Evenwicht in informatiesysteemontwikkeling

H.A. (Erik) Proper
Nijmeegs Instituut voor Informatica en Informatiekunde
Katholieke Universiteit Nijmegen[1]
E.Proper@acm.org

Traditioneel worden informatiesystemen ontwikkeld middels omvangrijke projecten. Het is echter maar de vraag inhoeverre een traditionele manier van denken in termen van projecten nog van toepassing is in onze moderne digitale samenleving. Dit artikel beoogt de achterliggende vragen nader te analyseren, en tracht de lezer aan te zetten tot *nadenken* over de rol van projecten in traditionele zin. Het doel van dit artikel is daarmee dus niet zo zeer om een concrete nieuwe werkwijze aan te dragen, maar eerder een discussie te starten over de essentiële uitdagingen bij informatiesysteemontwikkeling.

Informatiesysteemontwikkeling als gemeen probleem

Elke projectleider zal stellen dat een project goede uitgangspunten en einddoelen nodig heeft, die ook nog eens helder afgebakend en stabiel moeten zijn [1]. In de moderne digitale samenleving hebben projecten echter te maken met diverse vaagheden, onstabiliteit en onzekerheden ten aanzien van juist deze uitgangspunten en einddoelen [2]. Dit kan nogal wat gevolgen hebben voor ontwikkelprojecten.

Informatiesysteemontwikkeling kan gezien worden als het oplossen van een "probleem" [3]. Als een bestaande situatie niet (meer) voldoet aan de systeemeisen, dan is het probleem dat opgelost moet worden: "Zorg dat er een systeem komt dat wél aan de eisen voldoet". Dergelijke problemen komen in soorten en maten. Een relevant onderscheid dat hierbij gemaakt kan worden is het onderscheid tussen zogenaamde "gemene" en "tamme" problemen ([3], [4] en [5]). Een gemeen probleem laat zich typisch kenmerken door eigenschappen zoals:

- Je begrijpt een gemeen probleem pas goed als je er een oplossing voor hebt bedacht. Elke mogelijke oplossing brengt nieuwe aspecten van het probleem aan het licht, aspecten die verdere aanpassing van de oplossing vereisen.
- Gemene problemen hebben geen stopcriterium. Er is geen eenduidige en stabiele probleemdefinitie te geven. Als gevolg hiervan is het is niet duidelijk wanneer het probleem *echt* is opgelost.
- Oplossingen voor gemene problemen zijn niet simpelweg goed of fout. In plaats daarvan zijn ze "beter", "slechter" of "goed genoeg". Voor gemene problemen is het moeilijk om op een objectieve wijze de kwaliteit van een oplossing te beoordelen.
- Elke oplossing van een gemeen probleem krijgt slechts één kans. Elke realistische poging heeft direct consequenties. Je kunt niet eerst een realistisch prototype van de Betuwelijn bouwen om te zien of het allemaal wel zal werken in de praktijk. Hierbij komt meteen de in gemene problemen ingebouwde patstelling naar voren. Je kunt het probleem niet echt verkennen zonder oplossingen te proberen, maar elke oplossing die je probeert heeft onvoorziene bijeffecten die het probleem direct of indirect verder beïnvloeden.

Deze vier criteria zijn niet alle criteria voor gemene problemen, maar het zijn wel de meest onderscheidende. Het gemeen of tam zijn van een probleem zegt niets over de moeilijkheidsgraad van het probleem zelf. Het bewijzen van Fermat's laatste stelling was volgens deze definitie weliswaar een tam

[1] De Katholieke Universiteit Nijmegen heet vanaf 1 september 2004 de Radboud Universiteit.

probleem, maar men heeft er jaren over gedaan om een oplossing te vinden. Het onderscheid tussen tam en gemeen is overigens een glijdende schaal. Een probleem mag best “een beetje gemeen” of “enigzins tam” zijn.

Hoe zit het nu met het ontwikkelen van informatiesystemen? Komen we daar tamme of gemene problemen tegen? Veel informatiesysteemontwikkelingsprojecten in de digitale samenleving zullen al snel voldoen aan de criteria voor gemene problemen. Laten we daarom de belangrijkste criteria wat nader bekijken (zie voor een uitgebreidere bespreking [2]).

