Handboek projectmanagement

versie 1.2 - mei 2008

Wouter Baars

Advies:
Henk Harmsen
Rutger Kramer
Laurents Sesink
Joris van Zundert

http://www.projectmanagement-training.nl

This work is licensed under the Creative Commons Attribution-Non-Commercial-Share-Alike 2.5 License.

To view a copy of this license, visit:
http://creativecommons.org/licenses/by-nc-sa/2.5/

or send a letter to:
Creative Commons
543 Howard Street – 5th floor
San Francisco, CA 94105
USA

Inhoudsopgave

Voorwoord
Inleiding

1 De zes fasen van projectmanagement
2 Besturen van een project
3 Projectrapportage
4 De koopman en de politicus
5 Waterval versus cyclisch projectmanagement
6 DANS werkwijze voor software ontwikkeling
7 Programmamanagement

Bijlagen
1: Top 11 van de grootste vertragers van ICT projecten
2: Rollen binnen een project
3: Nuttige hulpmiddelen voor projectmanagement
4: Licentie van dit handboek
5: Over DANS en de makers van dit handboek
6: Voorbeeld actie- en besluitenlijst
7: Voorbeeld issuelog
8: Voorbeeld risicolog
9: Voorbeeld gespreksverslag
10: Voorbeeld projectplan
11: Voorbeeld begroting
12: Voorbeeld financiële verantwoording

Literatuur en internet bronnen

Voorwoord bij versie 1.3

Voorwoord

Dit handboek is ontstaan tijdens de invoering van projectmanagement bij DANS (KNAW). Bij de invoering van projectmanagement binnen die organisaties en latere andere organisaties, bleek al snel dat de traditionele 'watervalmethode' van projectmanagement maar voor een deel van hun projecten geschikt was. Met name voor software projecten en creatieve projecten bleek deze methodiek van projectmanagement te rigide. Een formele projectmanagement methodiek als Prince 2 viel al snel af als optie omdat die veel te zwaar is voor de relatief kleine tot middelgrote projecten die deze organisaties doen. De agile projectmanagement methoden RUP, DSDM blijkt ook te formeel en ingewikkeld voor veel organisaties. De zoektocht was dus naar een informele structuur voor projectmanagement die te gebruiken is bij kleine tot middelgrote projecten. Uiteindelijk bleek extreme programming geschikt, maar dan wel met enkele toevoegingen. In dit handboek wordt deze aanpak beschreven, naast de traditionele manier van projectmanagement.

In dit handboek wordt uit de doeken gedaan hoe projecten met een waterval aanpak of met een agile aanpak gerealiseerd kunnen worden.

Inleiding

Het handboek voor projectmanagement is bedoeld voor een ieder die te maken heeft of te maken krijgt met projecten bij of met DANS. De tekst is echter zo opgesteld dat andere organisaties die projectmatig werken er ook gebruik van kunnen maken. Dit geldt met name voor organisaties in de non-profit sector.
Het boek is opgebouwd uit een aantal delen. Het eerste deel, hoofdstuk 1 tot en met hoofdstuk 4 is een uiteenzetting van de basis van projectmanagement. Dat deel behandelt de theorie van de watervalmethode die geschikt is voor de meeste projecten.

Het tweede deel van dit boek, vanaf hoofdstuk 5, behandelt zogenaamde cyclische vormen van projectmanagement die beter geschikt zijn voor ICT-gerelateerde projecten. Deze methoden zijn met name geschikt voor creatieve ICT-projecten zoals softwareontwikkeling. In het voorlaatste hoofdstuk komt de werkwijze van DANS bij softwareontwikkeling aan de orde. Deze methode is een combinatie van elementen uit de watervalmethode en cyclische methoden.

Tenslotte wordt in het laatste hoofdstuk besproken hoe organisaties de dynamiek van het uitvoeren van meerdere projecten tegelijkertijd kunnen beheersen. De belangrijkste knelpunten worden besproken en de daarbij behorende strategieën voor de aanpak van deze knelpunten.

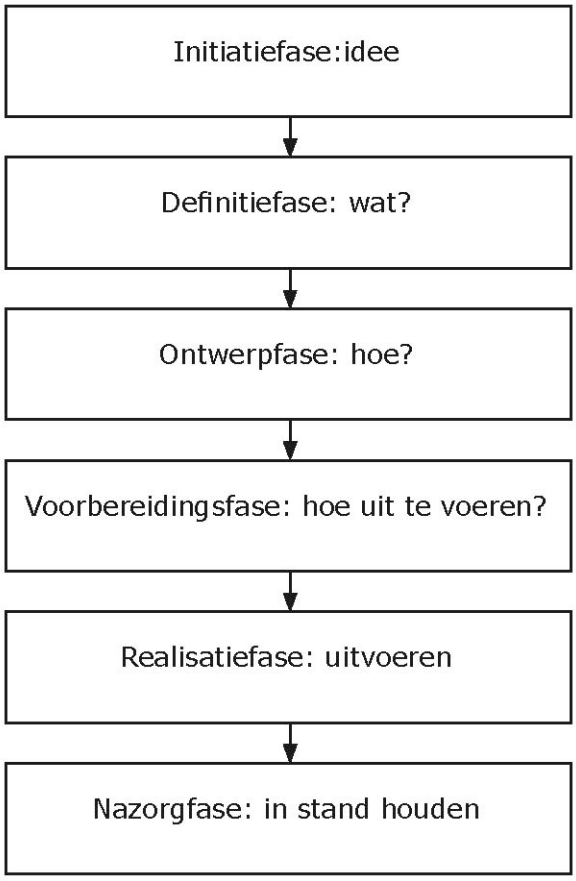
Bij dit handboek hoort een aantal standaarddocumenten die gebruikt kunnen worden om een project te sturen. Daarnaast zijn er verwijzingen opgenomen naar (open source) projectmanagementinstrumenten van derden. Verder is er een literatuurlijst voor diegene die zich verder wil verdiepen in het brede veld van projectmanagement.

1. De zes fasen van projectmanagement

In dit hoofdstuk wordt de traditionele wijze van projectmanagement geschetst. Het model dat hier besproken wordt, vormt de basis voor veel projectmanagementmethodes. In latere hoofdstukken zal nader ingegaan worden op een model dat speciaal geschikt is voor ICT gerelateerde projecten.

Om een project in zo goed mogelijke banen te leiden, is het nuttig het project op te delen in fasen. Door het faseren van het project, wordt het totale werk opgedeeld in kleinere delen, die daardoor makkelijker te overzien zijn. Hieronder wordt een faseringsmodel gegeven dat in de praktijk zijn nut heeft bewezen. Het werkt met zes fases:

1. Initiatiefase
2. Definitiefase
3. Ontwerpfase
4. Voorbereidingsfase
5. Realisatiefase
6. Nazorgfase



Figuur 1: Projectmanagement in zes fasen met per fase het centrale thema

Initiatiefase

De initiatiefase is het begin van het project. In deze fase wordt een idee voor een project nader onderzocht en uitgewerkt. Doel van deze fase is te onderzoeken of het project wel haalbaar is. Verder wordt er gekeken wie het project zou kunnen uitvoeren, welke partij(en) betrokken zouden moeten zijn bij het project en of er voldoende draagvlak is voor het project (bij betrokkenen).

De (beoogde) projectleider maakt in deze fase een projectvoorstel waarin hij bovenstaande zaken beschrijft. Een businessplan of subsidieverzoeken zijn voorbeelden van dergelijke projectvoorstellen. Diegenen die het project sponsoren zullen het projectvoorstel beoordelen en bij goedkeuring de financiering verzorgen. Vanaf die goedkeuring is het project officieel gestart.

De vragen die beantwoord moeten worden in de initiatiefase zijn:

- Waarom dit project?
- Is het project haalbaar?
- Wie zijn eventuele partners in dit project
- Wat moet het resultaat zijn?
- Waar liggen de grenzen van het project (wat hoort er niet meer bij het project?

Een belangrijke kwaliteit van een projectleider is dat hij goed 'nee' kan zeggen. Als mensen eenmaal enthousiast worden voor een project heeft dat de neiging steeds groter te worden. 'Als we nu toch bezig zijn, dan kunnen we net zo goed er ook voor zorgen dat...', is dan de gedachtengang. Een project waarbij men steeds meer wil en dat zich steeds uitbreidt, zal vrijwel zeker uit de planning lopen en waarschijnlijk nooit het doel bereiken.

In de initiatiefase gaan de projectpartners een (tijdelijke) relatie met elkaar aan. Om te voorkomen dat er verkeerde verwachtingen over de resultaten van een project ontstaan is het verstandig om expliciet af te spreken wat voor een soort project er gestart wordt:

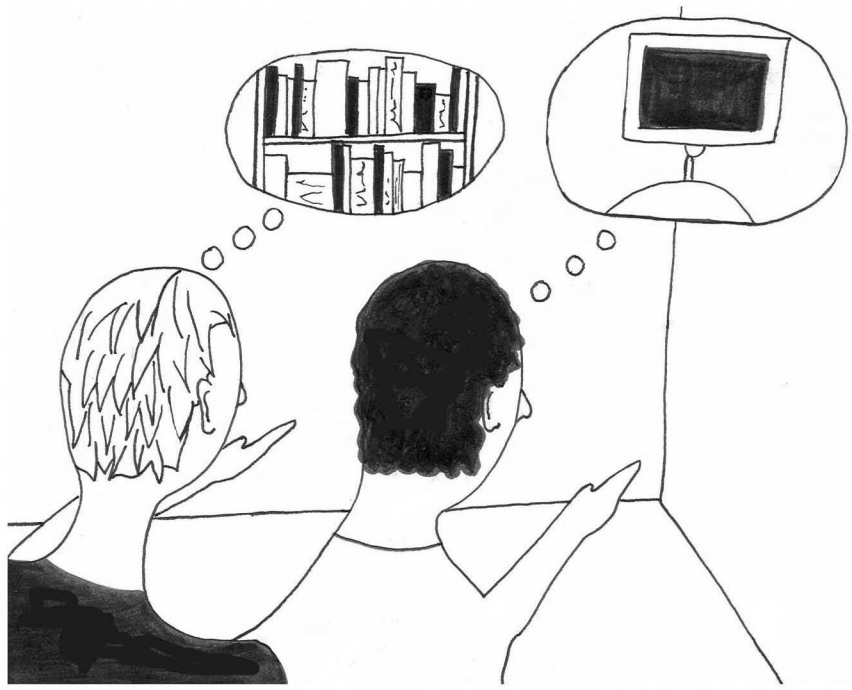
- een research & development project
- een project om een prototype of 'proof of concept' te leveren
- een project dat een werkend product op gaat leveren

De keuze voor een project bepaald in hoge mate het resultaat van een project. Een research & development project levert bijvoorbeeld een rapport op waarin onderzocht wordt of een toepassing technisch haalbaar is. Een project waarin een prototype ontwikkeld wordt levert alle functionaliteit van een applicatie maar hoeft nog niet geschikt te zijn voor bijvoorbeeld honderden gebruikers. Een project wat een werkend product oplevert dient ook rekening te houden met het regelen van onderhoud, handleidingen en het functionele beheer van een applicatie.

Veel misverstanden en/of conflicten ontstaan omdat de partijen over en weer dit niet helder voor ogen hebben. De klant verwacht een werkend product terwijl het projectteam denkt een prototype te moeten leveren. De financier denkt dat het project een werkend stuk software oplevert, terwijl het projectteam eerst moet onderzoeken of het überhaupt technisch kan.

Definitiefase

Nadat het projectplan, dat gemaakt is in de initiatiefase, is goedgekeurd, komt het project in de tweede fase: de definitiefase. In deze fase worden de eisen en wensen die aan een projectresultaat gesteld worden zo goed en compleet mogelijk bepaald. Het gaat erom de verwachtingen van betrokken partijen boven water te krijgen over wat men denkt dat het resultaat moet zijn. Hoeveel bestanden moeten er gearchiveerd gaan worden? Moet de metadata voldoen aan het Data Documentation Initiative (DDI) formaat of volstaat het Dublin Core (DC) formaat? Mogen bestanden in hun originele formaat gedeponeerd worden of worden slechts 'Preferred Standards' geaccepteerd? Moet de depositor van een dataset zorg dragen voor een correcte verwerking van een dataset in het archief of is dit de verantwoordelijkheid van een archivaris? Welke garanties worden gegeven op het projectresultaat? Enzovoort.



"Na de brainstorm waren alle betrokken leden van het projectteam 'nieuw archief' het eens over de wenselijke uitkomst."

Figuur 2: Verwachtingen van een project (Illustratie: Rachèl Harmsen)

Het is belangrijk om in een zo vroeg mogelijk stadium de eisen en wensen boven tafel te krijgen. Wijnen (Wijnen, 2004) onderscheidt een aantal categorieën projecteisen die kunnen dienen als geheugensteun:

- Randvoorwaarden
- Functionele eisen
- Operationele eisen
- Ontwerpbeperkingen

Randvoorwaarden vormen de context waarbinnen het project uitgevoerd moet worden. Voorbeelden hiervan zijn wetgeving, arbo-eisen, keurmerkeisen en dergelijke. Deze eisen zijn niet te beïnvloeden vanuit het project. *Functionele eisen* zijn eisen die gaan over hoe goed het projectresultaat moet zijn. Bijvoorbeeld hoe energiezuinig een nieuwe auto moet zijn, of hoeveel kamers een nieuw gebouw moet hebben. *Operationele eisen* gaan over de eisen aan het gebruik van het projectresultaat. Bijvoorbeeld dat na het realiseren van het softwareproject, het aantal storingen met 90% afneemt. *Ontwerpbeperkingen* tenslotte zijn eisen die te maken hebben met de realisatie van het project zelf. Bijvoorbeeld dat er in het project niet gewerkt wordt met giftige materialen, of niet gewerkt wordt met internationale leveranciers waarvan niet duidelijk is of ze kinderarbeid toepassen.

Bij de ontwikkeling van een webapplicatie voor een consortium van grote organisaties was men vergeten in de definitiefase af te spreken welke browser ondersteund zou worden door de applicatie. Het consortium ging er vanuit dat dit Microsoft Explorer zou zijn, omdat die door 'iedereen' gebruikt wordt. De programmeurs maakten de applicatie in Firefox, omdat zij daar zelf mee werkten en omdat die een aantal functies had, die erg handig waren tijdens het ontwikkelen. Omdat de meeste websites gemaakt voor Firefox er ook wel goed uitzien in Explorer, viel het niet direct op, maar tegen het einde van het project begon de klant te klagen dat de website 'er niet goed uitzag'. De programmeurs, die in Firefox keken, vonden natuurlijk dat de website er wel goed uitzag en begrepen de klacht niet. Toen het probleem van de twee browsers duidelijk werd, reageerden de programmeurs defensief:' Kunnen ze geen Firefox installeren, dat is toch gratis'. De organisaties waren echter gebonden aan de bureaucratisch werkende systeembeheerders die om de een of andere reden, wellicht terecht, weigerden Firefox naast Explorer te installeren. Waar de systeembeheerders dat wel wilden, was het een langdurig traject en waren er ook extra kosten voor de tijd die de beheerders ervoor nodig hadden. Uiteindelijk werd er besloten dat de applicatie ook geschikt gemaakt moest worden voor Explorer. Dat was nog behoorlijk wat werk waardoor het project (nog meer) uitliep dan het al deed. Over de extra kosten moest onderhandeld worden. Toen bleek ook nog dat de verschillende organisaties met verschillende versies van Microsoft Explorer werkten...

Het is heel belangrijk dat alle betrokkenen in een project kunnen meedenken in de definitiefase. Vooral ook de eindgebruikers die het projectresultaat gaan gebruiken. Het komt vaak voor dat de eindgebruikers niet de opdrachtgevers zijn voor het project, wat wellicht de reden is dat ze vaak over het hoofd worden gezien. De opdrachtgever, die voor het project betaalt, wordt wel uitgenodigd om mee te denken over eisen en wensen in de definitiefase. Toch is het voor het eindresultaat veel belangrijker de toekomstige gebruikers uit te nodigen. Als uitgangspunt is het een goede gewoonte om in de definitiefase een aantal bijeenkomsten te organiseren met alle betrokkenen bij een project.

Bij de ontwikkeling van een educatieve videogame, werden de gebruikers (jongeren) pas in een laat stadium betrokken bij het project. Toen de game al nagenoeg af was, werd aan een groep jongeren gevraagd de game te testen. De jongeren leken aanvankelijk mild en vriendelijk in hun oordeel. Bij doorvragen bleek echter dat ze de game 'eigenlijk ontzettend saai vonden en zeker niet gingen spelen'.

Waren de jongeren eerder in het project betrokken, dan was de game wellicht een succes geweest. Nu staat de game vrijwel onbezocht op een site op het internet.

Het resultaat van de definitiefase is een eisen- en wensenlijst van de diverse betrokken partijen bij het project. Nu is het natuurlijk zo dat elke eis en wens zijn keerzijde kent. Hoe mooier het project moet worden uitgevoerd, hoe meer tijd en geld het kost. Ook kan het zijn dat bepaalde eisen strijdig zijn. De ontwikkeling van een nieuw kopieerapparaat moet minder milieubelastend zijn, maar moet ook voldoen aan eisen aan brandveiligheid. Daarom moeten er volgens de norm brandvertragende, milieubelastende materialen gebruikt worden. Dat betekent dat er over sommige eisen en wensen onderhandeld moet worden.

Uiteindelijk zal er een definitieve eisen- en wensenlijst boven tafel komen die goedgekeurd moet worden door de beslissers over het project. Met dit goedgekeurde document kan de ontwerpfase starten.

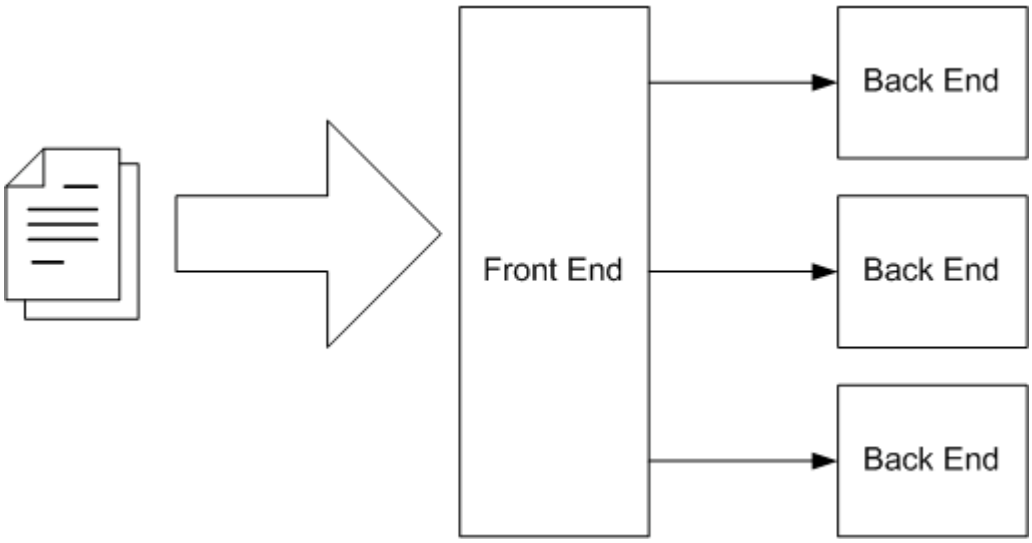
Met het afsluiten van de definitiefase wordt een belangrijk deel van de afspraken tussen klant en projectteam vastgelegd. De eisen- en wensenlijst vormt de richtlijn waar het projectresultaat aan moet gaan voldoen. Daar zal het projectteam op worden afgerekend. Het betekent ook dat de klant nu geen aanvullende eisen of wensen meer mag stellen.

Een deel van een nieuwe expositie van een museum bestond uit een computerinstallatie. Het maken van de installatie was een project. In dit project was er nooit een definitiefase geweest, zodat afspraken tussen het museum en de bouwers van de installatie niet duidelijk waren. Toen de computer van de installatie halverwege de expositie stuk ging, dacht het museum dat het onder de garantie van het project viel. Het projectteam dacht daar echter anders over. Overleg tussen directeuren was nodig om een regeling te treffen.

Ontwerpfase

Met de eisen- en wensenlijst uit de definitiefase kunnen ontwerpkeuzes worden gemaakt. In de ontwerpfase wordt er een ontwerp of worden er enkele ontwerpen gemaakt waarmee men denkt het projectresultaat te kunnen bereiken. Afhankelijk van waar het project over gaat, valt dan te denken aan maquettes, schetsen, flowcharts, site-trees, html-schermontwerpen, prototypen, foto-impressies, UML-schema's en dergelijke.

Uit de gemaakte ontwerpen wordt door de bestuurders van het project een keuze gemaakt voor het definitief te realiseren ontwerp. Dan start de voorbereidingsfase. Ook hier geldt dat een ontwerp als het eenmaal gekozen is, in een later stadium van het project niet meer gewijzigd mag worden.



Figuur 3: Voorbeeld: Globaal Ontwerp van DANS Archief Architectuur

Bij een jonge organisatie waar het er erg informeel aan toe ging, werd de afdeling design geleid door een kunstenaar. Er kon eigenlijk niet echt gesproken worden van een 'afdeling design', meer van een groep samenwerkende designers. Daarbij kwam dat iedereen het veel te druk had, het hoofd van de afdeling inclusief.

