5

Groot projectsucces is gevaarlijk

Door: Hans Mulder

Met enige regelmaat verschijnen berichten in de pers over problemen met informatiseringsprojecten. Sommige van die projecten zijn zelfs compleet mislukt. Met name de resultaten van zeer grote projecten laten te wensen over. Bij die projecten treden effecten op die te vergelijken zijn met onder meer de Deltawerken, de HSL of de Noord-Zuidlijn. Kennelijk gaat de complexiteit van dit type projecten het organisatievermogen van mensen te boven.

Er zijn sinds de jaren zestig verschillende onderzoeken uitgevoerd naar de oorzaak van het mislukkingen van informatiseringsprojecten. Daar is uiteraard lering uit getrokken, maar door de enorme ontwikkeling die IT de afgelopen decennia heeft doorgemaakt, is onderlinge vergelijking van projecten lastig. Ook omdat de aard en de scope van IT-projecten ingrijpend zijn veranderd. Met elk nieuw project begeeft men zich op *nieuw en onbekend terrein*. Aan het mislukken van projecten liggen daardoor steeds andere oorzaken ten grondslag. In de publieke opinie blijft echter hangen dat de dienstverlening van de IT-sector ondermaats is. Men beschouwt IT-projecten als zeer risicovol. Dit is een ongenuanceerde voorstelling van zaken, die IT'ers een slechte reputatie heeft bezorgd. Te makkelijk wordt verondersteld dat zij die problemen hebben veroorzaakt. Hoe lastig het ook is om IT-projecten verspreid over een aantal decennia met elkaar te vergelijken, er is wel degelijk lering uit te trekken. Met name uit grootschalig en langdurig statistisch marktonderzoek kunnen waardevolle conclusies worden getrokken (Gaikema et al 2019).

Het zou daarom goed zijn als bedrijven hun plannen aan dat soort onderzoek toetsen om daarmee betere inschattingen te kunnen maken over de slaagkans van toekomstige projecten

Grote projecten hebben een grote kans om te mislukken (Van Dijk, 2013), (Tweede Kamer der Staten-Generaal, 2014), (Groen, 2015), (Bronsgeest, 2016). Het marktonderzoekbureau Standish Group in Boston doet sinds 1985 onderzoek naar het slagen en falen van IT-projecten. Sinds 1994 werkt men op basis van casestudies (The Standish Group, 1994). De onderzochte 'Geslaagd' betekent dat een project binnen de planning en begroting conform de vooraf overeengekomen vereisten, is afgesloten. 'Mislukt' wil zeggen dat het beoogde systeem niet is opgeleverd of niet in gebruik is genomen. De niet-succesvolle projecten die wel in gebruik worden genomen, worden 'problematisch' genoemd. Deze problematische projecten duren langer dan gepland, kosten meer dan gebudgetteerd en bieden minder functionaliteit en/of kwaliteit dan overeengekomen.

Gedurende 10 jaar onderzoek naar het falen van IT-projecten zijn zo'n 50.000 projecten onder de loep genomen (Mulder, Johnson, 2016). Daaruit blijkt dat grote projecten vaker mislukken dan kleine. Grote projecten - boven tien miljoen dollar aan arbeidskosten - gaan qua complexiteit het organisatievermogen van mensen te boven. Kleine projecten zijn beter beheersbaar, de risico's zijn kleiner en in een team van hooguit vijf tot zeven mensen zijn de communicatielijnen kort. In een overzichtelijk proces met opeenvolgende 'stepping stones' worden projecten snel in gebruik genomen, getest en worden ervaringen van gebruikers direct teruggekoppeld. Zo kan de leerervaring wordt meegenomen in de volgende fase.