Het eerste criterium betrof het feit dat een probleem pas echt begrepen kan worden als er een eerste oplossing voor is bedacht. Terwijl men een informatiesysteem ontwikkelt, zal men enerzijds nieuwe belanghebbenden tegen kunnen komen, en zullen anderzijds de bestaande belanghebbenden door het concreter worden van het nieuwe systeem steeds beter gaan inzien wat hun feitelijke belangen en wensen zijn.

Het tweede criterium betrof het ontbreken van een goed stopcriterium. Bij het ontwikkelen van informatiesystemen ontstaat er een soort natuurlijke drang om te gaan schuiven met de eisen ten aanzien van het nieuwe informatiesysteem. Ten dele zal dit het gevolg zijn van veranderingen die plaatsvinden in de socio-economische context gedurende de ontwikkeling van het systeem. De wereld staat niet stil. Er zijn echter ook redenen dichter bij huis te vinden.

Het ontwikkelen van een informatiesysteem zal vrijwel altijd de directe context van het systeem beïnvloeden. Immers, om een ontwikkelproject überhaupt op te starten moet er in eerste instantie een concrete behoefte zijn om iets te veranderen aan een reeds bestaande situatie. Er moet een “knelpunt” zijn dat als ernstig genoeg wordt ervaren, om iemand er toe te brengen de moeite te nemen een ontwikkelproject op te zetten en te betalen. Het wegnemen van dat “knelpunt” zal ongetwijfeld het gedrag van de context beïnvloeden. Immers, de context van het systeem heeft moeten leren leven met beperkingen. Zodra die beperkingen zijn weggenomen, zal dat ongetwijfeld leiden tot ander gedrag van de context. Een verandering die vrijwel zeker *nieuwe* veranderingswensen met zich meebrengt, omdat er ergens anders een nieuw knelpunt zal opduiken. Nu kunnen we vervolgens hard roepen “ja maar dat nieuwe knelpunt is niet het originele probleem”, maar dat verandert niets aan het feit dat de definitie van het probleem is veranderd, en dat het dus maar de vraag is of we uiteindelijk het goede probleem hebben opgelost. Wat is het goede stopcriterium?

Het derde criterium voor gemene problemen had betrekking op het feit dat oplossingen niet simpelweg goed of fout zijn. Wanneer is een informatiesysteem goed? Als het conform de specificaties werkt? Maar wat als het systeem, om dat niveau van correctheid te bereiken, te laat wordt opgeleverd? Is een op tijd opgeleverd systeem met een aantal fouten in niet al te cruciale delen van het systeem, niet “beter” dan een systeem dat een maand later wordt opgeleverd zonder fouten? Stel dat het gaat om een informatiesysteem dat noodzakelijk is voor de introductie van een nieuw product op de markt. Een maand eerder met het nieuwe product op de markt komen dan de concurrentie kan wel eens van doorslaggevend belang zijn. Hoe erg zijn die overgebleven fouten dan nog?

Daarnaast blijft het, zie ook de vorige twee criteria, moeilijk om te komen tot een precieze specificatie van wat eigenlijk het goede informatiesysteem is. Als mens zijn we uitermate adaptief ingesteld. Zelfs als een systeem niet precies doet wat we er van verwachten, kunnen we er nog steeds erg nuttig gebruik van maken. Is dat systeem dan “slecht”? Het kan beter. Maar dat kan het bijna altijd. Wanneer is goed *goed genoeg*?

Volgens het vierde criterium zou een oplossing van een gemeen probleem slechts één kans hebben om zich te bewijzen. Dit criterium zal niet altijd van toepassing zijn bij het ontwikkelen van informatiesystemen. Als het te ontwikkelen informatiesysteem klein genoeg van omvang is, zou men zich kunnen veroorloven om te experimenteren met alternatieve oplossingen. Echter, bij grootschalige informatiesystemen, zoals de OV chipkaart, elektronische belastingaangifte, het betalingsverkeer, etcetera, is er slechts beperkt ruimte om op realistische schaal te experimenteren [3]. Na bijvoorbeeld een mislukte poging tot het introduceren van een OV chipkaart, moet er voor een volgende poging rekening worden gehouden met eventuele “overblijfselen” van deze eerdere poging. Het oorspronkelijke probleem bestaat daarom ook niet meer als zodanig. Je hebt maar één kans gehad om het originele probleem op te lossen.

