De voordelen van een 'low-cost' en 'low-hassle' DBMS

Eenvoud van Firebird loont

Marco Bommeljé

Dit artikel presenteert het open source DBMS Firebird. Open source. Krijgt u nu de neiging om dit artikel meteen terzijde te leggen? Niet doen. Het gaat hier niet om één van de duizenden open source projecten die omvallen als de lead programmer aan z'n vakantiebaantje begint. Firebird is een professioneel georganiseerd project, dat gebaseerd is op solide database-technologie die al meer dan vijftien jaar door talloze bedrijven met succes wordt toegepast.

Afbeelding 1. Aan dit beeldmerk herkent u de ware Firebird fans.

"I think a time will come when we will have much less concern for the physical aspects of the data base." zei Edgar F. Codd in 1982[1].

Twintig jaar na Codd's uitspraak zijn we evenwel nog steeds bezig met de fysieke aspecten van de database. Het DB/M themanummer over de Total cost of ownership van databases van afgelopen juni illustreert dat treffend. De TCO van een DBMS blijkt vooral te zitten in de inzet van technische experts "om het betreffende DBMS naar behoren te laten functioneren"[2]. Deze experts, ingewijd in DBMS-internals, worden ingezet om te goochelen met talloze configuratieparameters: voor schijfruimte, data placement, voor gebruik van werkgeheugen en netwerk-bandbreedte.

En dat terwijl het belangrijkste database-probleem, het beheersen van Pandora's doos met bedrijfsgegevens, menig bedrijf boven het hoofd dreigt te groeien[3]. In plaats van een dreigende data-chaos af te wenden door het organiseren van data- en metadata-management, kijken we naar de *performance monitor*, zoals konijnen in de lamp van de stroper. Het kan anders, het kan eenvoudiger, het kan goedkoper. Veel goedkoper en bovendien veel leuker.

In dit maakt u kennis met het open source DBMS Firebird. Zowel de organisatie als de technologie maken dat Firebird kan bijdragen aan het doorbreken van de de duivelse cirkel van toenemende complexiteit en kosten van DBMS-gebruik. Maar eerst Firebird's geloofsbrieven: iets over afkomst en geschiedenis.

Naar Borland en terug

Het Firebird project is in 2000 begonnen als *fork* (broncode-afsplitsing) van Borlands Interbase versie 6 open source editie. Interbase bestond toen al bijna vijftien jaar als commerciëel DBMS. Het onderscheidt zich door portabiliteit, eenvoud van beheer en bediening, *non-blocking concurrency control*, een rijke implementatie van SQL en uitgebreide voorzieningen voor *server based business rule processing*.

De Interbase-geschiedenis begint halverwege de jaren tachtig[4]. In die jaren ontwikkelde Jim Starkey de ideëen en de software die nu nog steeds het eigen karakter van Interbase en Firebird bepalen. Samen met zijn vrouw, Ann Harrison, heeft hij aan de wieg gestaan van Interbase en het laten opgroeien tot een volwassen DBMS. Later, eind jaren tachtig, werd Interbase overgenomen door Ashton-Tate dat op haar beurt in 1991 eigendom werd van Borland.

> We kijken naar de performance monitor, zoals konijnen in de lamp van de stroper

Onder Borlands vlag heeft Interbase geruisloos een grote schare supporters in de wereld van professionele software engineers opgebouwd, doordat een eenvoudige versie bij de Delphi ontwikkelomgeving werd meegeleverd. Weinig bekend is echter dat ook wereldwijd opererende bedrijven sinds jaar en dag Interbase gebruiken voor grootschalige, bedrijfskritische toepassingen[5].

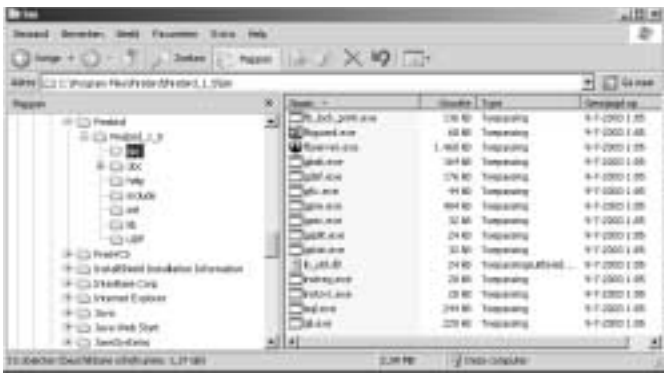
De Interbase open source editie uit het jaar 2000 is een eenmalige gebeurtenis geweest. Borland heeft sindsdien versies 6.5 en 7 met commerciële licenties uitgebracht. Interbase en Firebird bestaan nu dus naast elkaar en langzaam maar zeker groeien ze uit elkaar, tot verdriet van velen.

