ADF en APEX: een vergelijk

Wanneer komen ze het beste tot hun recht?

Oracle Application Express(APEX) is een technologie die al niet meer weg te denken is uit het Oracle-spectrum. APEX wordt aangeboden als een product waarmee snel webapplicaties kunnen worden ontwikkeld. Oracle Application Development Framework(ADF) is het strategische product van Oracle om webapplicaties te ontwikkelen. Maar hoe verhouden deze twee technologieën zich tot elkaar? Wanneer valt de keuze op de ene technologie, en wanneer op de andere?

Er is een aantal artikelen verschenen die de vergelijking tussen ADF en APEX maken, echter deze zijn nooit gebaseerd op concrete criteria. In dit artikel bieden we een handvat aan, om de keuze tussen ADF en APEX te kunnen maken. Dit gaan we doen aan de hand van criteria, per criterium behandelen we beide technieken. Wat duidelijk moet zijn is dat beide technologieën bestaansrecht hebben; de vraag is wanneer welke technologie het meest geschikt is.
Verder gaan we in dit artikel uit van een omgeving waarin bestaande Oracle software aanwezig is, als mogelijke technieken beperken we ons tot ADF en APEX. Als we APEX noemen in dit artikel, bedoelen we APEX versie 3.2, als we ADF noemen bedoelen we ADF versie 11.1.

Wat is ADF?

ADF is een 3-tier architectuur. In een 3-tier architectuur zijn de data, applicatie en client van elkaar gescheiden. Verder implementeert ADF het MVC- patroon. Hierin zijn de presentatie- en controllertier gebaseerd op Java Server Faces (JSF), meer specifiek ADF Faces (de JSF componenten van Oracle). De laatste laag, de modellaag, bestaat uit ADF Business components. Verder is er nog een aantal uitbreidingen op aangebracht zoals templating, declaratieve databinding, etcetera.
Voor ADF bestaat de ontwikkelomgeving uit JDeveloper en is de kerntechnologie JEE. De basistaal hiervoor is Java, voor databasetoegang wordt gebruikt gemaakt van SQL. Het is ook mogelijk om aanroepen naar PL/SQL te maken vanuit ADF. ADF bestaat sinds het begin van dit decennia.
ADF is door de fundering op Java object georiënteerd.
Opleveren is het 'deployen' van een zogenaamd Java archief, volgens de JEE standaarden.
ADF is in de markt gezet voor grote enterprise-applicaties, wat ook de doelstelling van JEE zelf is.

Wat is APEX?

APEX is een 2-tier architectuur. De database zorgt voor de business logica en de APEX-schema's zorgen voor de scherm-opbouw. De middenlaag bestaande uit de webserver (Oracle HTTP Server gebaseerd op Apache) is uitsluitend een doorgeefluik om webtoegang mogelijk te maken.
De term APEX staat zowel voor de ontwikkelomgeving als de runtimeomgeving en is geëvolueerd uit HTML DB dat sinds het begin van deze eeuw bestaat. De ontwikkelomgeving draait puur in de webbrowser en werkt uitsluitend met databasetabellen en packages die via de mod_plsql module van de Apache-webserver beschikbaar zijn. Alles is opgeslagen in de database er komt geen andere opslag aan te pas. Hooguit kunnen stylesheets, plaatjes en javascript op de webserver gezet worden voor betere performance.
APEX neigt naar objectoriëntatie door het gebruik van componenten in de user-interface, maar onder de motorkap is hier geen sprake van.
Opleveren is het draaien van een script op de productieomgeving van APEX.
APEX is in het begin vooral in de markt gezet als vervanger van Microsoft Access databases en spreadsheets, al begint Oracle hier wel op terug te komen gezien de populariteit van APEX. De opmerkingen over spreadsheets en Access databases zijn ook geheel verdwenen uit de APEX omschrijving, al zijn ze in de ontwikkelomgeving nog wel terug te vinden.