Voor een project moesten er ontwerpen gemaakt worden. Die ontwerpen waren erg belangrijk voor het slagen van het project. Een jonge ontwerper die lid was van het projectteam maakte de ontwerpen. Het hoofd van de designafdeling was eindverantwoordelijk voor de ontwerpen. Maar hij kwam nooit op de vergaderingen van het projectteam opdagen, als de ontwerpen besproken werden. De projectleider nodigde hem wel altijd uit en stuurde hem ook e-mails met de schetsen van zijn jonge collega. Er kwam nooit een reactie op. De projectleider en de jonge designer gingen er onterecht vanuit dat het hoofd design de ontwerpen goedkeurde. Vervolgens startte de realisatiefase. Toen het project bijna klaar was, kreeg het hoofd design het resultaat te zien. Hij reageerde woedend en vond dat het helemaal opnieuw moest gebeuren. Alleen was toen het budget nagenoeg op.

Voorbereidingsfase

In de voorbereidingsfase wordt alles geregeld dat nodig is voor de realisatie van het project. Eventuele leveranciers of onderaannemers worden ingeschakeld, een draaiboek wordt gemaakt, materialen en hulpmiddelen worden besteld, instructies aan het personeel worden gegeven, enzovoort. De voorbereidingsfase is klaar als het uitvoeren 'zo' kan starten. Alles moet dus duidelijk zijn voor de uitvoerende partijen.

In sommige, met name wat kleinere projecten, is een formele voorbereidings-fase wellicht overbodig. Het gaat erom dat duidelijk is wat er moet gebeuren in de realisatiefase, door wie en op welk moment.

Realisatiefase

In de realisatiefase wordt het project zichtbaar. In deze fase vindt de bouw van het projectresultaat plaats. Programmeurs zijn aan het coderen, designers zijn bezig beeldmateriaal te ontwikkelen, aannemers zijn aan het bouwen, de reorganisatie wordt daadwerkelijk in gang gezet. Het is de fase dat een project zichtbaar wordt

voor buitenstaanders. Voor hen lijkt het alsof het project nu pas gestart is. Het is de 'doe' fase. In deze fase is het van belang om de vaart er goed in te houden. Bij een project was over het hoofd gezien dat een van de belangrijkste teamleden elk moment vader kon worden en zich ongeveer een maand niet op het werk zou laten zien. Om het team niet stil te laten vallen, werd een externe specialist ingehuurd, die zijn werk overnam. Het team kon verder, maar de externe kracht drukte behoorlijk op de begroting.

Aan het einde van de realisatiefase wordt het resultaat gecontroleerd aan de hand van de eisen- en wensenlijst uit de definitiefase. Het resultaat wordt ook gecontroleerd aan de hand van de ontwerpen. Er wordt bijvoorbeeld getest of de webapplicatie inderdaad Explorer 5 en Firefox 1.0 en hoger ondersteunt. Of dat de afwerking van een gebouw inderdaad heeft plaatsgevonden volgens afspraak. Of dat de gebruikte materialen inderdaad de materialen zijn als bepaald in de definitiefase. Als aan alle eisen en wensen is voldaan en als het resultaat overeenkomt met de ontwerpen dan is deze fase klaar.

Betrokkenen moeten in hun achterhoofd houden dat het vrijwel nooit helemaal zal lukken om het projectresultaat te verkrijgen, dat exact voldoet aan de oorspronkelijke eisen en wensen van de definitiefase. Door onverwachte gebeurtenissen en door voortschrijdend inzicht zal het projectteam tijdens de realisatie van het project soms moeten afwijken van de eisen- en wensenlijst en de ontwerpdocumenten. Dit is een potentiële bron van conflicten, in het bijzonder als er een externe klant is die het projectresultaat heeft besteld. De klant zal zich dan namelijk beroepen op de afspraken die gemaakt zijn in de definitiefase.

De regel is dat na de definitiefase er geen wijzigen meer mogen zijn in de eisen en wensen. Dat geldt ook voor de ontwerpen: na de ontwerpfase mag er niets meer veranderen aan het ontwerp. Als dit toch moet (en dat moet soms), is het van belang dat de projectleider er voor zorgt dat de wijziging zo snel mogelijk besproken wordt met de betrokkenen (met name de beslissers of klant). Daarbij is het van belang dat de besloten verandering goed gedocumenteerd wordt, dit om latere misverstanden te voorkomen. Over de documentatie van het project volgt later meer.

Nazorgfase

Een fase die vaak over het hoofd gezien wordt, maar die erg belangrijk is, is de nazorgfase. In de nazorgfase wordt alles geregeld om het projectresultaat daadwerkelijk te doen landen. Voorbeelden van activiteiten die in de nazorg plaatsvinden zijn handleidingen schrijven, instructie en training voor gebruikers, helpdesk inrichten, onderhoud van het resultaat, evaluatie van het project zelf, schrijven van het projectverslag, een feestje om het bereikte resultaat te vieren, overdracht naar beheerders, opheffen van het projectteam en dergelijke.

De centrale vraag in deze fase is wanneer en waar het project ophoudt. Onder projectleiders wordt er vaak gegrapt dat de eerste 90% van een project snel gaat en dat de laatste 10% nog jaren voortduurt. Het is van belang dat bij het begin van het project al goed wordt nagedacht over waar de grenzen van het project liggen, zodat in de nazorgfase het project bij het bereiken van die grens afgesloten kan worden.

Soms is het voor de betrokkenen niet duidelijk of een project een prototype of een werkend product op zal leveren. Dit speelt met name bij vernieuwende projecten, waar de uitkomst onzeker is. De klant denkt dat hij een werkend product krijgt, terwijl het projectteam denkt bezig te zijn met het bouwen van een prototype. Dit uit zich vooral in de nazorgfase. Zo was er een softwareproject waarbij iets heel nieuws werd geprobeerd. Het was spannend of het allemaal resultaat zou

opleveren. Uiteindelijk was er goed resultaat. Het team leverde een stuk software op dat goed werkte. Althans in een testopstelling.
De klant, die niet zoveel wist van ICT, dacht echter dat hij een werkend product had gekregen. Het had immers op zijn bureaucomputer gewerkt. De software werkte ook wel, maar toen het geïnstalleerd werd bij zijn 50 werknemers, begon het prototype kuren te vertonen en was het af en toe instabiel. De programmeurs konden de software wel repareren, maar hadden daar geen tijd voor omdat ze alweer met een volgend project bezig waren. Bovendien hadden ze helemaal geen zin in het oplappen van iets wat zij zagen als een probeersel. Toen Microsoft een paar maanden later met servicepack 2 voor Windows uitkwam, deed de software het helemaal niet meer. De klant was boos omdat het "product'' wéér niet goed werkte. Omdat het een belangrijke klant was probeerde de projectleider het een en ander bij de programmeurs gedaan te krijgen. Maar de programmeurs verzetten zich. Het repareren van de bugs verstoorde hun nieuwe project te vaak. Bovendien was het in hun ogen maar een prototype. Om het geschikt te maken voor groot gebruik, moest de hele architectuur gewijzigd worden. Ze vroegen zich af wanneer het nu eens klaar zou zijn met de stroom van klachten van de klant.

Kern van het 6-fasenmodel is 'eerst denken en dan doen'. Elke fase heeft zijn eigen werkpakket. Elk werkpakket heeft zijn eigen aspecten waarop geconcentreerd moet worden. Op die manier hoef je dan bijvoorbeeld in de realisatiefase niet meer te discussiëren over wat er nu gemaakt moet worden. Dat is, als het goed is, bepaald in de definitiefase en ontwerpfase. Zie onder andere Wijnen en Kor (Wijnen, 2004, Kor, 2002) voor een uitgebreidere beschrijving van het 6-fasenmodel en de takenpakketten per fase.

2. Besturen van een project

Het hanteren van de zes fasen schept overzicht in een project en maakt het daardoor beter bestuurbaar. Maar wat houdt die besturing van het project dan in? Allereerst heeft een projectleider/projectteam te maken met de volgende ingrediënten:

1. Team
Er is een sprake van een projectteam, een groep mensen die het projectresultaat zal realiseren. De groep bestaat vaak uit mensen met verschillende achtergronden en inbreng van vaardigheden en kennis.

2. Doel
Er is een gewenst projectresultaat oftewel een doel. Na het uitvoeren van het project is er iets gerealiseerd. Er is een nieuw stuk software geschreven, er is een reorganisatie uitgevoerd of er is een brug gebouwd. Het projectdoel kan meer of minder vaag zijn en meer of minder vaststaan. Bij veel projecten moet het doel tijdens het project worden bijgesteld.

3. Beperkte middelen
Er is bij een project sprake van een beperkte hoeveelheid tijd en geld om het resultaat in te bereiken. Geen project is zonder enige mate van tijdsdruk.

4. Onzekerheid (risico's)
Kenmerkend voor een project is dat het niet van tevoren vaststaat of het allemaal gaat lukken. Als het gewenste doel al bereikt wordt, is het daarnaast ook onzeker of dit zal lukken met het beschikbare budget of binnen de gestelde tijd.

Niet ongebruikelijk is dat een project drie maal zo lang duurt en twee maal zoveel kost dan oorspronkelijk begroot. Ook niet ongebruikelijk is dat aan het einde van het project van het oorspronkelijke team nog maar 30 % van de mensen meedoen. Een projectmanager moet op een hoop zaken letten. Toch zijn er maar vijf parameters waarop de projectmanager zijn project stuurt:

- Tijd
- Geld
- Kwaliteit
- Organisatie
- Informatie

Deze vijf parameters worden ook wel eens afgekort met de term GOKIT. Hieronder wordt de GOKIT nader toegelicht. De GOKIT komt terug in projectplannen, de voortgangsbewaking en de projectverantwoording.

Tijd

De tijdsfactor in een project uit zich in de deadlines van taken en de hoeveelheid tijd die taken mogen kosten. De tijd beheersen is ervoor zorgen dat de taken op tijd gedaan zijn.

Tijd in projectplannen:

- Bepalen welke activiteiten in welke fase moeten plaatsvinden
- Inschatten hoe lang de activiteiten gaan duren
- De volgorde bepalen waarin activiteiten moeten plaatsvinden
- Inzet van mensen en materialen in de tijd uitzetten
- De activiteiten uitzetten in de tijd
- De (belangrijkste) deadlines bepalen

Tijd in voortgangsbewaking:

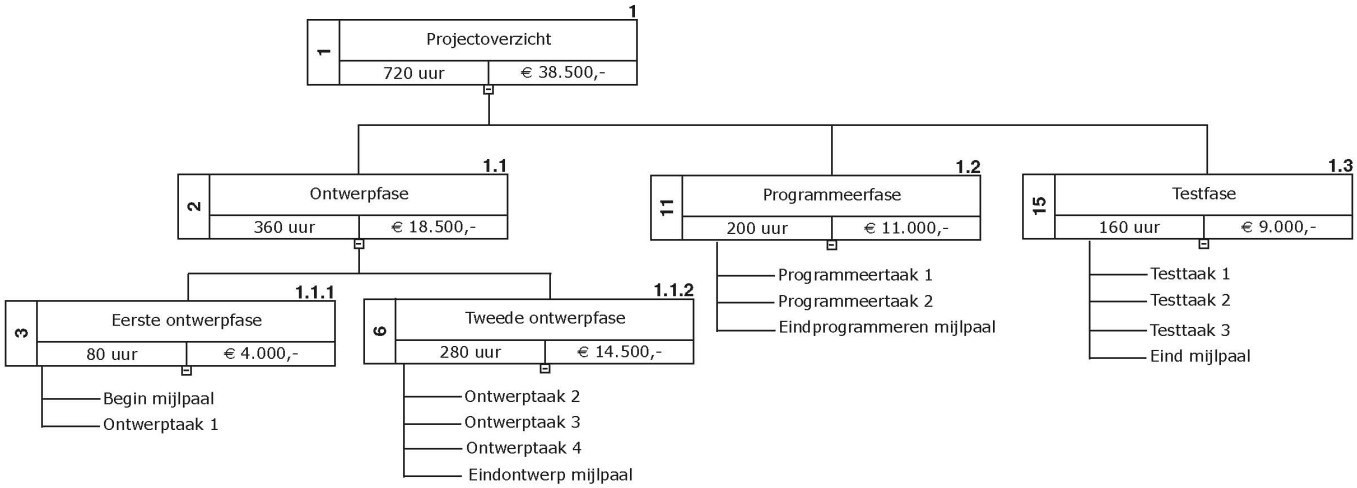
- Voortgang bewaken
- De deadlines bewaken
- Plannen bijstellen

Tijd in de projectverantwoording:

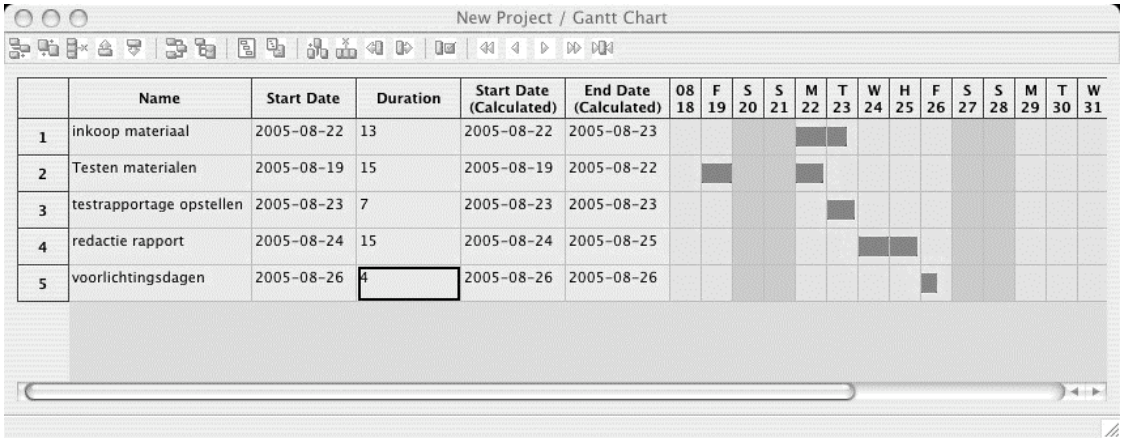
- Rapportage over het gerealiseerde tijdpad
- Analyse en uitleg waarom bepaalde taken veel sneller of langzamer liepen dan verwacht

Een tijdplanning is gebaseerd op een work breakdown structure (WBS). Een WBS is een decompositie van de taken die uitgevoerd moeten worden om het projectresultaat te bereiken. Om tot een tijdplanning te komen, wordt er vervolgens per taak bepaald hoeveel tijd er nodig is, wie de taak uitvoert en wanneer.

Een veelgebruikt hulpmiddel bij het plannen van tijd is een balkenschema of Gantt chart (zie Figuur 5). Er zijn verschillende softwarepakketten verkrijgbaar voor het maken en bijhouden van balkenschema's (zie bijlage 3).



Figuur 4: Een (stuk van een) WBS van een project



Figuur 5: Gantt chart of balkenschema, veel gebruikt voor het plannen van tijd.

Een organisatie groeide hard en deed steeds meer projecten. Doordat het bedrijf het steeds drukker kreeg - er was veel vraag naar hun product - had het personeel het gevoel dat ze van hot naar her moesten rennen om het werk gedaan te krijgen. Het personeel wilde dat er meer mensen werden aangenomen. Het management was daar uit kostenoverwegingen huiverig voor en voerde druk uit op het personeel om harder te werken. Maar hoeveel werk kan een team aan? Deze vraag bleek niet goed te beantwoorden omdat er in deze organisatie geen tijd werd geschreven.

Als een nieuw project werd gestart, was er wel een schatting gemaakt van de hoeveelheid uren die men dacht nodig te hebben, maar nooit werd er tijdens en na het project gecontroleerd of bij de uitvoering van het project daadwerkelijk zoveel uren nodig waren. Projectleiders werden wel aangespoord projecten goed in de hand te houden. De projectleiders protesteerden dat ze zonder urenregistratie geen grip hadden op de projecten. Ze hadden namelijk geen inzicht in de hoeveelheid verbruikte uren bij het uitvoeren van de taken van een project en dan kon er van bijsturen natuurlijk al helemaal geen sprake zijn.

Een projectleider besloot om met zijn team de uren te gaan registreren. Het project bleek uiteindelijk vier keer zoveel uren nodig te hebben dan oorspronkelijk begroot. Na eerst de projectleider op zijn kop te geven dat het project zo veel was uitgelopen, besloot het management toch maar een urenregistratie in te voeren.

Na enige maanden werden knelpunten zichtbaar. Bijna alle projecten bleken veel te krap begroot. Personeel dat voor 100 uren op een project stond, bleek in de praktijk vaak wel tot drie keer zoveel uren te moeten werken aan het project. De transparantie bracht nieuwe dilemma's. Aan de ene kant werd er gesteld dat er inderdaad te weinig personeel was om de projecten goed uit te voeren. Er zou dus meer personeel nodig zijn. De kosten van voldoende personeel waren aanzienlijk. Aan de andere kant kon je ook stellen dat projecten veel te goedkoop (voor te weinig uren) werden verkocht aan klanten. Het management was echter bang geen opdrachten binnen te halen als ze meer uren zouden offreren.

Geld

De geldfactor uit zich in de budgetten van projecten. Het beheersen van geld binnen een project is ervoor zorgen dat de kosten binnen de begroting blijven. Aangezien in de meeste projecten de kosten voor het overgrote deel bestaan uit arbeidskosten, zijn de factoren geld en tijd (hoeveelheid arbeidsuren) sterk verweven.

Geld in projectplannen:
- nagaan wat de tarieven zijn van de teamleden
- uren teamleden begroten
- budgetten aan teamleden toekennen voor bepaalde taken
- materiaal- en materieelkosten bepalen

Geld in voortgangsbewaking:
- geldstromen bewaken
- onderhandelen met leveranciers
- nagaan of de oorspronkelijke kosteninschattingen nog kloppen
- budgetten bijstellen
- onderhandelen met klant en/of opdrachtgever over bijstellingen budget

Geld in de projectverantwoording:
- financiële rapportage en verantwoording opstellen
- analyse definitieve financiële rapportage

Kwaliteit

Het projectresultaat zal aan bepaalde kwaliteitseisen moeten voldoen. Dat geldt dan ook automatisch voor de diverse tussenproducten van het project. Voor het sturen van het project is het met name van belang dat kwaliteitseisen in de definitiefase bepaald, afgesproken en opgeschreven worden. Voorkomen moet worden dat allerlei eisen aan de kwaliteit impliciet blijven. Een duidelijke lijst van eisen kan aan het einde van de realisatiefase gecontroleerd worden. Hiermee kan het projectteam bewijzen dat ze het project naar behoren heeft uitgevoerd.

Daarnaast kunnen er aan het uitvoeren van bepaalde taken van het project kwaliteitseisen gesteld worden, bijvoorbeeld dat een bepaalde taak alleen door gecertificeerd personeel mag worden uitgevoerd.

Kwaliteit in projectplannen:
- vaststellen van de gewenste kwaliteit van het projectresultaat en de tussenproducten (dit gebeurt met name in de definitiefase)
- vaststellen van de gewenste kwaliteit van het uitvoeren van de diverse activiteiten van het project

Kwaliteit in voortgangsbewaking:
- testen van de (tussen)resultaten
- behandelen eventuele kwaliteitsproblemen

Kwaliteit in de projectverantwoording:
- bevestigen dat de gewenste kwaliteit is bereikt
- behandelen van eventuele klachten (met name in de nazorgfase)

Perfectionisten zullen het moeilijk vinden om een project te managen. Een pragmatische houding ten aanzien van de kwaliteitsniveaus in projecten is: 'Goed genoeg is goed'. Wanneer in projecten de hoogst denkbare kwaliteit wordt nagestreefd, loopt het project grote risico's nooit af te komen.

Organisatie

Binnen een project moet het team gemanaged worden. In de meest smalle betekenis van teammanagement betekent dat: bepalen wie er wat doet van de activiteitenlijst. In de ruime betekenis houdt het ook alle zachte vaardigheden in die nodig zijn om met een groep mensen het doel te bereiken, zoals motivatie-technieken, communicatieve vaardigheden, leiderschapsstijlen en dergelijke. Deze zachte vaardigheden, hoe belangrijk ook, vallen verder buiten het kader van dit handboek

Organisatie in projectplannen:
- team samenstellen
- verantwoordelijkheden toekennen
- teamleden op taken inplannen
- afspraken maken over beschikbaarheid mensen met andere (project) managers en hoger management

Organisatie in voortgangsbewaking:
- team aansturen
- menselijke aspecten bewaken (zachte vaardigheden)
- onderhandelen tussen betrokken partijen in het project

Informatie

De factor informatie binnen projecten gaat over hoe, door wie en op basis waarvan beslissingen genomen worden. Wie mag of mogen er beslissen over welke zaken in het project? Is dat de projectleider, de opdrachtgever of een inhoudelijke expert uit het team? Wat wordt er gearchiveerd en door wie? Wordt er gebruik gemaakt van hulpmiddelen als een projectwebsite, issue tracker, e-mail notificatie of een gezamenlijke agenda?

Dergelijke informatievragen moeten worden beantwoord voordat een project kan starten. Organisaties die regelmatig werken met projecten hebben vaak een aantal hulmiddelen (bijvoorbeeld Word templates) klaarliggen voor het hanteren van de informatie binnen een project.

