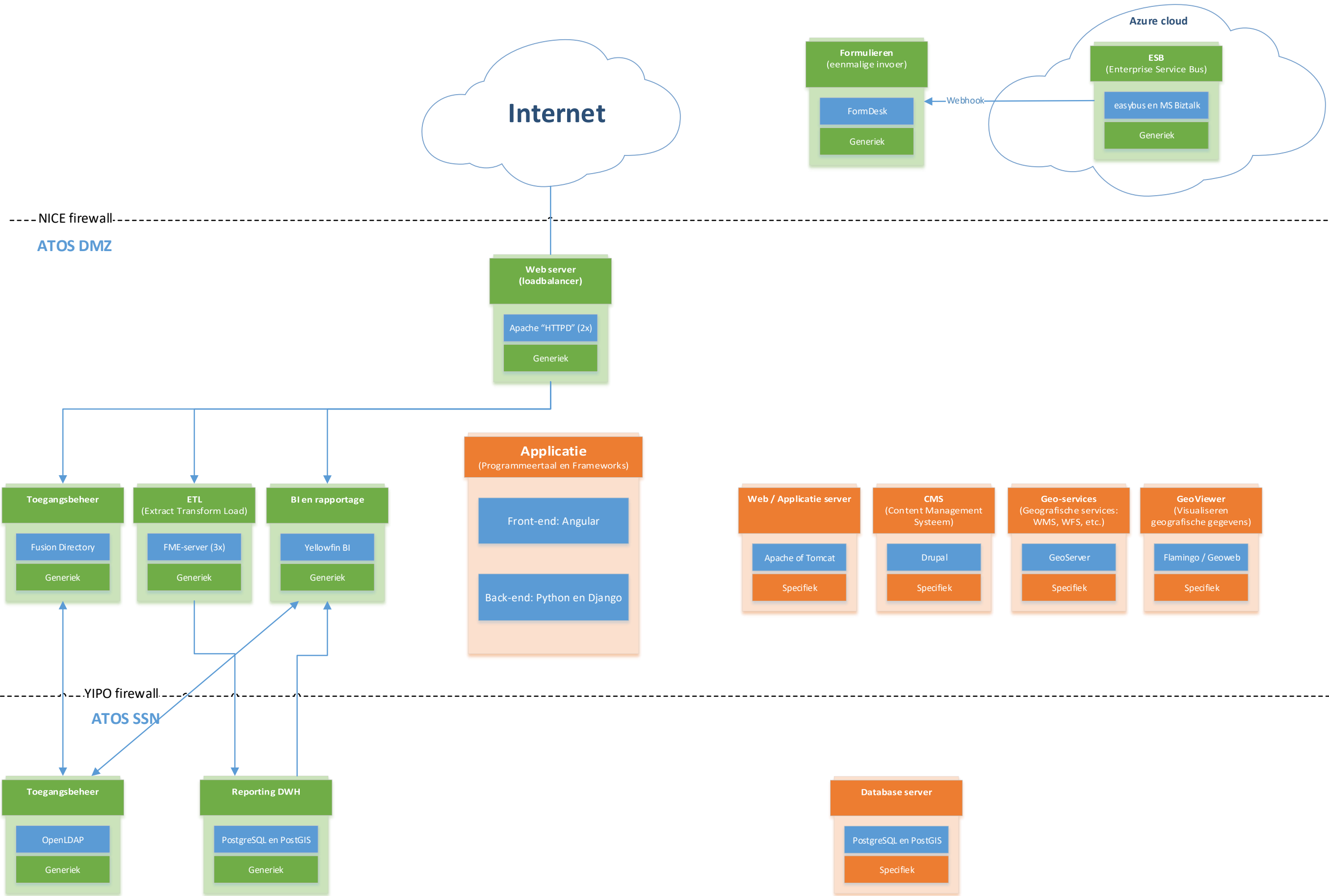
Een (nieuwe) applicatie biedt functionaliteiten voor gebruikers. Door deze functionaliteiten afzonderlijk te benoemen, is het mogelijk deze te vergelijken met de beschikbare basiscomponenten en voorkeurssoftware.