Uitgangssituatie

Op het moment dat we twee technologieën vergelijken, moeten we de uitgangssituatie duidelijk vast te stellen.
We gaan in dit artikel uit van de volgende situatie:

- Oracle-database als hart van de systemen
  – Hierdoor kunnen we ook uitgaan van kennis van (PL/)SQL, Forms en Designer
- Enige kennis van de Oracle applicatie server
  – Dit baseren we op het feit dat Webforms steeds meer standaard begint te worden, waarvoor de Oracle applicatie-server vereist is
- Weinig tot geen Java kennis

Vergelijken

Om een vergelijking te kunnen maken moeten we eerst een aantal criteria definiëren. Het definiëren van deze criteria is een studie op zich. Wij hebben een aantal criteria opgesteld, we presenteren deze criteria hieronder zonder verdere specifieke uitleg, dit valt buiten de scope van dit artikel. De criteria komen uit referenties en praktijkervaring. Hieronder bespreken we beide technologieën per criterium. De criteria staan in een willekeurige volgorde.

De criteria:

- Connectiviteit
- Leerbaarheid
- Onderhoudbaarheid
- Schaalbaarheid
- Flexibiliteit
- Productiviteit
- Applicatietypes
- Licentiekosten
- Aansluiting Oracle-database
- Continuïteit

Waar mogelijk gebruiken we een schaal van 1 tot 5, waarbij 1 het laagst is en 5 het hoogst.

Connectiviteit

Onder connectiviteit verstaan we het verbinden van een applicatie met externe systemen. Of deze verbinding lezen, schrijven of realtime data verwerken is, is niet van belang. Er zijn bijna geen systemen meer die compleet los staan van andere systemen; er is altijd wel sprake van datauitwisseling. Of dit nu uitwisselingen zijn met Content Management System (CMS), Enterprise Service Bus (ESB) of Business Process Execution Language (BPEL) processen, connectiviteit is een steeds belangrijkere rol gaan spelen.

APEX

Bij APEX is de Oracle-database verantwoordelijk voor de connectiviteit. Deze kan bijvoorbeeld bestaan uit databaselinks met andere Oracle-databases, advanced queues of file's lezen op het file systeem voor het importeren van data.
Advanced queues verdienen meer aandacht, deze kunnen namelijk samenwerken met het Java Messaging System(JMS) waardoor veel connectiviteit kan worden bereikt. Al is in het geval van communicatie met JMS wel Java kennis nodig. Verder biedt APEX een interface om webservices te gebruiken al is deze vrij beperkt.
In het kort is APEX vrij gelimiteerd in termen van connectiviteit en leunt vooral op de connectiviteit van de database. Wel biedt de database zelf database packages om te verbinden naar bepaalde systemen zoals een LDAP package voor het LDAP protocol, al werkt dit niet altijd probleemloos.
Hierdoor kunnen we concluderen dat APEX vrij laag scoort op connectiviteit, behalve op connectiviteit met andere Oracle systemen.
Op basis van dit voorgaande geven wij APEX een 2 voor connectiviteit.

ADF

Omdat ADF gebaseerd is op JEE zijn de mogelijkheden qua connectiviteit eigenlijk onbeperkt. Voorbeelden hiervan zijn webservices via JAX-WS, of een LDAP implementatie zoals Novell EDirectory, Microsoft Active directory via JNDI. Ook kan een willekeurige database zoals Sybase, SQL server, DB2 via JDBC[17] benaderd worden. Verder kunnen natuurlijk ook files bewerkt en ingelezen worden. Omdat de motor achter ADF Java is zijn al deze mogelijkheden ook toepasbaar binnen ADF.
Op basis van dit voorgaande geven wij ADF een 5 voor connectiviteit.

Leerbaarheid

Hieronder verstaan we twee soorten kennis, de minimale kennis van de technologie zelf om ermee te kunnen ontwikkelen (inleercurve) en de benodigde randkennis om de applicatie goed te laten werken.
Hiermee nemen we in acht de beginsituatie die eerder voorgelegd is; we gaan uit van weinig Java kennis een vooral Forms en PL/SQL kennis.

