



*IT-industrie kan leren
van luchtvaart bij
ontwikkelen van zichzelf
beherende systemen*

De grenzen van autonomic computing

De complexiteit van IT-omgevingen en van het beheer ervan neemt toe. IT-beheerders hebben de zware taak te voldoen aan alle door gebruikers gestelde eisen, tegen zo laag mogelijke kosten. De talrijke manieren waarop en vormen waarin problemen, events en alerts aan IT-beheerders worden gerapporteerd, maken het beheer nog complexer. Na analyse van de verschillende rapporten is menselijke tussenkomst nodig om de problemen te kunnen oplossen. Bram Dons bespreekt het concept van autonomic computing, waarbij sprake is van een zichzelf beherende ICT-omgeving.

Bram Dons

Er valt in de wet van Moore nog geen vertraging te constateren die een verdere technologische ontwikkeling in de IT-industrie in de weg zou staan. In plaats daarvan heeft de exploitatie van allerlei nieuwe technologieën ertoe geleid dat de complexiteit van systemen en netwerken in de afgelopen jaren nog steeds is toegenomen. Maar ook op het gebied van software hebben de nodige veranderingen plaatsgevonden. Zo hebben softwareontwikkelaars de afgelopen jaren een viertot zesvoudige hogere verwerkingskracht van applicaties weten te bereiken, wat geresulteerd heeft in nog meer geavanceerde softwareapplicaties en omgevingen. Op talrijke verschillende gebieden (systemen en componenten) heeft een exponentiële groei in aantallen en prestaties plaatsgevonden. De toepassing van moderne databasetechnologie en internet hebben belangrijk bijgedragen aan de spectaculaire groei van opslagsystemen. Netwerken worden in toenemende mate toegepast in omgevingen met heterogene en gedistribueerde systemen.

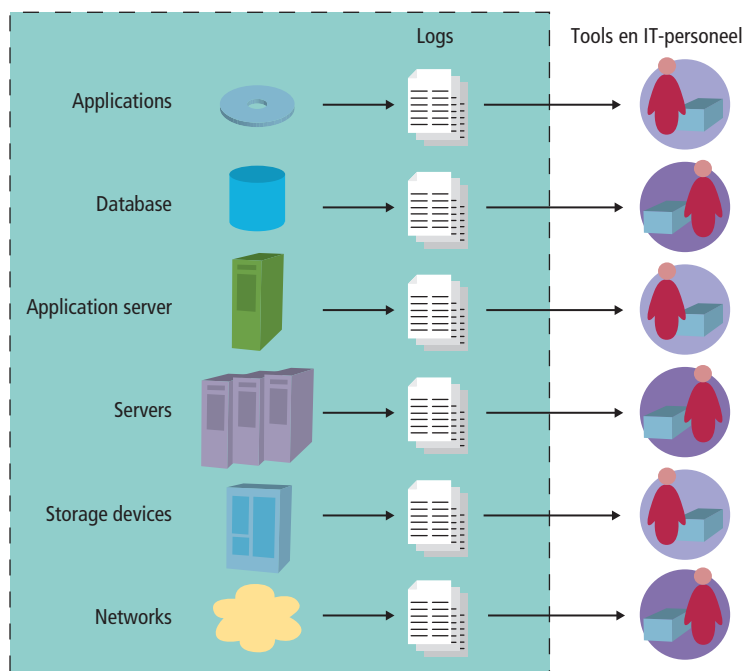
Al deze factoren hebben ICT-omgevingen gecreëerd waarin netwerksystemen

worden geconfronteerd met een sterk variërende en vaak onvoorspelbare werkbelasting. Voor onderhoud, installatie en tuning van deze kostbare en complexe systemen zijn steeds meer talentvolle en ervaren IT-beheerders nodig. Het dilemma waar de IT-beheerindustrie voor staat is dat er nu en zeker in de toekomst onvoldoende IT-specialisten beschikbaar zijn om alle computersystemen naar behoren te laten fungeren. Uiteraard wordt naar oplossingen voor dit toenemende beheerprobleem gezocht. Zoals in het verleden met zoveel geheel of gedeeltelijk handmatig uitgevoerde processen is gebeurd, kan een oplossing voor het IT-beheerprobleem worden gevonden in een volledige automatisering ervan. Dit heeft de afgelopen jaren bij een aantal toonaangevende leveranciers geleid tot het denken over een zogenaamd autonomic computing-model.

Wat is autonomic computing?