Vier pijlers

In de drie jaar dat Firebird bestaat, is een hechte community ontstaan. De gemeenschap rust op vier pijlers: de gebruikers, het Firebird project, de FirebirdSQL Foundation en IBPhoenix.
Software bestaat bij de gratie van gebruikers. Welnu, aan actieve gebruikers heeft Firebird geen gebrek. Een miljoen downloads onderstrepen dat[6]. Op de Firebird mail lists en nieuwsgroepen trekt een permanente stroom van vragen, klachten, *enhancement requests*, tips en adviezen voorbij. Het is de plaats bij uitstek om problemen aan te dragen, vragen te stellen en kennis op te doen. De hulpvaardigheid en betrokkenheid op dit online platform is ongekend groot. Ervaren DBA's en gebruikers, maar ook de software-ontwikkelaars van Firebird, beschouwen het als hun taak om vragen te beantwoorden. Wanneer een onervaren gebruiker zich meldt met een hulpvraag, springen mensen van alle kanten bij om hem of haar op weg te helpen. Wanneer iemand een probleem heeft met responsetijden, duikt men diep in een query of een databaseontwerp om een oplossing aan te reiken.

Arno Brinkman

De tweede pijler waarop de community steunt, is het project waarin de productontwikkeling van Firebird plaatsvindt. Het Firebird-project zelf heeft rond de 60 leden. Zij zijn rechtstreeks betrokken bij het ontwikkelen van software voor de server en de *client access* modules, het testen, het maken van installaties, porting, het schrijven van gebruikersdocumentatie en zo meer. Een aantal oudgedienden van Interbase is daarin prominent aanwezig. Daarnaast heeft het project een vaste kern van ervaren software engineers. Zij zijn afkomstig uit de meest uiteenlopende landen: VS, Rusland, Chili, Australië, Spanje, Tjechië, Frankrijk en Nederland. In Nederland zijn diverse mensen in het Firebird-project actief. Eén van hen, Arno Brinkman, heeft een uitzonderlijke bijdrage geleverd door Firebirds query optimizer vergaand te verbeteren.
Een enthousiast team voor softwareontwikkeling is evenwel niet voldoende om van een softwarepakket zoals Firebird een levensvatbaar 'product' te maken. Het vereist een organisatie die in staat is om werkzaamheden te plannen, beslissingen te nemen, juridische kwesties, zoals merkrecht, te regelen en nog veel meer.



Afbeelding 2. Een overzicht van de executables laat zien dat Firebird een small footprint heeft.

Zulke activiteiten spreken misschien minder tot de verbeelding, maar ze zijn essentieel voor het sucses en de toekomst van Firebird.
Dit alles realiseerden zich twee Australiërs met een lange loopbaan in de software-industrie, Helen Borrie en Mark O'Donohue. Zij hebben het initiatief genomen om de FirebirdSQL Foundation op te richten. Inmiddels heeft de stichting een centrale rol in de Firebird-gemeenschap verworven. De stichting is de belangrijkste schakel tussen de Firebird gebruikers en het Firebird project. Door lid of donateur te worden, kunnen ook zij die niet zelf de DBMS-code programmeren, invloed hebben op de wijze waarop

> De DBA wordt verlost van veel vreugdeloos technisch getob

Firebird zich ontwikkelt. Donateurs en leden zorgen voor fondsen die als stipendia worden ingezet om verdere ontwikkeling van Firebird te financieren. Nieuwe functionaliteiten kunnen zelfs 'op bestelling' ontwikkeld worden, mits deze al voor een komende versie gepland zijn.

