VELDWIJK

XXmL

In de psychologie bestaat de theorie van het gelijkblijvende risico. Die theorie zegt bijvoorbeeld dat automobilisten gevaarlijker gaan rijden naarmate de veiligheid van auto's toeneemt. In de ICT lijkt ook zo'n soort wet te bestaan: technologische vooruitgang leidt tot nieuwe methoden en hulpmiddelen die deze vooruitgang (groten)deels weer tenietdoen. Eens maakten 3GL talen het programmeren eenvoudiger, maar wel ten koste van de efficiency van programma-executie. Later vergrootten 4GL's de productiviteit nog verder en kwamen gestructureerd programmeren en het vooraf ontwerpen van applicaties in diskrediet. Weer later kwam met de PC als vervanger van de domme terminal de grootste ramp tot nu toe: *fat* client/server. Oorspronkelijk bedoeld om de rekenkracht van de PC te benutten en de servers te ontlasten, ontaardde client/server in een zo zware belasting van PC clients dat de toegenomen rekenkracht meer dan volledig werd opgesoupeerd. In de jaren negentig gold de regel dat je elke 2 á 3 jaar nieuwe client hardware nodig had om toepassingen te laten draaien op de nieuwste versie van Oracle*Forms, SQL Windows of Uniface. Organisaties konden zich hieraan ook nauwelijks onttrekken want alle serieuze tools waren *fat* client. Het einde van de nachtmerrie kwam jaren en miljarden later met de overgang naar de huidige *thin* client/server.

Ik schrijf deze inleiding omdat de geschiedenis zich weer lijkt te gaan herhalen. De verder toegenomen rekenkracht, opslagcapaciteit en netwerkbandbreedte lijken de komende jaren te worden opgegeten door XML. Voor alle duidelijkheid: ik heb op zichzelf niets tegen XML. Net als bij de servicebussen van twee columns terug geloof ik niet dat XML enig wezenlijk semantisch probleem zal oplossen, maar met het idee om gegevens in te bedden in metagegevens is niets mis. Het is zelfs oké om ieder individueel bericht (lees: record) in te bedden in een wolk van metagegevens en eenvoudige constraints, mits het daarbij gaat om niet al te massale uitwisseling van individuele berichten. XML is helemaal geweldig als het gaat om *loosely coupled* berichtenverkeer, dat wil zeggen, niet geheel vooraf gedefinieerde gegevensuitwisseling tussen systemen.

Echter, de toepassing van XML in klassieke toepassingen die grootschalig gegevens uitwisselen is pure dwaasheid en in een aantal omgevingen een ramp in wording. Er bestaan legio vormen van berichtuitwisseling die van nature (!) worden gekenmerkt door een periodieke overdracht van grote hoeveelheden gelijksoortige gegevens tussen systemen: denk aan het dagelijks verzenden van verkoopgegevens van filialen naar een centraal magazijn, het maandelijks aan de fiscus melden van betaalde salarissen of het periodiek controleren van samenloop tussen lonen en uitkeringen. Overal zie ik nu dat dergelijke processen worden *geXML'd*. Het bizarre daarbij is dat de gegevensuitwisseling zelf gewoon blijft bestaan uit batchbestanden, zij het dat die zijn opgebouwd uit individuele, gelijkvormige XML deelberichten, elk met de enorme XML-tag overhead. In het meest extreme geval dat ik tot op heden ben tegengekomen was een eenvoudige periodieke bestandsvergelijking geïmplementeerd in dynamische message queues, met als gevolg enorme performance-problemen en verlies aan greep op de volledigheid van de gegevens. Maar zelfs alleen het *XML'en* van klassieke batch-bestanden leidt tot enorme problemen. Allereerst blijkt in brede kring de opvatting te leven dat de XML berichten zelf in een database moeten worden gestopt. Vermoedelijk gebeurt dit omdat daarmee de associatie wordt gecreëerd met een XML database. Wat er in feite gebeurt is het ontoegankelijk opslaan van data. XML data in een RDBMS zijn in theorie niet performant te benaderen en in de praktijk helemaal niet te benaderen. De XML berichten zijn zo opgeblazen dat *gezipte* opslag in een BLOB al normaal is geworden. Bij een salarissysteem dat ik heb mogen auditen bestond 90 procent van de data uit dergelijke gezipte XML berichten. En onlangs heb ik mogen waarnemen dat het zo'n twee weken duurde om 10 miljoen 'geBlobte' XML berichten uit te pakken en in een min of meer toegankelijke database te stoppen.

Tenzij de organisaties die ik in mijn beroepspraktijk tegenkom niet representatief zijn, stevenen we af op een XML infarct van grote omvang. Ik verwacht dat dit als eerste zal gebeuren in het overheidsdomein. Daar is men hard bezig met het vormen van informatieketens tussen allerlei overheidsorganen en (semi)publieke organisaties als onderwijs- en zorginstellingen. Waar jaarlijkse gegevensuitwisseling zou kunnen bestaan uit enkele tientallen Gigabytes, zitten overheidsorganisaties met XML nu al tegen de Terabyte-grens. Ik ken een geval waar pakweg 200 Gigabyte aan normale data al is opgeblazen tot meer dan 20 Terabyte – een factor 100! En bedenk daarbij dat de overheid pas aan het begin staat van de vorming van gegevensketens.

Het goede nieuws bij dit alles is dat we anders dan bij client/server zelf voor de keuze staan om al dan niet gebruik te maken van XML. Als XML wordt gebruikt waarvoor het is bedoeld dan komt alles misschien nog goed. Zo niet dan rijden we met onze autogordels en kreukelzones recht het XXmL ravijn in.

René Veldwijk

Dr. R.J. Veldwijk (rene.veldwijk@faapartners.com) is partner bij FAA Partners, een onderdeel van de Ockham Groep.