10 GEBODEN VOOR SYS TEEMONTWIKKELING

Zoeken naar wat minimaal nodig is in plaats van kijken naar wat maximaal kan

door: WOUTER KELLER

Vele IT-ontwikkeltrajecten mislukken. Men wil een nieuw systeem maken of kopen aangezien het oude systeem verouderd is en moet worden vervangen. Meestal grijpt men dan de gelegenheid aan om alles te vernieuwen en meteen extra functionaliteiten te wensen. Resultaat: ontwikkeltrajecten die uit planning en/of begroting lopen of die gewoon mislukken. Mijn advies is daarom om vooral de wensen te beperken en op zoek te gaan naar dat wat minimaal nodig is om het oude systeem te kunnen vervangen. Dit in plaats van te kijken naar wat maximaal kan. En bij voorkeur op basis van configuratie van een standaardpakket in plaats van ontwikkeling van een nieuw systeem. Dat leidt tot de volgende tien geboden.

1 Zoek de minimumfunctionaliteit

Gebruikers struikelen vaak over elkaar om extra eisen in te leveren voor het nieuwe systeem. Levensgevaarlijk! Leidt tot IT-systemen van het type waterhoofd. De projectleider moet daarom voortdurend sturen op de vraag: wat hebben we minimaal nodig om met een eerste versie in productie te kunnen gaan. Of het vorige systeem te vervangen. Mini is troef, kies voor simpel. Beperk daarom je ambities tot een project van bijvoorbeeld maximaal een jaar. Na die eerste versie kun je altijd een nieuw project starten voor versie twee.

2 Configuur in plaats van codeer

Voor de meeste problemen zijn er goede standaardpakketten met ruime configuratiemogelijkheden om aan de wensen van de klant te voldoen. Maatwerk door code (Java et cetera) te krassen moet zoveel mogelijk worden voorkomen. Laat staan dat er een geheel nieuw systeem moet worden ontwikkeld (gecodeerd). Geeft allemaal later veel onderhoud en een inflexibel systeem met grote afhankelijkheid van de ontwikkelaars. Beperk je daarom tot configuratie van een standaardpakket met hulp van een externe consultant. Het mooiste is als je dat als klant na de sprints zelf kunt aanpassen, zonder hulp van de leverancier.

3 Gebruik sprints in een box

Werk met korte 'sprints' (à la Scrum) van een paar weken met een vaste tijd/geld-'box' (dus qua doorlooptijd en ontwikkelbudget) en variabele functionaliteit. Vooraf vastleggen van functionaliteit én een box werkt zelden, dan loopt alles uit of kost meer. Werk daarom met een vaste tijd/budget-box per sprint en met variabele functionaliteit op basis van eisen en wensen. Definieer de minimumrequirements (zie regel 1) als eisen per sprint: dit moet de sprint sowieso opleveren. Maak een kort lijstje van aanvullende wensen per sprint en laat de projectgroep (met key-users en consultant) hierop sturen met een bonus per sprint voor de consultant als (bijna) alles lukt. Splits grote programma's op in onafhankelijke (en eventueel parallelle) projecten zodat ieder project binnen een jaar in productie kan in een paar sprints.

4 Weg met het functioneel ontwerp

Vroeger (bij de watervalmethode) werd de klant eerst een functioneel ontwerp (FO) van tientallen of honderden pagina's voorgelegd, wat liefst met bloed moest worden ondertekend door de klant en daarna in beton werd gegoten. Dat is zinloos. Klanten kunnen dergelijke FO's zelden goed beoordelen. Hier geldt slechts de regel: zien is geloven. Werk daarom incrementeel in de sprints met key-users uit de eigen organisatie die voortdurend kunnen bijsturen. En met korte lijstjes eisen en wensen. Een paar A4'tjes is meer dan genoeg als (eerste) ontwerp. Zorg voor een tastbaar (productierijp) resultaat per sprint via demo's en prototype-sessies. Stuur tussen de sprints de eisen/wensenlijstjes voortdurend bij op basis van voortschrijdend inzicht.

5 Armoe helpt

Hou het ontwikkelbudget (en doorlooptijd) beperkt. Rijkdom is vragen om moeilijkheden bij IT-projecten. Rijkdom en ambitie vormen helemaal een groot risico. Pas dus op voor ambitieuze vergezichten en lange wensenlijsten. Zeg daar 'nee' tegen. Daarbij helpt armoede: 'dat staat het budget niet toe.' Zet eventueel alle luxewensen (verwarmde buitenspiegels) op een lijst en laat die minimaal een paar maanden liggen of vergeet die (geen geld/tijd voor). Streef niet bij voorbaat naar draagvlak maar naar snelle productie van een eerste versie. Ook daarbij geldt: zien is geloven. Evalueer dan en maak eventueel een versie 2.0. Maar zeg 'nee' tegen die verwarmde buitenspiegels.