Informatie in projectplannen:
- welke informatie moet bij wie, wanneer terecht komen en op welke manier
- welke informatie wordt er geregistreerd, verspreidt en/of gearchiveerd
- welke informatiehulpmiddelen worden er gebruikt

Informatie in voortgangsbewaking:
- zorgen voor overlegmomenten
- zorgen dat de juiste informatie bij de juiste personen komt
- kijken of gemaakte afspraken worden nagekomen

Informatie in de projectverantwoording:
- schrijven van de projectrapportage

In de bijlage 6 tot bijlage 9 is bij dit handboek een aantal voorbeelden van informatieformulieren bijgevoegd. De formulieren kunnen gebruikt worden bij de informatie-uitwisseling in een project:

- Issuelijst
- Actie en besluitenlijst
- Risicolog
- Gespreksverslag

De *issuelijst* is een lijst waarin alle punten die besproken moeten worden, worden verzameld. De issuelijst moet regelmatig worden besproken. Voor het bijhouden van de voortgang en het registreren van genomen besluiten is een model voor een *actie en besluitenlijst* bijgevoegd. Er is een *risicolog* bijgevoegd waarop de risico's die tijdens het project worden gesignaleerd, kunnen worden geregistreerd. Deze risico's moeten dan in het volgende overleg van het projectteam besproken worden en waar nodig geëlimineerd. Tenslotte is een standaard *gespreksverslag* bijgevoegd als voorbeeld van hoe gespreksverslagen opgesteld en gearchiveerd kunnen worden. In Bijlage 3 is een overzicht gegeven van nuttige hulpmiddelen van derden.

Een belangrijk aspect van de bewaking van informatie rond projecten, is dat de beslissingen die genomen worden reproduceerbaar zijn. Heel vaak komt het voor dat beslissingen mondeling genomen worden en niet gearchiveerd worden. Hoe duidelijk de beslissing op dat moment ook lijkt voor beide partijen, de beslissing moet later op schrift terug te vinden zijn. Als dat niet kan, is de ongedocumenteerde beslissing een bron van misverstanden of zelfs conflicten.

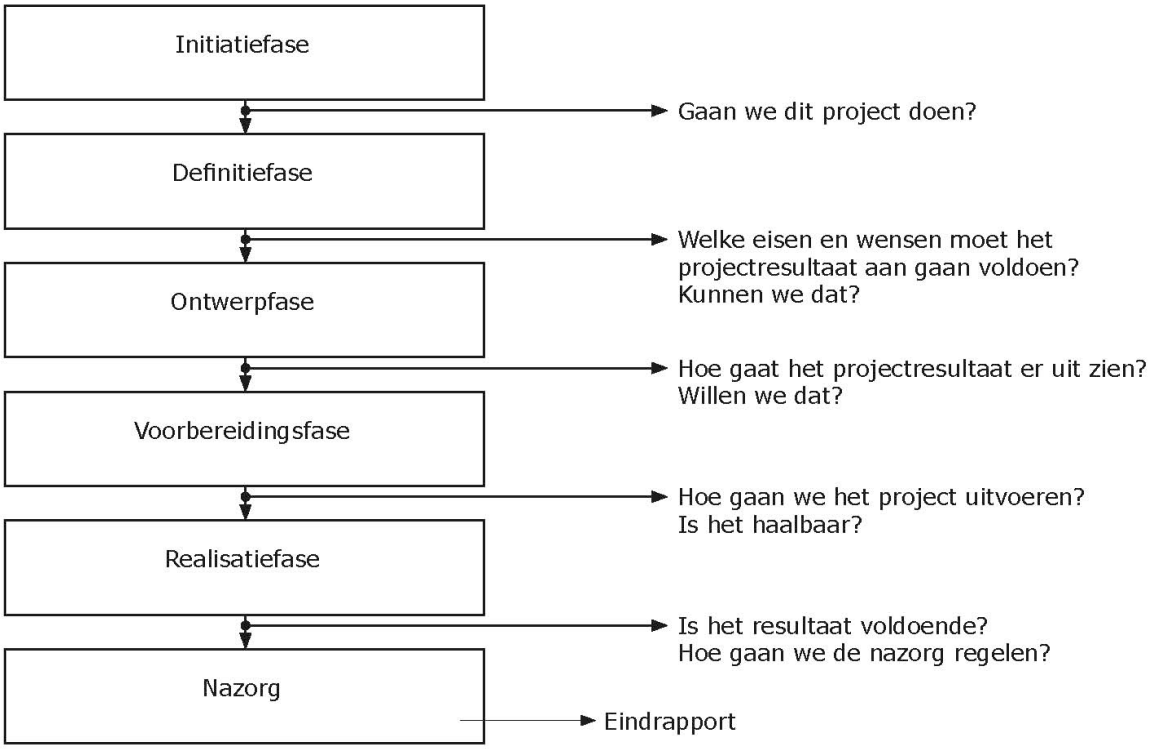
Veel projecten lopen vertraging op door allerlei interventies van buitenaf ('dit is even belangrijker', 'dit is politiek beter', ' de klant wil dat we eerst aan wat anders werken', enzovoort). Het kan verstandig zijn om voor jezelf een log bij te houden waarin je dit soort interventies bijhoudt, zodat het voor jou later duidelijk is waarom een project vertraging heeft opgelopen.

3. Projectrapportage

Er zijn vijf momenten, waarop cruciale beslissingen genomen worden over een project, namelijk na elke projectfase. Dat is niet alleen het moment om de stand van zaken van het project op te nemen en een tussenverslag te schrijven, maar ook om opnieuw te kijken naar de fases die nog moeten komen.

Dit zijn de momenten waarop de projectleider met de opdrachtgevers om tafel moet om beslissingen ten aanzien van het project door te nemen. Het zijn de momenten waarop de GOKIT-factoren – waar nodig - bijgestuurd moeten worden. Als bijvoorbeeld is gebleken dat er in de definitiefase veel nieuwe onverwachte eisen op tafel zijn gekomen, kan het zijn dat een project veel duurder uitvalt. Het heeft dan geen enkele zin om door te gaan met de oude begroting.

De momenten na de fases zijn vaak tevens 'go-no-go-momenten', Momenten waarop besloten wordt of het project definitief doorgaat of stilgelegd wordt.



Figuur 6: Vijf belangrijke beslismomenten bij een project

Wat vaak gebeurt bij organisaties die niet met faseringen van projecten werken, is het volgende: in het begin wordt een projectplan geschreven, waarin de GOKIT factoren worden beschreven. Er is een tijdpad uitgezet (T), er is een budget opgesteld (G), een team is geformeerd (O), een doel is beschreven (K) en de hulpmiddelen voor de informatievoorziening in en rond het project zijn bepaald (I). Tijdens het project controleert de projectleider nog wel of het project enigszins binnen het totale budget en tijdpad blijft, maar hij of zij stuurt niet echt bij. Tegen het einde van het project blijkt het allemaal toch weer duurder geworden of langer

te duren dan verwacht. Om te voorkomen dat het nóg veel duurder wordt of langer duurt, wordt het project afgeknepen. Helaas is daardoor het projectresultaat niet echt naar tevredenheid.

Had de projectleider wel met het 6-fasenmodel gewerkt, dan had het team waarschijnlijk al in de ontwerpfase of misschien al in de definitiefase geconcludeerd dat het oorspronkelijke tijdpad en budget onvoldoende zou zijn. Als de projectleider dan bijgestuurd had, had men voor een eenvoudiger ontwerp kunnen kiezen dat sneller en goedkoper te realiseren zou zijn. Of men had om meer tijd en/of geld kunnen vragen bij de opdrachtgever. In ieder geval was er al maanden eerder duidelijkheid geweest over de status van het project. En was er sprake geweest van daadwerkelijk sturen van het project.

Onzekerheid in projectplannen

Onzekerheid hoort bij projecten. Men weet bij aanvang van het project niet precies hoeveel tijd nodig is, noch hoe duur het project exact zal uitpakken. Bij sommige projecten is het zelfs onzeker of het beoogde doel überhaupt bereikt zal worden. Dan komt het soms ook voor dat in een snel veranderende wereld de uitgangspunten van een project al veranderd zijn voordat een project klaar is. Bijvoorbeeld door technologische ontwikkelingen of ontwikkelingen in de markt of politiek.

Bij het maken van projectplannen kan de projectleider slechts schattingen maken van de GOKIT-factoren van een project. Schattingen van tijd, geld, team, kwaliteitsdoelen en benodigde informatie. Door het project uit te voeren ontstaat er meer kennis over het project zelf. In de initiatiefase is er alleen nog een idee. In de definitiefase is het idee verder afgebakend door middel van concrete eisen en wensen. In de ontwerpfase zijn er mogelijke ontwerpen onderzocht en gemaakt waardoor er weer meer duidelijk is. In de voorbereidingsfase weet men hoe het ontwerp gemaakt moet worden, in de realisatiefase is het daadwerkelijke projectresultaat gebouwd en in de nazorg zijn de losse eindjes aan elkaar geknoopt.

Hoe verder het project is, des te meer duidelijkheid er ontstaat. Het heeft daarom geen zin om in het begin van een project een nauwkeurige begroting te maken voor de nazorgfase die pas later plaatsvindt. Het project kan nog alle kanten op. Het idee is nog nauwelijks uitgewerkt. Waarschijnlijk is ook pas op hoofdlijnen bekend hoe de nazorg er uit moet komen te zien. Dat is te weinig informatie om een realistische, nauwkeurige begroting te kunnen maken van de nazorgfase. Een begroting op hoofdlijnen is op dat moment het maximaal haalbare.

De werkwijze in projectplannen is dus als volgt: er wordt een globale begroting voor het hele project gemaakt en een concrete begroting voor de eerstvolgende fase. Als een projectteam bijvoorbeeld vlak voor de realisatiefase staat (na de voorbereidingsfase), is heel goed bekend wat er moet gebeuren. Op dat moment is het mogelijk een nauwkeurige begroting van de realisatiefase te maken.

De globale begroting voor het totale project zal na elke fase bijgesteld moeten worden. Na elke fase is er meer kennis en zijn er besluiten genomen waardoor de globale begroting nader ingevuld kan worden. Op die manier wordt na elke fase de begroting van de totale kosten van het project steeds nauwkeuriger.

Het maken van een globale begroting voor het hele project en een concrete voor de eerstvolgende fase is niet alleen van belang voor de GOKIT-factor geld. Ook voor de andere factoren geldt dat er van globaal naar concreet wordt gewerkt.
Samenvatting begrotingen maken:

- vindt plaats vóór elke fase
- globale begroting voor het hele project opstellen of bijstellen
- concrete begroting maken voor de volgende fase

- alle GOKIT-factoren moeten opnieuw worden bekeken en begroot voor een nieuwe fase.

Door deze manier van begroten (met name van tijd en geld) wordt realistisch omgegaan met de onzekerheid die er meer in het begin en minder aan het einde van een project is. Het levert wel een probleem op voor organisaties die gefinancierd worden door overheidssubsidie en/of maatschappelijke fondsen. Dit geldt in het bijzonder voor organisaties die vernieuwende en dus onzekere projecten doen.

De meeste fondsen en subsidieverstrekkers eisen een complete en vaststaande begroting van een projectvoorstel vóórdat zij overgaan tot financiering. Dat betekent dat een organisatie die een project gefinancierd wil krijgen in een vroeg stadium al met een complete en concrete begroting moet komen. Maar in het begin is het project pas in de ideeënfase. Op dat moment is het dus helemaal niet mogelijk een reële inschatting van kosten en tijdpad te maken.

Pas na de ontwerpfase, als het idee nader is uitgewerkt en een ontwerp is gekozen, is er met voldoende nauwkeurigheid te zeggen hoeveel het project zal gaan kosten en hoe lang de uitvoering gaat duren. Dat moment is echter pas enkele maanden na het aanvraagmoment van de subsidie.

Het resultaat van deze werkwijze van subsidieverstrekkers en fondsen is dat veel organisaties een bedrag aanvragen op basis van een grove inschatting van de projectkosten. Vervolgens worden de projectactiviteiten ingesnoerd naar het verkregen budget. Dit zet het projectteam al klem bij de start van een project, terwijl er dan nog veel flexibiliteit nodig is.

Bij het uitwerken van het idee in de definitiefase en ontwerpfase blijkt dan vaak dat het tijdpad in de subsidieaanvraag niet haalbaar is. Het budget is niet passend, te veel voor sommige posten en meestal te weinig voor andere posten. Als de subsidieverstrekker dan ook nog eist dat er bijvoorbeeld niet meer dan 5% van de posten mag worden afgeweken, komt het projectteam onder grote druk te staan. Zaken moeten gerealiseerd worden binnen een te klein budget en in te korte tijd. Het leidt tot een hoop schuiven met posten. In de projectverantwoording is vervolgens veel tekst en analyses nodig waarom het gewenste resultaat niet bereikt is.

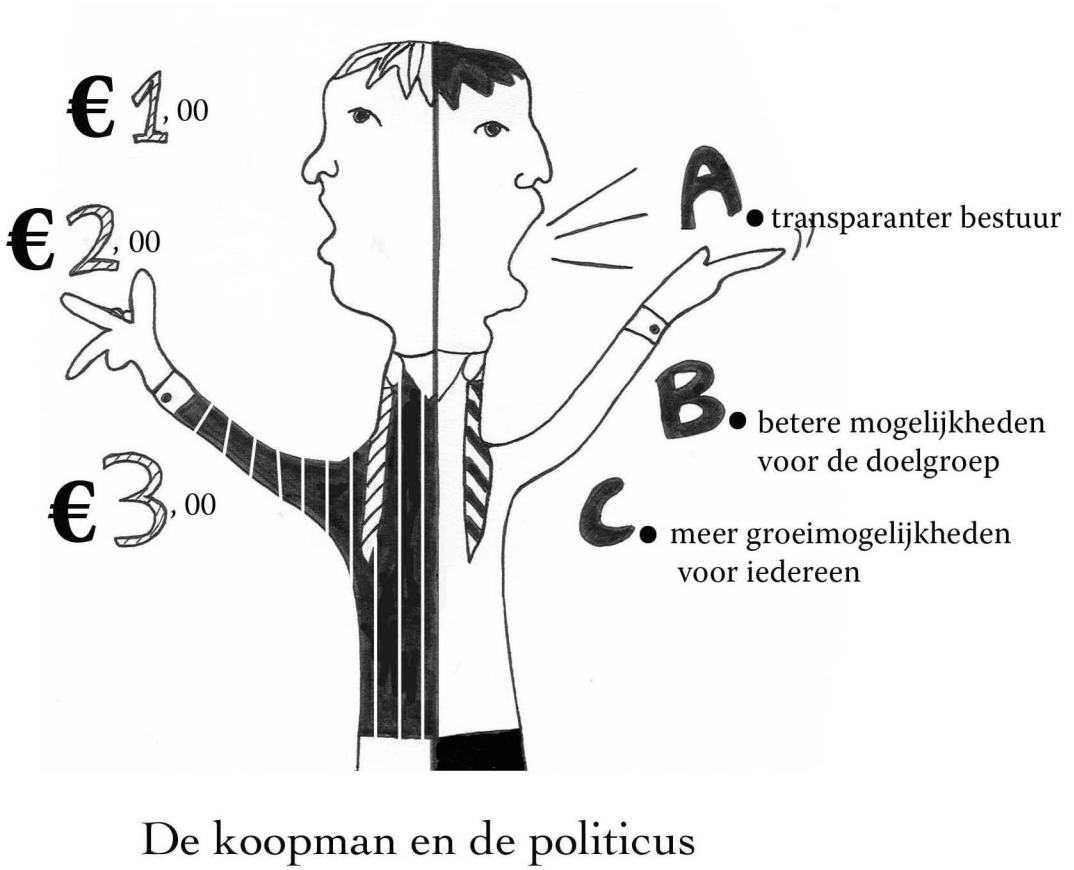
Deze situatie zou verbeteren als subsidieverstrekkers de financiering zouden koppelen aan de verschillende fases in plaats van in één keer vooraf te financieren. De initiële financiering zou dan zijn voor het doorlopen van de definitiefase en de ontwerpfase. Met een beperkt budget worden eisen en wensen nader uitgezocht en wordt een aantal alternatieve ontwerpen gemaakt. Op basis van deze ontwerpen wordt dan een vervolgaanvraag gedaan voor de realisatie en nazorg. Op deze manier worden projecten niet onnodig onder druk gezet. Bijkomend voordeel zou bovendien zijn dat de verwachtingen van betrokken partijen reëler wordt. Dat scheelt tijd, geld en teleurstellingen.

4. De koopman en de politicus

Het is voor projectleiders goed om rekening te houden met de omgeving waarbinnen het project plaatsvindt. Of beter gezegd: met de manier waarop beslissingen worden genomen binnen en over het project. Er zijn twee werelden waarbinnen een project zich kan bevinden: in de wereld van de koopman of in de wereld van de politicus.

In de wereld van de koopman draait alles om winstmaximalisatie. De koopman heeft veel belang bij stabiliteit. Het handelen is gebaseerd op onderling vertrouwen en het motto 'afspraak is afspraak' geldt. De relaties tussen de koopmannen zijn belangrijk en het gedrag dat mensen vertonen is authentiek. De macht is gedecentraliseerd.

In de wereld van de politicus is de meerderheid van belang om dingen gedaan te krijgen. Loyaliteit aan de groep is dus belangrijk, ook als de politicus op sommige punten zelf een andere mening heeft dan de groep tot welke hij behoort Omdat de meerderheid zelden bij één groep ligt, zijn tijdelijke coalities noodzakelijk. Soms met tegenstanders of zelfs met vijanden. De beslissingen komen voort uit een visie op de wereld. In de wereld van de politicus is het verzwijgen van bepaalde feiten voor de goede zaak wel eens nodig, het doel heiligt de middelen. De macht is gecentraliseerd.



Figuur 7: De koopman en de politicus (Illustratie: Rachèl Harmsen)

De meeste mensen voelen zich intuïtief meer aangetrokken tot de eerste wereld, de tweede wereld roept veel negatieve associaties op. "Wij doen niet aan politiek" is een veelgehoorde opmerking bij organisaties, wat vaak feitelijk onjuist is. Hoewel de meesten dus de wereld van de koopman het aantrekkelijkst vinden, heeft deze een belangrijk nadeel. Besluitvorming op basis van winstmaximalisatie werkt alleen bij beslissingen waar duidelijke geldstromen aanwezig zijn. Als het beslissingen betreft die gaan over dilemma's/vraagstukken zoals meer of minder investeren in onderwijs, milieu, gezondheidszorg, autowegen, onderzoek, defensie, kernenergie, enzovoort dan kan er niet een eenduidige balans worden opgesteld van winst en verlies. Voor dergelijke beslissingen geldt dat er geen ander beslissingsmodel is dan het politieke model. Het politieke spel moet dan dus gespeeld worden.

Maatschappelijke en gesubsidieerde organisaties bevinden zich per definitie in de wereld van de politicus. De financiering van deze organisaties en hun projecten is geheel of grotendeels afhankelijk van de politieke wil om deze organisatie te steunen. De effectiviteit van maatschappelijke organisaties is niet eenduidig uit te drukken in geldstromen. Dat geldt ook voor de resultaten van de projecten van maatschappelijke organisaties.

Een jonge ingenieur moest voor een gemeente ergens in het land een ambitieus windenergieproject uitvoeren. Via een ingenieus spaarsysteem konden de bewoners van de gemeente sparen voor een aantal windmolens, met als doel 30% van de benodigde elektriciteit van het dorp met eigen windmolens op te wekken. Daarvoor waren tien windmolens nodig. Het idee was afkomstig van een van de wethouders uit het dorp.

Het enthousiasme van de dorpsbewoners voor het spaarprogramma viel nogal tegen. Met moeite werd er een halve windmolen bij elkaar gespaard. Om het niet helemaal te laten floppen, besloot de gemeente het bedrag aan te vullen zodat er in ieder geval één windmolen geplaatst kon worden.

In de eerste versie van het eindverslag, schreef de ingenieur dan ook dat het resultaat erg was tegengevallen. Dat zou echter gezichtsverlies betekenen voor de wethouder, dus die drong aan op herformulering van de eindconclusie. De uiteindelijke tekst werd "Het project is een groot succes, de gemeente laat zien dat het betrokken is bij de leefwereld en heeft haar – weliswaar bescheiden - steen bijgedragen om klimaatverandering tegen te gaan." De jonge ingenieur was zich aanvankelijk niet bewust van het politieke kader van dit project. Om te voorkomen dat toekomstige projecten van de wethouder niet zouden kunnen starten, moest hij het politieke spel (mee) spelen.

Een project uitvoeren in een politieke omgeving is lastiger dan een project uitvoeren in een koopmansomgeving. Beslissingen in en rond een project zijn afhankelijk van het politieke spel en niet van wat het meest effectief zou zijn voor het project. De aanleiding om een project te starten is vaak politiek ingegeven en bepaalt daarmee het krachtenveld waarmee het projectteam te maken krijgt.

Een aantal organisaties moest gaan fuseren en samenwerken als gevolg van een reorganisatie. Deze reorganisatie was van bovenaf opgelegd en hield onder andere in dat een aantal kleine lokale filialen in dorpen vervangen werd door een centraal punt in de regio. Dat betekende dat medewerkers veel meer moesten gaan reizen. Ook het werk kwam inhoudelijk anders te liggen, er waren veel minder plekken voor hooggeschoolde medewerkers beschikbaar na de reorganisatie. Een deel van het personeel zou moeten uitkijken naar een functie buiten de organisatie of naar andere functies die inhoudelijk een stuk minder interessant leken. Er was dus veel weerstand tegen de reorganisatie, alhoewel het voor de klanten een grote verbetering van de service inhield als de reorganisatie zou lukken. Tenslotte was

het nog zo dat de reorganisatie door het personeel zelf uitgevoerd moest worden onder leiding van een projectleider.