De term *autonomic computing* werd een paar geleden door IBM gelanceerd in een gelijknamig onderzoeksprogramma. De term *autonomic* is niet door alle partijen overgenomen, vaak alleen omdat

dossier utility computing



Figuur 1 Verschillende soorten rapportagesystemen (bron: IBM)

deze van IBM afkomstig was. Microsoft bijvoorbeeld praat over het *Dynamic Systems Initiative*, terwijl andere leveranciers een meer generieke en minder suggestieve term als *self-managing* aanhouden. Hier houden we IBM's naamgeving *autonomic computing* aan.

Afgezien van de naam is er een algemeen geaccepteerde benadering van wat een autonoom, zichzelf beherend (ook wel: zelfbeherend) IT-systeem zou moeten kunnen doen. De term *autonomic* is afgeleid van de menselijke biologie. Ons autonoom werkende zenuwstelsel bewaakt onder meer de hartslag, controleert de bloedsuikerspiegel en houdt de lichaamstemperatuur constant. Op dezelfde manier zouden zelfbeherende autonome voorzieningen kunnen anticiperen op de eisen die aan een IT-systeem worden gesteld en zonder of met minimale menselijke interventie problemen in dat systeem kunnen oplossen. Het resultaat is dat IT-professionals zich dan volledig kunnen concentreren op andere taken (met als resultaat dat er minder IT-beheerders nodig zijn). Autonomic computing is een zelfhelende, zelfbeherende, op policy's gebaseerde infrastructuur. Daarvoor is een gedefinieerd proces nodig dat dynamisch veranderingen aanbrengt, al naar gelang de zakelijke behoeften. De daarvoor beno-

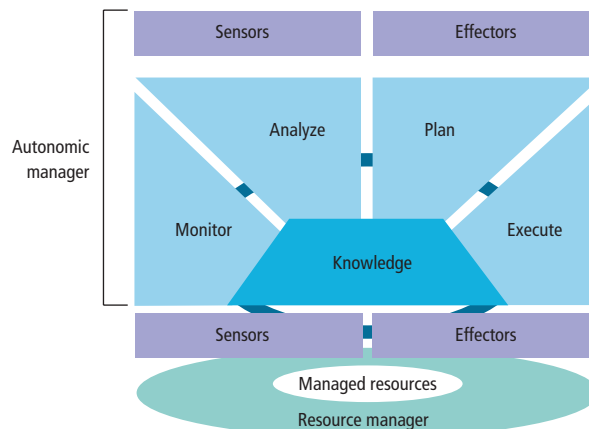
digde sleutelcomponenten omvatten:

- een algemene systeemadministratie;
- een autonoom werkende monitorfunctie;
- policybeheer;
- een meetsysteem voor transacties;
- een systeem voor de detectie van problemen en de oplossing daarvan.

Componenten van autonomic computing

In een autonome, zichzelf beherende omgeving worden uiteenlopende systeemcomponenten, van hardware (zoals opslagsystemen, desktops en servers) tot software (operating systems, middleware en zakelijke applicaties), opgenomen in een ingebedde *control loop*-functie (zie figuur 2). Alhoewel control loops bij al deze componenten uit dezelfde fundamentele onderdelen bestaan, kunnen hun functies in vier categorieën worden opgesplitst:

- *Self-configuring* houdt in dat elk nieuw device zichzelf automatisch in de bestaande IT-omgeving opneemt zodat niet langer een IT-beheerder nodig is die een speciale installatieprocedure moet volgen.
- *Self-healing* betekent dat het systeem zelf in staat is aan elke verstoring van het systeem het hoofd te bieden. Het detecteert het probleem, stelt de



Figuur 2 Autonomic computing-architectuur: control loops (bron: IBM)