APEX

Vooral de inleercurve voor APEX is bijzonder laag, zelfs binnen één dag cursus is het voor ontwikkelaars mogelijk om met APEX aan de slag te gaan. Omdat de hoofdtaal binnen APEX PL/SQL is, is deze kennis al aanwezig, verder is er enige HTML en Javascript kennis nodig. Ook lijkt de kennis om complexere applicaties te maken binnen APEX vrij laag, dit komt vooral doordat APEX veel complexiteit van webapplicaties verbergt, maar deze is er nogsteeds.
Op basis van dit voorgaande geven wij APEX een 5 voor de benodigde kennis.

ADF

In het geval van ADF moeten ontwikkelaars een andere taal leren, namelijk Java, al geeft Oracle veel handvaten met behulp van de developers-guide, specifiek gericht op Forms-ontwikkelaars.
Verder verwacht ADF/Java veel verdere rand kennis. Kennis van het HTTP protocol met de specifieke headers en de functies hiervan is vrij belangrijk. Ook verwacht ADF enige HTML en Javascript kennis. Verder is er kennis van Object Oriëntatie nodig, iets wat best een heel opstakel kan zijn voor ontwikkelaars die al heel lang procedureel denken. Verder wordt ook enige conceptuele kennis verijst van wat een applicatie server is, hoe webpagina's generiek werken, niet gerelateerd aan de specifieke technologie.
Op basis van dit voorgaande geven wij ADF een 2 voor benodigde kennis.

Al willen we hier als sidenote toch nog even melden deze extra benodigde kennis voor ADF uiteindelijk alleen maar betere applicaties oplevert.

Onderhoudbaarheid

Onder dit criterium verstaan we de mogelijkheid om een applicatie aan te passen na de initiële ontwikkeling. Hoe eenvoudig kan nieuwe functionaliteit toegevoegd worden, of bestaande functionaliteit aangepast worden. Nu komen we uit bij één van de meeste fundamentele verschillen tussen APEX en ADF. Dit is namelijk het verschil tussen een 2-tier architectuur (APEX) en een 3-tier architectuur (ADF).
APEX werkt direct op tabellen, terwijl ADF werkt op een zogenaamd domeinmodel. Onder domeinmodel verstaan we voor nu ook de Object-relational mapping (O/R mapping).
In het kort komt dit er op neer dat als er bijvoorbeeld een kolom van lengte wijzigt dit eenmaal in het domeinmodel aangepast wordt en er geen verdere aanpassingen nodig zijn, terwijl in een 2-tier architectuur dit per pagina aangepast moet worden. Verder kan dit domeinmodel ook taken van de database overnemen zoals (sub)totalen berekenen of dit wenselijk is laten we in het midden.
Ook kan dit domeinmodel validatie afdwingen, validatie op één plek die overal aangeroepen wordt, dit zijn standaard voordelen van het domeinmodel.
Dit kan allemaal ook gebouwd worden in een 2-tier architectuur maar hier is extra werk voor nodig. Hiertegenover staat dat het opzetten van een domeinmodel initieel extra werkt met zich meebrengt.
Verder spelen ook onderwerpen een rol zoals versiebeheer, debug-mogelijkheden, etc. een rol bij onderhoudbaarheid. Al geldt dat ook voor dit onderwerp alleen een kompleet boek geschreven kan worden, wij pakken een aantal duidelijk aanwezige aspecten.

