

Soft Skills zijn de harde kern van een goede oplossing onder IT Architectuur

juni 2007

Lotje Meijknecht
IT Architect
lotje.meijknecht@nl.ibm.com
IBM Nederland B.V.

Samenvatting

Veel oplossingen in de IT sluiten slecht aan bij de vraag. Welke skills zijn noodzakelijk voor een goede oplossing? Het antwoord luidt dat “Soft Skills de harde kern zijn van een goede oplossing onder IT Architectuur”. Deze noodzakelijke Soft Skills worden nader verkend.

Allereerst wordt de IT Architect gepositioneerd als ontwerper. Daarna worden aan de hand van voorbeelden vijf Soft Skills onderzocht. Dat zijn communicatie, analyse, creativiteit, empathie en het omgaan met onzekerheid. Communicatie en analyse blijken basis Soft Skills te zijn, die het mogelijk maken om creativiteit, empathie en omgaan met onzekerheid te kunnen gebruiken. Creativiteit blijkt het meest nodig te zijn in het begin van een opdracht, empathie tijdens het creëren van een oplossing en omgaan met onzekerheid in de laatste fase wanneer de ontworpen oplossing wordt gerealiseerd.

Omgaan met onzekerheid is waarschijnlijk de meest onbeminde Soft Skill. Men creëert liever een beeld van zekerheid. Voor de dagelijkse praktijk van de IT Architect betekent dit vooral dat deze zorg moet dragen voor de acceptatie van een bepaalde mate van onzekerheid bij zichzelf en bij de opdrachtgever.

Introductie

Er worden oplossingen gemaakt in de IT branche, maar oplossingen waarvan eigenlijk? Veel oplossingen blijken nog steeds beroerd aan te sluiten bij de bedoeling van de vrager of bij zijn of haar bedrijf. Er wordt ontworpen, overlegd en in tools geïnvesteerd, allemaal noodzakelijk, maar het levert niet zomaar het gewenste resultaat op: de werkelijke 'value add' waarvoor een opdrachtgever bereid is te betalen.

Het antwoord op de vraag wat een ICT oplossing nodig heeft om wel goed aan te sluiten bij de probleemstelling van een opdrachtgever ligt mijns inziens in het verlengde van de stelling dat **'Soft Skills de harde kern zijn van een goede oplossing onder IT Architectuur'**. Op deze stelling wil ik in dit stuk nader in gaan. Andere non-Soft Skills hebben hun eigen bestaansrecht maar worden in dit stuk buiten beschouwing gelaten.

Ik geloof zeker dat zo langzamerhand iedereen het nut van Soft Skills voor een IT Architect zal erkennen. Toch hebben veel mensen een te beperkte opvatting over wat Soft Skills zijn. Men denkt vooral aan communicatieve vaardigheden. Erg handig om te hebben zodat je soepel je doel kunt bereiken.

Maar Soft Skills omvatten meer dan communicatie. Het is niet zo dat je zonder Soft Skills hetzelfde doel bereikt, zij het langs een iets moeizamer pad. Een oplossing onder IT Architectuur die het zonder Soft Skills heeft moeten stellen kan alleen door puur toeval aansluiten bij de wensen van de opdrachtgever. Een oplossing is nog altijd een oplossing voor mensen en deze gaat niet werken als ze tijdens het ontwerp en de uitvoering niet voortdurend is ingepast in de menselijke situatie. Om dit te bereiken heb je Soft Skills nodig. Daarom vormen Soft Skills de kern van een oplossing onder IT Architectuur.

En Soft Skills vormen ook de harde kern van een oplossing omdat ze betrekking hebben op de moeilijkste aspecten van die oplossing. Al is iets erg ingewikkeld, met voldoende beta-denkers komt men er op het vlak van IT techniek wel uit. Moeilijk wordt het als verschillende belangen door elkaar gaan lopen, als men niet precies weet welke kant het op moet, als er keuzes gemaakt moeten worden op basis van onvolledige gegevens en kennis. Hier heb je Soft Skills voor nodig.

Wat zijn Soft Skills?

Het eerste woord dat naar boven komt als men spreekt over Soft Skills is communicatie. Hierbij horen spreek- en schrijfvaardigheid, luisteren, samenvatten, doorvragen. Hiervoor kun je trainingen volgen, natuurlijk moet je een boodschap over weten te brengen. Daarmee zijn Soft Skills heel keurig en veilig gepositioneerd. Maar dit is nog niet een kwart van het geheel.

Het gaat ook om meer ongreepbare kwaliteiten die niet zo keurig te definiëren zijn. Er is een deel dat met creativiteit te maken heeft. Elke nieuwe situatie heeft zijn eigenaardigheden en die moet je kunnen onderkennen; ook als de bestaande kennis en methoden niet direct uitkomst bieden; ook als terugvallen op gebaande paden eigenlijk makkelijker is. Het gaat er om of je in staat bent de gedachtegang, eventueel het probleem, van een ander te begrijpen en te doorgronden. Dat is sensitiviteit of empathie, het aanvoelen van *concerns* van de ander.

Dan nog iets, ben je in staat te zweven op onzekerheid? Kun je doorwerken terwijl je toch niet alles weet? De materie is vaak te complex om deze tot in alle details te beheersen. Je hebt geen zekerheid over de haalbaarheid en het uiteindelijke succes van de oplossing. Hoe ga je om met risico's?

Samenvattend komen we vijf elementen van Soft Skills tegen:

- Communicatie
- Doorgronden van problemen (specifieke plek, specifiek moment)
- Creativiteit
- Sensitiviteit, gevoel voor de situatie van een ander, empathie
- Omgaan met onzekerheden en risico's

Door Roel Wieringa worden ook deze kenmerken genoemd in zijn artikel 'De competenties van de ict-architect'¹, als onderdeel van de persoonlijkheidskenmerken die nodig zijn voor de IT Architect. De persoonlijkheidskenmerken vallen in zijn 'competentie-ijsberg' onder het oppervlak, en dat is het deel dat zeer sterk bepalend is voor het gebruik van overige kennis en vaardigheden. De definitie sluit daarmee goed aan bij die van Soft Skills in dit artikel. Er wordt nog een aantal persoonlijkheidskenmerken genoemd zoals besluitvaardigheid, zelfstandigheid, en de wens jezelf te blijven ontwikkelen. Hoewel ik niet specifiek op deze Soft Skills in ga, zouden ook deze aansluiten bij dit verhaal.

Wat is het geval?

Zie IT Architectuur als één van de vele vormen van techniek of technologie. Wat is techniek? Het is in eerste instantie het idee dat het mogelijk moet zijn om in een bestaande situatie verbetering aan te brengen. Je hebt problemen gezien of gevoeld, je ziet kansen voor verbeteringen. Je hebt de overtuiging dat wat je bedacht hebt ook een verbetering zal betekenen, ook als het geen perfecte oplossing is. Hiermee gaat de gedachtegang, ofwel het ontwerp, ofwel het vermoeden van toekomstig gebruik, vooraf aan het product.

Ontwerper versus opdrachtgever

In een romantische kijk op de geniale uitvinder begint de uitvinder met een verbetering van haar of zijn eigen situatie. Uitvinders ontwerpen en knutselen net zo lang totdat iets precies voldoet. In de huidige IT branche zijn probleemeigenaar en ontwerper niet dezelfde persoon. Ontwerpen blijft gebaseerd op een interpretatie van de 'te verbeteren situatie' van de probleemeigenaar. Hierin schuilt een existentiële onzekerheid voor iedere ontwerper. Hoe begrijp je de probleemeigenaar en houd je respect voor hem?

In de IT branche is de IT Architect vaak de ontwerper. De probleemeigenaar in de IT branche is de opdrachtgever² van een toekomstig IT systeem. Er is dus per definitie een kloof tussen IT Architect en opdrachtgever. Preciezer nog, er is niet alleen een kloof tussen de ene mens en de andere mens, een overbrugging moet ook nog eens lopen via een technische oplossing.