De projectleider had aanvankelijk grote moeite om het project op gang te krijgen. De teamleden die het werk moesten uitvoeren, vonden steeds aanleidingen om hun werk niet te doen. Er was altijd wel een probleem of tegenslag en er werd veel gediscussieerd. Uiteindelijk ontaardden de discussies meestal in een discussie of het project überhaupt wel een goed idee was. De projectleider verdedigde dan het project; het zou voor de klanten een grote verbetering inhouden, maar hij kreeg geen enthousiasme voor het project.

Toen de projectleider zich realiseerde dat een groot aantal van de medewerkers niet (volledig) achter het project stonden, besloot hij eerst aandacht te besteden aan het verminderen van de weerstand tegen het project . Hij deed dit door veel meer tijd te besteden aan het bezoeken van de verschillende filialen. Ook praatte hij veel meer met chefs en medewerkers, vaak informeel bij de koffieautomaat. Doordat hij een betere relatie kreeg met een aantal van de formele en informele machthebbers, kon hij het project nu beter vlottrekken als het vastliep. Het bleef een moeilijk project, maar de politieke aanpak werkte veel beter dan de meer rationele benadering die hij aanvankelijk had.

Een handleiding voor het spelen van het politieke spel valt buiten het kader van dit boek. Samengevat speelt het politieke spel zich af op het vlak van relaties en machtsverhoudingen. In een zakelijke omgeving staat het product of project zelf veel meer op de voorgrond.

Belangrijk voor projectleiders is dat zij zich realiseren dat als zij projecten doen bij maatschappelijke organisaties, er *altijd* in meer of mindere mate sprake is van politieke krachten. Om het project tot een succes te brengen doet de projectleider in deze situatie er verstandig aan zich niet te ontrekken aan het politieke spel, maar dat, naast de inhoudelijke sturing van het project, zo goed mogelijk te spelen.

5. Waterval versus cyclisch projectmanagement

Het 6-fasenmodel is een watervalmodel. Dat wil zeggen dat de fases na elkaar plaatsvinden. Net zoals je niet omhoog kan zwemmen tegen een waterval in, kan je in het pure watervalmodel na het doorlopen van een fase ook niet meer terug. Het is niet wenselijk als in de realisatiefase besloten wordt om het ontwerp aan te passen en daarvoor de uitvoering stil te leggen. Het watervalmodel is doorgaans minder geschikt voor softwareontwikkelingsprojecten. Hiervoor zijn een aantal oorzaken (zie o.a. McConnell, 1996, Kroll, 2004, Chromatic, 2003 en Stapleton, 2002).

- De ontwikkeling van software is een creatief proces.
- Het is bijna onmogelijk alle eisen en wensen (functionaliteiten) voor nieuwe software vooraf boven water te krijgen.
- Het inschatten van de benodigde hoeveelheid tijd voor het realiseren van een functionaliteit is erg moeilijk.
- Het is wenselijk dat alle tussenresultaten door de gebruiker gedurende het traject getest kunnen worden.

De ontwikkeling van software is een creatief proces

De ontwikkeling van software wordt door buitenstaanders eerder als 'engineering' dan als 'iets uitvinden' gezien. Toch heeft het wel raakvlakken met iets uitvinden. Iets uitvinden betekent dat je vooraf nooit precies weet wat er gaat komen.

De ontwerpers en programmeurs die nieuwe software gaan schrijven, moeten oplossingen bedenken voor de problemen die hen worden voorgeschoteld. Ondanks vele methodes en voorschriften van werken, blijft het schrijven van programmacode voor een groot deel nieuw en daardoor in hoge mate onzeker. De programmeurs kunnen voor hun oplossing kiezen uit wel miljoenen oplossingen, geschreven in tientallen programmeertalen, draaiend op duizendtallen verschillende hardwareconfiguraties, werkend op tientallen software platforms. Die vrijheid biedt een aantal voordelen, maar maakt ook dat het project moeilijker te beheersen is dan traditionele projecten als bijvoorbeeld constructie- of bouwprojecten.

Het is vrijwel onmogelijk alle eisen en wensen vooraf boven water te krijgen

De watervalmethode schrijft voor dat in de definitiefase de eisen en wensen als projectresultaat geformuleerd worden. Idealiter geval wordt van die eisen of wensen niet of nauwelijks meer afgeweken. Bij softwareontwikkeling volgens het watervalmodel wordt er in het begin van het project veel tijd besteed aan het analyseren van de benodigde functionaliteiten (eisen en wensen). Dit leidt tot een functioneel ontwerp (voor een nadere toelichting op functioneel ontwerpen, zie bijvoorbeeld [i]). Op basis van het functionele ontwerp gaat de softwarearchitect in de ontwerpfase een technisch ontwerp maken. In het technische ontwerp wordt beschreven hoe de gewenste functionaliteiten gerealiseerd moeten worden.

Dit lijkt allemaal vrij recht toe recht aan, maar stel de volgende situatie voor (voorbeeld aangepast van McConnell, 1996, p. 138): Er is een project voor de productie van een nieuwe auto. U bent een actieve autorijder en aan u wordt gevraagd de eisen en wensen te formuleren die er aan een auto zijn. U overlegt met andere autorijders en komt met een hele lijst:

- de auto moet plaats bieden aan 4 personen
- de auto moet een stuur linksvoor hebben, een rempedaal, een gaspedaal, een handrem
- de auto heeft 4 wielen
- de auto heeft witte koplampen en rode achterlichten enz. enz. (In de realiteit zou het een enorme lijst worden.)

De ontwerpers maken op basis van uw lijst een nieuw ontwerp, dat vervolgens, maanden later, gebouwd wordt. Dan, u loopt over straat en ziet een auto remmen. U realiseert zich dat u in deze lijst vergeten bent de remlichten te noemen! U belt onmiddellijk met de ingenieur van de autofabriek die zowat ontploft van uw telefoontje. U oppert nog dat het slechts om een kleinigheidje gaat. Maar voor de autobouwers is het rampzalig. De gemaakte auto's moeten helemaal opengemaakt worden, er moeten kabels van de rempedalen naar achteren getrokken worden, het ontwerp van de achterkant moet helemaal opnieuw om ruimte te bieden voor de remlichten, de achterkleppen die al geproduceerd zijn moeten worden weggegooid, en ga zo maar door.

Bovenstaand voorbeeld lijkt misschien wat absurd maar bij softwareontwikkeling is het bijna dagelijkse realiteit. Programmeurs gaan er ten onrechte vanuit dat de opdrachtgever, klant of gebruiker van nieuw te ontwikkelen software precies weet wat hij wil. De opdrachtgever gaat er ten onrechte van uit dat de softwarebouwers in de kortste tijd alles kunnen maken (en aanpassen). Dit probleem is de grootste oorzaak van conflicten tussen klanten en softwarebouwers.

En dat er veel conflicten zijn tussen klanten en softwareleveranciers blijkt wel uit het volgende. Een startende ondernemer wilde een rechtsbijstandverzekering voor zijn bedrijf afsluiten. Hij informeerde naar de mogelijkheden. De tussenpersoon vroeg in welke branche de ondernemer actief zou zijn, waarop hij 'ICT' antwoordde. "Helaas dan", reageerde de tussenpersoon, "er zijn zo vaak rechtszaken tussen ICT-leveranciers en klanten dat er geen verzekeraar meer is die voor ICT-bedrijven een rechtsbijstandsverzekering wil afsluiten."

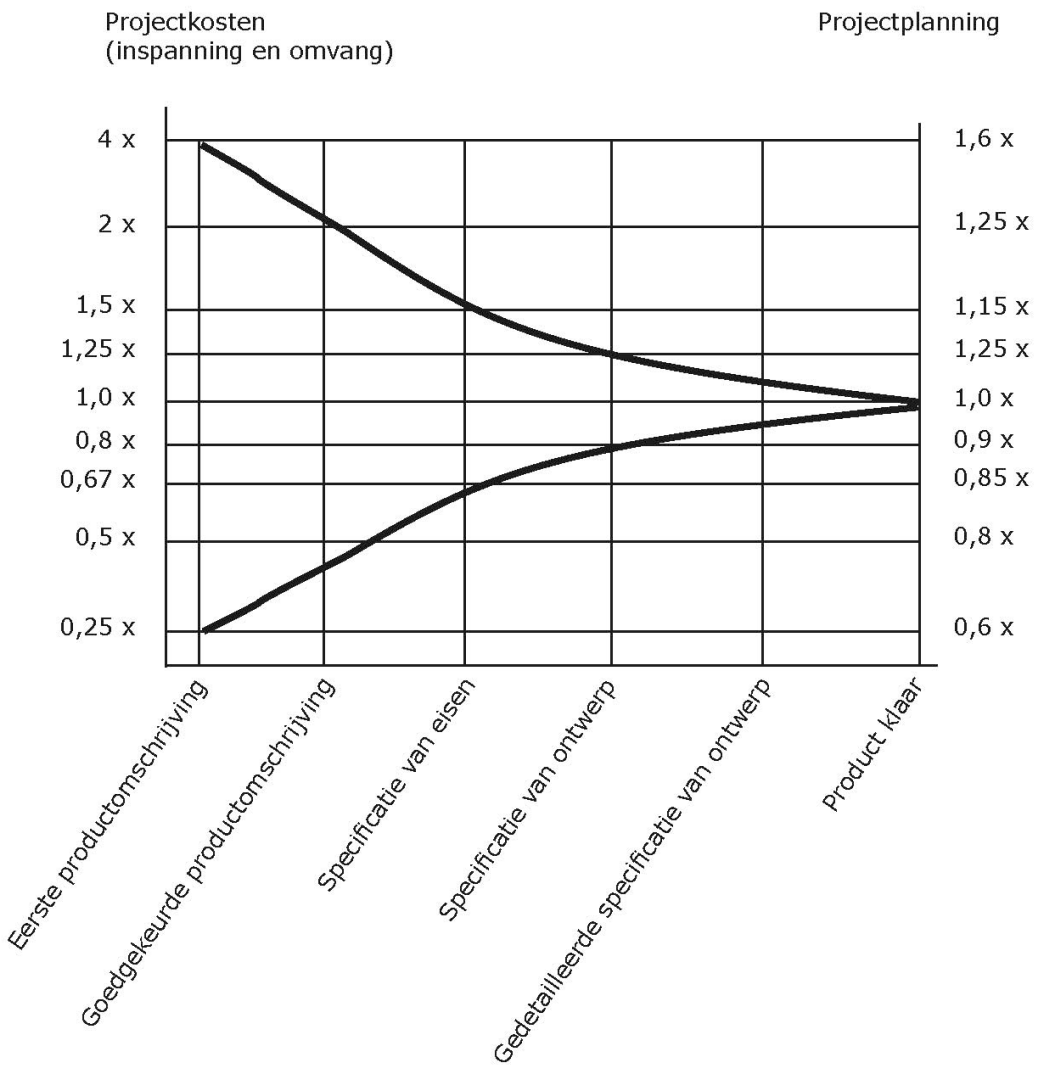
Het schatten van de benodigde hoeveelheid tijd is moeilijk

Het watervalmodel gaat uit van een aantal fasen. In het projectplan moet de projectleider een schatting maken van de hoeveelheid tijd (en dus geld) die nodig is voor elke fase. We hebben al gezien dat het inschatten van de factor tijd lastig is bij projecten in zijn algemeenheid. Met name als die schatting gegeven moet worden in het beginstadium van het project. Voor softwareprojecten is het schier onmogelijk. Stel dat het zou lukken een kwalitatief goede lijst van functionaliteiten op te stellen in de definitiefase. In theorie zou het projectteam dan per functionaliteit redelijk moeten kunnen inschatten hoeveel tijd er nodig is voor de realisatie. De praktijk laat echter zien dat er te veel onzekerheden zijn om een redelijke inschatting te kunnen maken (o.a. McConnell, 1996):

- Als functionaliteit gemaakt is, blijkt nogal eens dat de klant hem toch niet nodig heeft. De gebruikte uren zijn als nutteloos te beschouwen.
- Eisen en wensen veranderen gedurende het project.
- Moet de functionaliteit goedkoop of duur worden uitgevoerd? Er zijn vele manieren van realisatie en implementatie mogelijk.
- Hoe wordt functionaliteit technisch ontworpen? Het ontwerp bepaalt in grote mate de tijd die nodig is voor het maken ervan.

- Hoe goed moet de kwaliteit zijn van functionaliteit? Bijvoorbeeld moet een webapplicatie altijd 100% beschikbaar zijn, of mag hij een paar uur per jaar offline zijn?
- Het vinden en herstellen van fouten in software varieert van minuten tot weken.
- Hoe lang duurt de installatie en integratie van de nieuwe software bij de bestaande systemen van de klant?
- Het ontbreken van kennis over mogelijke oplossingen maakt het schatten van de benodigde hoeveelheid tijd ook moeilijk. Het is daarbij ook moeilijk in te schatten hoe lang het verkrijgen van de benodigde kennis duurt.

Uit bovenstaande lijst blijkt dat er heel veel factoren zijn die van invloed zijn op de hoeveelheid tijd die uiteindelijk nodig blijkt te zijn voor de realisatie van software. Uit onderzoek (McConnell, 1996, p. 168) is gebleken dat de schattingen voor de benodigde tijd voor het realiseren van functionaliteit in het begin van een project varieert tussen vier keer te weinig tijd en vier keer teveel tijd. Dat betekent bij een verkeerde schatting dat de benodigde hoeveelheid tijd 4 maal hoger of lager kan uitvallen. Die schattingen worden nauwkeuriger naarmate het project verder is. Na het maken van een functioneel ontwerp is de marge echter nog altijd tussen de 25% teveel of te weinig. Pas vlak voordat functionaliteit gerealiseerd wordt, kan een schatting met redelijk hoge zekerheid gegeven worden (zie figuur 8).



Figuur 8: Nauwkeurigheid van inschatten van benodigde tijd voor het realiseren van een functionaliteit (Bron: McConnell, 1996, p. 168).

Software is nooit 100% foutloos. Zelfs van bekende software pakketten waar velen mee werken als Word, Excel, Explorer, OSX, PHP, Flash zijn er lijsten van bekende bugs te vinden op het internet. Er komen geregeld nieuwe releases op de markt waarvan er fouten uit de software gehaald zijn.

Klanten verwachten soms foutloze software maar als dat nagestreefd zou worden in de praktijk zou dat betekenen dat software nooit af is. Bovendien komt het niet zelden voor dat de fout in de software niet in de eigen software zit.

Een educatieve game werd gemaakt in Flash. Afgesproken was dat de game als bestand aangeleverd zou worden en dat de klant hem zelf zou installeren op zijn server. Gedurende het project wilde de klant graag dat er een high score aan de game zou worden toegevoegd om het competitieve element tussen de spelers te verhogen. Daarvoor was een stuk scriptcode nodig in PHP. De gamebouwers maakte die scriptcode en testten die op hun eigen server die op Linux draaide. Het werkte. De game en de scriptcode werden aangeleverd aan de klant. Die klant had echter een Windows server en om de een of andere reden werkte het script niet goed meer. De high scores werden niet opgeslagen.

Uiteindelijk had de programmeur een week nodig om het probleem op te lossen. Het bleek zo te zijn dat de combinatie van PHP en Windows server soms niet goed werkt. In het script zelf zat helemaal geen fout! Door een hack kon hij het werkend krijgen en iedereen was tevreden. Alhoewel, wie moest die week extra werk betalen?

Bij een ander softwareontwikkelingstraject werd ook educatieve software gemaakt. Het probleem was dat de software bij de programmeurs prima werkte, maar bij de klant het niet goed deed. Na van alles geprobeerd te hebben, besloot de programmeur bij de klant op bezoek te gaan. Daar bleek dat de computer van de klant een oude pentium III was. Door de traagheid van de computer deed de software het niet goed. Bij de programmeurs was echter ook getest op een pentium III en daar werkte het prima. Na verder onderzoek bleek dat de computer van de klant zo traag was omdat het systeem vol zat met virussen en spyware.

Bovenstaande onzekerheid maakt het schrijven van projectplannen niet eenvoudiger. Ook het maken van afspraken tussen betrokken partijen wordt er lastig door. Iemand moet het risico dragen voor het meerwerk dat gedaan moet worden. Als er betaald wordt op uurbasis, dan ligt het risico bij de klant. Als er een fixed price is afgesproken (zoals bij subsidiefinancieringen), ligt het risico bij de softwarebouwer. Als er meer dan twee partijen bij zijn betrokken, ligt het nog complexer. Wie betaalt er dan voor de meeruren op een project?

Vaak ontstaat er een discussie over wie er verantwoordelijk is voor de vertragingen. Het aanwijzen van een schuldige is in veel gevallen heel moeilijk. En wellicht is er überhaupt geen schuldige omdat er geen sprake is van vertraging, maar van een verkeerde inschatting van de hoeveelheid uren in de eerste plaats. Het overschrijden van het aantal projecturen en de vraag wie dat moet betalen, is een bron van conflicten in de IT-wereld.

Het is wenselijk dat alle tussenresultaten door de gebruiker gedurende het traject getest kunnen worden

In de definitiefase en de ontwerpfase wordt van klanten gevraagd hun eisen en wensen zo goed mogelijk te formuleren. Dat is lastig om twee redenen. Ten eerste hebben klanten maar een beperkte voorstelling van de (on)mogelijkheden van ICT. Ze weten niet goed wat er kan of zou kunnen en dus weten ze ook niet goed wat ze wel of niet moeten wensen. Ten tweede hebben klanten vaak maar een beperkte kennis van hun eigen bedrijfsprocessen. Veel ICT-projecten gaan over het automatiseren van bestaande bedrijfsprocessen bij een klant. Hoewel de klant misschien al vele jaren volgens die processen werkt, blijkt vaak dat hij niet goed in staat is om zijn eigen bedrijfsproces te definiëren. Hij kan prima op zijn eigen manier werken, maar hij kan niet zeggen welke manier dat nu precies is. Deze precieze definiëring van processen is wel een voorwaarde voor het maken van software die de automatisering zal aansturen. De complexiteit voor klant neemt daarbij toe als hij ook nog eens niet bestaande processen moet gaan beschrijven.

Vaak zie je dat er in de definitiefase een eisen- en wensenlijst op tafel komt die onvolledig is. In de realisatiefase maken de programmeurs de software op basis van die onvolledige lijst. Wanneer de gebruiker dan de eerste versies van de nieuwe software voorgeschoteld krijgt, komen er meer eisen en wensen op tafel. "Dat ziet er goed uit, kan je het nu ook zo maken dat ik met een knopje kan inloggen zodat ik niet steeds mijn password hoef te herhalen". De programmeurs verzuchten dat de klant niet weet wat ie wil. De klant beroept zich op het 'feit' dat de softwareontwikkelaars professionals zijn en eerder hadden moeten inzien dat de klant dat wilde.

Bij een softwareproject voor de automatische afhandeling van aanvragen voor een dienst via het web was een uitgebreid functioneel ontwerp gemaakt. Lange lijsten van eisen en wensen van de klant waren opgesteld. Een aantal schermontwerpen en flow charts waren toegevoegd waarna de programmeurs aan de slag konden.

Wellicht omdat het project onder grote tijdsdruk stond of misschien omdat de organisatie van de klant behoorlijk rommelig was, bleek dat de ontwerpers een belangrijk onderdeel waren vergeten: handmatig beheer.
De aanvragen werden door de software verwerkt. Omdat het verwerken van de aanvragen automatisch zou moeten verlopen, dachten de programmeurs dat er geen handmatig beheer gewenst zou zijn. Die eis stond ook niet in het functioneel ontwerp. Toen de software werd opgeleverd om te testen, realiseerde de klant zich dat er uitzonderingen waren bij de aanvragen. Sommige aanvragen mochten niet automatisch verwerkt worden maar moesten handmatig verwerkt worden. De software werkte echter alleen automatisch...

In het watervalmodel wordt het gerealiseerde projectresultaat getest aan het einde van de realisatiefase. Dat is laat in het ontwikkelingstraject. In de tijd liggen er meerdere maanden, soms zelfs meer dan een jaar, tussen de definitiefase, ontwerpfase en de realisatiefase. Als er in een laat stadium fouten in het ontwerp boven water komen, is het duur of soms zelfs onmogelijk om de software nog aan te passen, zonder een geheel nieuw project te starten. Omdat we gezien hebben dat het eigenlijk onmogelijk is alle eisen en wensen vooraf goed te definiëren zou een werkwijze wenselijk zijn waarbij het mogelijk is om (tussen) resultaten eerder te testen.

Bij de keuze voor een softwarehuis werden enkele partijen vergeleken. De klant vroeg bij de concurrerende partijen wat er zoal mogelijk was. De ene partij was wat terughoudend en had zijn twijfels of bepaalde klantwensen wel zonder meer uitvoerbaar waren. De andere partij had een gedreven verkoper in dienst. Als de klant aan de verkoper vroeg of een bepaalde wens mogelijk was, belde hij met een van zijn programmeurs. " Kunnen wij een functionaliteit bouwen die X kan?" De programmeur dacht voornamelijk in termen van techniek en antwoordde daarop dat in principe alles mogelijk was. De programmeur en de verkoper maakten zich beiden niet druk om de haalbaarheid in termen van projectmanagement (tijd, geld, complexiteit, risico's, en dergelijke).