Recent heeft de Standish Group ontdekt dat succesvol zijn heel gevaarlijk is. Succesvolle projecten worden binnen tijd en budget afgerond en voldoen aan de vooraf gestelde eisen. De manager krijgt er een schouderklop en een bonus voor. Maar onderwijl is de organisatie veranderd en stelt de markt andere eisen dan bij de aanvang van het project. Ondanks het – volgens de norm - succes van dit project levert het daardoor geen of nauwelijks toegevoegde waarde. De achterliggende oorzaak hiervan is dat veel managers bang zijn om te falen. Ze kleuren netjes binnen de lijnen

door zich strikt aan de uitgangspunten van het project te houden en weten precies wat ze zullen opleveren.
Maar als je van tevoren al weet wat je krijgt, leer je onderweg niets en creëer je geen waarde om het bedrijf verder te helpen. Je bent op een ouderwetse manier aan het verbeteren/optimaliseren (wat er is) in plaats van innoveren (het anders doen).
Beter is het om de markt te volgen, af te wijken van het vertrekpunt en daarvan te leren. Die waardevolle leerervaringen kun je steeds opnieuw meenemen in volgende projecten. Dus durf te falen, maar faal snel en leer daar ook direct van.
Er zijn legio succesvolle projecten die geen enkele waarde toevoegen. Uit onderzoek blijkt dat projecten waarvan de uitkomst het verst van de businesscase verwijderd is het meest kans maken om meer waarde te bieden dan vooraf is ingeschat. Simpelweg omdat een statische businesscase niet past in een tijd waar technische ontwikkelingen de innovatie aanjagen. Innovatie, vernieuwing en onderhoud van informatiesystemen moet tegelijk plaatsvinden. Het gevaar van 'succesvolle' businesscases, is dat deze gebaseerd zijn op wat bekend is, terwijl innovatie vaak gaat over out-of-the-box denken, nieuwe combinaties en stapsgewijs doorbreken.

5.1 De waarde van het project

De waarde van een project komt pas tot uiting bij in gebruikname. De waarde in de praktijk dient voorop te staan en niet het succes van het ontwikkeltraject. Immers een succesvol project conform de criteria opleveren op tijd, budget en vooraf overeengekomen functionaliteit, voegt niet per se waarde toe aan een organisatie. Het omgekeerde kan trouwens ook het geval zijn. Een project dat bijvoorbeeld tien procent duurder is dan ingeschat, kan veel meer waarde opleveren.

Nieuwe techniek vormde de afgelopen decennia de drijvende kracht voor nieuwe mogelijkheden, waardoor informatiesystemen sneller en beter ontwikkeld en gebruikt konden worden. Tegenover deze snelle vernieuwing staat 'oude IT' beter bekend als legacy-systemen. Ze zijn niet

meer aanpasbaar aan de veranderende omgeving en moeten daarom als afgeschreven worden beschouwd.
Dat bepaalde systemen snel verouderen, soms zelfs binnen zeven jaar, is al in de jaren zeventig van de vorige eeuw door professor Manny Lehman opgemerkt. Hij verklaart dat software niet *veroudert* door gebruik, maar *slijt* door de noodzaak van voortdurende aanpassing aan de voortschrijdende techniek en aan nieuwe gebruikerseisen. Dit wordt zichtbaar in omgevingen waar informatiesystemen telkens aangepast moeten worden aan veranderende organisatie- en marktomstandigheden, nieuwe wet- en regelgeving en technieken. Naarmate het systeem groter wordt en meer koppelingen krijgt met andere software vergt aanpassing steeds meer inspanning.
Vanuit onderzoek weten we dat duurzame software alleen ontwikkeld kan worden als zich ook in de kleinste onderdelen geen combinatorische effecten – ook wel rimpeleffecten genoemd - meer voordoen.
Een combinatorisch effect bestaat wanneer de impact van een wijziging aan de software-architectuur afhankelijk is van de grootte van het informatiesysteem. Combinatorische effecten vertegenwoordigen een bijzonder nadelig soort koppeling in softwarearchitectuur. Dergelijke effecten zorgen ervoor dat informatiesystemen tijdens hun levenscyclus moeilijker onderhoudbaar worden naarmate de tijd vordert. Uiteindelijk kunnen ze niet meer kosteneffectief worden aangepast en moeten dan vervangen worden door een informatiesysteem met gelijkwaardige functionaliteit. Combinatorische effecten hebben verder een negatieve invloed op andere kwaliteitsfactoren zoals herbruikbaarheid en testbaarheid.
Momenteel krijgt duurzame software, zonder combinatorische effecten en technische schuld, meer en meer aandacht, waarbij de focus verschuift van oplevering van het project, naar langdurige exploitatie, zodat gesproken kan worden van een nieuw derde tijdperk.