APEX

Zoals eerder beschreven is APEX een 2-tier architectuur en heeft dus niet de voordelen van een domeinmodel. Wat wel mogelijk is, is om met updatable-views te werken. Zo kan een aardig deel van een domeinmodel nagemaakt worden, maar niet alles, onder andere de caching en het werken met objecten kunnen zo niet nagemaakt worden.
Omdat APEX onderliggend al zijn definities vastlegt in tabellen zijn er ook wel mogelijkheden om onderhoud te vergemakkelijken door queries rechtstreeks uit te voeren op deze tabellen, al is deze functionaliteit niet standaard aanwezig.
Verder mist functionaliteit zoals 'usage' controleren, het refactoren van code om de leesbaarheid te verhogen, uitgebreide debug-mogelijkheden, controleren of de veld definities gelijk zijn aan de database, allemaal punten die in de onderhoudsfase erg belangrijk zijn. Wel biedt APEX de tool UI Defaults om de layout te standaardiseren, ook wordt meestal bij een APEX applicatie de logica zo veel mogelijk buiten APEX zelf gehouden en aanroepen naar eigen packages geplaatst, deze packages zelf zijn beter onderhoudbaar.
Ook speelt versiebeheer een belangrijke rol bij onderhoudbaarheid, dat een zwak punt is van APEX, er is hier vrijwel geen ondersteuning voor.
Op basis van dit voorgaande geven wij APEX een 2 voor onderhoudbaarheid.

ADF

ADF gaat nog verder dan het domeinmodel en zelfs prompt, veldlengtes, displayhints, displayformaten, etc. kunnen vastgelegd worden op één centrale plek. Zoals in de inleiding van onderhoudbaarheid al gemeld werd, geeft dit domeinmodel duidelijk voordelen in de onderhoudsfase, dit gekoppeld met geavanceerde debug-mogelijkheden, refactor-mogelijkheden, controleren of het domeinmodel klopt met de database en 'usage' controles kunnen we van een duidelijke voorsprong ten opzichte van APEX spreken. In het geval van ADF is versiebeheer goed mogelijk.
Al is dit onderwerp vrij complex, de inrichting hiervoor kan gaan van versiebeheer tools zoals Subversion, tot tools zoals Maven die ook hele opleveringen met afhankelijkheden bijhouden.
Op basis van dit voorgaande geven wij ADF een 4 voor onderhoudbaarheid.

Schaalbaarheid

Onder schaalbaarheid verstaan we hoe een applicatie omgaat met toenemende gebruikers of bredere inzetbaarheid van een applicatie. Schaalbaarheid is vooral belangrijk bij applicaties met grote gebruikeraantallen, meestal wordt gesproken over lineaire schaalbaarheid. Dit betekent dat voor iedere gebruiker de load op de server ongeveer evenredig omhoog gaat.

APEX

Omdat APEX in de database draait is de schaalbaarheid van APEX databaseschaalbaarheid. En omdat dit een Oracle-database is kunnen we spreken van goede schaalbaarheid van de applicatie zelf. Een Oracle-database kan beter schaalbaar gemaakt worden door het inzetten van RAC(Real Application Cluster). Er zijn voorbeelden van APEX-omgevingen binnen een RAC, al hangt hier wel een stevig prijskaartje aan. Behalve de schaalbaarheid van de database biedt APEX zelf weinig opties om de schaalbaarheid of performance te verbeteren. APEX werkt vooral met gebruikersessies en niet met een overkoepelende sessies op hoger niveau. Hierdoor wordt veel applicatie afhankelijke informatie dubbel opgeslagen op iedere gebruikerssessie. Wel is het zo dat de database voornamelijk gebruikt wordt voor niet databasetoepassingen, oftewel het genereren van HTML en dit wegschrijven naar een HTTP socket.
Op basis van dit voorgaande geven wij APEX een 3 voor schaalbaarheid.