The Big Bad Wolf

De vierde pijler van de Firebird community is IBPhoenix, het bedrijf dat Ann Harrison met een handvol getrouwen heeft opgericht. Voor Interbase is Ann Harrison altijd de ultieme autoriteit gebleven waar het gaat om technologie en achtergronden. Toen Firebird als open source project ontstond, was zij dus degene tot wie iedereen zich wendde. Vanaf dat moment heeft zij haar ondernemerschap en haar enorme parate kennis belangeloos ten dienste gesteld van de Firebird-gemeenschap. Inmiddels beheert IBPhoenix ("We have answers") de website met de meest uitgebreide informatie over Firebird[7].
Ann Harrison is bovendien degene die Jim Starkey er met de haren bijsleept wanneer een discussie woedt over Firebirds interne architectuur. Dan stuurt zij een e-mail naar de andere hoek van de kamer, waarna hij een oordeel velt over het onderwerp in kwestie. Het oordeel van Jim Starkey is vaak niet mals, getuige zijn bijnaam 'the Big Bad Wolf'. Zo opperde iemand om een nieuwe configuratieparameter te introduceren waarmee gebruikers de werking van een intern proces in Firebird zouden kunnen beïnvloeden. Jim Starkey wijst het voorstel krachtig van de hand en besluit zijn repliek zo: "There are those who argue that users should be given enough rope to hang themselves, but those people generally don't have to support the corpses."
De bemoeienis van Ann Harrison en Jim Starkey met het Firebird is van onschatbare waarde, omdat zij waken over de conceptuele integriteit van de software[8]. Meer dan wie ook bewaken zij Firebirds motto van 'Simplicity and Performance'.
Al met al is Firebird een goed georganiseerde gemeenschap,

zeker in aanmerking nemend dat het project vrijwel volledig van vrijwilligers afhankelijk is[9]. Firebird heeft daardoor een bovengemiddelde levensverwachting, die nog zal verbeteren naarmate de FirebirdSQL Foundation erin slaagt om het aantal leden en donateurs uit te breiden.
Firebird gebruikers, ook bedrijven, kunnen de continuïteit van het DBMS zelf helpen veiligstellen. Ten eerste door lid of donateur te worden van de FirebirdSQL Foundation. Ten tweede door eenmaal opgebouwde expertise te delen met medegebruikers, dat wil zeggen ondersteuning te verlenen via de mail lists. En tenslotte kan men zelf actief worden in het Firebird-project. In wezen geldt dat gemeenschapszin tot eigen voordeel strekt.

> Laten we eerlijk zijn: heel vaak wordt database-tuning ingezet als lapmiddel voor zwaktes in het ontwerp

En dan nu: de techniek

Het is hoog tijd om aandacht te besteden aan het Firebird DBMS zelf. Hoe maakt Firebird het mogelijk dat we ons minder met de fysieke aspecten van de database hoeven bezig te houden? Welke technische zaken neemt Firebird ons uit handen? En hoe blijven responsetijden binnen de perken?
Firebird is, zoals Ann Harrison dat zegt, een *low hassle* DBMS. De DBA wordt verlost van veel vreugdeloos technisch getob. Enkele database-administratietaken waarvoor dat geldt, zijn recovery, performance tuning en schema-wijzigingen.

Recovery

Bescherming van gegevens bij calamiteiten (data protection) is een van de redenen waarom DBMS-technologie is uitgevonden. Ieder zichzelf repecterend DBMS biedt dan ook voorzieningen voor database recovery, het herstellen van database-gegevens na het optreden van storingen. Gewoonlijk maakt men onderscheid tussen software- en hardwarestoringen.
Een bijzondere eigenschap van Firebird DBMS is dat een database na een *soft crash*, een storing van programmatuur, onmiddelijk weer beschikbaar is. Doorgaans houdt database recovery in dat een database vanaf de transactielog bijgewerkt moet worden, voordat gebruikers weer kunnen werken. Dat kost tijd, juist wanneer dat niet wenselijk is. Firebird kent geen transactielog.
De database op schijf bevat steeds de meest recente, integere gegevens. Na herstart is Firebird meteen klaar voor *business as usual*. Een legendarisch verhaal binnen de Interbase en Firebird-gemeenschap is dat deze instant recovery de reden was dat het Amerikaanse leger Interbase koos als DBMS voor de aansturing van artillerie in o.a. tanks[10]. Bij het afvuren van het kanon ontstaat een electromagnetische puls waardoor alle electronica uitvalt. Het is fijn wanneer de boordcomputer van een tank dan weer snel paraat is.
Bij *media failure*, een eufemisme voor hardwarestoringen, zijn de gegevens op schijf meestal niet meer beschikbaar en zal men moeten terugvallen op gekopiëerde gegevens. Firebird biedt als veiligheidsmaatregel de mogelijkheid van *database shadowing*. Dit betekent dat het DBMS een synchrone kopie van de database op een andere schijfeenheid bijhoudt. Aangezien shadowing is ingebouwd in het I/O-subsysteem, werkt het zonder extra software-installatie of -configuratie. Bij een disk crash kan de DBA de shadow als database activeren. Ook hier is sprake van instant recovery[11].
Natuurlijk zal men tegenwoordig bij voorkeur kiezen voor RAID-schijfeenheden of disk mirroring als bescherming tegen gegevens-