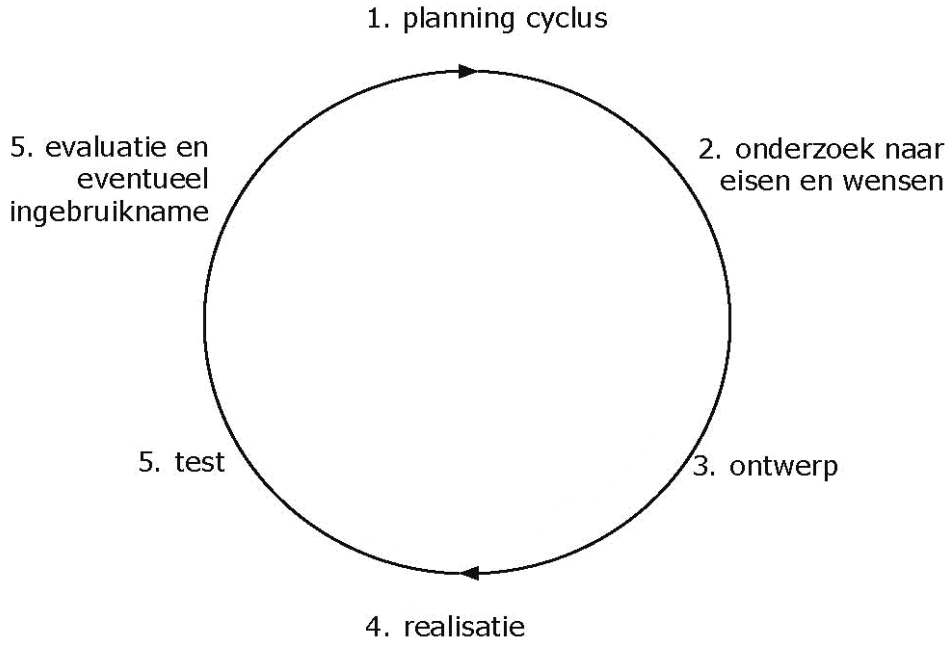
Het enthousiasme van de verkoper werkte beter dan de wat terughoudende houding van de andere partij. De klant koos voor de aanbieding van de gedreven verkoper. Het vers binnengehaalde project kwam vervolgens in handen van een projectleider en een groep programmeurs. Na verloop van tijd bleek dat het project niet voldeed aan de wensen van de klant. Dat had onder andere te maken met het feit dat de processen bij de klant veel ingewikkelder bleken dan op het eerste oog leek. Bij een zwaar weer gesprek tussen beide partijen beriep de klant zich op het feit dat de verkoper "had gezegd dat functionaliteit X geen enkel probleem zou zijn".

Cyclische projectmanagementmethodes

Door bovengeschetste problematiek zijn er de afgelopen jaren andere methodes voor projectmanagement ontstaan die met name worden toegepast bij ICT-ontwikkelprojecten. DSDM, RUP, eXtreme Programming (XP), RAD, agile projectmanagement, zijn enkele van de namen van deze relatief nieuwe stromingen binnen projectmanagement (McConnell, 1996, Kroll, 2004, Chromatic, 2003, Stapleton, 2002, [ii], [iii])

Hoewel bovenstaande projectmanagementmethodes op aspecten van elkaar verschillen, komen ze in essentie overeen. Omdat de weg naar het einddoel bij ICT-

projecten zo onzeker is gebleken, gaan deze methodes uit van het bereiken van het doel in een aantal korte cyclussen. Vandaar de benaming 'cyclisch' projectmanagement voor deze methodes.



Figuur 9: De activiteiten in een cyclus

Bij cyclisch projectmanagement tracht men bij het projectdoel te komen middels enkele korte cycli die na elkaar plaatsvinden. Elke cyclus duurt relatief kort, bij voorkeur korter dan een maand. Binnen een cyclus wordt een deel van het project uitgevoerd. Binnen een cyclus wordt geanalyseerd, ontworpen, gerealiseerd en getest. Dit is wezenlijk anders dan bij de watervalmethode waar die activiteiten allemaal in een eigen fase plaatsvinden. Bij de watervalmethode wordt bovendien maar éénmaal gedefinieerd, ontworpen, gerealiseerd en getest. Bij de cyclische methodes gebeurt dat vele malen na elkaar.

Tijdens de cycli worden verschillende onderdelen van de software gerealiseerd. De cycli staan daarmee maar deels los van elkaar. Nadat een cyclus is afgerond vindt evaluatie plaats. Mocht het zo zijn dat door voortschrijdent inzicht er nieuwe of andere eisen aan het project zijn ontstaan, dan worden de werkzaamheden van de komende cycli daarop aangepast.

Een cyclus begint bij het maken of bijstellen van de planning. Vervolgens worden de eisen- en wensen onderzocht van hetgeen dat gemaakt moet worden in deze cyclus. Er wordt een ontwerp gemaakt, dit ontwerp wordt geprogrammeerd en getest. Vervolgens vindt evaluatie plaats en bij sommige methodes ook in gebruikname van de nieuwe software. Daarna kan de volgende cyclus starten om het volgende deel van het project uit te voeren. (Voor een meer uitvoerige beschrijving van cyclische methodes en de verschillen tussen de methodieken zie onder andere Kroll, 2004, Chromatic, 2003, Stapleton 2002.)

De grote voordelen van het werken met een cyclische methode zijn:

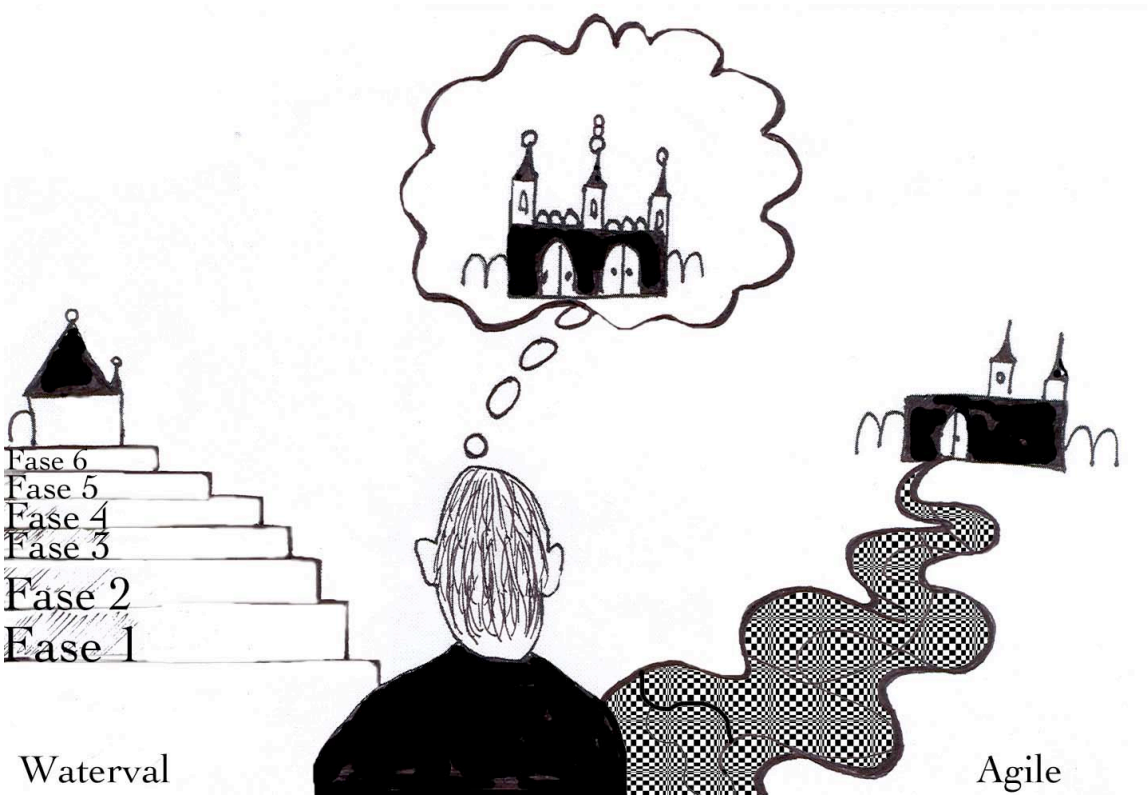
- hogere kwaliteit van het product en betere implementatie van functionaliteiten
- Realistischer begroting van tijd en geld
- Projectteam komt minder onder druk te staan
- Hogere kwaliteit

Uit vorige hoofdstukken is gebleken dat het moeilijk of wellicht onmogelijk is om in een vroeg stadium in het project de gewenste functionaliteiten goed te bepalen. In de cyclische methoden worden de gewenste functionaliteiten middels enkele korte cycli gerealiseerd. Per cyclus wordt een klein deel van de gewenste functionaliteit niet alleen onderzocht, maar ook ontworpen, gerealiseerd, eventueel geïmplementeerd én getest. Met name het kort opeenvolgen van ontwerpen, realiseren en testen leidt tot betere kwaliteit. Het maakt dat het team in staat is aanpassingen te doen. Als een ontwerp in de praktijk niet goed uitpakt, wordt dit binnen het tijdvak van de cyclus duidelijk. Er kan dan bijgestuurd worden. Binnen deze manier van werken kan de klant dus vragen om aanpassingen.

Een andere reden dat cyclisch projectmanagement tot betere kwaliteit leidt is omdat er in een cyclus intensief wordt samengewerkt tussen klant, ontwerpers en programmeurs. Er is een multidisciplinair team dat gezamenlijk oplossingen bedenkt en uitvoert. Bij de watervalmethode is de klant vooral in het begin aan het woord bij het formuleren van de eisen en wensen, daarna maken ontwerpers een ontwerp en vervolgens programmeren de programmeurs de software. De projectleider is de schakel tussen al die partijen en moet ervoor zorgen dat de informatie over en weer terecht komt. In de praktijk betekent het bijvoorbeeld dat veel programmeurs en ontwerpers de klant nooit zien. Die komt pas weer in beeld als de software af is.

Cyclische projectmanagement methodes zijn vooral goed voor projecten waarbij men het te bereiken doel vooraf niet goed kan vaststellen, zoals creatieve projecten of onderzoeksprojecten. Door te werken in een aantal cycli met een multidisciplinair team waarin de eindgebruikers ook vertegenwoordigd zijn, leert het team wat nu eigenlijk echt de bedoeling is van het project en hoe dat het beste bereikt kan worden. In elke cyclus zit een reflectiemoment en is er de mogelijkheid tot bijsturen.

Bij waterval projecten wordt vooraf een doel geformuleerd. Reflectie op het oorspronkelijke doel is in mindere mate mogelijk. Het oorspronkelijk geformuleerde doel wordt niet (geheel) bereikt. En geconstateerd wordt dat zowel het oorspronkelijk geformuleerde doel als het bereikte doel helemaal niet de daadwerkelijk gewenste doelen zijn.



Figuur 10: Door cyclisch te werken wordt een beter resultaat bereikt (Illustratie: Rachèl Harmsen)

De laatste reden waarom een cyclische projectmethodiek tot een beter resultaat leidt is dat de klant na elke cyclus acceptatietests uitvoert. Bovendien wordt de kwaliteit verder verstevigd door van meet af aan uit te gaan van tests als criterium voor goed werkende functionaliteit. Programmeurs moeten daarvoor voordat ze hun code gaan schrijven zogenaamde 'falende tests' definiëren (unit tests). Alleen als de code de falende test passeert, is hij goed.

Hoe werkt testgericht werken? Bij testgericht werken moet de programmeur voordat hij nieuwe code schrijft, bewijzen dat er geen bugs in de nieuwe code zitten. Dit doet hij door een test te bedenken die een eventuele bug aan zal wijzen (falende tests), voordat hij gaat coderen.
Bijvoorbeeld, stel dat een programma gemaakt moet worden voor het berekenen van de juiste hoeveelheid wisselgeld die je moet terugkrijgen van een snoepmachine. Allereerst moet er getest worden of er al een functie bestaat die wisselgeld geeft. Deze functie wordt 'geef_wisselgeld' genoemd. Een simpele test zal gemaakt worden en als die uitgevoerd wordt dan zal er blijken dat er nog niet een functie 'geef_wisselgeld' bestaat. Wanneer de programmeur vervolgens de functie aanmaakt maar nog geen inhoud geeft, zal deze test slagen.

De volgende stap is dat getest moet worden of de machine de juiste hoeveelheid geld teruggeeft als er iets gekocht wordt. Stop 60 cent in de machine en koop een item van 50 cent. De test zal niet slagen, want de functie is nog leeg. De programmeur zal vervolgens de code schrijven. In de functie 'geef_wisselgeld' schrijft hij dat de hoeveelheid terug te geven wisselgeld de hoeveelheid geld is die er in de machine is gestopt minus de kostprijs van het gekozen snoep. De test zou nu moeten slagen. Enzovoort, voor elk deel van de software moet vooraf een test bedacht worden (voorbeeld vertaald en aangepast uit Chromatic, 2003, p. 27 e.v.)

Niet alleen de code moet (technisch) getest worden, ook de functionaliteiten moeten getest worden (acceptatie tests). Daarvoor moet de klant voor het coderen bepalen onder welke condities de nieuw te bouwen functionaliteiten geaccepteerd gaan worden. Bijvoorbeeld hoe snel een computer moet reageren op een bepaalde actie van een gebruiker. Met name het vooraf aangeven onder welke condities de nieuwe functionaliteit geaccepteerd zal worden, blijkt moeilijk en tijdrovend. Toch is de actieve rol van klanten bij het testen zeer bepalend voor het succes van het project.

Realistischer begroting van tijd en geld
Bij een cyclische projectmethodiek worden de te realiseren functionaliteiten niet vastgepind aan het begin van het project. De beschikbare tijd wordt wel vastgelegd. Afgesproken wordt welke richting het project opgaat en onderweg wordt gekeken wat echt nodig, zinnig en realistisch is met betrekking tot de nieuw te maken software. Bij cyclische projecten worden dus niet de te realiseren functionaliteiten als vaststaand doel beschouwd, met als risico dat de benodigde uren (en dus benodigd geld) extreem uit de hand kunnen lopen. Om dat te voorkomen wordt de beschikbare tijd als uitgangspunt genomen en wordt gaandeweg gekeken wat men op realistische wijze kan maken in de beschikbare tijd.

Dit is ook de reden dat cyclische projectmanagementmethodes veel vriendelijker zijn voor het projectteam. Het team doet wat het kan in de gestelde tijd maar wordt niet onder druk gezet om onrealistische planningen te halen of te werken met veel te krappe begrotingen.

Ook het beheersen van externe leveranciers is gemakkelijker. Bij de watervalmethode zit je op een gegeven moment helemaal vast aan één leverancier tot het einde van het project, of die leverancier nu goed presteert of niet. Bij de cyclische werkwijze is het in principe mogelijk om per cyclus of zelfs per op te leveren component afspraken te maken en desnoods te wisselen van leverancier.

Voorwaarden voor cyclisch projectmanagement

Om cyclisch projectmanagement te kunnen toepassen moet aan een aantal voorwaarden voldaan zijn:

1. Actieve betrokkenheid van gebruikers/klanten
Bij cyclisch projectmanagement vindt het opstellen van eisen- en wensen, het ontwerpen, realiseren en testen in elke cyclus plaats. Dat betekent dat er veel beslissingen genomen moeten worden in een cyclus. Om in staat te zijn de wensen van de klant goed te laten uitkomen in de software, *moet* de klant actief meedoen in het projectteam. Hij moet zo goed mogelijk zijn wensen uitleggen aan de programmeurs en ontwerpers. Men moet denken aan wekelijkse of minimaal tweewekelijkse aanwezigheid in het projectteam.

Klanten houden zich binnen het project bezig met het bepalen van de gewenste functionaliteiten en de planning van de cycli. Ze denken mee over de acceptatietests, ze keuren tussenresultaten goed of af en denken mee over de algemene richting van het project. Overigens geldt ook voor de watervalmethode dat een actieve opstelling van de klant tot betere resultaten leidt.

2. Team mag besluiten nemen
Binnen een cyclus moet het projectteam geautoriseerd zijn om dat te doen wat het denkt dat het beste is. Als het projectteam niet deze macht heeft, werkt het cyclische projectmanagementmodel niet. Wanneer er tijdens een cyclus constant autorisatie nodig is van meerderen, zal dit leiden tot stagnatie. Bovendien weten buitenstaanders vaak niet goed wat er speelt omdat ze niet actief deelnemen in het projectteam, wat het voor hen moeilijk maakt zinnige beslissingen te nemen.

3. Projectresultaat (software) is opdeelbaar in kleinere delen
Bij cyclisch projectmanagement worden delen van het project uitgevoerd in een aantal cycli. Dat kan alleen als de te realiseren software opdeelbaar is in een aantal min of meer los van elkaar staande onderdelen.

4. Vanuit het management worden voornamelijk globale eisen ten aanzien van de software gesteld en niet direct concrete en specifieke eisen.
Een van de sterke punten van cyclisch projectmanagement is dat de klant, ontwerpers, programmeurs en eventueel testers nauw samenwerken binnen de cycli. Als er bij het project al direct in het begin heel specifieke en concrete eisen zijn gesteld aan de op te leveren software, dan belemmert dat de vrijheid van het projectteam om naar eigen inzicht ontwerpkeuzes te maken. Veel eisen aan een project blijken gaandeweg aangepast te moeten worden en moeten dus niet in het begin (te) vast liggen.

5. De activiteiten zijn voor de klant inzichtelijk
Als er binnen een cyclus veel technisch werk moet gebeuren, dat te moeilijk te begrijpen is voor de klant, bestaat er een risico dat hij niet in staat is goed mee te doen met het team. Hij heeft feitelijk niet veel in te brengen bij de ontwerpbeslissingen die genomen moeten worden.
Een dergelijk risico bestaat ook wanneer de voortgang voor de klant niet goed zichtbaar is. Bijvoorbeeld omdat er veel aan de code gewerkt is en minder aan de gebruikersinterface. Het is belangrijk dat de klant voldoende inzicht heeft in de inhoud en in de voortgang van een cyclus om ervoor te zorgen dat hij niet buiten spel komt te staan.

6. Een stap terug moet mogelijk zijn.
Ook bij cyclisch projectmanagement bewandelt het team soms een weg, waarvan men later moet concluderen dat het niet de juiste was. Dan moet het mogelijk zijn om een stap terug te doen. Als een nieuwe module is gemaakt in een cyclus en het blijkt dat hij niet bevalt, moet de oude module terug in gebruik kunnen worden genomen. Dit stelt met name eisen aan de archivering en documentatie van de projectmaterialen. CVS (Concurrent Versioning Systeem) of Subversion zijn hier bijvoorbeeld een nuttig hulpmiddel voor (zie bijlage 3 voor een lijst van hulpmiddelen).

7. Programmeurs moeten naast programmeren ook kunnen communiceren met klanten en vice versa. Teamleden moeten in staat zijn conceptueel te denken. Discipline is nodig om het testgericht werken vol te houden.

8. De organisatie waarbinnen het project plaatsvindt moet ook voldoende steun bieden aan deze manier van werken. Systemen voor tijdschrijven, archivering en planning zijn nodig om de projecten te ondersteunen. Deze registratiesystemen zorgen voor de transparantie welke nodig is voor een goede verdeling van resources over de projecten en in de tijd.

9. Projecten moeten voldoende prioriteit hebben, teamleden moeten worden vrijgemaakt voor projecten. Het werkt niet als teamleden in veel projecten tegelijk moeten werken. Als een organisatie onvoldoende ingesteld is op het werken met projecten dan zal de lenigheid van cyclisch projectmanagement eerder tot chaos leiden. Overigens is de watervalmethode ook gebaat bij een organisatie die ingesteld is op het werken met projecten (zie Wijnen, 2004, p. 111).

Een directeur van een softwarehuis die meer een visionair was dan een manager had bijna elke maand een goed idee en startte voortdurend nieuwe projecten in zijn bedrijf. Daardoor werden oude projecten nooit helemaal afgemaakt en deden de medewerkers soms wel vijf projecten tegelijk. De charismatische directeur had wel eens een boekje over rapid application development (RAD) gelezen en was er erg enthousiast over. Met name over het aspect 'rapid'. Hij had boven het kopieerapparaat de uitgangspunten van RAD opgehangen en ging er vervolgens vanuit dat iedereen wel zou gaan werken volgens die uitgangspunten. Het was immers een prachtige methode.

Risico's van cyclisch projectmanagement

Nu kan men opperen dat bij cyclische projectmanagementmethodes het zou kunnen voorkomen dat er onvoldoende tijd is voor het realiseren van de gewenste functionaliteiten. De hoeveelheid tijd is immers gefixeerd dus dat betekent dat er wellicht minder functionaliteit gemaakt zou kunnen worden dan aanvankelijk werd aangenomen.

Dit risico is inderdaad aanwezig, maar bestaat ook bij de watervalmethode. Bij de watervalmethode is er in de definitiefase een uitgebreide analyse van eisen en wensen. Op basis van die analyse zou je verwachten dat een betere planning van tijd mogelijk is. In de praktijk blijkt dit vaak niet het geval, om eerder genoemde redenen, en worden functionaliteiten uiteindelijk eveneens weggelaten, omdat het geld op is om ze te kunnen realiseren.

Bij cyclische methodieken wordt er op een pragmatische wijze omgegaan met eisen en wensen. Bijvoorbeeld door per cyclus de eisen en wensen in te delen volgens de MoSCoW regels (Stapleton, 2003):

- Must Have: eisen die essentieel zijn voor het systeem
- Should Have: belangrijke eisen die men graag wil
- Could Have: eisen die wenselijk zijn, maar die eenvoudig weggelaten kunnen worden
- Want to Have: but will not have this time round: eisen die wel kunnen wachten tot later

Ondanks dat het zou kunnen dat er voor bepaalde functionaliteiten geen tijd meer is, is de algemene consensus onder de projectmanagers van DANS dat cyclisch werken tot meer klanttevredenheid leidt dan de watervalmethode. In ieder geval worden de verwachtingen over en weer beter gemanaged en wordt er iets opgeleverd wat aantoonbaar werkt, al is dat wellicht minder uitgebreid dan aanvankelijk gehoopt.