ADF

Omdat ADF gebaseerd is op JEE, en de JEE specificatie schaalbaarheid en clustering omvat, is lineaire schaalbaarheid impliciet inbegrepen. Verder specificeert JEE ook clustering, een JEE applicatieserver moet geclusterd kunnen worden en vervolgens moet de schaalbaarheid nog lineair blijven, er zijn dus geen oplossing zoals RAC nodig. Verder specificeert ADF behalve gebruikersessie nog andere niveau's om data te cachen en zo beter schaalbaar te zijn. Er kan bijvoorbeeld data geladen worden op het niveau van de applicatie zelf, zoals een lijst met postcodes of landen die niet per gebruiker verschillen.
Verder komt het domeinmodel hier weer ter sprake. Dit domeinmodel, via de O/R mapping zal ook queries cachen, als een query tweemaal achter elkaar uitgevoerd wordt zal deze maar éénmaal de database ingaan, terwijl bij APEX dit niet het geval is, al zal de System Global Area (SGA) cache deze dan wel oppakken. Het grote verschil hierbij is ook dat APEX niet object gerelateerd is, en alleen de SGA cache gebruikt, terwijl ADF ook een object-cache gebruikt.
Op basis van dit voorgaande geven wij ADF een 5 voor schaalbaarheid

Flexibiliteit

Onder flexibiliteit verstaan we de mogelijkheid om buiten de technologie en ontwikkeltool heen te werken. Soms wil je net even iets anders dan wat standaard aangeboden wordt, is dit mogelijk? Kost dit veel moeite? Biedt de technologie hier zogenaamde hooks voor, plekken om in te grijpen als bepaalde events voorkomen.

APEX

Binnen APEX zijn alle componenten aanpasbaar en ook zijn alle templates die de pagina's genereren volledig aanpasbaar. Qua hooks biedt APEX niet veel, wel is er bijvoorbeeld page 0, waar bepaalde logica opgezet kan worden die voor iedere pagina uitgevoerd wordt. Ook biedt APEX verschillende punten per pagina waarop acties uitgevoerd kunnen worden, denk aan pagina inladen of bij het weg navigeren, maar dit is op pagina niveau, niet globaler zoals page 0.
Op basis van dit voorgaande geven wij APEX een 3 voor Flexibiliteit

ADF

Binnen ADF zijn ook alle componenten aanpasbaar en de pagina's zelf ook, verder is templating beschikbaar om pagina's flexibeler te maken. Verder biedt ADF ook nog een zogenaamde skinning functionaliteit, dit zorgt ervoor dat de layout van componenten generiek aangepast kunnen worden zodat ze er op iedere pagina hetzelfde uitzien. Verder biedt ADF ook nog erg veel hooks om in te grijpen zoals altijd als er een error voorkomt 'doe dit'. Altijd als er een query uitgevoerd wordt 'doe dat', of nog specifieker altijd als er een select query uitgevoerd wordt.
Deze zitten echter op hoger niveau dan pagina's, deze gaan applicatie breed. ADF ondersteunt ook de pagina acties die eerder bij APEX ook genoemd zijn. Echter soms merk je ook dat ADF tegenwerkt door al deze hooks en ingrijpmomenten, omdat de complexiteit omhoog gaat.
Op basis van dit voorgaande geven wij ADF een 4 voor flexibiliteit.

Productiviteit

Onder productiviteit verstaan we de hoeveelheid functionaliteit die een ontwikkelaar op kan leveren met de gekozen technologie, binnen een bepaalde tijdseenheid.

APEX

Omdat de benodigde kennis voor APEX lager ligt dan binnen ADF en de wizards vrij herkenbaar zijn is APEX voor Forms en PL/SQL programmeurs erg productief, deze kunnen snel applicaties opleveren. Voor kleinere applicaties gaat dit goed, echter voor grotere applicaties is wel degelijk meer kennis nodig. Dit punt is dan ook gekoppeld aan de benodigde kennis c.q. inleercurve.
Op basis van dit voorgaande geven wij APEX een 5 voor productiviteit.

ADF

Zoals we eerder al noemde is dit punt gedeeltelijk gekoppeld met benodigde kennis en inleercurve, deze is bij ADF een stuk steiler. Ook is er zeker op de eerste projecten altijd het idee dat ADF tegenwerkt omdat er veel bij komt kijken om echt te begrijpen wat er allemaal op de achtergrond gebeurd, als dit echter allemaal voorbij is gaat de productiviteit van ADF met stappen vooruit.
Op basis van dit voorgaande geven wij ADF een 3 voor productiviteit.