6 De stuurgroep als sponsor

De stuurgroep is de sponsor van het project, zowel in woorden (reclame) als daden (geld). De stuurgroep zorgt voor de resources voor de projectgroep. Dus uren voor externen (consultants) en uren voor key-users. Liefst meer interne uren (van goede, betrokken key-users) dan externe uren. Stuurgroepen moeten niet te veel zelf gaan sturen. De projectgroep moet voor iedere sprint een advies geven, dat de stuurgroep moet bekrachtigen. Dit advies bestaat uit een beknopt lijstje eisen en wensen voor de volgende sprint en eventueel de omvang van de bonus voor de vorige sprint. Daarnaast moet de stuurgroep helpen 'nee' te zeggen (zie regel 5). Goede maar strenge sponsors zijn even belangrijk als key-users.

7 Houd leveranciers kort

Leveranciers hebben eigen belangen, zoals meer uurtjes, meer factuurtjes. Leveranciers hebben dus belang bij uitloop. Zorg daarom voor een onafhankelijke projectleider die de leverancier kort houdt met vaste prijzen per sprint. Geef de consultant een bonus als (bijna) alle wensen in de sprint (zie regel 3) zijn gerealiseerd. Dat houdt de leverancier scherp. Accepteer alleen consultants die de harde kant (techniek) en de zachte kant (proces) kunnen combineren. Stuur slechte consultants binnen een maand naar huis. Pas op voor 'moeilijkmakers' zoals methodologen, architecten, Scrum-fundamentalisten en andere deskundigen: de beste oplossing is vaak de eenvoudigste, de beste adviseurs zijn vaak je eigen key-users.

8 Sluit lock-in uit

Zorg als klant voor eigen intellectueel eigendom (IE) van de ontwikkelde configuraties of code en zorg dat de configuratiebestanden, code en/of tools tijdig worden geleverd en getoetst, zodat de leverancier op ieder moment kan worden vervangen (of denkt dat dat kan gebeuren). Maak je zoveel mogelijk onafhankelijk van de huidige leverancier door het (configuratie)kunstje zelf te leren, of te zorgen dat het bij concurrenten kan worden uitgezet. Laat geen software ontwikkelen als het ook door configuratie van een standaardpakket kan (zie regel 2). Leg vooraf de exitregeling vast (inclusief migratie/conversie naar de nieuwe leverancier) mocht je naar een andere leverancier willen overstappen.

9 Migreer en integreer zo min mogelijk, test zoveel mogelijk

Migratie kan een rot klus zijn, waar geen eind aan komt en waar alles fout gaat. Hetzelfde geldt voor integratie, waarbij koppelingen tussen systemen voor veel ellende zorgen. Vraag je af wat je echt minimaal moet migreren/converteren naar het nieuwe systeem en welke koppelingen niet voorlopig handmatig kunnen. Het handmatig invoeren van oude maar warme content in het nieuwe systeem is een prima opleiding en systeemtest. Houd het oude systeem standby voor oude maar koude content. Ruim in iedere sprint voldoende tijd in voor testen (door de key-users) en bijbehorende aanpassingen (door consultant) per sprint. Geen acceptatie per sprint zonder grondig testen.

10 Hou training/documentatie kort en krachtig

Als het systeem weken training nodig heeft van de toekomstige gebruikers zit er iets fout. Accepteer geen gebruiksonvriendelijke systemen. Overleg met de key-users uit de organisatie hoe zaken het beste kunnen worden voorgesteld in het nieuwe systeem. Daarvoor zijn de sprints met demo's juist bedoeld. Korte trainingen (een of twee dagdelen) van de gebruikers vlak voor de productiegang, met herhalingen/opfris later, werkt beter dan lange cursussen maanden voor productie die iedereen vergeet. Werk bij voorkeur met train-de-trainer via de key-users die tevens betrokken zijn bij de sprints en het testen van het systeem. Hoe meer kennis de organisatie heeft, hoe onafhankelijker je bent van de leverancier.

Kortom, wil je slagen binnen tijd en budget (en dat is niet onmogelijk) dan moet je eerst uitgaan van het minimaal noodzakelijke. Pas daarna is er ruimte voor verwarmde buitenspiegels. Dat is de kunst van systeemontwikkeling.

Wouter Keller (wkeller@argitek.nl) is directeur-oprichter van het IT-adviesbureau M&I/Argitek en was tot 2001 lid RvB van het Centraal Bureau voor de Statistiek (CBS), onder andere belast met research en IT.

ag

Voor reacties en nieuwe bijdragen van IT-experts: Henk Ester, 070 3780397 h.ester@sdu.nl