De basiscomponenten kunnen worden hergebruikt en de voorkeurssoftware wordt per applicatie ingericht. Meestal is een applicatie een samengesteld geheel van basiscomponenten en voorkeurssoftware. De logica om de applicatie tot een goed functionerend geheel te maken, wordt ontwikkeld met Python, Django en Angular.

In de architectuurplaat op de volgende pagina is het bovenstaande gevisualiseerd. De relaties tussen de basiscomponenten zijn in de plaat opgenomen. Deze relaties bestaan nog niet voor de nieuw te bouwen applicatie en daarom zijn de overige functionaliteiten als separate blokken weergegeven.

Uitgangspunt: in de plaat is ervan uitgegaan dat de applicatie binnen de ATOS ict-infrastructuur (als technisch beheerder van onze omgeving) komt te draaien. Alleen op deze manier kan eenvoudig van de basiscomponenten gebruikt worden gemaakt.

5.1 BIJ12 architectuurplaat - schematisch



6 BIJ12 ontwikkelomgeving

Het ontwikkelen van (nieuwe) applicaties vergt een stabiele en afgeschermdede omgeving waarin kan worden gewerkt. Daarnaast zijn binnen BIJ12 een aantal basiscomponenten benoemd en ook deze dienen beschikbaar te zijn bij de ontwikkeling van (nieuwe) applicaties. Vanwege het bovenstaande en om te voorkomen dat derden een 'eigen' ontwikkelomgeving op moeten zetten, heeft BIJ12 besloten een generieke BIJ12 ontwikkelomgeving in te richten en beschikbaar te stellen voor BIJ12 projecten.

6.1 Opzet

De BIJ12 ontwikkelomgeving is een gecombineerde ontwikkel- en testomgeving die los staat van de ontwikkel- en testomgeving die de technisch leverancier gebruikt voor de doorontwikkeling van de applicaties die in beheer zijn. De omgeving kan worden gebruikt worden door BIJ12 en eventuele derde partijen voor ontwikkel- en testactiviteiten. De omgeving is momenteel toegankelijk via een Citrix portal.

In de architectuurplaat hieronder zijn de generieke onderdelen opgenomen. Het is natuurlijk mogelijk de omgeving uit te breiden met specifieke projectonderdelen. De kosten van dergelijke uitbreidingen zullen worden doorbelast aan het project dat hier gebruik van maakt.

Als tijdelijk geen gebruik wordt gemaakt van de ontwikkelomgeving, dan zullen overbodige onderdelen in verband met kostenbesparing worden afgeschaald. Te allen tijde wordt de omgeving operationeel en secure gehouden.

De groene blokken in de architectuurplaat zijn onderdeel van de beheeromgeving van ATOS en geen onderdeel van de ontwikkelomgeving anders dan dat er een connectie is om gebruik te kunnen maken van YellowFin.

In de rechterkant (de oranje blokken) van de architectuurplaat is de ontwikkelomgeving gevisualiseerd. Om te kunnen ontwikkelen, zijn een aantal onderdelen nodig. Allereerst is een desktop nodig waarop een ontwikkelaar alle rechten heeft om deze naar eigen inzicht in te richten. Deze functionaliteit wordt gefaciliteerd door VDI's (Virtual Desktop Infrastructure). Om de VDI's op een veilige manier te kunnen benaderen vanaf het internet en makkelijk op- en af te kunnen schalen, zijn ook de Citrix Netscaler, Citrix Infra server en proxy server noodzakelijk.

Vanaf de VDI is het mogelijk koppelingen te leggen met de basiscomponenten zoals die zijn opgenomen in de plaat. Alleen op de VDI heeft de ontwikkelaar volledige rechten. Op alle andere componenten zijn deze rechten beperkter, maar ruim genoeg om te kunnen ontwikkelen. De reden hiervoor is dat ATOS verantwoordelijk is voor de security en beschikbaarheid van deze omgeving. Dit kan alleen als admin rechten niet aan derden beschikbaar worden gesteld.

Mocht een server of een applicatie voor het project moeten worden toegevoegd (bijv. Drupal, Flamingo of iets anders) dan wordt uiteraard gefaciliteerd.

Regie en coördinatie van deze omgeving is belegd bij het IV-team.