Als sidenote hierbij, dit ligt wel heel erg aan de ontwikkelaar, uiteindelijk kan een ervaren ADF/Java ontwikkelaar sneller ontwikkelen dan een APEX ontwikkelaar, maar dit zal zeker in het begin niet het geval zijn. Hier zijn ook voorbeelden van in de praktijk. Uiteindelijk komt het toch altijd weer neer op de kwaliteit van de programmeur zelf.

Applicatietypes

Onder applicatietypes verstaan we het type applicaties waarop de makers van de technologie zich vooral gericht hebben. Eigenlijk is dit een afgeleide van een aantal van de eerdere criteria, met name de criteria connectiviteit, onderhoudbaarheid, schaalbaarheid en flexibiliteit. Allereerst is het mogelijk om in beide technologieën alle soorten applicaties te bouwen.
Echter is een technologie meestal specifiek gericht op een doelgroep. We gaan nu voor beide technologieën beredeneren welke applicatiedoelgroep dit is.
In dit onderdeel gebruiken we de term enterprise-applicaties, oftewel enterprise software. Hier zijn een aantal kenmerken van te onderkennen waarvan de belangrijkste de functie binnen de organisatie is, de software is gericht op het gehele bedrijf (enterprise) en niet op afdeling of andere niveau. Software die gericht is op de gehele organisatie vereist meestal veel connectiviteit, bestaat uit veel schermen en heeft een hoog aantal gebruikers.

APEX

APEX is in eerste instantie neergezet als spreadsheet en Microsoft Access vervanger, al lijkt APEX steeds meer te groeien naar de enterprise-applicaties. Een veel gevonden uitspraak is: 'Oracle Application Express is a web-based application platform and development environment that is included with all licences of the Oracle database. It has many characteristics that make it an ideal choice for smaller departmental applications that do not need an 'enterprise' solution'.
Terugkijkend op de onderwerpen connectiviteit en schaalbaarheid kunnen we hier zeggen dat de focus van APEX niet de enterprise-applicaties is, al wordt hier aangewerkt door Oracle in volgende versies.
Uit het voorgaande valt af te leiden dat APEX in eerste instantie opgezet is om de afdelingsapplicaties te vervangen en langzamerhand aan het groeien is naar de enterprise-applicaties

ADF

Omdat ADF de JEE standaard implementeert en JEE gericht is op Enterprise gerichte applicaties. 'Java Platform, Enterprise Edition (Java EE) builds on the solid foundation of Java Platform, Standard Edition (Java SE) and is the industry standard for implementing enterprise-class service-oriented architecture (SOA) and next-generation web applications'. Verder is de nieuwe Oracle EBS suite ook helemaal opgebouwd in ADF, denk aan honderdduizenden ADF componenten, hetzelfde geldt voor Webcenter 11, wat de schaal en focus van ADF weergeeft.
Ook hier kunnen we teruggrijpen op de onderwerpen schaalbaarheid en connectiviteit, hier zien we dat ADF erg goed scoort. Hierdoor kunnen we concluderen dat ADF vooral gericht is op enterprise-applicaties.

Licentiekosten

Onder licentiekosten verstaan we de kosten die verbonden zijn aan het publiek gebruik van een technologie. We gaan er vanuit dat er al een Oracle-database aanwezig is, deze licentiekosten rekenen we niet mee. Hier staat een 5 voor de minste licentiekosten en een 1 voor de hoogste licentiekosten. Omdat prijzen voor licenties nogal fluctueren en zeker ook omdat deze niet vast zijn, worden hier geen prijzen genoemd. Verder geeft Oracle, hetzij via partners of direct, soms korting op licenties.