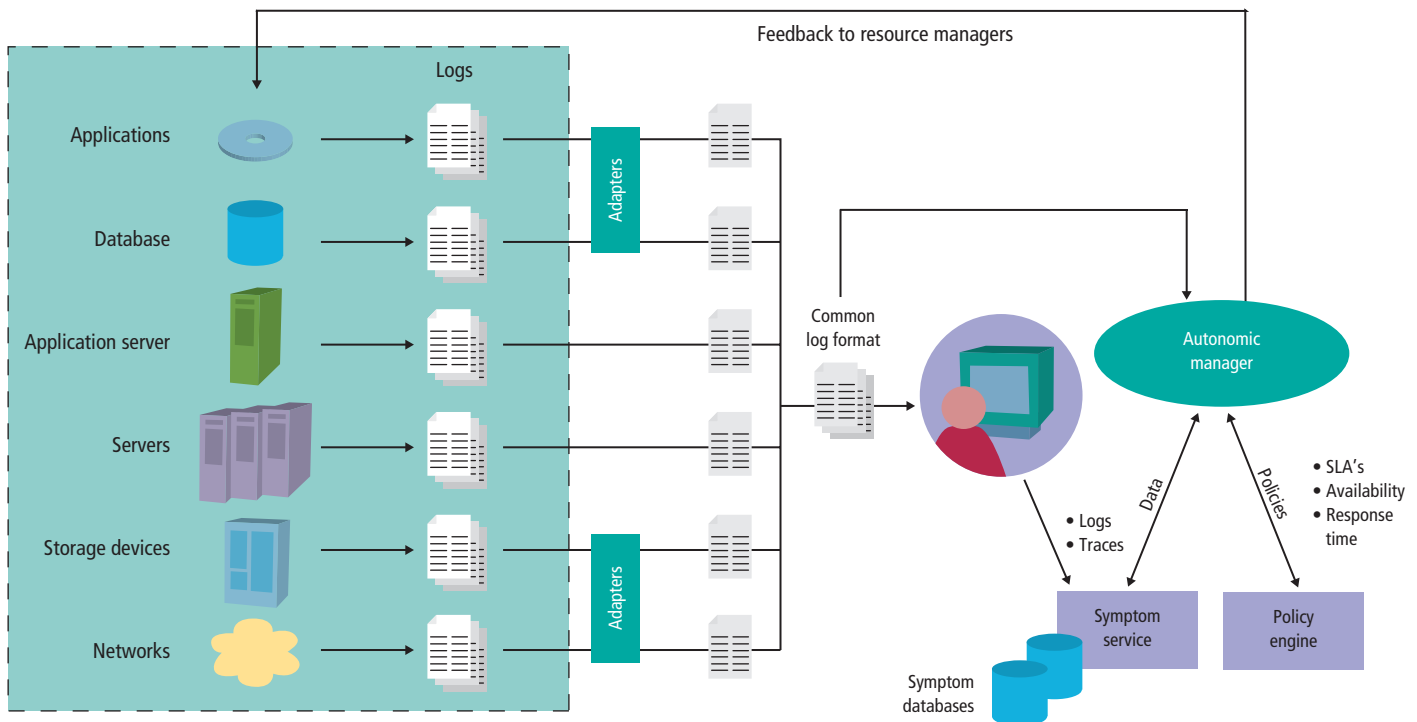
diagnose en herstelt de opgelopen schade (zie figuur 3).

- Bij *self-protecting* detecteert en beschermt het systeem zich tegen schade van bepaalde gebeurtenissen, bijvoorbeeld uitval van apparatuur of aanvallen van hackers en virussen.
- *Self-optimizing* betekent dat het systeem automatisch op de meest efficiënte manier taken en subtaken naar de verschillende devices distribueert.

Zoals altijd bestaat de uitdaging voor de IT-industrie eruit om al deze 'self-doing'-functies in een werkbare technologie te vertalen, technologie die op elk systeemniveau kan functioneren. Het goede nieuws is dat de ontwikkelaars niet helemaal vanaf het begin hoeven te beginnen, want hun voorgangers hebben al voor een uitgebreide *legacy* autonomic technologie gezorgd waarop de ontwikkelaars nu kunnen voortborduren. In de strategie van autonomic computing zijn de meeste 'self'-functies gebaseerd op het uitgangspunt dat de data en de verwerking gedistribueerd zijn.

De continuous control loop

De genoemde functies helpen niet veel bij individuele systeemcomponenten. Voor een autonoom gedrag op dat niveau is een aanvullende strategie nodig, de *continuous control loop*. De gedachte die aan de continuous control loop ten grondslag ligt, is dat elke component binnen het systeem – hardware of software – niet alleen weet hoe het bepaalde toegewezen taken moet uitvoeren, maar ook via interne mechanismen de eigen werking bewaakt en zo



Figuur 3 Autonomic computing self-healing systemen (bron: IBM)

nodig correcties uitvoert. Deze mechanismen omvatten een ingebouwd op hardware of software gebaseerd sensor-netwerk dat kennis heeft van de manier waarop sensordata moeten worden geïntegreerd en van wat het device moet doen als er iets fout gaat. Mettertijd zullen sensoren steeds meer zelflerend worden. Apparaten zullen op termijn in staat zijn de eigen sensoren te herschikken, bijvoorbeeld om een subactiviteit die meer fouten vertoont beter te kunnen monitoren.