Ik zou daarom willen stellen dat informatiesysteemontwikkeling in de digitale samenleving er toe neigt een gemeen probleem te zijn. Deze gemeenheid kan voortkomen uit zowel de menselijke, organisatorische als technologische aspecten van informatiesystemen.

Traditionele informatiesysteemontwikkeling

Traditioneel kan informatiesysteemontwikkeling worden opgedeeld in een aantal fasen. Nu zijn er verschillende manieren om deze indeling te maken ([6] en [7]), maar grofweg kunnen we stellen dat er vier belangrijke fasen zijn te onderkennen: definiëren, ontwerpen, construeren en implementeren ([8] en [9]).

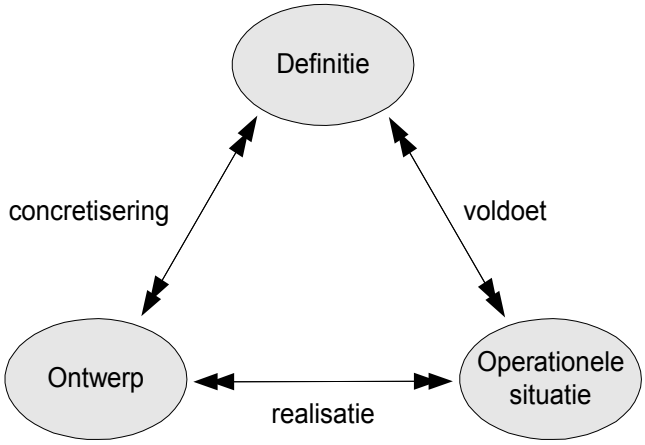
Er zijn verschillende strategieën om de vier genoemde fasen te doorlopen. Een lineaire strategie loopt, zoals de naam al suggereert, voor het gehele systeem stap voor stap alle fasen door. Dit wordt ook wel eens een waterval aanpak genoemd, omdat we als het ware van de ene fase naar de andere fase stromen. Een eerste variatie hierop is de incrementele strategie, waarbij de verschillende fasen per deelsysteem doorlopen worden. Deze strategie gaat er wel van uit dat we een zinnige opdeling van het systeem hebben kunnen maken in deelsystemen. Een derde variant is de iteratieve strategie. Hierbij wordt door veelvuldig op en neer springen tussen de verschillende fases een voldoende goed eindresultaat toegewerkt.

Wat al deze aanpakken gemeen hebben is dat ze uitgaan van wat ik zou willen noemen het "projectdenken". Men neemt als vertrekpunt de aanname dat men, idealiter, in staat is om heldere uitgangspunten en einddoelen te formuleren. Startende vanuit het projectdenken zou je het liefst willen zien dat je te maken hebt met situaties waarin je een nette, goed gestructureerde, lineaire aanpak kunt gebruiken. Omdat de praktijk weerbarstiger is dan dat, moest het projectdenken opgerekt worden. Dit heeft geleid tot de meer iteratieve strategieën ([8] en [9]). Ik zou echter willen stellen dat zelfs wanneer een iteratieve strategie gebruikt wordt, dit toch een ontkenning blijft van het gemene karakter van de problemen die men oplost. Men blijft immers het projectdenken, dat vraagt om heldere uitgangspunten en einddoelen, als kader gebruiken. Een moderne variant van de iteratieve strategieën zijn de zogenaamde "wendbare" (agile) systeemontwikkelaanpakken [10]. Dergelijke wendbare aanpakken gaan al minder uit van het traditionele projectdenken.

Evenwichtsdenken

Ik vraag me echter af of we er niet veel beter aan doen om in plaats van het projectdenken een ander startpunt te kiezen, waarbij ik overigens niet wil zeggen dat projecten afgeschaft moeten worden. Projecten zijn zeker nodig, ze bieden immers een goede manier om stabiele en behapbare brokken werk te verzetten. Ze zouden echter ingebed moeten worden in een groter geheel dat niet persé uitgaat van stabiele uitgangspunten en einddoelen. Er wordt hierbij ook wel eens gesproken over microprojecten, waarbij het gaat om het initiëren van vele kleine projecten, in plaats van één onbestuurbare kolos [11]. De ideëen achter wendbare systeemontwikkeling [10] vertonen sterke overeenkomsten met het idee van een portfolio van microprojecten in plaats van één groot project.