Als nadeel van XP wordt vaak genoemd dat er veel macht bij de programmeurs komt te liggen. Als ze die macht misbruiken kunnen ze zich verschuilen achter het gebrek aan technische kennis bij de klant. Om dit te voorkomen is een sterke projectleider nodig die zowel de belangen van de programmeurs als die van de klant behartigt. Deze projectleider assisteert de klant bij het kiezen en plannen van de cycli, het inzichtelijk maken van de technische achtergronden bij keuzes en het uithanden nemen van administratie en rapportage.

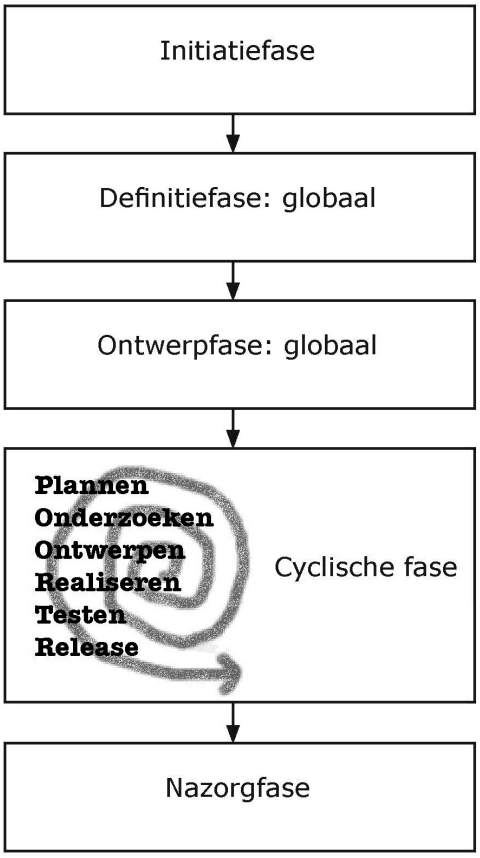
Tenslotte kan nog als nadeel van XP worden aangehaald dat er een stijle leercurve is met betrekking tot de introductie van de methodiek bij zowel programmeurs, management als klanten.

6. DANS werkwijze voor software ontwikkeling

In het voorlaatste hoofdstuk van dit handboek voor projectmanagement wordt de werkwijze geschetst die DANS toepast voor projectmanagement bij softwareprojecten.

Als groot nadeel van de cyclische werkwijze wordt wel genoemd dat teams gelijk aan de slag gaan. Er wordt dan te weinig nagedacht over wat er nu precies gewenst is. Verwachtingen van eventuele klanten of opdrachtgevers worden niet goed gemanaged. Afspraken over gewenste resultaten worden onvoldoende gemaakt. Dit is de andere kant van de cyclische werkwijze in tegenstelling tot de watervalaanpak waarbij in het begin alles wordt vastgelegd.

Om dit dilemma te omzeilen wordt binnen DANS voor softwareontwikkeling met het beste van twee methoden gewerkt. De projecten starten met de watervalmethode, zodat er goed nagedacht wordt over eisen, wensen en ontwerp. Na de ontwerpfase wordt overgegaan op cyclisch werken, waarbij flexibel omgegaan kan worden met eisen, wensen en ontwerpen. Voor het cyclische deel van de DANS-methode wordt gebruik gemaakt van eXtreem Programming (XP) (Chromatic, 2003, [ii], [iii]). Binnen de cycli vindt nadere definiëring, nader ontwerp, realisatie en testen plaats. Als de software vervolgens voldoende is ontwikkeld start de nazorgfase. Hieronder wordt deze werkwijze stap voor stap verder beschreven.



Figuur 11: Schematische weergave van de DANS-methode voor softwareontwikkeling

Initiatiefase

De initiatiefase begint bij een idee voor een project. Er is nog geen budget beschikbaar voor het project. Doel van deze fase is het schrijven van een projectplan op basis waarvan financiering intern of extern aangevraagd kan worden.

Activiteiten initiatiefase:
- uitwerken idee
- onderzoeken draagvlak
- contacten leggen met eventuele partners
- financieringsmogelijkheden onderzoeken
- eerste globale inschatting van de GOKIT factoren voor het project
- concrete inschatting GOKIT factoren voor de definitiefase
- begrenzen van het project
- schrijven projectplan
- aanvragen financiering of contractafspraken maken met een eventuele klant

Resultaat initiatiefase:
- goedgekeurd en gefinancierd projectplan
- (eventueel) contract met klant

Uitvoering/Beslissingen:
- (beoogd) projectleider
- opdrachtgever
- (eventueel) klant

Hulpmiddelen:
- Zie bijlage 10 voor een model van een projectplan

Als het mogelijk is om de financiering in trances te krijgen is dit te verkiezen boven financiering in één keer na de initiatiefase. Bij gefaseerde financiering wordt in de initiatiefase een financieringsverzoek gedaan voor een relatief klein bedrag in de initiatiefase voor het uitvoeren van de definitiefase en ontwerpfase. Op basis van de uitkomst van deze fases wordt aan het einde van de ontwerpfase een tweede financieringsverzoek gedaan voor de rest van het project.

Definitiefase

Nadat een projectplan is goedgekeurd, komt het project in de tweede fase, de definitiefase. In deze fase worden de eisen en wensen die aan een projectresultaat gesteld worden zo goed en compleet mogelijk bepaald. Het gaat er om de verwachtingen van de betrokken partijen over wat het projectresultaat moet zijn boven tafel te krijgen. Hierbij kan de lijst uit hoofdstuk 1 als geheugensteun dienen:

- randvoorwaarden
- functionele eisen
- operationele eisen
- ontwerpbeperkingen

Het is heel belangrijk dat alle betrokkenen in een project kunnen meedenken in de definitiefase. *Vooral* ook de eindgebruikers die het projectresultaat gaan gebruiken.

Activiteiten definitiefase:
- eisen en wensenlijst opstellen samen met opdrachtgever, (eventuele) klant, eindgebruikers, experts, projectteam
- eisen en wensen afwegen
- haalbaarheid eisen en wensen toetsen
- rapportage aan opdrachtgever en/of klant van eisen en wensen
- rapportage van de GOKIT factoren tot nu toe (gerealiseerd)
- nieuwe prognose van de globale GOKIT factoren voor de rest van het project
- prognose van de concrete GOKIT factoren voor de ontwerpfase

Resultaat definitiefase:
- goedgekeurde (voorlopige) eisen en wensenlijst
- goedgekeurde GOKIT rapportage en prognoses

Uitvoering:
- (beoogd) projectleider
- opdrachtgever
- (eventueel) klant
- eindgebruikers
- experts
- (toekomstige) programmeurs en ontwerpers
- systeembeheerder (in verband met eisen en wensen in de nazorgfase)

Beslissingen/goedkeuring:
- projectleider
- opdrachtgever
- (eventueel) klant

Hulpmiddelen:
- zie bijlage 10 voor een model van een projectplan

Bij de watervalmethode geldt dat er in principe na de definitiefase geen andere of aanvullende eisen meer gesteld mogen worden aan het project. De eisen en wensenlijst vormt als het ware de basis voor het contract tussen het projectteam en de opdrachtgever of klant. De eisen en wensenlijst is datgene waar het projectresultaat uiteindelijk aan moet voldoen.

Omdat we gezien hebben dat dit bij software projecten tot problemen leidt, wordt deze strikte manier van werken hier niet gevolgd. De definitiefase wordt hier gebruikt om een onderzoek naar de eisen en wensen te laten plaatsvinden zodat het project zo goed mogelijk richting krijgt. Ook wordt er beschreven wat er allemaal *niet* gemaakt wordt. Dit om de verwachtingen tussen klant en makers helder te krijgen. Mocht door het voortschrijdende inzicht in de cyclische fases blijken dat bepaalde eisen en wensen anders ingevuld moeten worden, dan is dat bij deze werkwijze mogelijk (uiteraard in overleg en goed gedocumenteerd)

Ontwerpfase

Met de eisen- en wensenlijst uit de definitiefase kan het projectteam keuzes maken over hoe de software eruit moet komen te zien. De ontwerpfase van ICT projecten heeft een ontwerpdocument als resultaat. In het ontwerpdocument wordt het concept nader uitgewerkt en in grote lijnen wordt er een technisch ontwerp uitgewerkt. Doel is om te onderzoeken hoe de software er zowel technisch als functioneel uit zou kunnen zien (een voorbeeld van een functioneel ontwerpdocument is te vinden op [iv]).

In dit kader is het handig om in de ontwerpfase te werken met dummy's die voorgelegd worden aan de opdrachtgevers en/of klanten en eindgebruikers. Een dummy is een snel in elkaar gezet, niet werkende of slechts deels werkend stuk software dat vooral dient om het ontwerp te evalueren. Dummy's hebben het voordeel boven schema's op papier dat ze er uitzien als het beoogde eindproduct.

Bij de watervalmethode is het na de goedkeuring van de ontwerpdocumenten door de klant in principe niet meer mogelijk op ontwerpbeslissingen terug te komen. Bij de DANS-werkwijze kan dat wel. De ontwerpdocumenten dienen als eerste uitgangspunt voor de bouwers, maar mocht door voortschrijdend inzicht blijken dat er wijzigingen nodig zijn, dan kunnen die doorgevoerd worden.

Als in de cyclische fasen besloten wordt het ontwerp aan te passen, dan moet die wijziging niet alleen geprogrammeerd worden, maar ook bijgewerkt worden in een zogenaamd projectlog.

Als naar mening van het projectteam het ontwerp voldoende is uitgewerkt, start de cyclische fase.

Activiteiten ontwerpfase:
- Ontwerpdocumenten opstellen
- Mock-ups (dummies) maken en evalueren met klant
- rapportage gekozen ontwerp
- rapportage van de GOKIT factoren tot nu toe (gerealiseerd)
- nieuwe prognose van de globale GOKIT factoren voor de rest van het project
- prognose van de concrete GOKIT factoren voor de cyclische fase

Resultaat ontwerpfase:
- goedgekeurd ontwerpdocument
- goedgekeurde GOKIT rapportage en prognoses

Uitvoering:
- projectleider
- designers
- programmeurs
- (eventueel) eindgebruikers voor evaluatie ontwerpen

Beslissingen/Goedkeuring:
- projectleider
- opdrachtgever
- (eventueel) klant

Hulpmiddelen:
- zie [i] voor een handleiding voor het maken van een ontwerp document
- zie de bijlage 10 voor een model van een projectplan

Cyclische fase

De werkwijze in de cyclische fase is ontleend aan XP. In deze fase wordt een aantal cycli na elkaar uitgevoerd. Een cyclus duurt maximaal een tot twee weken. In *elke* cyclus vinden de volgende activiteiten plaats:
- plannen
- onderzoeken van functionaliteiten
- ontwerpen van functionaliteiten
- realiseren van functionaliteiten
- testen van functionaliteiten

- opleveren van functionaliteiten

Plannen
Aan het einde van de ontwerpfase is een inschatting gemaakt van het aantal cycli dat nodig is voor het bereiken van het projectdoel. Dit gebeurt op basis van het functionele en technische ontwerp. Een cyclus duurt nooit lang, ergens tussen de twee en vier weken. In een cyclus komen altijd de activiteiten plannen, onderzoeken, ontwerpen, realiseren, testen en opleveren voor. Per cyclus wordt dus maar gewerkt aan enkele functionaliteiten, soms maar aan één.

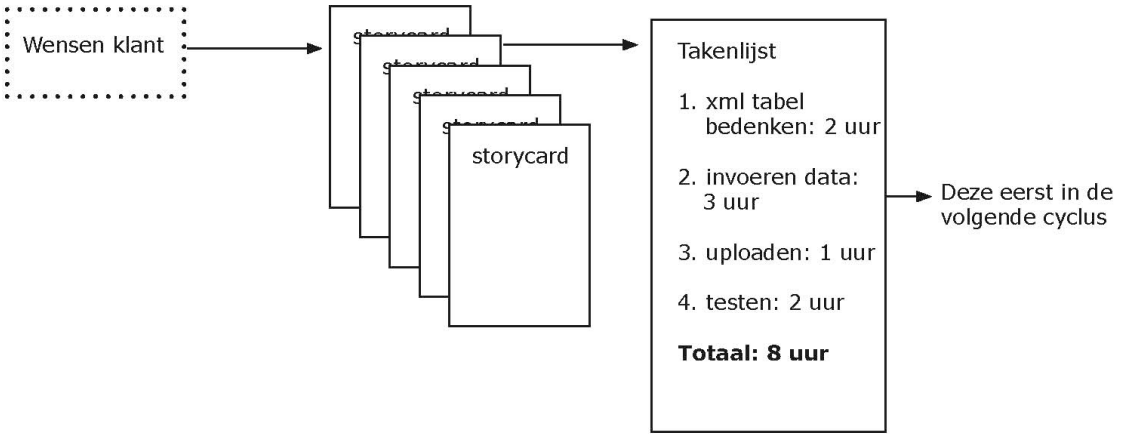
De spelregels voor het plannen zijn als volgt. De gewenste functionaliteiten worden door de projectleider samen met de klant op zogenaamde storycards opgeschreven. Begonnen wordt met de functionaliteiten die bepaald zijn in de definitie- en ontwerpfase op storycards te schrijven. Op basis van die storycards maken de programmeurs een takenlijst, een lijst van dingen die zij moeten doen om die functionaliteit te realiseren. Die taken mogen in de regel niet langer duren dan een paar uur. Als een taak langer duurt, moet hij gesplitst worden in verschillende subtaken of moet de storycard gesplitst worden in meerdere storycards. Als een storycard te veel werk is om in één cyclus plaats te vinden, moet hij worden teruggegeven aan de klant. De klant moet de wensen dan opsplitsen en verdelen over meer storycards.
Per taak geeft de programmeur aan hoeveel tijd hij er voor nodig denkt te hebben. Zo ontstaat een ureninschatting per storycard. Op basis van de ureninschattingen van de storycards kan de klant nu samen met de projectleider aangeven welke van de functionaliteiten ze als eerste gerealiseerd willen zien in de eerstvolgende cyclus.

Hoe lang duurt het om een website te maken en deze te vullen met 50 pagina's tekst en een aantal foto's? Een snel antwoord van de ontwerper dat het 'ongeveer twee weken duurt' is veel te grof. Het zou best wel eens veel langer kunnen duren. Wordt deze taak uitgesplitst dan blijkt dat er een CSS gemaakt moet worden, overleggen met de opdrachtgever over het ontwerp, het ontwerp programmeren in xhtml, de teksten inkorten voor het internet, de pagina's vullen met de teksten, de foto's geschikt maken voor het internet, de foto's plaatsen, controle door de klant en de laatste foutjes eruit halen.

Door het werk uit te splitsen in kleinere delen, blijkt het veel meer werk dan aanvankelijk gedacht. Ook realiseert de opdrachtgever zich dat hij ook een aantal dingen moet doen. Als nu per taak ingeschat wordt hoeveel uur werk het is, zal het een veel betere totaalschatting opleveren (en zal blijken dat het niet in twee weken kan).

Als de programmeurs aan de slag gaan, houden ze hun uren bij die ze nodig hebben voor het uitvoeren van een taak. Vaak blijkt dat ze meer uren nodig hebben dan ze aanvankelijk zelf geschat hadden. Doordat ze een terugkoppeling krijgen op hun eigen inschattingen, leren de programmeurs steeds betere schattingen te maken.
Uit de praktijk blijkt dat naarmate een project een tijdje loopt er een redelijk constante factor verschil is tussen de schattingen van de programmeurs en de werkelijk benodigde aantallen uren. Deze factor wordt de 'dragfactor' genoemd. De dragfactor kan na een paar cycli bepaald worden op basis van het gemiddelde verschil tussen de geschatte en benodigde uren. De dragfactor wordt gebruikt in planningen van toekomstige cycli en in rapportages. Met name het aantal uitstaande storycards met takenlijsten maal de dragfactor is een redelijk betrouwbare indicatie van de hoeveelheid tijd die nog nodig is voor het realiseren van de rest van het project.

Het aantal uren voor het project is beperkt, dus dat betekent dat er keuzes gemaakt moeten worden. Doordat er een redelijk inzicht is in het aantal benodigde uren voor het realiseren van een storycard, kan een goede afweging gemaakt worden tussen de verschillende storycards. Sommige storycards zullen simpeler uitgevoerd moeten worden, andere storycards komen als geheel te vervallen. Belangrijk uitgangspunt bij het bepalen van de volgorde is dat risicovolle storycards als eerste behandeld moeten worden. Dit om zo snel mogelijk de belangrijkste risico's uit te schakelen. Daarnaast gelden de MoSCoW-regels of een simpeler prioriteitstelling van 1 tot 5.



Figuur 12: Plannen van de functionaliteiten in de cycli

Onderzoeken en ontwerpen functionaliteiten
Door het maken van de takenlijsten voor de planning vindt het eerste onderzoek van de functionaliteit plaats. Door het vooraf bedenken welke deeltaken nodig zijn bij het realiseren van een functionaliteit, krijgt de programmeur meer inzicht in de functionaliteit zelf. Voor het verder onderzoeken en ontwerpen van de gewenste functionaliteit is de betrokkenheid (aanwezigheid) van de klant essentieel. Hij moet aangeven wat hij precies wil. De klant zal in het begin van het project veel contact hebben met de programmeurs. Later in het project kan dat afnemen, al moet er voor gewaakt worden dat programmeurs niet gaan denken voor de klant en dan verkeerde aannames doen.

Realiseren van de functionaliteiten
Nadat de klant en de makers een ontwerp voor de gewenste functionaliteit zijn overeengekomen, wordt deze gebouwd. Programmeurs werken daarbij altijd in paren ('pairprogramming' in XP termen). Dit lijkt wellicht improductief, maar pairprogramming biedt vele voordelen:

- kennis van twee programmeurs wordt gecombineerd
- minder tijd nodig voor overdracht van kennis of code binnen een project
- minder fouten tijdens het programmeren
- sneller knelpunten oplossen

Er is nog een ander voordeel: het grootste probleem is beginnen met programmeren. Eenmaal begonnen dan loopt het verder wel. Door paarsgewijs te programmeren, worden de programmeurs eerder door elkaar aangespoord te starten met werken.

Joel Spolsky was programmeur bij Microsoft en realiseerde zich dat hij maar zo'n drie uur per dag effectief werkte [v]. Vaak nog minder. Het lijkt erop dat meer collega's dat hadden. De rest ging op aan koffiedrinken, e-mails lezen, internetten, praten met collega's en de prachtige kantoortuin bekijken. Door te werken met een ander wordt je gemotiveerd en is de start makkelijker.

Overigens vraagt pairprogramming veel van de concentratie van programmeurs en niet alle programmeurs vinden het prettig. Bovendien werken niet alle combinaties van programmeurs prettig samen. Om deze nadelen te verzachten kan een team bijvoorbeeld besluiten om niet meer dan een halve dag per dag pairprogramming toe te passen (zie verder [vi] voor een discussie over pairprogramming).

Testen en opleveren

Essentieel is dat elke cyclus tot een release van een nieuw deel van de software leidt en dat elk deel dat opgeleverd wordt, getest is. Testen bestaat uit een unit test (test door de programmeurs) en een acceptatietest (test door de klant). Ook hierbij is de inzet van de klant dus nodig.

Achterliggend idee van het testen in elke cyclus is de theorie dat het exponentieel duurder wordt om fouten te herstellen naarmate het langer heeft geduurd voor ze gevonden zijn. Uitgangspunt bij het opleveren van software in elke cyclus is dat de klant zo snel mogelijk waar voor zijn geld krijgt en dat er snel feedback mogelijk is van de gebruikers naar de programmeurs. De klant ziet duidelijk de voortgang van het project. Dit is vooral psychologisch van belang en kan de verhouding met de klant aanzienlijk verbeteren.

Op een punt verschilt de werkwijze van DANS essentieel met de werkwijze van XP. XP schrijft voor dat er niet vooraf een ontwerp gemaakt mag worden voordat met programmeren wordt begonnen. Dit om maximale flexibiliteit te bereiken en niet al dingen vast te zetten, die later niet handig blijken te zijn. Naar mening van de auteur en adviseurs van dit handboek is het echter wel nuttig om te beginnen met het maken van een ontwerp in de definitiefase en ontwerpfase. De functionaliteiten die in die fases worden bepaald, mogen bij de DANS werkwijze wel worden aangepast in de cyclische fase in tegenstelling tot de werkwijze van de watervalmethode.

Activiteiten cyclische fase:

- doorlopen van een aantal cyclussen waarin onderzocht, ontworpen, gerealiseerd, getest en opgeleverd wordt
- opstellen van storycards
- keuzes maken uit de storycards
- plannen van de cycli
- voortgang bewaken (GOKIT)
- (aan het einde) prognose van de concrete GOKIT-factoren voor de nazorg fase

Uitvoering/beslissingen

- projectleider
- opdrachtgever of klant
- (eventueel) eindgebruikers voor testen en ontwerpen
- programmeurs en ontwerpers

Hulpmiddelen:

- zie bijlage 3 voor hulpmiddelen voor het registreren en bijhouden van de storycards en takenlijsten

Nazorgfase

Nadat voldoende resultaat is bereikt in de cyclische fase, start de nazorgfase. In deze fase moet het projectresultaat geborgd worden. Het hangt van het soort project en de afspraken met de opdrachtgever of klant af wat die borging inhoudt. Bij een onderzoeksproject volstaat wellicht een eindrapport, bij de ontwikkeling van een nieuw product zal meer nazorg nodig zijn.