De mythe van de schone lei

Er is een zeer natuurlijke neiging bij de ontwerper (en wellicht ook wel bij de opdrachtgever) om van de oplossing uit te gaan, van de technische mogelijkheden. In het technische domein heerst helderheid, voorspelbaarheid, en enige vertrouwdheid. Maar de omgeving (organisatie, samenleving, publieke opinie, natuur) is zeker niet beheersbaar. Als een ontwerper er direct van uit moet gaan dat de omgeving waaruit de vraagstelling naar voren is gekomen ook de omgeving moet zijn waarin een oplossing zich zal moeten settelen, wordt een probleem vele malen complexer.

Toch is het betoog van dit verhaal dat een ontwerper dit wel moet doen (eventueel gefaseerd zoals in de techniek gebruikelijk is). Een oplossing gaat niet werken als een ontwerper geheel binnen het technische kader blijft. Binnen het technische kader kom je slechts tot een puur technische oplossing, die niet hoeft te passen in de omgeving van het probleem en de opdrachtgever. Stephen Toulmin³ noemt dit 'de mythe van de schone lei', alsof het inderdaad mogelijk is te werken aan een probleem zonder te denken aan probleemgeschiedenis en de omgeving waarbinnen probleem en oplossing hun plek moeten vinden.

Het artikel 'De opgave van de IT Architect'⁴ belicht ook beide aspecten: 'De praktijk wijst uit dat het vaak moeilijk gevonden wordt om aan te sluiten bij de gedachtewereld van de verschillende belanghebbenden en dat daarom aan de techniek en technische kennis een te hoge prioriteit wordt gegeven. Hiermee wordt voorbijgegaan aan het feit dat naast vakkennis en ervaring juist de persoonlijke vaardigheden het verschil uitmaken tussen oplossingen die de toets van haalbaarheid, maakbaarheid en bruikbaarheid (aspecten van de menselijke maat) kunnen doorstaan en oplossingen die gedoemd zijn te mislukken.'

¹ Wieringa, R.J.; 'De competenties van de IT Architect' in: *Informatie 4* (2006)

<http://www.informatie.nl/artikelen/2006/04/competentiesVanDelctarchitect.html>

² In de rest van dit stuk zal ik ten behoeve van de leesbaarheid de term opdrachtgever gebruiken en daar rollen klant, eindgebruiker, opdrachtgever, en alle andere varianten mee samenvatten.

³ Toulmin, S.; *Kosmopolis*, Verborgene agenda van de moderne tijd; Kok Agora, Kampen 1990; pagina 239

⁴ Muller, M.; Neecke, H.; Hammer, D.; Perdeck, M.; 'De opgave van de IT-architect met betrekking tot de "menselijke maat"'; voor studiegroep [Menselijke Maat in IT](#); 2004

Analyse van de achtergronden

Er zijn twee kloven te overbruggen. De kloof tussen opdrachtgever en ontwerper en de kloof tussen oplossing en zijn omgeving. Wat kan de IT Architect hier van zijn kant aan doen? Ik wil hier de bouwstenen van Soft Skills nader bepalen zodat we deze materie zo helder mogelijk kunnen structureren.

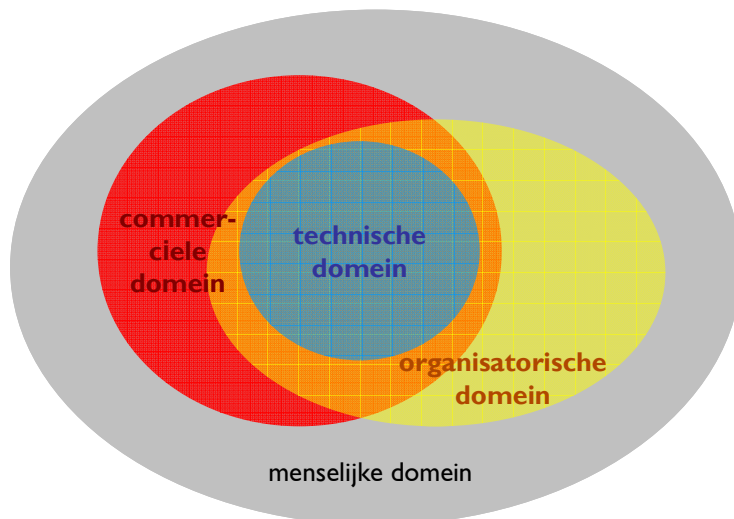
Gelaagdheid in een praktijksituatie

In een praktijksituatie komt men meestal vier domeinen tegen in de probleemstelling van de opdrachtgever:

1. Eerst bestaat het probleem alleen in het technische domein. Op technisch gebied speelt de selectie van interfaces, migratie van ICT componenten, het uitstippelen van migratiepaden, etc.
2. Dan komt het zakelijke belang van de opdrachtgever aan bod. Zo kan bijvoorbeeld de timing van de introductie van nieuwe producten invloed uitoefenen op keuzes die tot nu toe alleen op technische gronden genomen werden.
3. Vervolgens blijkt vaak dat er afdelingen met parallelle visies te zijn. Men komt dan in een organisatorisch domein. Hoe gaan verschillende entiteiten binnen het bedrijf van de opdrachtgever met elkaar om en wie wil welk probleem oplossen?
4. Het vierde domein is de overall aanwezige menselijke factor, hoe kunnen de specifieke mensen die betrokken zijn bij een opdracht met elkaar overweg?

Het is niet zo dat de beschreven domeinen allemaal door één en dezelfde persoon moeten worden behandeld. De verantwoordelijkheid van de IT Architect ligt in primaire zin in het technische domein. Maar omdat de oplossing moet aansluiten bij de situatie van de opdrachtgever, zullen de drie niet-technische domeinen van het probleem de technische oplossing beïnvloeden.

De volgorde waarin de domeinen of lagen aan de orde domein in een opdracht kan telkens anders liggen. Mijn ervaring is dat de technische laag vaak de meeste focus krijgt, ook al omdat het meestal het beginpunt van een opdracht is. Ik wil de gelaagdheid van de probleemdomeinen en hun onderlinge samenhang op de volgende manier schetsen:



Figuur 1: De verschillende domeinen van een probleem en hun onderlinge samenhang. De kern wordt gevormd door het technische domein. De omliggende domeinen geven andere aspecten van hetzelfde probleem weer.

Benodigde Soft Skills

Welke kwalificaties heeft een IT Architect nodig bij de interpretatie van een probleem, tijdens het ontwerpen en tijdens bouwen van een architecturale oplossing? Hoe verhoud je je professioneel ten opzichte van een probleem? Kun je in de huid van een ander kruipen, de situatie aanvoelen en relevante verbeteringen voorstellen? Kwalificaties voor een IT Architect op het gebied van vakkennis en ervaring wil ik in dit verhaal niet bespreken, mij gaat het om het deelgebied van Soft Skills.

Zelfs met alle technieken en methodes die me ter beschikking staan en met alle kennis die ik heb, kan ik als IT Architect niet zomaar met een passende oplossing komen. Eerst moet ik begrijpen waar het om gaat in de domeinen in een specifieke situatie en dan kan ik daarbij een passende oplossing ontwerpen.

Aansluiten bij de specifieke situatie vraagt om het verkennen en begrijpen van die situatie. Het is een probleemsituatie in een menselijke omgeving, mensen geven mij de opdracht om iets te verbeteren. Dus moet ik de normale gereedschappen van die omgeving gebruiken. En dat zijn Soft Skills. Met deze Soft Skills kan ik de grenzen tussen het technische domein en de andere domeinen, als geschetst in

, overschrijden en van een puur technische, *stand-alone* oplossing iets maken dat in zijn omgeving past en thuis hoort. Daarmee vormen Soft Skills het fundament van een oplossing die is ingepast in een specifieke organisatie.