Automatisch space management en tuning

In een 'verse' Firebird database (opgebouwd met een restore vanaf een backup) zijn metadata, tabelgegevens en indexen aaneengesloten en in de genoemde volgorde opgeslagen. Door schrijfactiviteit verandert dit. Nieuwe pages worden aan de database-file toegevoegd wanneer geen vrije pages voor nieuwe gegevens beschikbaar zijn. Vrije database pages ontstaan door verwijderen of wijzigen van gegevens. In de loop van de tijd ontstaat een diffuse, sponzige interne structuur. In het algemeen levert dit geruime tijd een stabiele responsetijd.
Firebird beschikt over diverse middelen om responsetijden optimaal te houden en de database-ruimte te beheren.
De query engine optimaliseert geïndexeerde leesoperaties door eerst alle data page pointers te verzamelen voordat de data pages daadwerkelijk worden opgehaald. De data pages worden dan in volgorde van hun page number benaderd, waardoor een effect van virtuele clustering kan worden bereikt, d.w.z. meer lezen en minder bewegen. Een ander voordeel van deze strategie is dat meer dan één index gebruikt kan worden per tabel. Hoe meer *index covered* zoekargumenten, hoe beter.
In een actieve database worden back versions van teruggedraaide transacties opgeruimd door een *garbage collection* mechanisme. Afhankelijk van de versie is dit een aparte thread dan wel onderdeel van user threads.
Daarnaast verzorgt het GBAK backup-programma het opnieuw opbouwen van indexen, het opnieuw vaststellen van de OIT (de oudste transactie waarvoor back versions bewaard moeten blijven) het opruimen van verouderde back versions en het defragmenteren van de database en database pages.
Het maken van een backup en het vervolgens opnieuw opbouwen van de database met de restore-optie is daarom de enige operationele beheeractiviteit die regelmatig uitgevoerd moet worden.

Afbeelding 3. De Firebird Manager als Nederlandstalig control panel applet.

verlies door een disk crash. Nochtans is database shadowing zoals Firebird dat biedt, een goedkoop alternatief dat hoe dan ook eenvoudiger is en sneller werkt dan procedures voor *rollforward recovery* met behulp van backups en transactielogs zoals die bij menig ander DBMS vereist zijn.