In het eerste IT-tijdperk zijn de technische mogelijkheden van mainframe, mini- en desktop- computer bepalend voor de IT-oplossing. De nadruk ligt in dit tijdperk op het ontwikkelen van apparatuur en software die

werkt. Dat was niet vanzelfsprekend. Er werden wel 'moderne' computers op de markt gebracht maar die gebruikten vrijwel alle performance om hun eigen besturingssysteem te laten functioneren. IT-leveranciers en opdrachtgevers waren opgelucht wanneer hun informatiesystemen in de praktijk werkbaar bleken. Dat betrof toen vooral statische systemen met een stabiel gedrag van input en output, en zo min mogelijk invloeden van buitenaf. Denk bijvoorbeeld aan een systeem met hypothecaire berekeningen. Het eerste tijdperk was leerzaam voor iedereen. Tekortkomingen werden opgelost in volgende generaties systemen en methodieken.

In het tweede tijdperk lag de nadruk nog steeds op het ontwikkelen en implementeren van informatiesystemen, maar de kwaliteit van die systemen was aanzienlijk beter.
Dat kwam niet zo goed uit de verf. Weliswaar bood nieuwe techniek nieuwe mogelijkheden, maar de eisen en verwachtingen van opdrachtgevers namen ook toe. Verder werden meer risicovolle projecten gestart die impact hadden op de bedrijfsvoering, zoals voorraadbeheer, logistiek en marketing. Omdat deze bedrijfsvoering kon veranderen, veranderde de rol van de IT'er van expert naar analist en werden parameters geplaatst in de informatiesystemen. Enterprise Resource Planningsoftware (ERP) in de jaren tachtig is daarvan een goed voorbeeld. Daarmee kan de input, output en verwerking van het systeem via parameters worden aangepast zonder de programmatuur aan te passen. Ook werd duidelijk dat het niet alleen de IT was die moeilijkheden veroorzaakte. De kern van alle problemen zat (en zit nog steeds) vooral in de samenwerking tussen mensen, afdelingen en organisaties. Wel is het zo dat omvangrijke IT-projecten en -toepassingen veel 'last' hebben van die moeizame samenwerking, en daardoor een grote kans op falen hebben.
Softwareprojecten werden in dit tweede tijdperk groter, veel complexer en naarmate de omvang van projecten toenam, nam ook de behoefte aan meer beroepsspecialisatie toe. Kwaliteitszorg werd een afdeling. Het analyseren van informatie, opstellen van vereisten en specificaties werd een specialisatie. Het was duidelijk dat projecten een hoger niveau

van governance nodig hadden, wat ook hogere kosten voor de projecten met zich meebracht. IT-management keek naar de bouwsector om te zien hoe men erin slaagde om zowel wolkenkrabbers als normale woningen te creëren. Projecten werden uitgesplitst in fasen, zoals een waterval, en werden gedocumenteerd met een plan. In deze periode dacht IT-management dat bijna elk project een professionele, gecertificeerde projectmanager zou moeten hebben. Het resultaat was een sterke stijging van de vraag naar deze beroepsgroep.
Toen duidelijk werd dat de gestructureerde watervalaanpak, met hoge overheadkosten en lange levertijden, geen betere resultaten en meer waarde leverde, begon Agile. Projecten werden opgesplitst in kleine deliverables, die hogere succespercentages, meer waarde en kwalitatief betere software opleverden. Teams werden zelfsturend, producteigenaren kwamen dichter bij de teams te staan en een snelle levering van producten beheerste de dagelijkse werkzaamheden van teams. Scrum werd de koning van Agile, met projectverantwoordelijkheden voor de producteigenaar en de Scrum-master. Projectmanagers probeerden een plek te vinden om waarde aan het agile-proces toe te voegen. Ontwikkelingen in de techniek, zoals tools voor het maken en beheren van kleine projecten (nu bekend als microservices), hebben de rol van de projectmanager verder beperkt. Scrum werd ook geïntegreerd in DevOps om een naadloze levering te creëren tussen softwareontwikkeling, implementatie en services die worden geleverd via de softwarepijplijn.