Zelf zou ik informatiesysteemontwikkeling daarom in eerste instantie willen zien als een continu evolutionair proces, met als doel het bewaren van evenwicht tussen drie polen: *definitie*, *ontwerp* en de *operationele situatie.* De definitiepool richt zich op het gewenste systeem. Met andere woorden, het wat, waarom en hoe goed. Dit betreft overigens niet alleen de expliciete wensen en eisen (die op papier staan), maar juist ook de impliciete eisen en verwachtingen. De ontwerppool richt zich op het hoe en waarmee van het systeem. De operationele pool betreft de actuele informatievoorziening, waarbij het overigens best mogelijk is dat er in een actuele situatie nog geen gebruik gemaakt wordt van een (gecomputeriseerd) informatiesysteem.



Figuur 1: Evenwicht in informatiesysteemontwikkeling

Zoals in Figuur 1 is geïllustreerd spannen de drie polen (metaforisch gezien) samen een spanningsveld op. De drie polen zijn onderling verbonden middels een drietal krachten; drie elastiekjes. Het elastiekje tussen de definitiepool en de operationele pool heeft betrekking op de vraag in hoeverre de operationele situatie *voldoet* aan de gestelde de definitie. Het elastiekje tussen de definitiepool en de ontwerppool richt zich op de vraag of het ontwerp een *concretisering* is van de definitie; een goede vertaling van het wat naar het hoe. Tenslotte richt het elastiekje tussen de ontwerppool en de operationele pool zich op de vraag of de operationele situatie een goede *realisatie* is van het ontwerp. Het evenwichtsmodel in Figuur 1 is van toepassing op alle soorten informatiesysteemontwikkeling, dus zowel bij nieuwbouw, correctie, uitbouw als renovatie.

Het evenwicht bewaren

Het evenwichtsmodel zegt nog niets over de eventuele processen die het krachtenveld kunnen beïnvloeden. Idealiter is er een volledige balans tussen de drie polen. Bij een onbalans ontstaat er een behoefte aan verandering. Mits we een dergelijke verandering scherp genoeg kunnen formuleren in termen van uitgangspunten en einddoelen, en de verandering klein genoeg is qua omvang en te verwachten doorlooptijd, kan er een project opgestart worden om deze verandering daadwerkelijk te verwezenlijken.

In elk van de drie verbindingen kan er sprake zijn van een onbalans. Zo kan het zijn dat de behoeften van de belanghebbenden niet voldoende afgedekt worden door de operationele situatie. In zo'n geval kan de balans hersteld worden door de operationele situatie aan te passen, en/of door de behoeften bij te stellen. Dat laatste klinkt misschien raar, maar u zou zich een situatie kunnen voorstellen waarin de diverse belanghebbenden er onrealistische wensen op nahouden, die bijvoorbeeld met de huidige stand der technologie niet praktisch realiseerbaar zijn, terwijl de verschillende belanghebbenden elkaar misschien ook nog eens tegenspreken. Het kan in een dergelijke situatie heel effectief en verfrissend zijn om een project op te starten dat er voor moet zorgen dat de belanghebbenden in gaan zien wat realistische en haalbare wensen zijn, gelet op de stand der technologie, de eventuele kosten van verbeteringen en de belangen van andere belanghebbenden.

Wanneer de definitie en het ontwerp met elkaar in onbalans zijn, moet óf het ontwerp geactualiseerd worden, óf moeten wederom de behoeften van de belanghebbenden bijgesteld worden. Een onbalans tussen realisatie en ontwerp kan er op duiden dat de operationele situatie nog verder geactualiseerd moet worden. Het kan er echter ook op duiden dat het ontwerp achterloopt op de operationele situatie. In de praktijk komt men dit laatste nogal eens tegen als er veranderingen aan een systeem zijn aangebracht zonder dat het ontwerp hierop is aangepast.

