

De auteur

Wilbert de Wolf is enterprise architecture consultant met een jarenlange brede ervaring op het gebied van software, informatie en infrastructuur.

Als consultant is hij nauw betrokken bij het voorbereiden en uitvoeren van strategische projecten waaronder mogelijke leads, overnames of bidtrajecten. Hij ontwikkelt een effectief netwerk, volgt ontwikkelingen in de markt en draagt zelf initiatieven aan, toetst deze op haalbaarheid en adviseert een MT o.a. op het gebied van investeringsbeslissingen. Hij heeft uitstekende project- en onderhandelingsvaardigheden en het vermogen diverse stakeholders waaronder CEO's te overtuigen en committeren.

LinkedIn: <http://www.linkedin.com/in/wilbertewolf>

E-mail: wilbertewolf@yahoo.com

Informatiedynamica als next-generation paradigma

Samenvatting

Toekomstige systemen volgens het informatiedynamica-paradigma worden gekenmerkt door chaos. Architecten missen de nodige wetenschappelijk valide methodieken om chaos te beschrijven en beheersen. Methodieken die wel terug te vinden is in andere disciplines zoals de chemische procesindustrie. Denk hierbij aan meet-en-regeltechniek. De vraag is wel of begrippen uit de klassieke thermodynamica opgaan voor informatiedynamica. De theoretische grondslagen hiervoor zijn al wel gevormd. Nu nog de praktische uitvoering. De procesindustrie maakt hiervoor veelvuldig gebruik van simulatietechnologie.

Natuurlijk zullen grote pakketleveranciers hierop inspringen aangezien de business voldoende uitdagingen ziet. Maar hiervoor moet het IT-collectief nog wel een slag maken. De kennis ligt voor het oprapen. Ik nodig architecten uit om deze kennis tot zich te nemen ten einde hier praktische oplossingen te bieden.

De ICT-architectuur gaat een spectaculaire ommezwaai beleven. Het is pas recentelijk dat ICT-specialisten architectuurraamwerken zoals TOGAF en DYA inzetten om hun producten op een gestructureerde wijze vorm te geven. Maar de huidige technologische ontwikkelingen vragen om andere methodieken. Andere denkwijzen die tot voorkort nog onbekend waren. Bedrijfsgrenzen vervagen. Er ontstaat een tendens waarbij federale samenwerking tussen bedrijven onderling worden gelegd ten behoeve van een gezamenlijk economisch doel. De onderlinge concurrentiestrijd wordt voor heel even onderbroken als blijkt dat een federale aanpak tot meer winst zal leiden. Het probleem is dat veel van de huidige hulpmiddelen nog afstammen van vroeger. Softwaresystemen waren applicatiegericht daar waar tegenwoordig veel meer servicegericht wordt gewerkt. De acroniemen zijn reeds bekend: SOA, EDA, ESB, maar de hulpmiddelen ontbreken voor architecten. Er wordt getracht om de architectuurraamwerken in te passen maar dat zal onvoldoende blijken. Dat komt omdat er samen met deze servicegerichte systematiek nieuwe fenomenen opduiken.

Om te kunnen begrijpen welke fenomenen dat zijn moeten we een blik werpen op de nabije toekomst.

Binnenkort wordt de markt overspoeld met *context aware utilities* [ref_02]. Dit zijn apparaten en toepassingen die weten wie je bent en zijn afgestemd op wat je wilt. Zo rij je in een auto die in contact staat met alle auto's binnen een beperkte straal. Inhalen wordt een makkie aangezien jouw auto tegemoetkomers op ruime afstand kan herkennen dankzij een ingebouwde inhaalassistent [ref_06]. Vervolgens kom je thuis zonder sleutel te gebruiken. Je bent immers al op afstand herkend. Je dagelijkse krant ligt klaar op de Microsoft Surface tafel en jouw favoriete kopje koffie wordt gezet. Maar ook in je werk gaat het zo. Laptop, PDA, mobile het maakt niets meer uit. Al jouw data bevindt zich in de "cloud". Al die apparaten plukken data uit de lucht. Multimedia in een notendop.

Business mensen begrijpen dit. Wanneer ik dit aanklaart binnen de top 10 gerenommeerde bedrijven dan zitten alle beleidsmakers op het puntje van hun stoel. De meest lastige vraag die ICT de komende tijd voorgeschoteld gaat krijgen is wanneer we er klaar voor zijn. En vreemd genoeg is de infrastructuur er klaar voor maar de kennis om het te bouwen staat nog in de kinderschoenen.

Overigens staat niet iedereen te springen om een explosieve groei van deze *context aware utilities*. Ze koppelen namelijk allemaal via internettechnologie. Een populair mechanisme hiervoor is RFID [ref_01] waarbij elke utility uniek geïdentificeerd kan worden met internet als transportmechanisme. Men vreest wel dat het internet verstopt gaat raken als gevolg van het geklets tussen utilities wat gekscherend "The Internet of things" wordt genoemd.