5.2 Go with the flow, de komst van een 3e projectloos tijdperk

Projectmanagers zullen altijd een plaats hebben, maar misschien niet in de ontwikkeling van software. Momenteel zien we het begin van wat Standish Group[3] de Flow Periode noemt.
Deze Flow periode wordt gekenmerkt door de ontwikkeling van software in een continue procespijplijn. Niet alleen met zogenaamde 'software-ontwikkelstraten, maar ook 'specificatiestraten'. Wat Scrum is voor Agile, is DevOps voor Flow. De Flow periode kent geen projectbudgetten,

3 http://blog.standishgroup.com/post/88

projectplannen, projectmanagers of Scrum-masters. Er zal wel een budget zijn om de pijplijn te beheren, maar de huidige projectoverheadkosten zullen drastisch dalen. Dit wordt bereikt door het verminderen en elimineren van de meeste van de huidige projectmanagementactiviteiten. Werk zal aan het begin van de pijplijn gespecificeerd worden en volledig bruikbaar uit de pijplijn komen. Verandering zal continu plaatsvinden, maar in kleine stappen die alles actueel, waardevol en acceptabel houden voor gebruikers.

Jim Johnson van de Standish Group citeert vaak een uitspraak van Steven Wright: "Als je ouders geen kinderen hadden, dan is de kans groot dat jij ze ook niet krijgt." De uitspraak is ook van toepassing op projecten. Als u geen projecten heeft, zult u geen mislukte of uitgedaagde projecten hebben. U hoeft de kosten van het project niet te schatten. U krijgt geen projectmanager toegewezen aan het project. Je zult geen conflicten hebben over budgetten.

Dit betekent natuurlijk niet dat watervalprojecten verdwijnen. Ook agile-projecten blijven bestaan. We zullen echter zien dat organisaties de komende twintig jaar overstappen van projectgebaseerde softwareontwikkeling naar een pijplijn.

Dit betekent wel dat er een nieuwe generatie ontwikkelaars zal komen om die pijplijn te voeden, beheren en te laten produceren. We zullen Flow verder moeten definiëren en moeten kijken welke mensen, vaardigheden en principes nodig zijn voor governance van de software-flow. Vertrek vanuit een functioneel probleem - dat via een uniek project resulteert in een onvergelijkbaar systeem qua inhoud en omvang – maakt plaats voor Flow waar men vertrekt vanuit de constructie van de organisatie en de kleinst mogelijke software-elementen.

In de toekomst zijn er geen unieke grote projecten meer, maar kleine multidisciplinaire teams van materiedeskundigen en ontwikkelaars. Zij plegen continu onderhoud om de organisatie en systemen te laten aansluiten op de steeds sneller veranderende omgeving. De verwachte ontwikkelingen in het allesomvattende internet, in combinatie met data sciences betekenen dat informatiesystemen continu aangepast worden,

als het ware in een 'flow' of 'pipeline'. Vergelijkbaar met water dat continu via kraan geleverd wordt in plaats van keer op keer via flesjes en kratjes. In dit derde tijdperk komt de nadruk allereerst te liggen op het flexibel en economisch gebruik van informatiesystemen. Die moeten zich in kwaliteit, aanpasbaarheid en levensduur onderscheiden van hun voorgangers. De essentie is: al in het ontwerp van organisaties en systemen rekening houden met toekomstige aanpassingen, zonder deze vooraf te kennen. Dit kan door een hoge mate van aanpasbaarheid zodat enerzijds de impact van technologie en marktomstandigheden snel en makkelijk geïmplementeerd kunnen worden en anderzijds door de organisatie en de producten met 'strategische lenigheid' blijvend te innoveren.

Processen en systemen snel aanpassen en onderwijl de organisatie lenig houden raakt zowel governance-, architectuur- en ontwerp-aspecten. Organisaties kunnen alleen evolueren als in de kleinste onderdelen, waaronder de bedrijfsprocessen en de software, geen rimpeleffecten meer voorkomen. Voor de oplossing wordt gekeken naar andere wetenschappen en industrieën. We zijn tegenwoordig in staat om tegen minimale kosten, miljarden transistors in een laptop te stoppen en deze uit te breiden op het niveau van processoren. Dat is mogelijk omdat elke transistor dezelfde eigenschappen heeft. Je kunt ze dus aan elkaar klikken zonder dat er met elke uitbreiding een combinatorisch rimpeleffect optreedt. Als processen en software op die manier ontwikkeld worden, kunnen aanpassingen die uit verandering van de omgeving voortvloeien eenvoudig en snel worden doorgevoerd. Op die manier hebben veranderingen minder ingrijpende gevolgen.