Een apparaat zal zoveel mogelijk proberen problemen lokaal af te handelen, maar het heeft altijd de mogelijkheid om hulp van een groter systeem in te roepen. Het systeem kan ook recursief worden uitgevoerd, waardoor het grotere systeem al zijn eigen interne routines kan gebruiken op het moment dat het verzoek om hulp komt. Het kan dan een correctieve actie ondernemen of op zijn beurt hulp van een nog groter systeem aanroepen. Op die manier ontstaat een 'diep' recursief autonoom werkend systeem.

Blauwdruk voor autonomic computing

Het idee om technologie te gebruiken voor het beheer van technologie is niet nieuw. Veel ondernemingen in de IT-

industrie hebben al op dit concept gebaseerde producten ontwikkeld. Efficiënter systeembeheer met autonomic computing is echter alleen haalbaar wanneer de verschillende technologieën voor IT-beheer met elkaar kunnen samenwerken. Om dit in een heterogene infrastructuur mogelijk te maken moeten leveranciers instemmen met op standaarden gebaseerde autonome systemen. In een dergelijke architectonische blauwdruk voor een autonomic computing-systeem wordt een overzicht gegeven van de fundamentele concepten, constructies en gedragingen voor het bouwen van een zelfbeheerend autonoom werkend systeem.

Een blauwdruk van hoe een systeem er idealiter uitziet is één ding, hoe dit in de praktijk moet worden geïmplementeerd is heel iets anders. Gelukkig bestaan er al veel bouwstenen voor een *autonomic framework*, hoewel ze nog niet met goed met elkaar samenwerken.

Software toolkit voor Autonomic Network Services

Aan de basis van een autonomic framework staan netwerkservices. De meeste van de huidige netwerkservices bezitten al een autonoom werkende zelfconfiguratie. Bepaalde vormen van autonomic net-

werkservices vinden we al in het Domain Name System (DNS) en het Dynamic Host Configuration Protocol (DHCP). Op netwerkgebied kennen we ook Network Information Services (NIS) en opvolger NIS+. Voor lokalisatie van netwerkbronnen en netwerkdiensten zijn er respectievelijk de Directory Service (LDAP) en het Services Location Protocol (SLP).

Een methode om tot een autonomic computing-omgeving te komen wordt door IBM geboden met de Autonomic Computing Toolkit. Met behulp van deze toolkit kunnen ontwikkelaars zelfconfigurerende en andere autonome voorzieningen aan hun software toevoegen. Ontwikkelaars kunnen met de nodige inspanning de toolkit in verschillende client/server- en peer-to-peer-netwerken integreren. In IT-omgevingen past IBM's *autonomic manager* logisch gezien als een *add-on* in bestaande DHCP-, DNS-, LDAP- en andere serverplatforms. Het kan ook als raamwerk dienen voor *early adopters*, om te zien wat de impact van autonomic technologie op hun IT-omgeving is. De toolkit is gebaseerd op het CIM Object Model en werkt met MOF-bestanden.

Een autonomic computing-systeem kan echter niet bestaan in een hermetisch

Utility computing

Autonomic computing is niet identiek aan *on demand computing* of *utility computing*. Deze twee concepten zijn breder en hebben betrekking op zowel de applicatielaag en de zakelijke processen als de gebruikers, terwijl autonomic computing alleen de infrastructuur betreft.

afgesloten IT-omgeving. Daarvoor zijn componenten nodig, afkomstig van verschillende leveranciers, die zijn gebaseerd op open standaarden zodat ze door autonomic managers kunnen worden beheerd. De toolkit probeert zich daarom te conformeren aan de bestaande enterprise-beheerstandaarden. Om de ontwikkeling van open standaarden te faciliteren nemen diverse grote leveranciers deel aan organisaties, waaronder Distributed Management Task Force (DMTF), Organization for the Advancement of Structured Information Standards (OASIS), Global Grid Forum (GGF), Internet Engineering Task Force (IETF), World Wide Web Consortium (W3C), Java Community Process (JCP) en Storage Networking Industry Association (SNIA).

Autonomic policy's

Autonomic computing gaat over de verschuiving van de lasten van het systeembeheer van de mens naar de technologie. Er is echter één aspect waar software-ontwikkelaars weinig of geen aandacht voor hebben, namelijk de vraag hoeveel bevoegdheden we straks naar deze autonome systemen moeten delegeren. Hoe weten we of dergelijke autonome systemen zich gaan gedragen op de manier waarop wij dat willen? En wie heeft het in de laatste analyse bij een optredend probleem voor het zeggen, het autonoom werkende systeem of toch nog de mens?