Een opdracht voor een IT Architect kent verschillende fases, die ik 'Inzicht verschaffen', 'Oplossing bieden', en 'Plaatsing in de omgeving' wil noemen. Welke Soft Skills zijn er in verschillende fases van een opdracht van een IT Architect nodig? Welke Soft Skills zijn er nodig om vanuit het technische domein inzicht te krijgen in de andere drie domeinen van een probleem? Deze vier domeinen die zich voordoen zijn uitgezet tegen de drie fases. Dit geeft de volgende tabel:

Domein, zie	Technisch	Zakelijk/Commercieel	Organisatorisch	Menselijk
Fase				
Inzicht verschaffen				
Oplossing bieden				
Plaatsing in de omgeving				

Tabel 1: De domeinen van een probleem uitgezet tegen de fases van een opdracht voor een IT Architect – nog niet ingevuld met Soft Skills.

Binnen Soft Skills zijn communicatie, begrip, creativiteit, sensitiviteit en omgaan met professionele onzekerheid de sleutelbegrippen. Elk vakje is uiteindelijk in te vullen met een van Soft Skills. Aan het eind van mijn betoog wil ik op deze matrix terugkomen, en kijken hoe de ingevulde variant zich verhoudt tot de werkelijkheid.

1) Inzicht verschaffen

U hebt net een lange inleiding gelezen. Deze bevat een theoretische grondslag voor mijn betoog dat het aansluiten van IT oplossingen bij de vraag van de opdrachtgever niet zonder Soft Skills kan. In de komende drie hoofdstukken wil ik hier meer invulling aan geven door drie fases die een opdracht van een IT Architect gemiddeld heeft, nader te belichten. Dit eerste hoofdstuk gaat over het 'Inzicht verschaffen in complexe problemen'. Daarna volgen '2) Oplossingen bieden', en '3) Plaatsing in de omgeving'.

Het begrijpen van een complex probleem vraagt om een ontwerper

Eenvoudige problemen kunnen direct helder op papier gezet worden en kennen een *straight forward* oplossing. Maar niet alle problemen zijn eenvoudig; dit verhaal richt zich op complexe IT problemen. Complexiteit is aan de orde wanneer veel factoren van invloed zijn op het probleem, wanneer er onbeheersbare kanten aan een probleem zitten, of wanneer inzicht in de onderliggende problemen ontbreekt. En waar is bij dergelijke problemen het beginpunt van een oplossing te vinden? Daarvoor heb je als probleemeigenaar ruggespraak nodig met een ontwerper. Wat doet een ontwerper?

“Een ontwerper moet een probleem dat een opdrachtgever zelf nog amper heeft kunnen doorgronden, kunnen verwoorden en inzichtelijk maken”

Simpel gezegd, als een opdrachtgever zelf deskundig is en de problemen kan doorgronden, heeft hij niet iemand nodig die vorm geeft aan zijn gedachten en zorgen. De toegevoegde waarde van een ‘probleemoplosser’ (de IT Architect, zullen we in het volgende onderdeel van dit hoofdstuk zien) schuilt niet in eerste instantie in het harde eindproduct maar in zijn vermogen problemen te verhelderen. Werken aan een complex probleem vraagt allereerst om een heldere probleembeschouwing. Inzicht in de bestaande situatie kunnen opbouwen, kijken naar achtergronden, zoeken naar mogelijke oorzaken en onderliggende problemen. Dit alles met sensitiviteit voor zowel technische als niet technische aspecten.

Voorbeelden

Er was eens een opdracht is om een *Enterprise Service Bus* neer te zetten bij een netwerkbeheerder. Als een klant van de netwerkbeheerder iets bestelt en vervolgens vraagt waar haar product blijft, wil men kunnen opzoeken wat de status is. Dit is een commercieel aspect dat past bij de nieuwe klantgerichtheid en dat inderdaad met een *Enterprise Service Bus* kan worden aangepakt. Logica voor *monitoring* moet in dat geval wel aan de bus worden toegevoegd, en een dergelijke bus hoeft niet zonder meer organisatiebreed te zijn. Aan de andere kant: als je toch met *monitoring* begint, wil je dat misschien *end-to-end* over je bedrijfsprocessen invoeren en dat pleit er wel voor om gelijk de organisatiebrede implementatie mogelijk te maken. Wat is nu het eigenlijke probleem? Kan men binnen de organisatie geen berichten van A naar B sturen? Wil men weten waar een bestelling zich bevindt? Of heeft men geen goede contacten en berichtuitwisseling met de andere leveranciers?

Het blijkt om de tweede vraag te gaan: men wil weten waar een bestelling zich bevindt. Hier kom je achter door veel praten en informeren en door op te letten welk probleem het meest door verschillende belanghebbende wordt genoemd. Het gaat hier om Soft Skill communicatie en creativiteit. Daarna maak je een samenvatting met behulp van creativiteit. Iets wat verspreid in de organisatie wel geweten wordt, is zo voor het eerst samengevat en aan iedereen helder gemaakt.

Een ander voorbeeld. De eerste berichten moeten over de organisatiebrede bus gaan lopen. Wat zijn eigenlijk de eerste berichten? Het bedrijf bevindt zich in een transformatieproces rondom de productencatalogus. Nu bestaat er per product een applicatie, straks zal het merendeel gebruik maken van een in te voeren CRM pakket. In het technische domein gaat het erom waar verouderde technologie zit en welk interface voorlopig nog met de bestaande techniek uit de voeten kan. In het commerciële domein gaat het erom welk product als eerste met nieuwe technologie moet worden aangeboden. In het organisatorische domein gaat het erom hoe je een proces als een afgerond geheel kan onderscheiden.

Sluiten we aan bij de bestaande organisatie of bij een nieuwe organisatie? Er bestaat geen organisatiebreed migratieplan voor de korte tot middenlange termijn: van nu tot over 1 of 2 jaar. Er bestaat alleen een visie voor de langere termijn en activiteiten per afdeling. Alleen op basis van een dergelijk migratieplan kunnen bovengenoemde keuzes gemaakt worden. Als ik alleen binnen mijn project dergelijke keuzes moet maken, moet ik onvergelykbare grootheden met elkaar vergelijken.

In gesprek de opdrachtgever, hebben we het over de selectie van specifieke interfaces. Er is bij mijn gesprekspartner niet veel interesse. ‘Dat kun je wel uitzoeken, daar of daar’. De interesse breekt pas door als we het over de concretisering van lange termijnvisie naar migratieplan hebben. Ik laat zien dat dit een vraag is die aan de selectie van interfaces ten grondslag ligt. Het beeld dat ik schets waarin een concreet migratieplan een grote rol speelt, wordt aanvaard. Het kan natuurlijk niet volledig nieuw voor mijn opdrachtgever zijn, maar in deze constellatie wordt het noodzakelijk om een migratieplan te maken.

Refererend aan

op bladzijde 1 blijken het commerciële domein en het organisatorische domein veel invloed op de definitie van het probleem uit te oefenen. Het technische domein gaat zelfs verschuiven. Het was ‘selectie van berichten over een *Enterprise Service Bus*’, en wordt zoiets als ‘*Enterprise Service Bus* berichten die een concreet geplande transformatie van de productapplicaties ondersteunt’. Het hoeft niet eens het laatste beeld te zijn, maar we hebben beiden het gevoel dat we op de goede weg zitten. Doorslaggevend is de Soft Skill creativiteit geweest, die het mogelijk maakte om alle onderdelen van het probleem op deze manier te rangschikken zodat het beginpunt van een oplossing gevonden was.