Hoe ziet de toekomst eruit? Net als in andere volwassen industrieën, moet de vrijheid van individuele ontwerpers en ontwikkelaars worden ingeperkt. Zij moeten op uniforme wijze werken met genormaliseerde proces- en software-constructies, die na opname in het totaalontwerp – net zoals de printplaten met transistors – gerobotiseerd worden geprogrammeerd.

Literatuur

- Bronsgeest, W. (2016), Meer vorm dan inhoud, Enschede, Gildeprint.
- Dietz, J.L.G., J.B.F. Mulder, Enterprise Ontology, A human-centric approach to understanding organization, 2019, Springer, Berlin.
- Dijk, A. van (2013), Success and failure factors in ICT projects - with Dutch case studies as examples, School of Engineering and Information Sciences, Middlesex University.
- Gaikema, M, Donkersloot, Johnson, Mulder (2019), Increase the success of Governmental IT-projects, Systemics, Cybernetics and Informatics, ISSN: 1690-4524, volume 17.
- Groen, N. (2015), De bodemloze put. Maastricht, Mosae verbo.
- Mannaert, H, J. Verelst, P. De Bruyn, Normalized Systems Theory, From Foundations for Evolvable Software Toward a General Theory for Evolvable Design, 2018, Koppa, België.
- Mulder, J.B.F., J. Johnson, (2016), The Next Step of IT Project Research in Practice: The CHAOS University System, Systemics, Cybernetics and Informatics, USA.
- The Standish Group, (1994), CHAOS REPORT 1994. Boston, USA.
- Tweede Kamer der Staten-Generaal, (2014), Eindrapport - Parlementair onderzoek naar ICT-projecten, Den Haag: Nederlandse Overheid. Retrieved from http://www.tweedekamer.nl/images/33326-5-Eindrapport_181-239826.pdf.

537865-L-bw-KNVI
Processed on: 8-11-2019

Reflectie: Wat is succes?

Door: Jos van Dijk

De titel van Hans Mulder 'Groot projectsucces is gevaarlijk' roept de vraag op 'Wat is succes?'.
Succes betekent 'welslagen' of 'goede uitkomst', in vroegere tijden ook 'overwinning'. Oorspronkelijk afgeleid van het Latijnse woord 'successus' wat ook opvolging betekend. We herkennen dat in het woord successierechten en het Duitse woord voor succes 'erfolge'.
Daarmee is succes min of meer gedefinieerd. De vraag is vervolgens wat betekent welslagen of wat is een goede uitkomst? Dat zijn al snel morele vragen. Want wat is goed? En voor wie? En is het ook goed voor de overwonnene of opgevolgde?
Achterberg en Vriens definiëren organisaties als 'sociale experimenten om te blijven voortbestaan'. In organisaties zijn twee krachten actief. Enerzijds is er de noodzaak om relevant te blijven, wat vraagt om verandering en aanpassing. Anderzijds is er de behoefte aan continuïteit en stabiliteit. Een behoefte die veelal remmend werkt op verandering.
Het begrip 'relevant' gaat verder dan het leveren van een gevraagd product of dienst. Het gaat ook over betekenisvol werk, de bijdrage aan de samenleving, het milieu en deugdzaam handelen. Hoe weeg je dit tegen elkaar af, wat sluit je in, wat sluit je uit? De vraag is vervolgens 'hoe ver reikt relevantie als die ten koste gaat van de continuïteit?'.
Een paar voorbeelden: In hoeverre mag flexibilisering van de arbeid te koste gaan van zekerheid voor de medewerkers? Wat zijn de effecten van bonusregelingen voor de relevantie op lange termijn? Als het juridisch klopt, deugt het dan ook?
Achterberg en Vriens gebruiken hiervoor de ideaaltypen 'arme' en 'rijke' organisaties. Arme organisaties sluiten veel uit en houden weinig rekening met andere belangen dan het eigen financiële gewin. Rijke organisaties maken afwegingen over veel belangen en waarden. Governance gaat allereerst over de waarden waarmee wordt bestuurd. Dat geldt voor de wijze van besturen alsook voor de beoogde resultaten van de besturing. Hoe rijk is jouw organisatie?