APEX

In het geval van APEX wordt APEX meestal in de productie database, in een apart schema geïnstalleerd (of in een aparte database instantie) Hier komen geen licentie kosten bij kijken. Kort gezegd, APEX is gratis indien de Oracle-database aanwezig is, waar we vanuit zijn gegaan. Waar wel naar gekeken moet worden is als APEX echt als enterprise-applicatie gebruikt gaat worden er meer gekeken moet worden naar Oracle Real Application Clusters (RAC), en hiervoor praat je over de licentiekosten die op dit moment hoger liggen dan de zwaarste applicatieserver licentie. Verder zijn we uitgegaan van een bestaande Oracle-database binnen de omgeving, toch willen we nog even noemen dat APEX gedraaid kan worden op een XE database (gratis) of een SE database wat rond één achtste van de prijs van een enterprice-editie database ligt.
Op basis van dit voorgaande geven wij APEX een 4 voor licentiekosten.

ADF

Met ADF heb je twee keuzen, of ADF draaien op een Oracle Weblogic, dan zijn de ADF libraries gratis. De kosten hiervan liggen op ongeveer één derde van de kosten voor een enterprice-database licentie. Verder is er ook nog de Weblogic enterprice editie, wat als extra clustering, high availablity en betere monitoring biedt. Deze kosten hiervoor liggen iets lager dan de kosten van een RAC cluster.
De standaardeditie is bijna altijd wel voldoende, voor de enterprise editie moet gedacht worden aan grote bank applicaties.
De andere optie is ADF draaien op een andere applicatie server zoals JBoss, dan moeten alleen de libraries betaald worden wat op ongeveer de helft van een Oracle Weblogic server uitkomt.
Op basis van dit voorgaande geven wij ADF een 2 voor licentiekosten.

	APEX	ADF
Connectiviteit	2	5
Benodigde kennis	5	2
Onderhoudbaarheid	2	4
Schaalbaarheid	3	5
Flexibiliteit	3	4
Productiviteit	5	3
Applicatietypes	'rand' afdelingsapplicaties, groeiend naar de enterprise-applicaties	Enterprise-applicaties
Licentiekosten	4	2
Aansluiting op de Oracle database	5	5
Continuïteit	5	5

Aansluiting Oracle-database

Over dit punt kunnen we vrij kort zijn, omdat beide producten Oracle producten zijn is de aansluiting op de database goed geregeld, denk aan sequences, PL/SQL raadplegen, stored procedures, functions, triggers die data vullen etc. Dit is vooral een punt waar naar gekeken moet worden als er geen APEX of ADF gekozen wordt bovenop een Oracle-database.
Op basis van dit voorgaande geven wij beide dan ook een 5 voor aansluiting op de Oracle database.

Continuïteit

Onder dit criterium verstaan we de mate waarin het zeker is of de technologie nog doorontwikkeld een ondersteuning blijft krijgen van de leverancier. Dit is net als het voorgaande punt niet echt een punt om APEX en ADF te vergelijken maar vooral een belangrijk punt als naar kleinere technologieën, opkomende technologieën of opensource technologieën gekeken wordt. Oracle heeft aangegeven dat APEX en ADF doorontwikkeld worden en volledig gesupport blijven.
Op basis van dit voorgaande geven wij hier dan ook beide een 5 voor continuiteit.

Conclusie

De conclusie geven we in de vorm van een tabel (zie hierboven), een soort van cheatsheet voor APEX en ADF om te kunnen beoordelen welke technologie het meest geschikt is in welke situatie. De criteria die een rol spelen zijn natuurlijk specifiek per situatie. Op basis van de criteria voor een bepaalde situatie hopen we een beter beeld te kunnen schetsen over welke technologie het meeste geschikt is.
De stap naar ADF is steiler, maar dat deze zeker voor enterprise-applicaties zich terug gaat betalen in de tijd. Dit ook vooral 80% van de levensduur van een enterprice-applicatie in onderhoud zit. En op dit aspect komt ADF een stuk beter uit de bus dan ADF. Voor afdelingsapplicaties rondom de enterprise-applicaties is APEX zeker een mooie technologie al moet altijd in het oog gehouden worden dat er ook degelijk software engineering kennis aanwezig moet zijn al lijkt de technologie dit te verbergen.
Oracle heeft al wel APEX 4 aangekondigd, als deze versie definitief uit is, kan de hieronder beschreven balans verschuiven. In APEX 4 worden een aantal van constateringen ten opzichte van APEX anders, er worden een aantal 'achterstanden' rechtgetrokken.