Tuning

Van oudsher bestaat database tuning voornamelijk uit het organiseren van beschikbare resources, met name schijfruimte en geheugen. Het doel daarvan is om de computercapaciteit zodanig in te zetten dat de databasetoepassing het snelst werkt.
Bij het organiseren van schijfruimte wordt traditioneel veel tijd besteed aan *space allocation*, het toewijzen van databaseobjecten aan specifieke fysieke locaties. Door tabelgegevens, indexen, metadata en transactielogs op verschillende harde schijven met elk een eigen I/O-controller te plaatsen, kan een vorm van parallellie bij disk access bereikt worden.
De meeste DBMS-en vereisen dat de DBA zich intensief bezighoudt met space allocation op straffe van beroerde responsetijden. Bij menig DBMS gaat space allocation zelfs nog veel verder. Daar moet voor iedere tabel en index worden nagedacht over de benodigde schijfruimte; de wijze waarop deze kan groeien; de fysieke organisatie, partitionering, clustering en zo nog veel meer. Het is allemaal heel interessante materie en een DBA kan er gemakkelijk zijn levensvervulling van maken, maar voor hen die met Firebird werken, hoort het toch echt tot de categorie 'oude ambachten'.
Een Firebird-database bestaat uit één of meer bestanden. Deze database-ruimte wordt als één geheel door Firebird beheerd. Een DBA heeft zelfs geen mogelijkheden om databaseobjecten aan een fysieke plaats te koppelen. Wat overblijft aan space allocation betreft de database zelf en de actieve bestanden van DBMS en OS, met name Firebirds temp file directory en de page file van het besturingssysteem.
Voor *memory allocation*, het toewijzen van geheugen aan specifieke DBMS-functies, geldt hetzelfde verhaal. Firebird kent dat niet. De enige configuratieoptie voor geheugen is de omvang van het cache memory per database.
Is database tuning nu echt niet meer mogelijk? Natuurlijk wel, maar niet met *tweaking nuts & bolts*. Laten we eerlijk zijn: heel vaak wordt database tuning ingezet als lapmiddel voor zwaktes in het ontwerp van database en/of toepassingsprogrammatuur. Door daar de problemen aan te pakken, wordt de meeste winst in responsetijden behaald.

Schema evolution

Dit brengt ons vanzelf op het volgende punt: wijzigingen van de database structuur oftewel *schema evolution*. Bij veel DBMS-en is het wijzigen van een database schema een moeizame operatie die arbeidsintensieve gegevensconversies met zich meebrengen. Firebird voorkomt veel conversie-leed doordat veel metadata wijzigingen on line kunnen plaatsvinden. Zolang de aard van de wijziging geen verlies van gegevens impliceert, is het mogelijk om de structuur van tabellen en de datatypen van kolommen te wijzigen zonder dat conversie van gegevens nodig is. Het datatype van een kolom kan zo bijvoorbeeld van INTEGER in VARCHAR gewijzigd worden. De achtergrond daarvan is dat de Firebird engine de metadata gebruikt om user data te lezen en te schrijven.

Bij onderhoud op een database schema helpt bovendien dat Firebird een fundament van de relationele theorie adequaat heeft geimplementeerd: het domein. Veranderingen in betekenis of gebruik van databasegegevens hebben vaak betrekking op definities van DOMAINs. In een database zonder DOMAINs is de impact van zulke wijzigingen moeilijk te overzien en is het aanbrengen van dergelijke wijzigingen een complexe taak.
Met het DOMAIN als database-object heeft de DBA een krachtig, eigenlijk onmisbaar instrument is voor het beheer van een databaseschema. Eén on-line uitgevoerd ALTER DOMAIN statement is de spreekwoordelijke handomdraai, waar in menig ander DBMS een uitgebreide impact-analyse en conversie nodig is.
Een Firebird-database kan op die manier stapsgewijs evolueren

> Het werkt zonder dat ernstig kijkende mannen in witte jassen voortdurend aan knoppen hoeven te draaien

met een minimale verstoring van de beschikbaarheid. Overigens moet wel worden opgemerkt dat in de praktijk nog genoeg situaties zullen voorkomen waarin wel conversies nodig zijn om een database bij veranderende gebruikerswensen te laten aansluiten.

Afbeelding 4. Het Windows-taakoverzicht laat zien dat de Firebird-server in rustige tijden slechts 3,6 MB aan geheugen opeet.

Niet wachten op vragen van het management

Firebird is ontworpen voor databasetoepassingen met zowel OLTP (online transaction processing) als MIS (management information system) aspecten. Zulke toepassingen vormen voor het DBMS een zogenaamd *mixed environment*.
Performantie in een mixed environment is vooral een kwestie van *concurrency*, gelijktijdige gegevenstoegang voor verschillende gebruikers. De responsetijden aan de balie mogen niet te lijden hebben van het genereren van de kwartaalrapportages. Ofwel: management queries die een consistent snapshot van de database vereisen, mogen on-line gebruikers niet blokkeren.