Zo zijn er enorm veel uitdagingen. Om utilities te voorzien van data zijn er twee mogelijkheden. Of de data wordt gepersisteerd op het apparaat zelf of op de oude vertrouwde centrale server. In het tweede geval heeft dit tot gevolg het dataverkeer tussen server en apparaat de bottleneck gaat vormen. Dat komt omdat

apparaten steeds sneller worden terwijl het dataverkeer begrensd is aan beschikbare netwerkcapaciteit. Het resultaat is een latente applicatie en dat willen we niet.

Een mogelijke oplossing is om datatransacties met moeder-server af te schaffen voor niet-kritische software systemen. Kost toch allemaal teveel tijd. Uitwisseling van data gebeurt enkel tussen apparaten onderling. Soms vertelt jouw applicatie de waarheid, soms niet. Stel dat je een boek bestelt via een applicatie op jouw PDA. Je krijgt de bevestiging dat het boek voorradig is, maar die informatie is afkomstig van een naburige computer die denkt de waarheid in pacht te hebben. Dit wordt *subjective consistency* genoemd. Ondertussen zal jouw applicatie de bestelling blijven toetsen door contact te maken met andere naburige apparaten en het blijkt dat het boek eigenlijk niet op voorraad is. Je wordt hiervan geïnformeerd door jouw pda. Dit wordt *eventual consistency* [ref_03] genoemd.

Dus de trend is dat apparaten en softwarecomponenten uitgerust worden met veel meer autonomie dan vroeger. Het dataverkeer wordt fijnmazig en diffuus. In het geval van *Event Driven Architecture* (EDA) worden ontelbare databerichtjes (events) afgevuurd en geconsumeerd door evenzoveel onbekende afnemers. Maar deze ruwe communicatievorm wordt uitgebreid en verfijnd doordat events ook weer zelf events kunnen genereren middels *complex event processing* [ref_12].

Maar het gaat verder. In de toekomst worden events geladen met eigen intelligentie. Dergelijke agents kunnen onderling zaken met elkaar doen. Zo kan ik straks mijn spaargeld de opdracht geven zichzelf weg te zetten zodanig dat het mij de meeste rente oplevert. Het kader is hiervoor al opgezet. Een CALMA-framework [ref_02] voor het context-aware paradigma en BDI voor het opzadelen van agents met persoonskenmerken zoals *Beliefs*, *Desires*, en *Intentions*.

Agents hoeven niet te worden gepresenteerd als next-generation-services. Alsof na de services we verder gaan met agents. Agent en service bestaan onafhankelijk van elkaar. Een agent wordt voorgesteld als stakeholder met autonoom gedrag en mogelijkheden tot samenwerking ten behoeve van het zijn individuele doel. Een dergelijk *Agent-Based Computing* (ABC) concept [ref_13] wordt toegepast om op een flexibele manier computerintelligentie en autonomie te modelleren voor complexe softwaresystemen. Aan de andere kant worden services ontwikkeld middels *Service Oriented Computing* (SOC) om applicatielogica te exposeren middels message-based interfaces over een netwerk zoals het internet. Een groot gebrek van SOC is dat het niet de kwaliteiten van een ABC herbergt zoals autonomie en flexibiliteit.

Van nature is de content van ABC en SOC goed overdraagbaar. Door in een systeem zowel agents en services te integreren, kan ABC zichzelf versterken middels SOC en vice-versa waardoor krachtigere computersystemen ontstaan. Hiermee is het *agent service-oriented design* (ASOD) concept geboren.

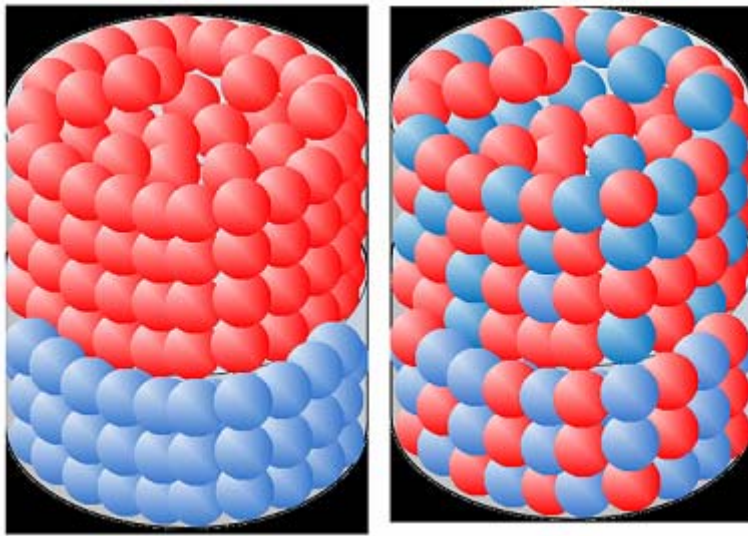
Maar goed, zover is het nog niet en de vraag is of ICT'ers het wel gaan redden met de huidige kennis op het gebied van SOA. Vooralsnog is SOA een rudimentaire oercommunicatie tussen providers en subscribers. Meer zoals dieren elkaar toebrullen. Echter om intelligentie en autonomie neer te leggen op het niveau van agents is een verfijnd gesprek noodzakelijk. Talloze agents die geciviliseerde gesprekken moeten voeren met elkaar. Hoe moeten we hiermee omgaan zonder dat het een onbeheersbaar kippenhok wordt? Beter gezegd, hoe moeten we de communicatie regelen zonder dat het uitloopt op een chaos?