De meeste problemen in de nazorgfase ontstaan omdat er bij het begin van het project geen duidelijke afspraken zijn gemaakt tussen klant of opdrachtgever en projectteam. Denk bijvoorbeeld aan de volgende punten:

- Hoe lang moet de nazorg duren?
- Waar bestaat de nazorg uit?
- Hoe snel moeten fouten hersteld worden?
- Zit er garantie op het projectresultaat?
- Wie is er verantwoordelijk voor bugs die gevonden worden ná het project?
- Moet er documentatie bij het projectresultaat worden geleverd?
- Is er training en/of opleiding nodig voor gebruikers?
- Wie is er verantwoordelijk voor updates?
- Wie wordt de eigenaar van de code, wie mag de code wijzigen?
- Wie betaalt de bovenstaande punten?

Het is belangrijk te beseffen dat een projectorganisatie gericht is op tijdelijke activiteiten en derhalve niet ingericht is om (lang) ondersteuning te bieden op de software die het zelf heeft ontwikkeld. Voor de langere termijn moet een andere vorm voor ondersteuning gevonden worden. Er zijn speciale (commerciële) organisaties voor het beheren van software. Deze organisaties bieden helpdeskondersteuning, trainingen, serverbeheer, applicatiebeheer en dergelijke. Voor kleine non-profit initiatieven zullen deze organisaties veelal (te) duur blijken.

Een ander traject dat bewandeld kan worden om de continuïteit van de software te waarborgen, is de software open source maken. Hier wordt een organisatie voor ingericht zodat een groep van ontwikkelaars en gebruikers in staat is om de software te onderhouden en te ondersteunen.

Activiteiten nazorgfase:
- rapportage van de GOKIT-factoren van het project
- opstellen en indienen eindverantwoording
- ontbinden team
- overdracht naar beheersorganisatie

Resultaat nazorgfase:
- projectverantwoording
- overdrachtsdocumenten

Uitvoering:
- projectleider
- teamleden
- systeembeheerder

Beslissingen/Goedkeuring:
- projectleider
- opdrachtgever
- (eventueel) klant

Hulpmiddelen:

- zie bijlage 10 voor een model van een projectplan

7. Programmamanagement

Tot zover is er in dit boek voornamelijk geschreven over het managen van een enkel project. In dit hoofdstuk wordt er dieper ingegaan op de vraagstukken die ontstaan wanneer een organisatie meer projecten tegelijk uitvoert. Dit vraagstuk speelt zich vooral af op het vlak van de relatie tussen het (hogere) management team en de projectleiders en staat verder los van de keuze voor een watervalmodel of een cyclische projectmanagement aanpak.

Coördinatie van projecten

In eerdere hoofdstukken is beschreven dat de GOKIT factoren de parameters zijn waarop projecten gerapporteerd en gestuurd worden. Ook bij het afstemmen van meerdere projecten spelen deze GOKIT factoren weer een belangrijke rol:

Geld: om te zien of projecten financieel mogelijk zijn
Organisatie: om onderlinge afspraken te maken over de hiërarchie tussen projecten onderling en projecten en de overige afdelingen
Kwaliteit: om te zien of de doelstellingen van een project passen binnen de strategie van de organisatie
Informatie: Om af te spreken hoe, wat en wanneer er gerapporteerd wordt over een project aan het management team?
Tijd: om te schatten hoeveel inzet van mensen er nodig is in een bepaalde periode om tot een goede verdeling te komen van de medewerkers over de projectteams.

Voor aanvang van een project en na elke projectfase moet de projectleider een inschatting geven van de GOKIT factoren voor de rest van het project. Daarnaast evalueert hij na elke fase de GOKIT factoren tot dan toe. Deze gegevens worden overlegd aan een programmamanager of aan het management team om al dan niet gezamenlijk met de projectleider en externe partijen (klanten, financiers) besluiten te nemen over het project. Hieronder staan een aantal van de belangrijkste besliscriteria beschreven, met name op het vlak van de afstemming van projecten.

Geld
Bij de beoordeling van geldzaken door een programma manager gaat het om de volgende zaken:

- Is het project als geheel en de volgende fase in het bijzonder voldoende gefinancierd?
- Wat zijn eventuele financiële risico's van het project? Moet er een go-no-go moment worden afgesproken?
- Hoe is de liquiditeitsprognose van het project? Ontstaat er een probleem als de inkomsten van een project later zijn dan de uitgaven (bijvoorbeeld als de subsidie pas betaald wordt na afloop van een langdurig project)?

Door alle budgetten op te tellen kan de programmamanager een jaarplanning maken van de projecten. Daarbij let hij er op of er voldoende projectomzet in de pijplijn zit zodat de projectmedewerkers gefinancierd zijn. Bij onvoldoende projectomzet, moeten er nieuwe projecten geworven worden. Bij teveel omzet zal er extra personeel ingehuurd moeten worden en/of moeten projecten worden uitgesteld of zelfs afgeblazen.

Het geeft een beeld of de projecten als geheel financieel rendabel genoeg zijn. Stel dat je als organisatie 10 projecten doet van elk 2000 uur. Je hebt 20 FTE

aan projectmedewerkers in dienst. Verderop zullen we zien dat daarmee ongeveer alle uren van die medewerkers zijn verdeeld. Stel dat deze 20 FTE per jaar een miljoen kosten aan loonkosten. Wanneer deze projecten minder dan een miljoen opbrengen (aan subsidie, interne financiering, commerciële opbrengsten) is er een probleem. Mogelijk moeten de tarieven worden aangepast of moeten onrendabele projecten geschrapt worden.
Voor projecten geldt dat de afwijking tussen begroting en reëel benodigde bedragen groot kunnen zijn. Met name als er sprake is van gesubsidieerde projecten kan de afwijking substantieel zijn (zie Projectrapportage). Om dit soort onzekerheid op te vangen moet de programmamanager een bedrag reserveren voor het opvangen van tegenvallers bij één of meer van de projecten.
Begrotingen zijn soms ook te ruim. In dat geval is het wel zaak dat projectleiders bereid zijn om delen van hun budget in te leveren. Anders kunnen projecten alleen maar binnen budget blijven óf te duur zijn. Als te duur ingeschatte projecten een deel inleveren aan de projecten die aanvankelijk te goedkoop zijn ingeschat zou de organisatie gemiddeld (redelijk) neutraal moeten uitkomen.

Wat doe je als organisatie als je 100.000 euro subsidie hebt gekregen voor een project en het blijkt dat je maar 80.000 euro nodig hebt? Dat geld komt wel op, er zijn zoveel nuttige dingen die nog gedaan kunnen worden voor dat geld. Bijvoorbeeld dat andere project dat 150.000 euro kreeg, maar dat eigenlijk 200.000 euro nodig had. Gelukkig is niet te achterhalen wat die programmeurs allemaal deden in de uren die ze schreven op de projecten. Dus een beetje schuiven met uren en het klopt allemaal weer.

Logisch dat organisaties zo omgaan met hun gesubsidieerde projecten, ze kunnen niet veel anders. Het is een direct gevolg van het feit dat organisaties bij een aanvraag een totaal budget moeten opstellen waar ze later niet meer van mogen afwijken. Een gevolg van deze situatie is dat de (financiële) projectrapportages van gesubsidieerde projecten met een korrel zout genomen moeten worden. Vanuit het oogpunt van projectmanagement is dat jammer omdat nu niet goed te achterhalen is hoeveel projecten echt gekost hebben. Vanuit het oogpunt van accountability van het subsidiegeld is het ook geen goede zaak.

Tijd
Als alle begrotingen van de uren van de projecten die een organisatie in een komende periode gaat uitvoeren bij elkaar opgeteld worden, geeft dat een indicatie van de werkdruk. Op jaarbasis moet het totaal aantal FTE in dienst meer zijn dan alle uren opgeteld uit de projectplannen.

Bijvoorbeeld: 10 projecten 'verbruiken' samen 20.000 uur. In de organisatie werken 20 FTE. Per FTE wordt er in onze organisatie 1700 uur gewerkt. Daarvan is 70% vrij voor projecten en 30% voor algemeen werk (vergaderen, email, reistijd, overige werkzaamheden, studieverlof). Netto komt dat neer op: 20 x 1700 x 70% = 23800 uur per jaar beschikbaar voor projecten. In dit voorbeeld is er dus globaal gezien voldoende capaciteit om deze projecten uit te voeren komend jaar. Veel meer kan er niet bij zonder meer mensen in dienst te nemen (ook hier is een marge nodig).

Bovenstaande controle is een globale controle op jaarbasis. Daarnaast is een fijnere controle van de capaciteit nodig, gespecificeerd naar de taken of rollen die projectmedewerkers hebben. Die 20 FTE moet verder onderverdeeld worden naar verschillende rollen binnen een project (bijvoorbeeld: programmeurs, projectleiders, designers, systeembeheerders). Het kan best zo zijn dat er in totaal

voldoende mensen in dienst zijn om de plannen voor komend jaar uit te voeren, maar bijvoorbeeld te veel projectleiders en te weinig programmeurs.

Tenslotte moet de werklast (aantal verwachte uren) van de projecten niet alleen per jaar overeenkomen met het aantal beschikbare uren van de medewerkers, dit moet ook gelden voor kortere periodes. Als in het bovenstaande voorbeeld alle projecten gepland zijn in de eerste drie maanden van het jaar, is er toch een knelpunt, alhoewel er voldoende mensen in dienst zijn op jaarbasis.

De operationele verdeling van mensen op maand of zelfs weekbasis gebeurt op basis van de plannen van de fases van de projecten of tijdboxen als het gaat om cyclische projecten. Daarnaast is een wekelijks of maandelijks overleg van programmamanager met alle projectleiders nodig om de mensen operationeel in te plannen.

Ten aanzien van tijd wordt een aantal klachten veelvuldig gehoord in projectorganisaties:

- "Er is geen goed beeld van de werkdruk die er heerst. De sales afdeling of het management blijven er maar nieuwe projecten bij doen terwijl we nu al niet goed toekomen aan de projecten die we nu moeten uitvoeren."
- "Als één van de projecten uitloopt, dan heeft dat heel veel gevolgen voor andere projecten. Omdat we de resources moeten delen, betekent dat vertraging in het eerste project ook alle andere projecten vertraagt."

De eerste klacht komt veel voor bij snel groeiende organisaties waar men vooral oog heeft voor het werven van zoveel mogelijk omzet. Door bovenstaande controle mechanismen in te voeren, ontstaat er overzicht over de capaciteit op jaarbasis en op operationele basis van maand tot maand of zelfs week tot week.

De tweede klacht heeft een paar oorzaken. Allereerst is het bij organisaties waar deze klacht voorkomt vaak zo dat men niet goed in staat is oude projecten af te sluiten. Elke keer als het project klaar zou moeten zijn, komt er weer een nieuwe eis van de klant, een nieuwe bug of een uitbreiding op het project. Om dit tegen te gaan is het belangrijk om bij het *begin* van het project zo duidelijk mogelijke afspraken te maken wanneer het project klaar is. Zie ook eerdere hoofdstukken over dit probleem.

Daarnaast geldt er bij de tweede klacht ook dat er sprake is van een (te) hoge werkdruk. Omdat het zo druk is, worden er geen marges tussen de projecten gepland. Hierdoor heeft een kleine uitloop van een vorig project direct gevolgen heeft voor volgende projecten.

Tenslotte kan gezegd worden dat het tweede probleem minder vaak voor komt bij organisaties die cyclisch projectmanagement toepassen, omdat daar met vaste tijdboxen gewerkt wordt.

Om de productiviteit te verhogen wordt personeel vaak op te veel projecten ingezet. Een poging de verloren uurtjes te vullen met andere projecten leidt echter tot ernstige vertragingen. Het volgende voorbeeld illustreert dat (overgenomen en aangepast uit Goldratt, 2002):
Stel dat iemand een project A moet uitvoeren dat bestaat uit drie taken a en een project B moet uitvoeren dat bestaat uit drie taken b. Elke taak kost 5 eenheden tijd.
Als hij ze uitvoert in de volgorde aaa, bbb dan is taak A klaar na 5 + 5 + 5 = 15 eenheden. Taak B is ook na 15 eenheden klaar, gemeten vanaf het moment dat project B start.

Als hij ze echter niet na elkaar maar tegelijk, dat wil zeggen door elkaar moet uitvoeren, ziet het plaatje er anders uit. Stel dat hij een taak a steeds afwisselt met een taak b. De volgorde van werken is dan abababab.

Nu duurt het 5 + 5 + 5 + 5 + 5 = 25 eenheden voordat A klaar is en ook 25 eenheden voordat B klaar is. Daarbij het feit dat het switchen tussen projecten zelf ook tijd kost nog niet meegenomen.

Volgens (Goldratt, 2002) is de praktijk dat veel personeel binnen organisaties ingezet wordt op veel verschillende projecten tegelijk een van de belangrijkste oorzaken van het feit dat projecten (veel) te lang duren. Projecten waar een doorlooptijd van drie maanden voor staat duren in de praktijk soms wel meer dan twee jaar. Als de projecten één voor één na elkaar werden gedaan, waren ze elk na ongeveer drie maanden klaar geweest.

Uit bovenstaand voorbeeld, maar bijvoorbeeld ook uit de kennis van de wachtrijtheorie blijkt dat het niet verstandig is om het personeel te vol te bezetten. Uit kostenoverwegingen op korte termijn zijn management teams er vooral op gefocust om iedereen zo veel mogelijk te laten werken. Hierdoor gaat er veel snelheid in projecten verloren. Het kan best zo zijn dat het verhogen van de bezettingsgraad van het personeel met 10% leidt tot een verhoging van de gemiddelde doorlooptijd van projecten met 40%. Deze kosten van vertraging zijn echter veel minder zichtbaar, zeker in non-profit organisaties.

Tijdschrijven

Het is aan de programmamanager om bij aanvang van een nieuw project bovenstaande controles op financieel vlak en op gebied van urencapaciteit uit te voeren. Dit moet hij tevens doen tussen de fases van een project, dan worden de begrotingen immers bijgesteld.

Maar niet alleen het vooruitkijken is van belang, ook het evalueren van de uiteindelijk gerealiseerde begrotingen (van tijd en geld) moet gebeuren. Zijn de projecten inderdaad rendabel of kostendekkend geweest, zoals begroot? Had dat ene project inderdaad 450 uur nodig of is dit anders gelopen? Deze vragen moeten gesteld en beantwoord worden om de kwaliteit van het projectwerk in de toekomst te verbeteren.

Om deze evaluatie te kunnen uitvoeren is de aanwezigheid van een tijdschrijfsysteem noodzakelijk. Ook voor de wekelijkse sturing van projecten door projectleiders is een tijdschrijfsysteem overigens wenselijk.

In organisaties is er vaak weerstand tegen een tijdschrijfsysteem. Mensen voelen zich gecontroleerd en hebben een hekel aan administratieve klusjes. Een andere bron van weerstand is wellicht dat opeens zichtbaar wordt dat er (vaak) zo weinig vooruitgang geboekt is. Het implementeren van een geschikt tijdschrijfsysteem is een project op zich.

Een belangrijk argument voor het invoeren van een tijdschrijfsysteem is de transparantie die ontstaat. Medewerkers klagen (vaak) over een te hoge werkdruk. Het gebruik van een tijdschrijfsysteem waarin zowel de urenplanningen als de urenverantwoording staat maakt die werkdruk zichtbaar. Wanneer er nu een project 'over de schutting' gegooid wordt door het management of door de sales afdeling kan het projectteam zwart op wit aantonen dat het de komende tijd vol zit en dat het nieuwe project nog even moet wachten.

Waar moet een tijdschrijfsysteem aan voldoen?

- Een tijdschrijfsysteem moet door de hele organisatie gebruikt worden, ook door de boekhoudafdeling.
- Een tijdschrijfsysteem moet overal benaderbaar zijn (bv.webbased).
- Projectleiders moeten snel de rapportages van uren kunnen zien (er mag niet meer dan 1 week tussen tijdschrijven en interne rapportage zitten).
- In het tijdschrijfsysteem moeten de (uren-)planningen verwerkt zijn.

- In het tijdschrijfsysteem moet geschreven kunnen worden op werkpakketten in projectfases van projecten. Het schrijven op algemene projectnamen is niet afdoende.

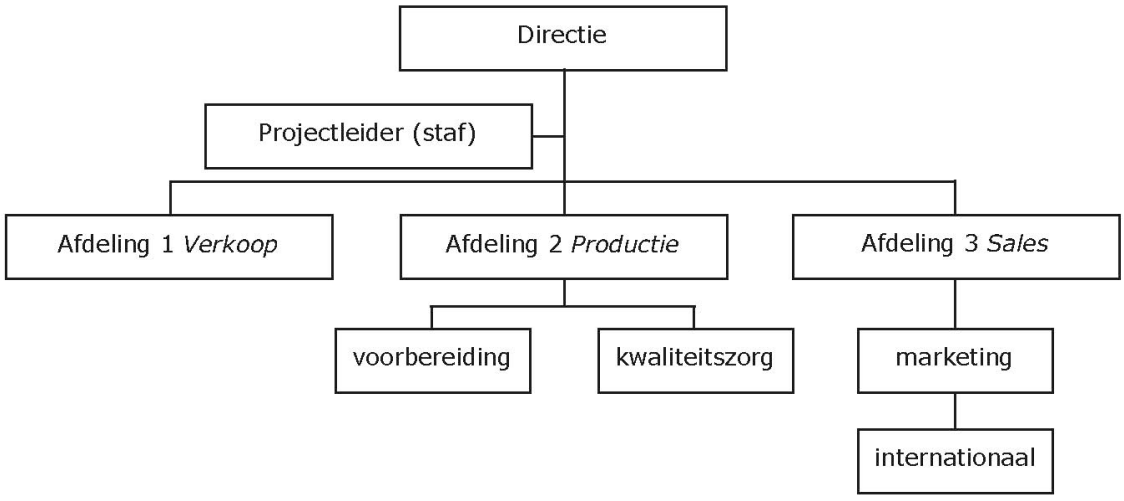
De projectleiders hemel
Voor veel projectleiders is het lastig om de teamleden binnen de begroting te laten werken. Uitvoerenden lijken niet veel om de klok te geven en ook al is er een afspraak dat een klus niet langer mag duren dan een paar uur, er is altijd wel een reden waarom het toch langer duurde.

Bij een architectenbureau hebben ze dit probleem aangepakt door de verantwoordelijkheden om te draaien. De technische tekenaars van dat bureau moeten als interne freelancers hun werk gaan werven bij de projectleiders. Hierbij is onderlinge concurrentie toegestaan. Dus als een tekenaar zegt dat hij het in minder tijd kan dan zijn collega, krijgt hij de klus. Bovendien is 8 uur ook echt 8 uur. Dus als het niet klaar is, moet de technisch tekenaar het in zijn eigen tijd afmaken. Heerlijk om daar projectleider te zijn, misschien wat minder prettig om projectmedewerker te zijn.

Organisatie van meerdere projecten naast elkaar
In (Wijnen, 2004, p. 187 en verder) wordt er onderscheid gemaakt tussen drie structuren van projectmanagement ten opzichte van de moederorganisatie:

- de overleg- of coördinatiestructuur;
- de matrixstructuur;
- de zuivere projectstructuur.

In de overleg- of coördinatiestructuur bestaat het "projectteam" meestal uit alleen de projectleider zelf. Deze projectleider heeft niet veel gezag over andere medewerkers. Hij is bezig met een licht project, vaak onderzoekswerk om een advies aan het management uit te brengen. Als deze projectleider de inzet nodig heeft van andere medewerkers in de organisatie dan kan hij ze informeel om hulp vragen of moet hij dat via hun bazen regelen.

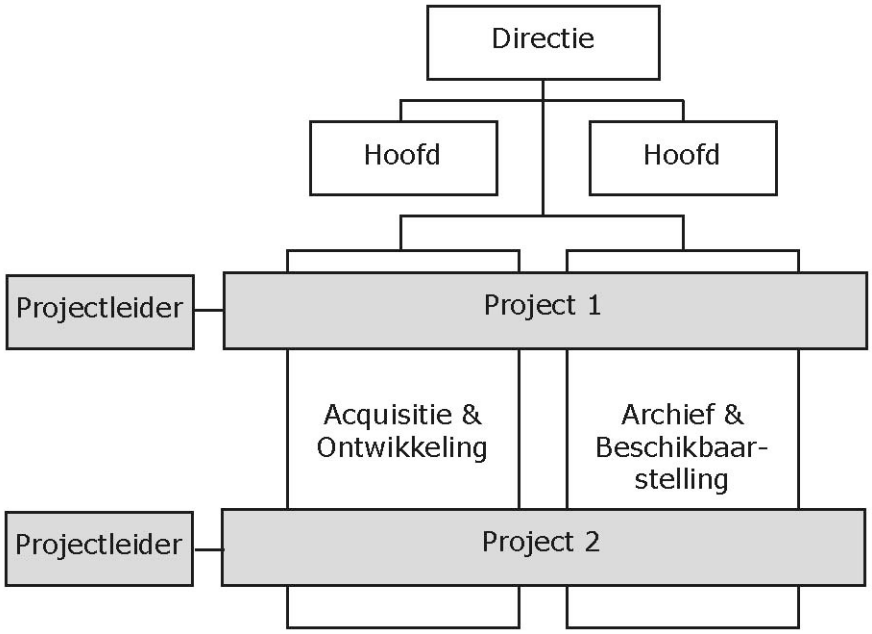


Figuur 13: Een project door middel van de overlegstructuur. De projectleider (vaak staflid) staat geheel buiten de afdelingen.