Vormgeven aan de gedachten van een ander

De 'probleemoplosser' moet een beeld gaan vormen van wat er aan de hand is, terwijl hij zorgt dat de verschillende verhalen in de juiste relatie tot elkaar komen te staan. Verschillende belangen moeten worden afgewogen en een plek krijgen. Het maken van een beeld dat voor de meeste partijen begrijpelijk en acceptabel is, dat is precies het punt waarom ik deze 'probleemoplosser' een ontwerper wil noemen. Een ontwerper is een vormgever van gedachten of van een probleemanalyse (en vervolgens van een oplossing natuurlijk, dat komt in de volgende paragrafen aan bod), zodanig dat dit ook bij de opdrachtgever overkomt, dat het inzichtelijk èn acceptabel is.

Welke gereedschappen gebuikt deze ontwerper dan? Vooral creativiteit en communicatie hebben hier een rol gespeeld. In tweede instantie ook het afwegen van belangen en het creatieve proces van beeldvorming. Dit zijn allemaal Soft Skills, en zonder deze skills zou de ontwerper niet op een dergelijke manier tot een helder beeld van de bestaande situatie, 'het probleem' gekomen zijn.

Wie is de ontwerper in de IT?

De voorgaande paragraaf sprak over probleem en ontwerper van een probleemdefinitie in algemene zin. Wie vervult de rol van ontwerper binnen de IT industrie? Generaliserend gezegd is dat de IT Architect.

"De IT Architect is verantwoordelijk voor het overall ontwerp, voor, tijdens en na de bouw"

In rolbeschrijvingen is de IT Architect verantwoordelijk voor het inpassen van een oplossing in een bestaande situatie en verantwoordelijk voor het overall ontwerp voor en tijdens en na de bouw. Dit wordt in de IBM rolbeschrijving⁵ tot uitdrukking gebracht onder de term *Leadership*, bijvoorbeeld *'Lead strategy, design and implementation of a solution'*. In het artikel *'Competenties van de ict-architect'*⁶ schrijft Roel Wieringa 'een ict-architect analyseert bedrijfsbehoeften en ontwerpt daarbij een passende ict-oplossing'. En verderop staat dat de ict-architect 'met de opdrachtgever en uitvoerder [moet] kunnen praten'. Dus ook hier is de IT Architect verantwoordelijk voor een passend ontwerp en een passende oplossing.

De IT Architect degene die een 1 tot 4 jaren visie moet neerzetten. Hij moet niet alleen leven met de waan van de dag zoals een project manager dat moet kunnen, en zich ook niet alleen laten leiden door de vijfjaren plannen en een tienjaars horizon. Juist de IT Architect moet die omslag maken van visie naar de dagelijkse praktijk, het concretiseren van droombeelden zonder dat bijvoorbeeld een 'virtuele wereldbibliotheek' wordt platgeslagen tot een simpel zoekalgoritme. Dit vraagt om creativiteit, beeldend vermogen, en gevoel voor de situatie. Dit zijn typisch de kenmerken van een ontwerper.

2) Oplossingen bieden

Er bestaat een menselijke wens om de situatie te verbeteren, om een probleem van zichzelf of van een ander op te lossen. Daarvoor gebruiken we techniek. Maar zelfs als je tot een inzichtelijke probleemstelling bent gekomen, wordt er nog zo geworsteld met de oplossing. Waarom? Het antwoord op een probleemstelling blijft vaak steken in termen van systemen, terwijl het een probleem behandelt dat 'gegroeid' is en een organische structuur heeft. Begrijp me goed, mijn hart gaat echt sneller kloppen bij de intenties van de techniek, maar hoe zorgen we dat dit ook in een oplossing zichtbaar, of voelbaar blijft? Hierover wil ik het in dit hoofdstuk hebben.

⁵ IBM Global, Characteristics of the IT Architect; <http://w3-03.ibm.com/hr/careerplanner/caita008.html> en The IBM IT Architect Profession Guide, version 2.2, March 2007; <http://w3-03.ibm.com/hr/careerplanner/ITArchitectProfessionGuideV22March2007.pdf>

⁶ Wieringa, R.J.; 'De competenties van de IT Architect' in: *Informatie 4 (2006)*; <http://www.informatie.nl/artikelen/2006/04/competentiesVanDelctarchitect.html>

Groeien en maken

Wat is technologie of techniek eigenlijk? Is het het tegenovergestelde van natuur? Op hoofdlijnen zou je kunnen zeggen van wel. Natuur gaat over groeien, en techniek gaat over maken⁷. Het is tijd voor een duik in de geschiedenis.

We gaan terug naar de mens die samenleeft met de natuur. Daar zie je het romantische beeld van de schaapherder. Hij maakt zelf zijn kaas en heeft daarmee een heerlijke lunch ergens onder een boom, vruchten en verse melk erbij en een warm wollen vest voor als het kouder wordt. De herder is gelukkig met zichzelf, in harmonie met de omgeving, brengt aan niemand schade toe, een schattig herderinnetje op de achtergrond... Op dit beeld zijn helaas aanvullingen nodig. De natuur is niet altijd harmonieus, er is schaarste, ziekte, de schapen worden opgegeten door wolven, het vuur van de mensen heeft voeding nodig, natuurrampen doen zich voor, en dan hebben we het nog niet eens over twisten en jaloezieën.

Het ligt in de aard van de mens⁸ om op problemen in zijn omgeving te reageren. Deze schaapherder gaat dingen maken, beginnend met een huis voor beschutting, vuur voor het bakken van brood, als hij in het toekomstige Nederland leeft wil hij vast ook een terp of een dijkje om droog te blijven in de winter, er komen gereedschappen, rechtssystemen, oorlogen, machines, en nog veel meer. Zelfs de natuur wordt dan gezien als ingenieus systeem, waarmee de mechanisering van het wereldbeeld start. Uiteindelijk ontstaat in de 17^{de} eeuw het beeld van de mens als machine. Maar is er eigenlijk iets mis met het oorspronkelijke streven van de mens om zijn eigen situatie in een 'meedogenloze' natuur te verbeteren? Een verbetering is op zich zelf geen hoogmoedig streven, het gaat niet uit van overheersing maar van het dienen van een mens die het zwaar heeft.

“De IT Architect moet de kloof tussen natuurlijk groeien en gemaakte techniek overbruggen”

Uitgaande van het streven om een mens te dienen die het zwaar heeft, ontwikkelen we systemen voor de problemen die een mens tegenkomt. Ook in de ICT branche is dit zo. Problemen zijn ontstaan en gegroeid en bevinden zich vaak in een complexe omgeving. Hier spelen begrippen als voortschrijdend inzicht, aanpassen, gefaseerd migreren, op gevoel beslissen, reageren op nieuwe situaties, 'in dialoog met de omgeving' een rol. De eigenheid van elke situatie is van belang. Net als de rijkdom aan variëteiten en soorten in de natuur en de dynamiek van ontstaan, aanpassen en afsterven. Mensen maken systemen als antwoord op een gegroeid probleem en ten behoeve van de beheersbaarheid van de situatie.

Dit zijn eigenschappen die haaks staan op wat we in de computerwereld willen bereiken: standaardisatie, voorspelbaarheid, service levels, centralisatie, kortom beheersbaarheid. Je kunt zeggen dat de vraag leeft in de gegroeide situatie, het antwoord in de gemaakte situatie.

De IT Architect moet omgaan met de Paradox van de Systemen

De gemaakte situatie bevat nooit in een keer alle natuurlijke nuances en variaties die er in de gegroeide situatie voorkomen. We hebben vaak de neiging om alle nuances van een probleem in een keer op te willen lossen, en om een oplossing te maken die onafhankelijk is van tijd en plaats. Dit is een vorm van volledigheidsdwang of een vlucht in de techniek. Het levert bijzonder ingewikkelde systemen op, ze lijken wel ontstaan in een heel parallel universum. En zo wordt het systeem dat bedoeld was om de situatie beheersbaar te maken en een probleem op te lossen toch onbruikbaar. Het roept onmacht op, en dat terwijl het bedoeld is om de onmacht te verminderen⁹.