Chaos

Om te kunnen begrijpen welke invloed chaos heeft op al onze toekomstige systemen moeten we eerst het begrip chaos uitdiepen.

Chaos gaat gepaard met het begrip entropie. Entropie is een begrip die alom bekend is in de chemische procesindustrie. Entropie is het beste te omschrijven als de kans dat een bepaalde situatie optreedt afgezet tegen het totaal aantal mogelijke situaties. Hoe kleiner die kans wordt hoe groter de entropie. In 1872 slaagde Boltzmann er in om een verband te geven tussen de begrippen entropie en de kans dat een bepaalde situatie optreedt. De onderzoeker J.D. Fast [ref_05] gaf er de volgende betekenis aan: Stel een koker voor met boven rode en onder blauwe knikkers. De koker wordt geschud.

Onder toeneming van entropie mengen de knikkers tijdens het schudden. Maar vreemd genoeg treedt nooit meer de geordende begintoestand op. Dat komt omdat er zoveel meer onregelmatige dan regelmatige configuraties mogelijk zijn voor die knikkers. De kans op een geordende begintoestand is praktisch 0.



Maar waarom is dit nu zo belangrijk? Wat levert ons die Boltzmann theorie nu op? Immers informatica heeft betrekking op elektronica terwijl de verhandeling van Boltzmann slaat op fysieke materie. De wetenschapper Claude Shannon had jaren terug dezelfde gedachte. Hij probeerde aan te tonen of er een verband bestond tussen entropie in realiteit en de virtuele wereld. Het bleek een meesterzet. Shannon kwam erachter dat beide entropiebegrippen nagenoeg dezelfde wiskundige betrekking opleverde hetgeen hem deed besluiten om het begrip informatie-entropie in het leven te roepen.

Vervolgens storten veel onderzoekers zich tegenwoordig op informatie-entropie [ref_09]. Wat echter hierbij opvalt is het hoge theoretische gehalte zonder dat er praktische toepassingen tegenover staan. De business gaat hier niet op wachten. ICT schreeuwt om praktische oplossingen voor het controleren van niet-deterministische systemen waar chaos een onderdeel van uitmaakt.

Toepassingsgebied

De Boltzmann entropie slaat terug op de klassieke thermodynamica. De Shannon entropie moeten we onderbrengen in een geheel nieuw paradigma, de informatiedynamica. Omdat de formules van Boltzmann entropie en Shannon entropie aan elkaar gelijk zijn gelden de wetten voor de thermodynamica dan ook voor informatiedynamica? Bestaat er zoiets als "de wet op behoud van informatie"? Of het feit dat informatie de neiging heeft om zonder extra moeite steeds incoherenter te worden en nooit coherenter. Dan moet gedacht worden aan termen als informatieconvectie ofwel: hoe kan ik een informatie- of kennisinfrastructuur zo inrichten dat de informatie zich zonder deze expliciet rond te pompen als vanzelf verspreid? Net zoals je vroeger van die dikke convectieradiatoren en buizen had die ook geen pomp nodig hadden.

Er is een kenmerkend verschil hoe we denken over het huidige SOA-paradigma en hoe we moeten denken over informatiedynamica. Het SOA-paradigma concentreert zich op twee basisprincipes, safety en liveness [ref_10] wat erop neerkomt dat een bericht altijd maar 1 keer ontvangen kan worden door een subscriber en het bericht ook altijd gegarandeerd afgeleverd wordt aan de subscriber. In een eenvoudig systeem zal dit standhouden maar informatiedynamica is veel weerbarstiger. Die gaat gepaard met veel meer onzekerheid en onvolledigheid. Tot nog toe worden event-based systemen voorgesteld als een berichtensysteem waarbij de semantiek uniek opgesloten zit in het bericht. Een gepubliceerd bericht bevat kennis en semantiek. Een subscriber interpreteert het bericht en neemt de kennis tot zich.

Bij informatiedynamica wordt de semantiek over veel meer kennisdragers gespreid. Afhankelijk van de constitutie van de kennisdragers kunnen semantische conclusies getrokken worden. Informatiedynamica heeft meer een realtime karakter.