Referenties

- *APEX vs ADF: Round 1, Dimitri Gielis, http://dgielis.blogspot.com/2007/12/apex-vs adf-round-1.html*
- *Apex vs. ADF study, H.Tonguç Yılmaz, http://tonguc.wordpress.com/2007/12/20/apex-vs-adf-study/*
- *APEX and/or ADF - demonstrating two similar yet different applications - part 2, Lucas Jellema, http://technology.amis.nl/blog/2681/apex-andor-adf-demonstrating-two-similar-yet-different-applications-part-2*
- *APEX and/or ADF - demonstrating two similar yet different applications - part 3 - the implementation in ADF, Lucas Jellema, http://technology.amis.nl/blog/2735/apex-andor-adf-demonstrating-two-similar-yet-different-applications-part-3-the-implementation-in-adf*
- *APEX versus ADF, Nathalie Roman, http://iadvise.blogspot.com/2008/04/apex-versus-adf.html*
- *JDeveloper Vs Oracle APEX, Patrick Wolf, http://www.inside-oracle-apex.com/jdeveloper-vs-oracle-apex/*
- *Apex vs JDev - first impressions, Marc Thompson, http://marc-on-oracle.blogspot.com/2008/02/apex-vs-jdev-first-impressions.html*
- *Jave EE 5, Sun Microsystems, http://java.sun.com/javaee/technologies/javaee5.jsp*
- *Tier architectuur, Wikipedia, http://en.wikipedia.org/wiki/Multitier_architecture*
- *Java server faces, Sun Microsystems, http://java.sun.com/javaee/javaserverfaces/*
- *JEE overview, Sun Microsystems, http://java.sun.com/javaee/*
- *APEX 'MS Access on Steroids', Jules Lane, http://www.ukuug.org/events/seminars/databases/APEXforUUG2.pdf*
- *Migrating to APEX from Desktop Systems, Apress - John Edward Scott and Scott Spendolini*
- *Title: 8 Reasons to Use Apex for Departmental Applications, Ingo Peters, Domingo Informatics Inc, http://www.domingoinformatics.ca/uploads/2009.06.17-OOUG%20Apex%20Talk.pdf*
- *Jack-WS, Sun Microsystems , 2.0 http://java.sun.com/developer/technicalArticles/J2SE/jax_ws_2/*
- *Java Naming and Directory Interface, Sun Microsystems, http://java.sun.com/products/jndi/*
- *The Java Database Connectivity, Sun Microsystems, http://java.sun.com/javase/technologies/database/*
- *ADF developers guide for Forms/4GL developers, Oracle, http://www.oracle.com/technology/documentation/jdev/b25947_01/index.html*
- *Domain model, wikipedia, http://en.wikipedia.org/wiki/Domain_model*
- *RAC APEX, Michael Forster / Dietmar Aust (Duits)*
- *, http://www.opal-consulting.de/downloads/DOAG_SIG_AbiT_20070531_V05.zip*
- *Architectural Review of APEX, P. Dorsey, http://www.dulcian.com/papers/ODTUG/2009/2009_Dorsey_APEX.htm*
- *Enterprise software, http://en.wikipedia.org/wiki/Enterprise_software*
- *Subversion, Versiebeheer, http://subversion.tigris.org/*
- *Maven, Software project management, http://maven.apache.org/*
- *Model View Controller, http://en.wikipedia.org/wiki/Model–view–controller*
- *Specificeren van software-kwaliteit, Bemelmans, 1992*
- *ISO 9126, http://en.wikipedia.org/wiki/ISO_9126*

Anton Gerdessen is Consultant bij Transfer Solutions. Hij werkt vooral met Java technologie, maar houdt zich ook bezig met de raakvlakken met 'oudere' Oracle technologie.