Elke actie die ondernomen moet worden om de balans weer te herstellen kan in potentie uitgevoerd worden middels een project. De uitgangspunten en einddoelen van een dergelijk project kunnen uitgedrukt worden in termen van de mate waarin de balans verbeterd moet worden. Het is hierbij de uitdaging om zo exact mogelijk uit te kunnen uitdrukken wat het is om in balans te zijn. Met andere woorden, we moeten er

voor zorgen dat er meetinstrumenten komen om, voor elk van de elastiekjes uit het evenwichtsmodel, precies vast te stellen wat de mate van balans in een bepaalde situatie is.

Het evenwicht toetsen

Om in staat te zijn vast te stellen in welke mate een operationele situatie daadwerkelijk een realisatie is van het ontwerp, zijn instrumenten en methoden nodig om te *testen/toetsen* of een operationele situatie zich gedraagt conform het ontwerp. Er bestaat reeds over een uitgebreid instrumentarium voor het testen van software- en hardwareonderdelen van een informatiesysteem [12]. Daarnaast bieden methoden zoals Total Quality Management [13] aanknopingspunten om te toetsen in hoeverre het menselijke deel van een informatiesysteem zich conform het ontwerp gedraagt. Een belangrijke voorwaarde is wel dat de talen die gebruikt worden om het informatiesysteem te ontwerpen een goed gedefinieerde semantiek hebben; ze dienen een *exacte* basis te hebben. We moeten immers wel precies weten waar we op willen testen/toetsen.

Bepalen in welke mate een ontwerp een concretisering is van de definitie, en of een operationele situatie voldoet aan de gestelde definitie, is uitdagender dan het op het eerste gezicht lijkt. Allereerst zijn er specificatietalen nodig om de definitie eenduidig te kunnen vastleggen. Idealiter kunnen we alle soorten kwaliteiten, dus zowel de functionele als niet-functionele, waaraan het systeem zou moeten voldoen in deze talen uitdrukken. Vervolgens kunnen we dan instrumenten en methoden ontwikkelen om te controleren of de operationele situatie, of het ontwerp hiervan, voldoet aan de specificatie. Hier zal overigens niet altijd een simpel ja/nee antwoord mogelijk zijn.

De benodigde specificatietalen kunnen we zeker ontwikkelen. Echter, de aanname dat we de definitie eenduidig kunnen vastleggen (expliciet maken) is wat naïef. Waar moet die eenduidige definitie vandaan komen? Er zijn twee factoren die het verkrijgen hiervan nogal bemoeilijken. Ten eerste zit de definitie van het systeem primair “in de hoofden” van de belanghebbenden verstopt (ze is impliciet). En om de zaak nog erger te maken, zullen die belanghebbenden zich vaak niet eens precies bewust zijn van hun specifieke wensen en eisen. We worden gedwongen om een onderscheid te maken tussen een deel van de definitie dat reeds expliciet is gemaakt, dus op papier staat, en een deel dat impliciet is gebleven. Zodra een belanghebbende zich bewust wordt van een aspect van zijn of haar belangen, en dit kan verwoorden, kunnen we het toevoegen aan het expliciete deel van de definitie. Pas dan zijn we in staat om middels ons instrumentarium te toetsen of de bestaande situatie, of een ontwerp daarvan, hieraan voldoet. Toetsen of een ontwerp of een bestaande situatie voldoet aan het impliciet gebleven deel van de definitie is uiteraard niet zomaar mogelijk. Wel kunnen methoden worden ontwikkeld om, op basis van een ontwerp of een operationele situatie, belanghebbenden te helpen in het expliciet maken van hun behoeften.

Tenslotte is de kans groot dat we te maken hebben met tegenstrijdige wensen ten aanzien van het informatiesysteem. Het verkrijgen van een eenduidige definitie vereist dus ook onderhandelingen tussen de verschillende belanghebbenden en hun belangen.