⁷ Als je in detail kijkt, is het groeien in de natuur ook een vorm van maken natuurlijk. Maar vergeleken met het menselijke maken is het 'maken' in de natuur een evolutie met langzame aanpassingen en natuurlijke selectie. Het maken van de mensen zou je in vergelijking hiermee een revolutionair proces kunnen noemen. En dat is precies de reden waarom ik verschil tussen groeien en maken wil maken: het langzame ontwikkelen is iets wat ontbreekt bij menselijk maken en daarom moeten we andere middelen zoeken om iets in een omgeving te laten passen.

⁸ Misschien ligt het voornamelijk in de aard van de westerse mens om zijn situatie te willen verbeteren in plaats van er beter mee te leren leven, maar dat is verder geen onderwerp voor deze scriptie.

⁹ Dit is geen betoog tegen herbruikbaarheid van ICT oplossingen of componenten daarvan, het is een betoog voor inpassing van (reeds bestaande) oplossingen in een eigen situatie.

Hoe zit dat op dit moment trouwens met neurale netwerken, zelfhelende systemen, enz.? Is dit werkelijk groeien en op deze wijze een natuurlijke probleem-oplossing, of alleen maar het vergroten van de complexiteit van gebouwde systemen waarin de link met de oorspronkelijke vraag verloren gaat?

Een voorbeeld: Volledigheidsdrang in een metamodel

Tijdens een bepaalde had ik een aantal documenten gemaakt in een standaardvorm zoals ik gewend ben die te gebruiken. Mijn opdrachtgever zag het nut van dergelijke standaarddocumenten in en we raakte in gesprek over methoden, IT Architectuur artefacten, notatiestandaards en metamodellen hiervoor. Ik werd betrokken bij een pilot-project hiervoor. We startten met een handige set standaarddocumenten. Toen gingen we onderscheid maken tussen standaarddocumenten en de onderdelen waaruit deze documenten zijn opgebouwd. Verschillende documenten kunnen over dezelfde onderdelen gaan, zij het vanuit een perspectief. Daarnaast is het soms goed om iets op conceptueel niveau te schrijven, dan weer logisch, etc. En zo kregen we een metamodel voor IT Architectuur met artefacten en onderdelen, met layers, levels en dimensies, onderlinge relaties, views en nog een derde dimensie daaronder en nog een set attributen om geheel volledig te zijn. Iedereen raakte de weg kwijt. Wat was begonnen als een simpel model werd een systeem dat niemand echt kon begrijpen en de oplossing schoot het doel voorbij.

Menselijke maat moet in een architectuurontwerp verklonken liggen

Om het onderscheid tussen groeien en maken in positieve termen te vatten, kun je zeggen dat we natuur en techniek met elkaar in harmonie brengen wanneer we de menselijke, dus de natuurlijke maat als basis voor onze IT systemen nemen. Varianten van deze stelling vond ik terug bij de studiegroep De menselijke maat in IT, met name in artikel 'De opgave van de IT Architect'¹⁰ en in de visie 'De mens heeft al sinds de oorsprong moeite met de menselijke maat'¹¹. Hierin wordt ervoor gepleit om gericht te blijven op de probleemstelling van mensen en niet te vluchten in puur technische oplossingen. De eindstelling luidt: "[...] een belangrijk aspect van het werk van de architect [is] namelijk haar of zijn vermogen om door aandacht voor mensen kwalitatief goede en geaccepteerde oplossingen tot stand te brengen. Hierbij spelen vaardigheden op het gebied van communicatie en inlevingsvermogen een doorslaggevende rol."

Er is in de klassieke architectuur, de bouwkunst, ook langs deze weg gezocht naar ontwerpen voor bruikbare gebouwen. Want ook de bouwkunde kent natuurlijk zijn gebouwen die bijvoorbeeld prachtig zijn om te zien, maar een ramp om in te leven. De oudste geschriften zijn van Vitruvius¹² uit de eerste eeuw voor Christus. Hij stelt de natuur als maatgever voor de architectuur, bijvoorbeeld in maatvoering. In deze traditie stelde Le Corbusier in de 20ste eeuw ook de Modulor¹³ op: een maatvoering voor gebouwen gebaseerd op de gulden snede. Als je gebouwen van Le Corbusier bezoekt, zijn ze inderdaad aangenaam om in te zijn. Je kunt er je weg vinden, het licht is er mooi, de ruimtes zijn afgestemd op het gebruik, en het is aangenaam om naar het gebouw te kijken. Hoe kunnen we dit naar ontwerpen in de huidige IT branche vertalen?

Dit zijn aanknopingspunten voor de Soft Skill die we tijdens het ontwerpen wellicht het meest nodig hebben en dat is empathie. Communicatie is in deze fase belangrijk, je zou kunnen zeggen dat communicatie de Soft Skill is die het hanteren van alle andere Soft Skills mogelijk maakt. Creativiteit is net als communicatie van belang, maar niet zo prominent als in de vorige fase 'Inzicht Verschaffen'. Het gaat in beiden fasen om dezelfde beeldvormende kwaliteiten die noodzakelijk zijn. Het vormgeven van een oplossing kan alleen het logische gevolg zijn van een goed vormgegeven probleemanalyse.

Gegrepen door de probleemstelling

Wanneer een IT Architect begint aan zijn opdracht, is het vaak een taai werk. Nieuwe mensen leren kennen en daar een samenwerking mee opzetten, inwerken in de technische aspecten en rondzwemmen in vraag en probleemstelling. Inwerken vraagt extra energie, maar kan er ook voor zorgen dat de IT Architect betrokken raakt bij de opdracht. Wat levert deze extra inspanning op?

¹⁰ Muller, M.; Neecke, H.; Hammer, D.; Perdeck, M.; 'De opgave van de IT-architect met betrekking tot de "menselijke maat"'; voor studiegroep "De Menselijke Maat in IT"; 2004

¹¹ Neecke, H.; 'De mens heeft al sinds de oorsprong moeite met de menselijke maat' voor studiegroep "Menselijke Maat in IT"; juni 2003

¹² Start voor informatie over Vitruvius: <http://en.wikipedia.org/wiki/Vitruvius> & Wittkower, R.; *Architectural Principles in the Age of Humanism* (1973)

¹³ Start voor informatie over Le Corbusier en de Modulor: http://en.wikipedia.org/wiki/Le_Corbusier

“Werken aan een probleemstelling waar je hart in ligt, daar wordt de oplossing beter van”

Door een probleem in zijn eigenheid te proberen te begrijpen, ontstaat er ook een bepaalde band met het probleem. Dit maakt werken aan een opdracht boeiend. Je krijgt oog voor de technische, organisatorische, menselijke en de sociaal-culturele aspecten. Juist voor een IT Architect geldt een brede invloedssfeer op zijn probleemdefinitie, want de IT Architect moet ook de inpassing van de oplossing in zijn omgeving verzorgen. Nogmaals: de omgeving is geen onderdeel van de opdracht. Maar de omgeving beïnvloedt de vorm van je oplossing (en voorafgaand daaraan je probleemstelling), anders zou deze nooit anders dan door puur toeval kunnen passen in de omgeving.

Voorbeeld: De doelen van de twee afdelingen verenigd in een oplossing

Bij de eerder genoemde netwerkbeheerder waren er twee afdelingen met een heel andere visie op wat er nodig is. Ik begreep de beweegredenen van een centrale IT afdeling. Net nieuw, opgezet om de gezondheid en effectiviteit van de hele bedrijfs-IT te verbeteren, waren ze voor iedereen een verstoring van de bestaande gang van zaken en moeten ze zich overal bewijzen. Ik wilde graag meewerken aan het verbeteren van de IT huishouding, dus zou er ook een *Enterprise Service bus* moeten komen die algemene standaardisering van digitale communicatie mogelijk maakt en afdwingt.