Na enige aarzeling krijgen we van softwareleveranciers als antwoord: "Natuurlijk, de mens heeft altijd het laatste woord. Uiteindelijk is het de IT-manager die de policy's van het systeem bepaalt, het systeem zelf stelt deze niet in!" Echter, om de essentie van een dergelijke uitspraak te kunnen beoordelen, is het toch goed even terug te gaan naar de jaren zeventig en tachtig, toen de veelbelovende expertsystemen en programma's met kunstmatige intelligentie 'booming' waren. Het idee was om menselijke ervaring in diagnostiek in te voeren. De programmeurs konden dan eenvoudig deze kennis met hun if-then-else-regels in hun programma opnemen.

De programma's deden ook precies wat van ze verwacht werd, alleen het resultaat was net zo goed als de ingebouwde regels zelf waren. Zodra zich een situatie voordeed die niet in de regels stond, faalde het programma.

Dit soort problemen kunnen we ook verwachten bij op policy's gebaseerde autonome systemen. Wat gebeurt er bijvoorbeeld als door verschillende afdelingen of organisaties onafhankelijk van elkaar gecreëerde autonome systemen via het netwerk met elkaar in aanraking komen en blijkt dat hun policy's conflicterend zijn? Wat gebeurt er als een systeem automatisch het netwerkverkeer omleidt om een congestieprobleem op te lossen, met als resultaat een verstopping op een ander netwerk? Wie voert de arbitrage in deze situaties, de menselijke operator? Misschien wel, of misschien niet, wanneer het systeem was uitgerust met software voor conflictsituaties. Maar dan rijst de vraag: wie is er verantwoordelijk voor de creatie en het beheer van deze high-level software en de instelling van de policy's? De leveranciers van deze systemen, de ondernemingen zelf of de overheid soms?

Een voorbeeld van een gebrek aan eindverantwoordelijkheid is de grootschalige verstoring van de Noord-Amerikaanse elektrische stroomvoorziening (ook een *grid*) in augustus 2003. Wanneer verschillende autonome voorzieningen gekoppeld zijn, kan een fout in een van de systemen een domino-effect hebben. Grootschalige autonome computersystemen zullen geen uitzondering op deze regel zijn, in elk geval wanneer ze zijn samengesteld uit individuele subsystemen die policy's volgen die niet op elkaar zijn afgestemd.

Human-centered autonomic computing de toekomst?

Het is misschien verstandig dat softwareleveranciers eerst eens kijken naar de al eerder opgedane ervaring binnen de vliegtuigindustrie in de jaren zeventig en tachtig. Bij de eerste golf van automatisering in de cockpit hadden vliegtuig-

ontwerpers nog het vooruitstrevende en ambitieuze idee om een vliegtuig volledig door de computer te laten besturen, zodat de piloot alleen nog maar naar een monitor hoefde te kijken en alleen nog bij ernstige fouten mocht ingrijpen. Bij de eerste cockpitontwerpen kwamen er al snel achter dat een slecht of overgedimensioneerd geautomatiseerd besturingssysteem bij een fout de veiligheidssituatie aanzienlijk zou kunnen verslechteren, zelfs met catastrofale afloop.

In het geval van de cockpitautomatisering leidde dit in de vliegtuigwereld tot een honderdtachtig graden ommeswaai in het denken over wat computers wel of niet mochten doen. Halverwege de jaren tachtig neigden vliegtuigontwerpers naar het concept van *human-centered automation*. Daarin mogen computers alleen het vliegtuig besturen met uitdrukkelijke toestemming van de vliegeniers. Alleen als de elektronica detecteerde dat iets ernstig fout dreigde te gaan, moest het daarvoor waarschuwen en mocht het direct de besturing overnemen.

Human-centered automation brengt met zich mee dat het systeem de mensen bewaakt en alleen ingrijpt wanneer dat strikt noodzakelijk is. Het betekende in de luchtvaart in ieder geval dat de toepassing van automatisering weer in de juiste proporties werd gezet. De besturing van het vliegtuig kwam daardoor weer zo goed als volledig bij de mens te liggen. Misschien kan deze voor de IT-ontwikkelaars toch wat afwijkende gedachtegang ook dienen als sjabloon voor human-centered autonomic computing in de IT-industrie. Zullen we in de IT-industrie in staat zijn om het hiervoor geschetste beheerprobleem met autonomic computing op te lossen of zullen we bij de toepassing daarvan in de IT-industrie in dezelfde valkuil trappen als destijds de vliegtuigindustrie deed? De tijd zal het leren...

*Bram Dons is onafhankelijk IT-analist.
E-mail: b.dons@it-trendwatch.nl.*