Het evenwicht sturen

Nu rest de vraag hoe het evenwicht bestuurd kan worden, zowel wat betreft de inhoudelijke als de procesmatige sturing. Bij inhoudelijke sturing gaat het om de uiteindelijke inrichting van het ontwikkelde informatiesysteem, terwijl het bij procesmatige sturing gaat om de inrichting van de ontwikkelprocessen. Het is hierbij zaak dat deze sturing zoveel mogelijk bewust mogelijk gebeurt [14]. Een beslissing om op een bepaalde manier het informatiesysteem in te richten heeft invloed op de kwaliteitseigenschappen van het resulterende systeem. Gelijkelijk zal een beslissing om het ontwikkelproces op een bepaalde manier in te richten van invloed zijn op de kwaliteit van dat proces. Men zou “bewuste sturing” kunnen zien als als een combinatie tussen drie ingrediënten:

1. De semantiek van beslissingen dient gezien te worden als de impact die ze hebben op de kwaliteitseigenschappen van het resultaat. Op basis hiervan kan een goede kosten-baten analyse gemaakt worden ten behoeve van de besluitvorming.
2. Inrichtingsbeslissingen dienen zoveel mogelijk expliciet genomen te worden. Bij informatiesysteemontwikkelprojecten worden nogal eens verborgen beslissingen genomen. Onbewust wordt er gekozen voor een bepaalde technologie, methode of techniek, of wordt er een motivatie van een project als gegeven beschouwd zonder nadere toetsing.

3. Communiceren en onderhandelen tussen de verschillende belanghebbenden over inrichtingsbeslissingen is essentieel om tot een goede besluitvorming en concretisering van de gekozen oplossingsrichtingen te komen.

Tot besluit

Het doel van dit artikel was om kritisch stil te staan bij de traditionele benadering om grootschalige informatiesystemen te ontwikkelen middels (omvangrijke) projecten. Hierbij is alleerst beargumenteerd dat het ontwikkelen van dergelijke systemen veel trekken heeft een gemeen probleem. Vervolgens is het evenwichtsdenken geponeerd als mogelijke alternatief voor het projectdenken.

Referenties

[1] Project Management Body of Knowledge, November 2001. http://www.pmi.org

[2] H.A. Proper. *Informatiekunde; Exacte vaagheid.* Inaugurele rede. Katholieke Universiteit Nijmegen, Nijmegen, Nederland, EU. November 2003. ISBN 909172866

[3] D. Budgen. *Software Design*. Addisson-Wesley, Reading, Massachusetts, USA, 2nd edition, 2003. ISBN 0201722194

[4] H. Rittel and M. Webber. Dilemmas in a general theory of planning. *Policy Sciences*, 4:155-169, 1973.

[5] J. Conklin. Wicked problems and social complexity. White paper, CogNexus Institute, 2003. http://cognexus.org

[6] I. Sommerville. *Software Engineering*. Pearson Education, Harlow, United Kingdom, EU, 6th edition, 2001. ISBN 020139815X

[7] J.C. van Vliet. *Software Engineering – Principles and Practice*. John Wiley & Sons, New York, New York, USA, 2nd edition, 2000. ISBN 0471975087

[8] M. Franckson and T.F. Verhoef, editors. *Managing Risks and Planning Deliveries*. Information Services Procurement Library. ten Hagen & Stam, Den Haag, Nederland, EU, 1999. ISBN 9076304831

[9] H.A. Proper, editor. *ISP for Large-scale Migrations*. Information Services Procurement Library. ten Hagen & Stam, Den Haag, Nederland, EU, 2001. ISBN 9076304882

[10] R.C. Martin. *Agile Software Development, Principles, Patterns, and Practices*, Prentice Hall, Englewood Cliffs, New Jersey, USA, 2002. ISBN 0135974445

[11] CHAOS: A Recipe for Succes. Research report, The Standish Group International, 1998. http://www.standishgroup.com

[12] R. Black. *Managing the Testing Process: Practical Tools and Techniques for Managing Hardware and Software Testing*. John Wiley & Sons, New York, New York, USA, 2nd edition, 2002. ISBN 0471223980

[13] J. Oakland. *Total Quality Management*. Butterworth-Heinemann, London, United Kingdom, EU, 1993. ISBN 0750609931

[14] G.E. Veldhuijzen van Zanten, S.J.B.A. Hoppenbrouwers, and H.A. Proper. System development as a rational communicative process. In *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics*, volume XVI, pages 126-130, Orlando, Florida, USA, July 2003.

De auteur

Erik Proper is als Hoogleraar Informatiekunde verbonden aan het Nijmeegs Instituut voor Informatica en Informatiekunde van de Katholieke Universiteit Nijmegen. http://www.cs.kun.nl/~erikp