In tweede instantie kon ik ook veel begrip opbrengen voor de klantenafdeling die met de argumenten tijdigheid en marktgerichtheid een bus in eigen beheer wilde opzetten. Ze wilden meer controle hebben over het berichtenverkeer. Betekent dat dan geen standaardisering? Marktgerichtheid hield op dat moment voor hen vooral in dat ze *business process monitoring* wilde gaan gebruiken, althans ze wilden kunnen nagaan wat de status was van opdrachten die met berichten over de bus verstuurd werden tussen hun afdeling en anderen. Moet je daarvoor *business process monitoring* in eigen beheer gaan opzetten?

Door meer begrip voor beide situaties te krijgen, zag ik dat dit betekent dat ook de klantenafdeling een centrale bus nodig heeft omdat het om communicatie met andere (interne en externe) afdelingen gaat. Meer controle betekent niet meer dat alles in eigen beheer moet, maar juist meer in samenwerking. Dit is een creatieve ombuiging geweest op basis van inleven in de situatie.

Het heeft natuurlijk nog veel inspanning gevraagd om telkens weer andere mensen in dezelfde twee afdelingen, of dezelfde mensen in nieuwe fasen van het project te overtuigen. Het gevoel van het soort controle dat men binnen de afdeling wilde hebben, bleef sterk gericht op eigen assets. Deze ‘machtvraag’ kwam weer duidelijk naar boven toen het over kwesties als beheer ging. Ik kon beide partijen begrijpen, en zo kon ik deze weg uit de probleemstelling vinden. Het is empathie die de doorslag geeft.

En weer loskomen van de oplossing

Een typische taak voor de IT Architect is het overzicht bewaren en de inpasbaarheid van de totale oplossing bewaken. Daarvoor is het nodig dat hij of zij zich ook weer los kan maken van de oplossing die ontworpen is en de oplossing zo objectief mogelijk beoordeelt. Maar hoe maak je je los van je eigen ontwerp? Hoe ga je uiteindelijk boven de systemen staan zodat de je de inpasbaarheid kunt valideren?

“Loskomen van je ontwerp kan alleen door gevoel voor de situatie van de opdrachtgever”

Niet je eigen gedachtevorming en ontwerp is leidend, maar de probleemstelling van de opdrachtgever. Het loslaten van eigen ideeën en ze aan de toets van de opdrachtgever en de praktijk overlaten. Het vraagt om de skills die de camera's in de film *The Matrix*¹⁴ hebben als ze vanaf het afvuren van een schot de kogel volgen en meedraaien om de impact ervan te filmen. Het gaat om het verplaatsen van je empathie, weer terug naar de situatie van de opdrachtgever.

Voorbeeld van individuele karakters

Bij het verifiëren van je oplossing kom je als IT Architect ook heel nadrukkelijk in aanraking met de verschillende karakters van de mensen die in de organisatie van de opdrachtgever werken. Er zijn altijd mensen die een oplossing wel begrijpen en zien zitten. Maar soms begrijpt de opdrachtgever zelf het uiteindelijk concept niet goed. Het is vreemd om iets voor iemand te ontwerpen, dat ook door zijn

¹⁴ Meer informatie via Internet Movie database <http://www.imdb.com/title/tt0133093/>.

medewerkers op waarde wordt geschat, maar dat telkens bij hem zelf niet lijkt aan te slaan. Als er iets op papier stond was dat onbegrijpelijk en liep de concrete implementatie te veel vertraging op. Als ik het niet opgeschreven had omdat de implementatie voor ging, viel het nog veel minder in goede aarde.

Het bleek zeer duidelijk in review rondes waar de opdrachtgever de oplossing om telkens andere redenen niet zag zitten. Ik vond dat raar, want de bezwaren dacht ik met een logisch verhaal weg te nemen. Toch werkte het niet. En het verhaal telkens aanpassen naar de woorden van de opdrachtgever ging uiteindelijk ook niet omdat we dan eindeloos heen en weer schoten tussen de zelfde synoniemen. Er schuilt kracht in opnieuw verwoorden, maar er is ook een grens aan. Het enige wat ik nog kon doen was vertrouwen winnen door het verhaal dat oorspronkelijk zijn verhaal was niet van hem 'af te pakken'. Voor zijn gevoel moest het zijn verhaal blijven, en dit heb ik alleen begrepen met empathie. Uiteindelijk heeft hij een presentatie gemaakt op basis van verschillende presentaties en verhalen van mij. Het werd heel anders dan ik verwachtte, en dat heeft de verdere presentatie naar buiten gevormd.

3) Plaatsing in de inherente onzekerheid van de omgeving

Er was een probleem en er is een oplossing ontworpen. Zal deze gaan werken? Dat blijft de spannende vraag. Wat kan ik als IT Architect hieraan doen na de fase van het ontwerp en wat blijft de invloed van de omgeving? Dit zijn allemaal onzekere factoren. Het valt op dat onzekerheid in voorbeelden minder sterk is aan te wijzen. Als het over acceptatie van de oplossing gaat, is het nog het makkelijkst te zien. Als het om onzekerheid van de omgeving en de intrinsieke onzekerheid van de ontwerper gaat, blijft het een kwestie van persoonlijke waarneming. Onzekerheid is er wel, maar het niet vaak onderwerp van gesprek.

Acceptatie van de oplossing

Past de oplossing wel? Hoe zullen het bedrijf en de markt erop reageren? Zal men de oplossing uiteindelijk gaan incorporeren in de bedrijfsvoering of blijft het een soort pilot? Het vraagt bijzonder veel overleg met alle betrokken partijen om dit te blijven nagaan.

“Een IT Architect moet blijven wikken en wegen”

Een oplossing is ontworpen, maar dan is de IT Architect nog niet klaar met zijn werk. Tijdens de bouw 'moet de IT Architect het ontwerp in de gaten blijven houden' wordt er altijd gezegd. Je moet er bij betrokken zijn om te kijken of de uitvoering zo is als je bedoeld hebt, of alles helder is overgekomen. Dat vraagt ook om begrip van de wereld van de bouwers, zodat je begrip voor je ontwerp kunt kweken. Andersom is dit ook de eerste toets van je ontwerp. De bouwers hebben het twijfelachtige voorrecht de eerste gebruikers van je ontwerp te zijn, en hun reactie is op die gronden zeer waardevol. In deze fase moet je bekijken welke aanpassingen nog gemaakt moeten en kunnen worden, en welke niet.

Er is een ontwerp voor een specifieke omgeving neergezet, een ontwerp dus voor dit moment en deze plaats, deze mensen. Als 'dit moment', 'deze plaats', 'deze mensen' echt punten in de ruimte zijn in plaats van iets grotere units, is de oplossing natuurlijk te beperkt. 'Dit moment' zal een bepaalde duur moeten hebben. Wikken, wegen en bijschaven van de oplossing is nodig om de tijdelijkheid en plaatselijkheid van de oplossing voldoende *body* te geven, zonder dat het ingepaste ontwerp in gevaar komt.

Praktijksituatie: Onzekerheid bij anderen wegnemen

Elke belanghebbende heeft een bepaalde kijk op het probleem en wil dat in probleemstelling en ontwerp zien terugkomen. Elke belanghebbende wil kijken of de oplossing specifiek voor zijn situatie gaat werken. Hiervoor moet de IT Architect het probleem telkens op nieuwe wijze inzichtelijk kunnen maken en bijna eindeloos kunnen herformuleren waar het in de probleemstelling en ontwerp om gaat. En tijdens die bijna eindeloze reeks herformuleringen raak je zelf ook weer aan het wikken en wegen.