Een publisher stuurt niet een bericht maar een signaal. Een signaal is opgebouwd uit een serie pulserende berichten. De publisher stuurt dus niet een afzonderlijk bericht, maar herhaalt het bericht continu. Daarbij niet uitgesloten dat de berichten onderweg gemodificeerd kunnen worden tot andersoortige of nieuwe berichten vergelijkbaar met *complex event processing*. Vervolgens wordt het signaal geïnterpreteerd door de subscriber waarbij aan de sterkte van het signaal ook betekenis gegeven wordt. Een dergelijk concept heeft bovendien een aantrekkelijk bijkomend voordeel. Een afzonderlijk bericht heeft voor een buitenstaander geen betekenis. Slechts het signaal heeft betekenis. Dit is erg interessant met het oog op beveiliging. Een subscriber wordt geacht om het signaal te destilleren uit de serie berichten die binnenstroomt.

EDA is een eerste opstap naar de toekomstige informatiedynamica. Er zit echter een verschil in toepassingsgebied. Informatiedynamica bevindt zich meer op het vlak van internetmarketing en bijvoorbeeld informatieverspreiding bij rampen. In het kader hieronder wordt een toepassingsgebied geschetst waarbij informatiedynamica in volle glorie prijkt.

Toepassingsgebied: **Een neus voor fabrieksgassen**

Een chemische fabriek stoot een hoeveelheid schadelijke gassen uit. De dichtheid van de schadelijke gassen is vastgesteld op een wettelijk maximum. Het systeem wat de uitstoot meet stuurt pulserend signalen de ether in. Een signaal representeert hierbij het schadelijke gas. Vanwege een technisch gebrek blijkt de fabriek het toelaatbare maximum te overschrijden. De sterkte van het signaal neemt toe. Vervolgens zijn er context-aware-utilities die deze signalen op kunnen pikken (*subscribers*) en afhankelijk van de dichtheid van het signaal besluiten te alarmeren. Eigenlijk bouw je een kunstmatig reukorgaan in, bijvoorbeeld in een mobiele telefoon. Voorwaarde is dat de ontvanger kennis moet hebben van het signaal om het te kunnen interpreteren. Je moet weten hoe een appeltaart ruikt om hem te kunnen herkennen.

Om dergelijke toepassingen te kunnen ontwerpen en beheersen is het noodzakelijk de te bouwen systemen vooraf te simuleren. Een discipline die binnen de chemische procesindustrie volstrekt normaal is [ref_11]. Een simulatie betreft een schaalmodel van de werkelijkheid. Door simulaties toe te passen wordt inzichtelijk welke eveneffecten onderkend worden voor het toekomstige systeem en hoe hierop te anticiperen. Welke resources noodzakelijk zijn voor de gekozen schaalgrootte. Kortom, de effectiviteit van het systeem kan voorspeld worden. Zo wordt in de procesindustrie middels simulatietechnieken onderzocht hoe een systeem het beste opgestart of stilgelegd moet worden.

Dat simulatie overigens geen overbodige luxe is binnen de IT-sector wordt geïllustreerd in het volgende praktijkvoorbeeld.

praktijkvoorbeeld: De SOA big-bang

Een grote dienstverlener binnen de rijksoverheid heeft onlangs een SOA-integratie uitgevoerd. De architectuur was volledig uitgedestilleerd en volgens de beste normen afgeleid van een project startarchitectuur. Echter net voor deployment van het systeem kwam aan het licht dat de conversie en initiele vulling van het systeem behoorlijk roet in het eten heeft gegooid. Om de SOA te vullen door het publish/subscribe-mechanisme bleek dat de doorlooptijd meer dan 2 maanden ging duren. Architecten hadden niet voorzien hoe het systeem aan de gang gebracht moest worden. De bedoeling was dat gedurende een frozen period het systeem gevuld ging worden met realtime data. De onvoorziene lange doorlooptijd had ondervangen kunnen worden door het uitvoeren van een uitgebreide systeemsimulaties vooraf.

Maar hoe ziet informatiedynamica er uit voor een computersysteem? Wat zijn kenmerken van een informatiedynamisch systeem (IDS). Is het mogelijk een beeld te scheppen welke technieken noodzakelijk zijn om informatiedynamica te beschrijven? En met het oog op het chaotische karakter, is het mogelijk om dergelijke systemen te controleren en beheersen?

Een IDS is in rudimentaire vorm gelijk aan een ESB. In beide gevallen gaat het om publishers die informatie versturen aan subscribers die informatie ontvangen. Het grote verschil zijn de krachten die spelen binnen een IDS. Die verschillen nogal ten opzichte van een ESB. Een eerste aanzet om een IDS te regelen waar diverse krachten een rol spelen is door te bepalen waar ik aan kan komen en vooral waar ik vanaf moet blijven. Het is onmogelijk om een IDS regelen zonder goede afspraken te maken.

Als iedereen maar wat doet wordt het zeker een chaos. Welke vrijheden kun je je permitteren als deelnemer van een IDS?