MGA

Firebird werkt met een systeem dat *multi-generational architecture* (MGA) genoemd wordt. Dat werkt als volgt. Bij elke table row wordt het transactienummer van de schrijfactie opgeslagen en alsmede een verwijzing naar een zogenaamde *back version*, die de vorige waarden van de gewijzigde kolommen bevat. Back versions worden in de database zelf opgeslagen, zo mogelijk op dezelfde database page als de row data zelf.
Dit biedt een vorm van r*ead consistency* voor leesacties. Het transactienummer van een leesactie wordt vergeleken met het transactienummer van de data rows die worden gelezen. Is de eigen transactie eerder gestart, dan worden back versions geraadpleegd om de juiste kolomwaarden samen te stellen.
Een consistente representatie van de database is zo gewaarborgd voor leesacties zonder dat dit schrijfacties blokkeert.
Firebird gebruikt dit *versioning* mechanisme niet alleen voor *snapshot consistency*, maar ook voor *concurrency contro*l. Van elke transactie wordt de status bijgehouden in speciale Transaction Information Pages (TIP) van de database. Firebird signaleert een *update conflict* wanneer een rij die voor wijziging in aanmerking komt, geschreven is door een actieve transactie. Dit mechanisme betekent ook dat een COMMIT-operatie uitsluitend de TIP hoeft te schrijven.

Interactieve applicaties

Deze multi-generational architecture vereenvoudigt het ontwerpen van toepassingsprogrammatuur op een essentieel punt: leesacties hoeven niet afgestemd te worden op schrijfacties. Dat is belangrijk voor interactieve systemen waarin gebruikers door gegevens bladeren of navigeren of een rapportage aanvragen. Zulke gebruikersactiviteit werkt verlammend wanneer het DBMS een systeem met read locks hanteert. De read locks blokkeren algauw zoveel gegevens dat throughput van schrijfacties afneemt en wachttijden toenemen[12]. De multi-generational architecture zorgt ervoor dat Firebird DBMS in vergelijkbare situaties goede responsetijden blijft bieden.

Conclusie

Het open source DBMS Firebird kan met een gerust hart worden ingezet als 'low cost en low hassle DBMS' voor de meeste databasetoepassingen. De community is door haar omvang, toewijding en professionele organisatie een verzekeringspolis voor de continuïteit van database-gebaseerde bedrijfsprocessen.
Firebird is krachtig en betrouwbaar genoeg voor bedrijfskritische toepassingen, terwijl het eenvoudig genoeg is om goed te werken zonder dat ernstig kijkende mannen in witte jassen voortdurend aan knoppen hoeven te draaien[13]. Overigens is Firebird-expertise in ruime mate beschikbaar, zowel via de mail lists, als in Nederland door geroutineerde DBA's en ontwikkelaars.
Het belangrijkste voordeel van Firebird is evenwel dat de database administrator zich veel minder met de fysieke aspecten van de database hoeft bezig te houden. Daardoor is het mogelijk dat hij zich toelegt op het bepalen van de informatie-inhoud van de database, dat wil zeggen metadata management, het ontwikkelen en beheren van gegevensdefinities en informatiemodellen. Suksesvolle aansluiting van informatievoorziening op de bedrijfsvoering is daar direct bij gebaat.

Marco Bommeljé is partner bij Bommeljé Crompvoets en partners bv

Noten

1 *Data Base Newsletter, March & May 1982, An interview with Edgar F. Codd, onlangs opnieuw gepubliceerd door de Business Rules Community naar aanleiding van diens overlijden.]*
2 *DB/M jaargang 2003, nr 4.*
3 *Ik doel hier op de integratie van applicaties en gegevens op bedrijfsniveau. Het blijkt een welhaast onoplosbaar probleem. De percentages geslaagde datawarehousing en EAI-projecten komen in geen enkel onderzoek boven de 50% uit (zie o.a. het rapport van Michael Klotz "Integration: what they don't tell you.", eBI LLC, 2002). Dit probleem is naar mijn inzien alleen oplosbaar door terug te gaan naar de basis en data en metadata management strak te organiseren, om te beginnen op conceptueel nivo. De verlossing komt in ieder geval niet van één van de modieuze techno-afkortingen uit de alfabet-soep die over ons wordt uitgestort.*
4 *Interbase is oorspronkelijk ontwikkeld voor minicomputers, een grotendeels vergeten categorie van multi-user computers die zich bevinden in het midden van het spectrum tussen mainframe en personal computer. Minicomputers hadden veelal een leveranciersspecifiek besturings-*

systeem. Bekende minicomputers waren o.a. DEC, Wang, Data General, Hewlett Packard, Datapoint en vele anderen. Zie: http://www.wikipedia.org/wiki/Minicomputer
Minicomputers hadden een zeer beperkte CPU-, geheugen- en schijfcapaciteit. Interbase was ontworpen voor een machine met een 16-bits CPU en met 64K RAM geheugen. Dat verklaart waarom Interbase en Firebird zo'n kleine footprint hebben en spaarzaam met resources omspringen.]