Welke Soft Skills zijn dat, herformuleren, wikken, wegen, bijschaven en overtuigingskracht? Als IT Architect behoor je tenminste een paar oplossingen te definiëren en die tegen elkaar af te wegen. Het gaat erom dat je openstaat voor zijn omgeving en zodoende in gesprek blijft met alle partijen die bij de

oplossing betrokken zijn en kijkt naar mogelijke verbetering van de oplossing. Het gaat ook om acceptatie want er had misschien ook een heel andere oplossing ontworpen kunnen worden, en wie weet had dat nog gewerkt ook, zij het met heel andere voordelen en nadelen.

Dit besef is ook verwerkt in de *mission statement* van de faculteit van Techniek, Bestuur en Management van de TU Delft¹⁵: “De problemen waarvoor de samenleving, zich geplaatst ziet, worden steeds complexer van aard. Kenmerkend voor deze tijd is, dat de technologie in deze problemen een steeds belangrijker rol speelt. De technologie maakt de problemen niet alleen complexer, maar speelt ook een steeds belangrijker rol in de oplossingen. Bovendien worden de oplossingen minder eenduidig. “De *Oplossing*” bestaat niet meer, “een *oplossing*” wel, met de nodige voordelen, maar ook de nodige nadelen.”

Acceptatie van het risico: Leven met onzekerheid

We leven in een maatschappij die niet goed om kan gaan met onzekerheden. We proberen ze uit te bannen. Verrassing moet er zoveel mogelijk uit worden gefilterd, en we willen liever vasthouden aan het model dan aan de werkelijkheid die meer verrassingen bevat. Dit zie je ook terug in de modellering die we vaak doen. Een model is in eerste instantie bedoeld om een aspect van de werkelijkheid inzichtelijk of bespreekbaar te maken. Maar het wordt al snel een hele constructie waaraan altijd vastgehouden moet worden. Het krijgt allemaal lagen en dimensies om zo echt mogelijk te lijken.

De rede en de retórica

Er is vaker gesteld dat de moderne maatschappij, die van de techniek en de logica, een risico mijdende maatschappij is. Er zijn wel tijden geweest dat men beter met onzekerheid kon omgaan. Stephen Toulmin schrijft in zijn boek *Kosmopolis*¹⁶ over de 16^{de} eeuw toen de wetenschappen rede en retórica beide in even hoog aanzien stonden. Ik durf te wedden dat u als lezer uw wenkbrauwen nu optrekt: retórica een wetenschap? Zo wordt het nu niet gezien, het wordt gekoppeld aan misleidingen en drogredeneringen. Maar retórica gaat over het overtuigen van mensen en niet over logica. Als iets op een bepaald moment en op een bepaalde plek waar is, en op een andere plek of moment niet meer, is het dan geen valide oplossing meer? Is dat niet juist waarom het gaat bij inpassen van een oplossing in de situatie van de opdrachtgever? En van de andere kant: de zekerheid van modellen is ook maar een schijnzekerheid. Een model blijft altijd een model, en zal nooit de werkelijkheid worden. Dit is een vorm van onzekerheid waarmee je in je werk moet om kunnen gaan, voor jezelf en voor de opdrachtgever.

“Uitbannen van kansen maakt de risico’s groter”

Als je risico’s vermijdt, loop je een veel groter gevaar wanneer er toch wat gebeurt. Dan word je volstrekt overvallen. Je was namelijk op de gecreëerde zekerheid gaan vertrouwen, er is in geïnvesteerd, de formule ‘*impact = risico x schade*’ blijft geldig. Er is bijvoorbeeld gebouwd in de uiterwaarden, er draaien ongedocumenteerde applicaties in het hart van veel bedrijven. De Duitse socioloog Ulrich Beck¹⁷ heeft het begrip onzekerheid al in 1986 in het centrum van zijn analyse van de laatmoderne maatschappij geplaatst. De bewoners van deze maatschappij zijn ervan doordrongen dat onzekerheid niet langer uit te bannen is, zoals de modernen altijd hadden willen doen geloven. Risico’s zijn inherent aan het laatmoderne leven en overall heerst een fundamentele onzekerheid over de gevolgen van ons handelen in de nabije en verre toekomst. We zijn daardoor in de paradoxale situatie terecht gekomen dat we moeten handelen zonder dat we over de noodzakelijke fundamenten beschikken.

Deze situatie heeft de wiskundige en hoogleraar *Aeronautical Engineering* Henk Tennekes¹⁸ ertoe gebracht een boek te schrijven met een verzuchting als titel ‘Dan leef ik liever in onzekerheid’. Deze verzuchting is echter ook een opluchting. De wetenschap heeft gelukkig niet overall grip op. En is nog plaats voor verwondering en voor echte vragen.

¹⁵ <http://www.tbm.tudelft.nl/live/pagina.jsp?id=30118494-6cd0-471e-b74f-984332533500&lang=nl>

¹⁶ Toulmin, S.; *Kosmopolis*, Verborgene agenda van de moderne tijd; Kok Agora, Kampen 1990

¹⁷ Beck, U: *Risikogesellschaft. Auf dem Weg in eine andere Moderne*. Suhrkamp, Frankfurt a.M. 1986.

Zie ook Wikipedia Deutsch onder *Risikogesellschaft*

¹⁸ Tennekes, H: *Dan leef ik liever in onzekerheid*, Aramith Uitgevers, Bloemendaal 1990.

Voorbeeld: De impact van een overstroming

Ik wil hier een voorbeeld uit de klassieke Nederlands techniek aanhalen. In het begin van de 15^{de} eeuw is er een aantal grote overstromingen in Nederland geweest, in het bijzonder de Sint Elizabeths vloed in 1421 waardoor uiteindelijk de Biesbosch is ontstaan. De doorbraak van de Rijn naar de Maas is toen ook ontstaan: de Lek. Dat geeft wel aan dat het vreselijke overstromingen waren. Doordat men er echter vaker mee te maken had, bleef de economische schade toch beperkt.

Als we nu kijken wat er bij de overstroming in 1953 gebeurde, was het risico veel groter. Zeeland en de Zuid-Hollandse eilanden stonden al onder water. Zuid-Holland werd bedreigd. Er is een prachtig verhaal over, dat aangeeft dat het echt maar een haar gescheeld heeft of het zuidelijk deel van Holland was ook ondergelopen¹⁹. In dat geval zou de schade voor Nederland waarschijnlijk onherstelbaar zijn geweest. Heel Rotterdam, de economische motor en zeer dichtbevolkt gebied, zou onder zijn gelopen terwijl men nog bezig was aan de wederopbouw na de tweede wereldoorlog.

Voorbeeld: De strategie van een Service Bus en leven met de keuzes die gemaakt zijn

Vaak wordt er een centrale *Enterprise Service Bus* ontworpen omdat een centrale IT afdeling dit als een strategie heeft ontwikkeld. Is dit daadwerkelijk goed voor het bedrijf? Gaat het de centrale IT afdeling om standaardisatie en doelgerichte IT? Gaat een bus hier ook aan bijdragen? Op basis van aannames werken we allemaal verder. Er zijn referenties om op te vertrouwen, maar de vraag kan pas echt beantwoord worden als we alles al gebouwd hebben en de investering gedaan is.

De IT Architect kan met een oplossing niet alle onzekerheid uitsluiten. Niet de onzekerheid die van andere niveau's wordt aangedragen, maar ook niet alle onzekerheid binnen het eigen ontwerp. Er worden aannames gedaan, er worden keuzes gemaakt, alles kun je documenteren. Maar ergens zul je ook met de gevolgen moeten leven. En behalve dat de IT Architect hier zelf van doordrongen moet zijn, moet hij hiervoor ook de acceptatie van de opdrachtgever krijgen.