Om hiertoe inzicht te verkrijgen is het goed om specifiek in te zoomen op de chemische procesindustrie. Hiertoe wordt een IDS voorgesteld door een metafoer, een chemische reactor. In bijlage 1 wordt technisch beschreven hoe vrijheden worden toegekend aan krachten die spelen in een reactor. Een IDS kenmerkt zich net als een reactor door een groot scala aan diverse parameters die in toom gehouden moeten worden. Door dergelijke parameters constant te houden blijven er uiteindelijk een paar over waarvoor meer vrijheden gelden. Maar de vrijheden zijn beperkt tot het stabiliseren van processen die binnen de IDS spelen, waarmee de IDS geregeld kan worden.

In bijlage 2 wordt in analogie met de reactor uit bijlage 1 besproken hoe dergelijke vrijheden toegekend kunnen worden voor een IDS.

Vooralsnog leven we in het SOA-tijdperk. EDA komt net om de hoek kijken als voorbode voor informatiedynamica. Vandaag de dag zijn legitieme vragen:

- Kan ik een scenario bedenken hoe ik *publishers* en *subscribers* het beste kan koppelen bij het ontwikkelen van een ESB? Moet dit in een specifieke volgorde? Bijvoorbeeld zwaar-publicerende publishers eerst koppelen of juist eerst de kleintjes?
- Kan ik een scenario bedenken waarbij ik (complexe) events gecontroleerd aan het systeem doseer zodanig dat het systeem niet meteen platgelegd wordt als gevolg van een stortvloed aan events?
- Hoe moet ik eigenlijk mijn ESB gaan vullen met valide data middels *publish-subscribe*? Hoe lang wordt de doorlooptijd voordat een representatieve hoeveelheid data in het systeem zit? Kan ik dit voorspellen?
- Kan ik een systeem waar events andere events kunnen genereren doorrekenen en wat betekent dit voor mijn totale systeembelasting? Hoeveel resources denk ik nodig te hebben?
- Kan ik beschikbare bandbreedte (resources) gereguleerd ter beschikking stellen aan goedbetalende klanten? Kan ik dit "regelen" middels een meet-en-regelproces?

Dergelijke vragen zijn een voorbode van de roep tot beheersen, in toom houden, controleren. Daarom is het helemaal niet zo onlogisch vrijheden als een groot goed te beschouwen voor computersystemen. In een periode waarbij desktop-applicaties nog de boventoon voerden waren noodzaak tot beheren en controleren ook al aan de orde. Al was het maar dat we bovenmatig ons best gedaan hebben om data en dataverkeer zo grondig mogelijk te beheersen. Transacties zijn aan de orde van de dag. Deze noodzaak was er toen en is er nu en in de toekomst ook. Webtechnologie heeft behoorlijk veel deuren opgezet. IDS zal nog wel een aantal jaren op zich laten wachten. Dat is een goede motivatie om initiatieven zoals hier besproken te initiëren en deelgenoot te maken van het werkpakket voor architecten.

Conclusie

Dit artikel beschrijft hoe informatiedynamica in de toekomst door architecten aangepakt kan worden. Uiteindelijk is meet- en regeltechniek slechts een aspect van het totale arsenaal aan engineeringactiviteiten die de procesindustrie te bieden heeft. Daarvoor is het echter wel noodzakelijk dat systemen zich meer gaan ontwikkelen in de richting van informatiedynamica. Het huidige SOA-paradigma staat hier nog ver vanaf. Maar dat wil niet zeggen dat dergelijke technieken niet kunnen worden ontwikkeld. Voorlopig kunnen architecten nog vooruit met de frameworks zoals TOGAF of DYA die hen nu ter beschikking staan. Immers SOA zoals we dat kennen is goed uitvoerbaar zonder dat chaos direct een probleem gaan vormen. Toch is het belangrijk om nu al met engineering aan de slag te gaan. Pakketleveranciers onderkennen dit door de SOA-governance beter te faciliteren en in te kleden. Maar hierbij gaat de IT-industrie voorbij aan lessons-learned uit de procesindustrie. En dat is jammer. Simulaties, meet- en regeltechniek en andere engineeringactiviteiten kunnen de uitdaging vormen voor het werkpakket van architecten. De complexiteit zal de komende jaren een grote vlucht maken. Door hier nu al over na te denken wordt een basis gelegd voor de toekomst.