5 *Enkele global corporates die Interbase gebruiken zijn Motorola, Nokia, MCI, Northern Telecom, Bear Stearns, Deutsche Telekom, the US Army, NASA, Boeing, etc. Een langer overzicht wordt bijgehouden door de Deutsche Interbase User Gruppe (http://www.dibug.de/Informationen/Anwenderberichte/anwenderberichte.html).*

6 *De thuisbasis van Firebird is www.firebirdsql.org. De ontwikkelaars in het project maken gebruik van de Internet werkplaats Source Forge (http://sourceforge.net/projects/firebird). De projectstatistieken van Source Forge meldden op 8 augustus jl. 970,718 downloads. Naar verwachting zal bij publicatie van deze DB/M dit aantal het miljoen benaderen. De Source Forge projectstatistieken laten een constante, hoge project-activiteit zien. Firebird hoort al tamelijk lang tot de meest actieve 50 á 70 projecten van de 66.500 projecten die Source Forge herbergt. Source Forge geeft overigens geen gedetailleerde informatie over wat de cijfers in de projectstatistieken precies weergeven.*

7 *Veel van de informatie in dit artikel is afkomstig van de IBPhoenix website www.IBPhoenix.com. Men vindt daar ook de lezenswaardige ontstaansgeschiedenis van Interbase en Firebird verteld door Ann Harrison en Jim Starkey zelf.*

8 *De conceptuele integriteit wordt door Brooks, auteur van het klassieke boek The Mythical Man Month, de belangrijkste kwaliteit van een softwareproduct genoemd.*

9 *Een klein aantal van de vrijwilligers krijgt stipendia van de FirebirdSQL Foundation. Andere open source projecten worden in hoge mate door belanghebbende bedrijven gefinancierd. De Mozilla-organisatie werd tot voor kort volledig financiëel en materieel op de been gehouden door AOL. Sinds kort moet Mozilla op eigen benen staan, maar weet zich daarbij sterk gesteund door AOL, IBM, Oracle, Sun en andere bedrijven.*

10 *Zie het interview met Paul Beach op www.interbase-world.com.*

11 *Aangezien database shadowing alleen kan met lokale schijfeenheden, biedt het geen bescherming tegen calamiteiten zoals aardbevingen en neerstortende vliegtuigen. De data loss kan in dergelijke situaties beperkt worden door te gaan werken met replicatie-technieken, maar die zijn per definitie complex.*
Firebird's database shadowing kan wel worden ingezet als een elegante mogelijkheid om vrijwel volcontinue beschikbaarheid te bereiken. Op Firebird-architect list wordt gediscussieerd over mogelijkheden om dit als backup-strategie uit te werken.

12 *Een DBMS dat met read locks werkt, vereist daarom een zeer zorgvuldig ontwerp van lees-transacties in toepassingsprogrammatuur. Sommige data access componenten die gebruikt worden in applicatieprogrammatuur bevatten intelligentie om het ophalen van grote result sets van bevragingen uit te stellen totdat ze nodig zijn, bijvoorbeeld om op het scherm af te beelden. Dat is mooi omdat dit een oplossing biedt voor het responsetijdprobleem van de gebruiker die de bevraging uitvoert. Het is minder mooi, omdat het a-sociaal is tegenover andere gebruikers die niet meer verder kunnen als gevolg van openstaande read locks.*

13 *De witte-jassen opmerking is gejat uit correspondentie op een van de mail-lists. Daarin beklaagde iemand zich over de onmogelijkheid om Firebird te 'verkopen' aan de directie van zijn bedrijf, omdat het zo eenvoudig en goedkoop was. Daarop bood iemand van IBPhoenix aan om tegen betaling als deskundige in een witte jas op te treden.*