Professionele onzekerheid, intrinsieke onzekerheid

We hebben van alles over de IT Architect gezegd, uiteindelijk blijft het een bepaalde persoon. De IT Architect moet voor zichzelf overzicht creëren om boven de materie kunnen gaan staan.

Je weet nooit alles, daarvoor is het echt te veel, en toch moet je een oplossing ontwerpen. Je moet dus met je eigen onwetendheid, en de onzekerheid die dat mogelijk oproept, kunnen omgaan. Niet alleen de onzekerheid of je feitelijk tot een goed ontwerp of een goede oplossing kunt komen. Maar ook het vertrouwen op eigen kunnen ondanks dat je niet alles weet. Ervaring geeft wel het wat vertrouwen, maar per opdracht verschilt de situatie en een deel van de onzekerheid is daarmee intrinsiek. Omdat er altijd probleeminterpretatie plaats vindt tijdens het ontwerp, zal er ook altijd professionele onzekerheid zijn, geheel los van het ervaringsniveau van een IT Architect. Ervaring leert je niet om zeker te zijn, ervaring leert je om te gaan met de onzekerheid.

Voorbeeld

Ik kwam het eens in een heel basale vorm tegen. Er werd een enorme berg informatie bij mij gedumpt zou ik haast willen zeggen. Zeker ook omdat een afdeling wilde laten zien dat zij niet degenen waren die openlijk niet meewerkten. Ik kon maar beperkt informatie opnemen, door de tijdsdruk en de ongestructureerdheid van alle informatie. Daar kon ik makkelijk in verdrinken. Ik kon ook proberen vorm te geven aan mijn eigen gedachten en op schrijven wat ik wel wist. Het klinkt eenvoudig maar iets op papier krijgen, zonder je af te laten leiden door de dingen die je niet weet en daar eventueel aannames voor in de plaats te zetten, is een kunst.

Omgeving verandert als je er zelf in werkt

Een probleem oplossen is reflexief, je verandert de uitgangssituatie zodra je er in gaat werken. Als je een probleem anders omschrijft, moet je een ander ontwerp maken om het probleem op te lossen. Een mooi voorbeeld in dit geval is het schrijven van een scriptie, daar ben ik op dit moment mee bezig. Ik heb van

tevooren een ontwerp gemaakt, op hoofdlijnen mijn verhaal uitgewerkt. Terwijl ik dat deed, kwam ik op nieuwe ideeën en tijdens het schrijven ook weer. Argumentaties werden beter doordacht en moesten daardoor op een andere plek worden gebracht, waardoor ook de structuur van dit verhaal veranderde.

Terwijl je een ontwerp op papier zet verandert het, ook dit veroorzaakt een vorm van onzekerheid die inherent is en waar een IT Architect mee om moet kunnen gaan.

Slotakkoord

Er zijn meerdere Soft Skills per fase en domein aan bod gekomen in de afgelopen hoofdstukken. Per fase en domein is er een Soft Skill aan te wijzen die een dominante rol speelt. De Soft Skill die het meest in het oog springt in elke fase heb ik in de tabel uit de inleiding ingevuld. Creativiteit tijdens de fase 'Inzicht verschaffen', empathie tijdens de fase 'Oplossingen bieden', en omgaan met onzekerheid tijdens 'Plaatsing in de omgeving'

Domein, zie	Technisch	Zakelijk/Commercieel	Organisatorisch	Menselijk
Fase				
Inzicht verschaffen	analyse	creativiteit	creativiteit	communicatie
Oplossing bieden	analyse	empathie	empathie	communicatie
Plaatsing in de omgeving	analyse	omgaan met onzekerheid	omgaan met onzekerheid	communicatie

Tabel 2: De domeinen van een probleem uitgezet tegen de fases van een opdracht voor een IT Architect – nu met Soft Skills die in per domein en per fase dominant zijn.

Kijkend naar de Tabel vallen de volgende dingen op:

Communicatie werkt als een basis Soft Skill, het is de toegang tot de andere Soft Skills. Met behulp van communicatie kun je de andere Soft Skills gebruiken. Analyse werkt ook als een toegang tot de andere Soft Skills. Om het in technische termen te vatten: analyse en communicatie, ofwel kijken en praten, vormen de input en output van een 'IT Architect' als een soort systeem. De interne processing van de IT Architect bevat op het gebied van Soft Skills creativiteit, empathie en omgaan met onzekerheid.

Het is opvallend dat creativiteit uitblinkt in de beginfase en empathie in de oplossingsfase. Velen onder u zijn wellicht geneigd om dit intuïtief andersom te zien. Bekijk het een zo: in de beginfase weet een IT Architect nog niet veel. Dan heeft deze zijn creativiteit nodig om zich een beeld te vormen. Empathie kan pas worden gebruikt als je meer weet van de situatie en de hoofdrolspelers daarin. In de beginfase moet van de gegeven opdracht toch een eigen probleemstelling gemaakt worden, en dat is wellicht nog fundamenteeler dan het maken van een oplossing bij een probleemstelling. Dit omdat kiezen voor een bepaalde probleemstelling vaak al een (vorm van) een oplossing suggereert.

De Soft Skills creativiteit, empathie en omgaan met onzekerheid vormen een steeds wijder wordende trechter, zoals ze nu geordend zijn in de loop van een opdracht. Bij creativiteit ben je nog redelijk naar binnen gericht en in je zelf bezig, bij empathie ga je afstemmen met de buitenwereld en bij omgaan met onzekerheid geef je de buitenwereld een plek.

Omgaan met onzekerheid heeft als karakteristiek dat men er niet graag over spreekt. Misschien een aardig uitgangspunt voor vervolgonderzoek. Voorlopig zou ik er over willen zeggen dat men alle onzekerheden ook niet expliciet hoeft te benoemen tijdens een opdracht. Belangrijk is dat de ontwerper zelf de onzekerheid erkent en instemming verwerft bij de opdrachtgever dat er onzekerheid is: amor fati.

Referenties

Beck, U: *Risikogesellschaft. Auf dem Weg in eine andere Moderne*. Suhrkamp, Frankfurt a.M. 1986.
Zie ook Wikipedia Deutsch onder *Risikogesellschaft*

Sllager, K.; *De Ramp* (2003)

Muller, M.; Neecke, H.; Hammer, D.; Perdeck, M.; '[De opgave van de IT-architect met betrekking tot de "menselijke maat"](#)'; voor studiegroep "[De Menselijke Maat in IT](#)"; 2004

Neecke, H.; '[De mens heeft als sinds de oorsprong moeite met de menselijke maat](#)' voor studiegroep "Menselijke Maat in IT"; juni 2003

Tennekes, H.: *Dan leef ik liever in onzekerheid*, Aramith Uitgevers, Bloemendaal 1990.

Toulmin, S.; *Kosmopolis, Verborgene agenda van de moderne tijd*; Kok Agora, Kampen 1990

Wieringa, R.J.; '*De competenties van de IT Architect*' in: *Informatie 4* (2006)
<http://www.informatie.nl/artikelen/2006/04/competentiesVanDelctarchitect.html>

IBM Global, *Characteristics of the IT Architect (IT Architect profession)*
<http://w3-03.ibm.com/hr/careerplanner/caita008.html>

The IBM IT Architect Profession Guide, version 2.2, March 2007
<http://w3-03.ibm.com/hr/careerplanner/ITArchitectProfessionGuideV22March2007.pdf>

Missie statement van de Faculteit Techniek, Bestuur en Management van de TU Delft
<http://www.tbm.tudelft.nl/live/pagina.jsp?id=30118494-6cd0-471e-b74f-984332533500&lang=nl>

Start voor informatie over Vitruvius: <http://en.wikipedia.org/wiki/Vitruvius> &
Wittkower, R.; *Architectural Principles in the age of Humanism* (1973)

Start voor informatie over Le Corbusier en de Modulor: http://en.wikipedia.org/wiki/Le_Corbusier