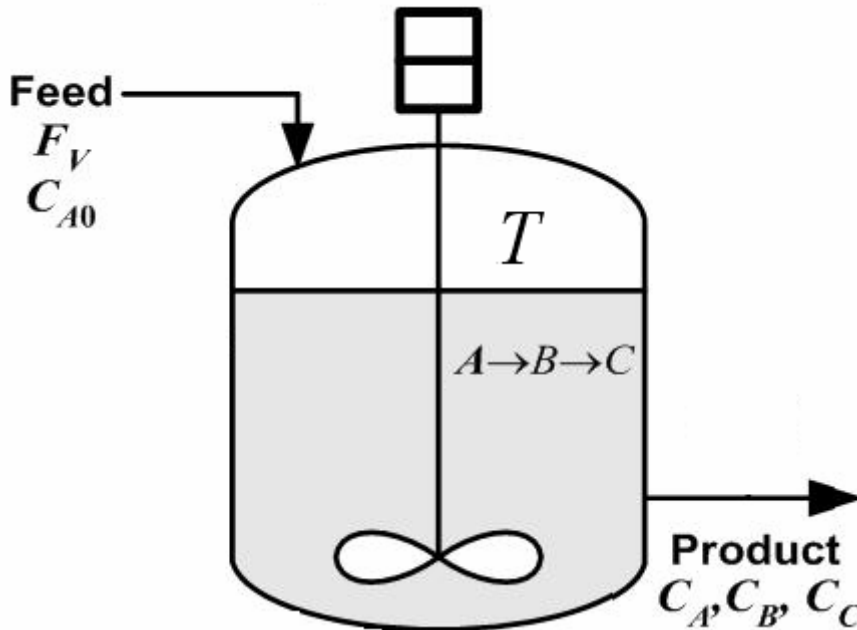
Kijk, het is niet zo dat de procesindustrie ons alle antwoorden gaan bieden. Het gaat hier niet om een wonderbaarlijk eureka. Een techniek die elke andere techniek overbodig maakt. Ik nodig architecten uit om hun horizon te verleggen. De tijd is rijp dat architecten de theoretische beschouwingen over dit onderwerp onder de arm nemen en hier praktijkoplossingen voor gaan aandragen. Ik nodig de TOGAF-community en andere architect-groeperingen uit om openingen te introduceren. Hiervoor is alignment met het beta-collectief onontbeerlijk. Architecten kunnen te rade gaan bij groeperingen waarvoor exacte wetenschappen gewoengoed zijn. Architecten kunnen aanschuiven op universiteiten en hogescholen. Ze kunnen synergie zoeken bij de faculteiten natuurwetenschappen zoals chemie en natuurkunde. Het is nieuw. Het is onbekend, maar het is niet ondenkbaar.

BIJLAGE 1 CISTR als metafoor voor IDS

Een computersysteem waar informatiedynamica aan ten grondslag ligt is goed te vergelijken met een chemische reactor. Onderstaande tabel beschrijft de overeenkomsten tussen de metafoor, de reactor, en computersysteem.

Chemische reactor	Informatiedynamisch computersysteem (IDS)
Boltzmann entropie	Shannon entropie
moleculaire deeltjes	events / agents
chemische reactie	complex-event processing
invoer en uitvoerstromen	informatieconvectie

In de procesindustrie wordt een CISTR (Continuous Ideal Stirred Tank Reactor) gebruikt om reactoren te modelleren [ref_07].



Een CISTR is een reactor welke gevoed wordt door een mengsel met samenstelling, C_A . In de reactor vindt een chemische omzetting plaats waarbij de samenstelling van het mengsel verandert. C_A wordt omgezet in C_B en C_B wordt weer omgezet in C_C . Verder wordt er goed geroerd waardoor aangenomen mag worden dat overal in de kookpan condities als temperatuur en samenstelling hetzelfde zijn. Uiteindelijk verlaat het mengsel de kookpan.

In stationaire toestand verandert er niets. Maar wat gebeurt er indien de invoerkraan verder wordt opgezet, de temperatuur zou wijzigen of het reactiemechanisme verandert? Dergelijke verstoringen maken het systeem instabiel en oncontroleerbaar.

De eerste stap die noodzakelijk is om instabiliteit de kop in te drukken is door het aantal vrijheidsgraden te bepalen.

Het aantal vrijheidsgraden (DOF = Degrees Of Freedom) bepaalt welke parameters gewijzigd kunnen worden minus het aantal vergelijkingen die de parameters moeten verklaren.

$$\text{DOF} = (\text{aantal parameters}) - (\text{aantal vergelijkingen})$$

Nu zijn er 3 mogelijkheden:

$$\text{DOF} = 0$$

Hierbij hebben we een systeem waarbij het totaal aantal parameters gelijk is aan het aantal vergelijkingen. In dit geval is er sprake van een *exactly specified process*.

$$\text{DOF} > 0$$

We hebben meer variabelen dan vergelijkingen. Daarmee is het systeem niet stabiel. Meerdere variabelen zijn in staat het systeem te ontwrichten zonder dat we dit kunnen herstellen. In dit geval is sprake van een *underspecified process*.

$$\text{DOF} < 0$$

Blijkbaar zijn er meer vergelijkingen dan het aantal parameters. Hierdoor zijn de parameters niet eenduidig te voorzien van een geldige waarde. In dit geval is sprake van een *overspecified process*.

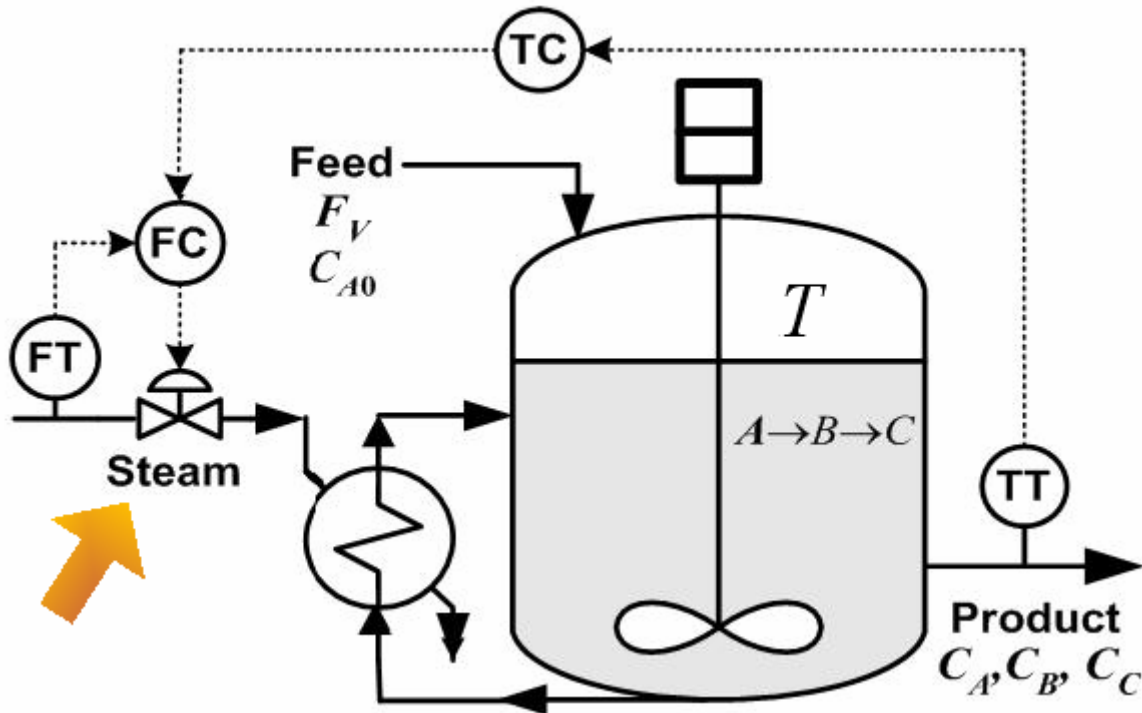
In de metafoor geldt:

Parameter	Vergelijking
C_A , C_B en C_C	Wiskundige formule die een chemische omzetting beschrijft. Dit levert 3 vergelijkingen op
F_V en C_{A0}	Aanname: Deze parameters zijn constant. Levert 2 vergelijkingen op. $F_V = x$, en $C_{A0} = y$ met $x, y \in \mathbb{N}$
T	Geen vergelijking

Het totaal aantal vrijheidsgraden in dit voorbeeld: $DOF = 6 - 5 = 1$

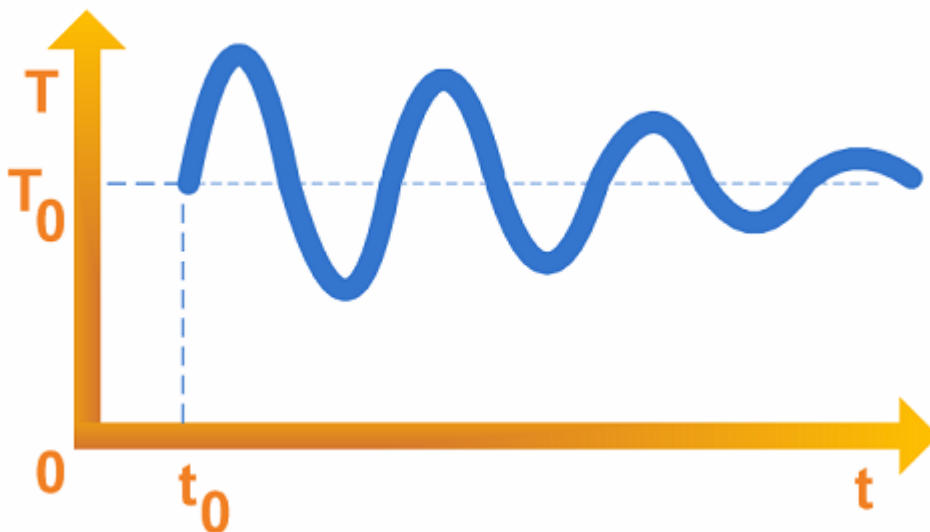
Omdat alle overige parameters zijn verklaard wordt temperatuur T gebruikt om het proces te regelen.

Hiertoe wordt een thermometer TT geplaatst in de reactor. Vervolgens wordt een warmtewisselaar aangebracht door het plaatsen van een stoomklep. Indien het systeem te koud is opent de stoomklep zich en verwarmt de reactor.



Door het aanbrengen van regelkleppen, meetinstrumenten is de reactor volledig bestuurbaar. Elke verstoring in de reactor kan opgevangen worden door instellingen van de regelkleppen te wijzigen.

In de praktijk verloopt een dergelijke regeling middels demping. Net zoals het bijregelen van warm/koud tijdens het douchen. Eerst te koud, iets meer warm erbij, dan weer te warm enzovoort. Onderstaande figuur toont een weergave van dit regelproces. De procesregeling zorgt er voor dat een verstoring van de temperatuur automatisch weer wordt gecorrigeerd.



BIJLAGE 2

Afleiden vrijheden voor een IDS

In analogie met de chemische reactor wordt eerst de DOF afgeleid. Hiervoor zijn wel enkele definities noodzakelijk.

Definitie: Event Publish Velocity (EPV)

Deze grootheid bepaalt hoeveel events er per seconde toegediend kunnen worden aan het systeem.

Definitie: Event-Process-Factor (EPF)

Deze factor bepaalt hoe snel een event "presteert" in het systeem. EPF is een dimensieloze grootheid die tussen 0 en 1 ligt. Bij EPF = 1 wordt elk gepubliceerd event terstonds opgepikt door alle subscribers. Bij EPF = 0 duurt het tot in de eeuwigheid voordat een subscriber het event binnenkrijgt.

Een algoritme om EPF voor eventtype A te bepalen:

$$EPF_A = \text{Aantal-events-A per seconde} / \text{Totaal-aantal-events per seconde}$$

ofwel

$$EPF_A = EPV_A / EPV_{\text{Totaal}}$$

Onderstaand kader illustreert het opzetten van een meet- en regelproces voor een ESB. De eerste stap is dat de DOF bepaald moet worden.

Voorbeeld: Meet- en regelproces voor een ESB

Gegeven:

Stel een ESB met twee klanten A en X. Klant A publiceert events van het type A en klant X publiceert events van het type X. Klant A is een goedbetalende klant en klant X een slechtbetalende. Voor EPF_A is met klant A afgesproken dat deze gegarandeerd 83% dient te zijn.

Vraag: Is de ESB zo in te regelen dat klant A gegarandeerd 2 keer meer events kan publiceren dan klant X?

Antwoord:

Hiertoe zet ik een meet-en-regel op het systeem die het aantal beschikbare resources in de gaten houdt en indien de doorvoer van A in het gedrang komt, dan kan de doorvoer van klant-X afgekneld worden ten gunste van klant-A.

Afleiding DOF:

	Aantal	Waarde
Parameter	3	EPV_A, EPV_X, EPF_A
Vergelijkingen	2	$EPV_A = 2 * EPV_X$ $EPF_A = 0,83$
DOF		$3 - 2 = 1$

Er is 1 vrijheidsgraad onderkend. Hiervoor stel ik de toevoer van klant-A constant door met de toevoer van klant-X meer open of dicht te draaien op grond van meetresultaten. Dit betekent dat een meetinstrument op EPF_A gezet moet worden en een klep op de toevoer van klant-X, EPV_X

De beschreven procesregeling heeft tot doel het constant houden van procesparameters. Indien een toevoer van berichten een destabilisering van het systeem tot resultaat heeft dan biedt procesregeling op basis van DOF en demping uitkomst.

Referenties

[01]

Lu Yan, Yan Zhang, Laurence T. Yang, Huansheng Ning (2008), *The Internet of Things*, Auerbach Publications

[02]

Seng Loke (2007), *Context-Aware Pervasive Systems*, Auerbach Publications

[03]

Pet Helland (2008), *The irresistible forces meet the movable objects*, Microsoft

[04]

C.E. Shannon (1948), *A Mathematical Theory of Communication*, The Bell System Technical Journal vol. 27

[05]

J.D. Fast (1970), *Entropy*, Palgrave Macmillan, revised edition

[06]

G. Hegeman (2008), *Influence infrastructure on overtaking (thesis)*, Universiteit Delft

[07]

G. Stephanopoulos (1983), *Chemical Process Control: An Introduction to Theory and Practice*, Prentice Hall

[08]

S.W. Angrist, L.G. Hepler (1967), *Order and Chaos*, Basic Books Inc.

[09]

I. Csiszár, G.O.H. Katona, G. Tardos (2007), *Entropy, Search, Complexity*, Springer

[10]

G. Mühl, L. Fiege, P. Pietzuch (2006), *Distributed Event-Based Systems*, Springer

[11]

S.T. Karris (2008), *Introduction to Simulink with Engineering Applications*, Orchard Publications

[12]

B. M. Michelson (2006), *Event-Driven Architecture Overview*, Patricia Seybold Group

[13]

N.T. Nguyen, A. Grzech, R.J. Howlett, L.C. Jain (2007), *Agent and Multi-Agent Systems: Technologies and Applications*, Springer