Bij de matrixstructuur is de organisatiestructuur zodanig opgezet dat de teamleden zowel in een projectteam werken (parttime) als in hun functie binnen de lijnorganisatie (parttime). Ook als mensen binnen meer projecten tegelijk werken zou je kunnen spreken van een matrixorganisatie.

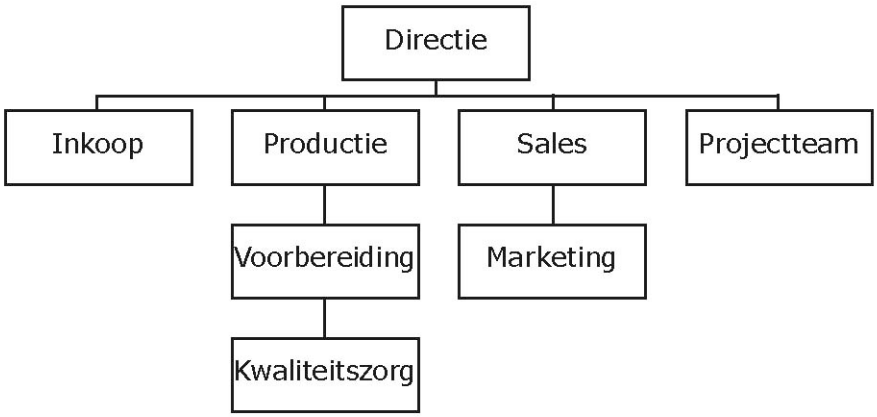
Groot voordeel ten opzichte van de overlegstructuur is dat er sprake is van een projectteam dat bestaat uit meerdere mensen. Hierdoor kan dit projectteam meer realiseren dan in de situatie van de overlegorganisatie, omdat daar de projectleider er voornamelijk alleen voor staat. De projectleider heeft meer gezag in deze situatie. Hij heeft daadwerkelijk de leiding over zijn team. Voor de teamleden kan er op dit punt wel verwarring ontstaan omdat ze nu twee bazen hebben: de projectleider en het hoofd van de afdeling waar ze in werken. Als medewerkers in meerdere projecten tegelijk werken kan het zelfs zo zijn dat ze meer dan twee bazen hebben.

Het is belangrijk dat de afdelingshoofden en projectleider(s) onderling duidelijke afspraken maken over de inzet van mensen. De praktijk leert nog wel eens dat dit niet gebeurt. Van alle kanten wordt er aan medewerkers getrokken en die doen (noodgedwongen) eerst het werk van de baas die het hardst schreeuwt. Dat dit niet altijd het werk is dat de hoogste prioriteit heeft voor de organisatie als geheel, moge duidelijk zijn.



Figuur 14: Projecten georganiseerd volgens het matrix model. Leden van een projectteam hebben twee bazen: de projectleider en het afdelingshoofd.

Bij de zuivere projectstructuur, worden de medewerkers uit de organisatie gehaald en werken zij voor de duur van het project alleen maar in het project. Deze vorm is het meest geschikt voor grote, zware projecten. Het projectteam is in hoge mate autonoom en de teamleden hebben alleen de projectleider als baas. Belangrijk nadeel van deze structuur is dat het duur en ingrijpend is voor de organisatie. Medewerkers worden namelijk gedurende lange tijd van hun oorspronkelijke werk gehaald.



Figuur 15: Schema van de zuivere projectstructuur. Het projectteam staat redelijk autonoom buiten de rest van de organisatie

Het hangt voornamelijk van de projecten af, welke organisatiestructuur het beste is. Voor kleine projecten is de overlegstructuur goed geschikt en voor grote, langdurige en zware projecten is de zuivere projectstructuur het best geschikt. In de praktijk worden veel projecten georganiseerd door middel van een matrixstructuur omdat veel projecten tussen deze twee uitersten in zitten. Bij het matrixmodel is de coördinatie van de projecten wel het lastigst.

Bijlage 1: Top 11 van de grootste vertragers van ICT projecten

In deze bijlage staan de 11 grootste factoren van vertraging van projecten beschreven. Voor een uitgebreide analyse van deze en andere vertragingsfactoren zie McConnell en Goldratt (McConnell, 1996, Goldratt, 2002).

1. Functionaliteitenaanwas
Functionaliteiten aanwas is het fenomeen dat naarmate het project vordert, er steeds nieuwe functionaliteiten bij bedacht en gevraagd worden. Hierdoor komt de software nooit af.

2. Goudranden
Goudranden is het fenomeen dat programmeurs en designers allerlei details van de software of ontwerpen te mooi willen maken. Er wordt veel tijd gestoken in het verbeteren van details, terwijl die verbeteringen niet gevraagd zijn door de klant of opdrachtgever. Vaak voegen de details weinig toe aan het gewenste resultaat.

3. Kwaliteitscontroles overslaan
Door tijdsdruk komen programmeurs of projectteams wel eens in de verleiding om het testen over te slaan. Dit leidt vaak tot meer vertraging dan de tijd die bespaard is met het overslaan van de test. Hoe langer het duurt dat een fout wordt gevonden in software, des te exponentieel meer tijd het kost om hem te herstellen.

4. Te optimistische planningen
Een te optimistische planning leidt tot veel druk op het projectteam. Het team zal aanvankelijk trachten de (niet reële) deadlines te halen. Hierdoor wordt er slordig gewerkt en zullen er meer fouten gemaakt worden, die weer tot vertragingen leiden.

Wees in dit kader vooral beducht op het van bovenaf opleggen van planningen. Soms wordt de wens een project snel(ler) af te ronden vooral ingegeven door strategische motieven, maar als dat niet redelijkerwijs kan, moet dat ook niet geprobeerd worden. Het zal er niet sneller door gaan en het uiteindelijke product zal er niet beter van worden.

5. Werken aan veel projecten tegelijk
Door het werk te versnipperen over veel verschillende projecten (of andere taken), ontstaan er wachttijden waardoor projecten veel vertraging oplopen.

6. Slechte ontwerpen
Het ontbreken van, of slecht uitgevoerde ontwerpen, leidt tot vertragingen omdat er dan in een later stadium veel revisies nodig zijn.

7. Het 'oplossing-voor-alles' syndroom
Het gebruiken van de juiste software voor een project is belangrijk. Het ene softwareplatform is beter geschikt voor bepaalde toepassingen dan het andere. Het is echter een valkuil om te denken dat het gebruiken van bepaalde software tot hele grote productiviteitsverbeteringen leidt.

8. Onderzoeksgeoriënteerde projecten
Projecten waarbij software gemaakt moet worden én onderzoek gedaan wordt, zijn moeilijk te managen. De onzekerheden van het onderzoek zijn heel groot. Onduidelijk is wanneer en of er voortgang geboekt zal worden bij het onderzoek. Wanneer de software ontwikkeling afhankelijk is van onderzoeksresultaten, komt deze regelmatig stil te liggen.

9. Matig personeel
Personeel dat onvoldoende bekwaam is, zal het project vertragen. Daarbij speelt niet alleen de technisch inhoudelijke kennis over het projectonderwerp een rol, maar ook de kennis en vaardigheid om met elkaar het spel van het project te spelen.

10. Klant komt afspraken niet na
Klanten realiseren zich niet altijd dat van hen een behoorlijke bijdrage gevraagd wordt als ze een project laten uitvoeren. Waneer een klant niet of niet op tijd reageert op de onderwerpen waar hij bij moet meedenken, komt het project stil te liggen. Of erger, het team gaat verder zonder dat er goed meegedacht is door de klant, wat later weer tot conflicten kan leiden.

11. Spanning tussen klant en ontwikkelaars
De spanning die kan ontstaan tussen klant en ontwikkelaars, bijvoorbeeld omdat het project niet snel genoeg verloopt, leidt zelf ook tot meer vertraging omdat de benodigde vertrouwensbasis en werksfeer verstoord is.

Bijlage 2: Rollen binnen een project

In deze bijlage worden de verschillende rollen gedefinieerd van mensen die deel uitmaken van een project.

1. Projectleden/projectteam
De projectleden zijn de teamleden in het project. De mensen die het project daadwerkelijk uitvoeren en die taken hebben binnen het project. Vaak hebben teamleden verschillende expertisegebieden. Teamleden kunnen intern zijn (eigen personeel) en/of extern (van projectpartners, klanten, gebruikers of ingehuurd personeel).

2. Projectleider
De projectleider is diegene die het projectteam aanstuurt en de eindverantwoordelijkheid heeft over het projectresultaat. Afhankelijk van hoe het wordt afgesproken, kan het natuurlijk zo zijn dat de projectleider verantwoordelijkheden delegeert aan teamleden en/of dat externe managers de verantwoordelijkheid hebben over een onderdeel van het project.
Binnen cyclische projecten behartigt de projectleider zowel de belangen van de klant als de programmeurs. Hij zorgt ervoor dat de klant voldoende technische uitleg krijgt en helpt de klant met het kiezen en prioriteren van functionaliteiten.

3. Projectmanager
De termen projectleider en projectmanager worden vaak door elkaar gebruikt. Gebruikelijk is dat een projectmanager meerdere projecten leidt en een projectleider vaak maar één. De projectleider bevindt zich dan ook vaak 'dichter op de werkvloer' dan een projectmanager, die zich doorgaans meer met sturing en cijfers bezighoudt. Andere betekenissen en afbakeningen komen ook voor en de termen worden vaak door elkaar gebruikt.

4. Programmamanager
De programmamanager is diegene die een aantal projecten in een organisatie beoordeeld. Aan hem rapporteert de projectleider/projectmanager. Vaak is de programmamanager lid van het management team.

5. Klant
De klant is de partij die het projectresultaat besteld heeft. Hij of zij kan actief in het project deelnemen of meer afstand houden. De klant is soms ook de gebruiker van het projectresultaat maar dat is niet altijd het geval. Stel dat een universiteit een webapplicatie wenst voor zijn medewerkers en studenten, dan is de universiteit de klant en zijn de medewerkers en studenten de gebruikers.

6. Gebruikers
De gebruikers zijn diegenen die het projectresultaat gaan gebruiken.
Het is belangrijk om deze mensen te betrekken in de definitiefase, ontwerpfase en bij het testen van het project.

7. Projectpartner
De projectpartner is een derde partij (organisatie) waarmee het project wordt uitgevoerd. Wanneer er meerdere partijen deelnemen aan het project is het uiteraard belangrijk om de verantwoordelijkheden en taken goed af te bakenen.

8. Opdrachtgever/klant/sponsor
De partij(en) die het project financieel mogelijk maakt. Dit is vaak (maar niet altijd) ook de partij die het projectresultaat gaat gebruiken. De opdrachtgever zorgt voor de financiering van het project en is derhalve ook de partij aan wie verantwoording wordt afgelegd over het projectresultaat.

Bijlage 3: Nuttige hulpmiddelen voor projectmanagement

1. http://sourceforge.net/
Website waar Open Source Software te vinden is, waaronder software voor het managen van projecten. Onderstaande open source software kan hier gedownload worden.

2. Xplanner
Xplanner is een open source software tool voor het administreren en managen van de cycli door middel van storycards (volgens de eXtreme Programming werkwijze).

3. Open source CVS (Current Version System) beheersapplicaties die veel gebruikt worden zijn: CVS, Subversion en Gnu arch.

4. MS Project, Fasttrack en andere
MS project is het meest bekende programma voor het voeren van de administratie van een project en het maken van een Gant chart (balkenschema).
Fasttrack is een ander bekend pakket en daarnaast zijn er vele open source pakketten. Deze programma's zijn eigenlijk alleen nuttig bij projecten die worden uitgevoerd volgens de watervalmethode.

5. http://www.bugzilla.org/
Bugzilla is een open source programma voor het registreren, bewaken en archiveren van issues en bugs. Wordt met name binnen softwareontwikkeling gebruikt.

Bijlage 4: Licentie van dit handboek

De Creative Commons Attribution-NonCommercial-ShareAlike 2.5 Licentie is van toepassing op dit werk. Ga naar http://creativecommons.org/licenses/by-nc-sa/2.5/ of stuur een brief naar Creative Commons, 559 Nathan Abbott Way, Stanford, Californië 94305, VS om deze licentie te bekijken.

Kort samengevat komt het hier op neer:

De gebruiker mag:
- het werk kopiëren, verspreiden, tonen en op- en uitvoeren
- afgeleide werken maken

Onder de volgende voorwaarden:
- **Naamsvermelding**. De gebruiker dient bij het werk de naam Wouter Baars, http://www.wouterbaars.net, http://www.projectmanagement-training.net en Dans, http://www.dans.knaw.nl/ als oorspronkelijke auteurs aan te geven.
- **Niet-commercieel**. De gebruiker mag het werk niet voor commerciële doeleinden gebruiken
- **Gelijk delen**. Indien de gebruiker het werk bewerkt kan het daaruit ontstane werk uitsluitend krachtens dezelfde licentie als de onderhavige licentie worden verspreid.
- Bij hergebruik of verspreiding dient de gebruiker de licentievoorwaarden van dit werk kenbaar te maken aan derden.
- De gebruiker mag uitsluitend afstand doen van een of meerdere van deze voorwaarden met voorafgaande toestemming van de rechthebbende.

Bijlage 5: Over DANS en de makers van dit handboek

DANS is de nationale organisatie die zorgt voor de opslag en blijvende toegankelijkheid van onderzoeksgegevens in de alfa- en gammawetenschappen. DANS werkt daartoe samen met onderzoekers en bevordert samenwerking tussen hen onderling. DANS heeft de vorm van een netwerk, met een centrum dat verantwoordelijk is voor de organisatie van de data-infrastructuur. Dat centrum bestaat uit een team van ca. vijftien mensen die werken op het DANS-kantoor in Den Haag of bij één van de onderzoekcentra in het land, zie http://www.dans.knaw.nl

Auteur:
Ir. Wouter Baars is ontwikkelaar van software en educatie. Hij studeerde bedrijfskunde aan de Technische Universiteit van Eindhoven. Na zijn studie heeft hij in diverse projecten op het gebied van oude en nieuwe media gewerkt. Hij werkte als projectleider voor onder andere Waag Society, KPN, De Digitale Universiteit, de Hogeschool van Amsterdam, Noterik multimedia en de Europese Commissie. Naast zijn werk als ontwikkelaar geeft hij les in projectmanagement (http://www.projectmanagement-training.net) .Meer informatie over zijn werk is te vinden op: http://www.wouterbaars.net

Adviseurs:
Dr. Henk Harmsen is adjunct directeur van DANS (Data Archiving & Networked Services), een nieuw initiatief van KNAW en NWO op het gebied van het archiveren en beschikbaarstellen van onderzoeksdata in Nederland. Henk studeerde alfa-informatica aan de Universiteit van Amsterdam en promoveerde aan de Vrije universiteit van Amsterdam. Hij heeft als bibliothecaris, hoofd van automatisering en hoofd bedrijfsvoering op een breed vlak ervaring opgedaan. Meer informatie is te vinden op:
http://www.dans.knaw.nl/nl/over_dans/organisatie/henk_harmsen/

Ir. Rutger Kramer heeft Technische Informatica gestudeerd aan de Technische Universiteit Delft. Tijdens zijn afstudeerwerk was hij betrokken bij het ECPA Sepia project waarvoor hij bij het Nederlands Instituut voor Wetenschappelijke Informatiediensten (NIWI - KNAW) meewerkte aan een metadata invoer applicatie. Na zijn afstudeerstage bleef hij bij het NIWI als Technisch Wetenschappelijk Programmeur. In die hoedanigheid werkte hij aan verschillende projecten, zoals EVAMP en XPAST, waarbij ontsluiting van digitaal erfgoed materiaal centraal stond. Als Informatiekundige bij DANS houdt hij zich bezig als IT liaison en projectmanager voor interne en externe R&D projecten. Hij is betrokken bij het Easy Store DMS project voor DANS en het verzorgen van database ontsluiting voor de afdeling Kunstgeschiedenis van de Letterenfaculteit van de Universiteit Utrecht http://www.dans.knaw.nl/nl/over_dans/organisatie/rutger_kramer/

Drs. Laurents Sesink is informatiekundige bij de afdeling Acquisitie en Ontwikkeling van DANS (Data Archiving & Networked Services). Laurents studeerde geschiedenis aan de Universiteit Utrecht en historische informatiekunde aan de Universiteit Leiden.
Hij heeft als voormalig senior specialist digitaliseringsdiensten, technisch wetenschappelijk programmeur, coördinator ontwikkelgroep, senior consultant/projectleider en beleidsmedewerker een brede achtergrond op het terrein van wetenschappelijke en bestuurlijke informatievoorziening.

http://www.dans.knaw.nl/nl/over_dans/organisatie/laurents_sesink/
Drs. Joris van Zundert is onderzoeker en ontwikkelaar bij het Huygens Instituut, onderdeel van de Koninklijke Akademie van Wetenschappen (KNAW). Hij studeerde Nederlandse Taal en Cultuur aan de Universiteit Utrecht. Naast en na zijn studie ontwikkelde hij een professionele loopbaan als zelfstandig ontwerper en ontwikkelaar van internetapplicaties. Opleiding en praktijkervaring combineerde hij later in dienst van Stichting Wetenschap en Techniek Nederland, het Nederlands Instituut voor Wetenschappelijk Informatievoorziening en het Huygens Instituut. Joris van Zundert ontwikkelt in diverse projecten internettoepassingen en digitale gereedschappen speciaal gericht op (literair historisch) wetenschappelijk gebruik en onderzoek.
http://www.huygensinstituut.knaw.nl/index.php?option=com_content&task=view&id=131&Itemid=57&lang=du

Bijlage 6: Voorbeeld actie- en besluitenlijst

Projectnaam: <vul hier de projectnaam in>
Datum: <vul hier de datum in van bijwerken>
Eigenaar: <vul hier de naam in van de persoon die dit document beheert>
Fase: <vul hier in: initiatiefase, definitiefase, ontwerpfase, voorbereidingsfase, realisatiefase of nazorgfase>

Actielijst

Nr.	Onderwerp	eigenaar	Datum gepland	Datum gereed	status
1	Hier komen de taken die uitgevoerd moeten worden in een bepaalde periode te staan	Naam verant-woordelijke	5-1-2006	3-1-2006	☺
2					☹
3					😐
4					

Besluitenlijst

Nr.	Omschrijving	Datum
1.	Hier komen de besluiten te staan die genomen zijn tijdens het overleg	Datum van het besluit
2.		
3.		

Bijlage 7: Voorbeeld issuelog

Projectnaam: <vul hier de projectnaam in>
Datum: <vul hier de datum in van bijwerken>
Eigenaar: <vul hier de naam in van de persoon die dit document beheert>

Nr.	Type	Omschrijving Issue	naam	datum	prioriteit	Beslissing	status
1.	RFC	Hier een korte omschrijving van het issue dat is ontstaan			1= hoog 3= laag	Hier de beslissing beschrijven	ok
	AS					T = toegewezen	
	V					A = afgewezen	
	Z					U = Uitgesteld tot...	
	R						

Type	Prioriteit	Beslissing	Status
RFC= Request for change (algemeen)	1 = direct actie nemen	T = toegewezen	ok = issue is afgehandeld
AS = afwijking van specificatie (t.o.v. ontwerp)	2 = later actie nemen	A = afgewezen	open = openstaand
V = vraag	3 = geen actie	U = Uitgesteld tot...(datum/gebeurtenis)	
Z = Zorg			
R = Risico			

Bijlage 8: Voorbeeld risicolog

Projectnaam: <vul hier de projectnaam in>
Datum: <vul hier de datum in van bijwerken>
Eigenaar: <vul hier de naam in van de persoon die dit document beheert>

Nr.	Omschrijving risico	prioriteit	maatregel	status
1.	Hier komt een korte omschrijving van het risico dat men denkt of denkt te hebben	1= hoog 3= laag	Hier de maatregel beschrijven	ok

Prioriteit	Status
1 = direct actie nemen	ok = risico is afgehandeld
2 = later actie nemen	open = openstaand
3 = geen actie	

Bijlage 9: Voorbeeld gespreksverslag

Notulen van Projectnaam: <vul hier de projectnaam in>
Datum: <vul hier de datum in van het gesprek/ de vergadering>
Notulist: <vul hier de naam in van de persoon die dit document heeft opgesteld>
Aanwezig: <vul hier de namen in van de aanwezigen bij het overleg>
Afwezig met kennisgeving: <vul hier de namen in van diegenen die afwezig waren met kennisgeving
Afwezig zonder kennisgeving: <vul hier de namen in van diegenen die afwezig waren zonder kennisgeving>

Agenda

<Hier komt de lijst van de zaken zoals die op de agenda stonden tijdens het overleg, bijvoorbeeld:>

Goedkeuring gespreksverslag vorige vergadering
Bespreken actielijst
Bespreken besluitenlijst
Bespreken issuelog
Bespreken risicolog (nieuwe risico's en knelpunten)
Voortgang project
Aanpassing planning
Overleg met management
Rondvraag

1. Goedkeuring gespreksverslag vorige vergadering
Piet heeft aangegeven dat in het verslag zijn opmerkingen over de ontwikkelingen van nieuwe software bij de concurrent niet goed zijn opgenomen. Hij zal een korte email sturen naar de notulist met een correcte weergave van zijn ideeën. De overige aanwezigen geven aan dat ze akkoord gaan met het verslag.

2. Actielijst
Er zijn acties afgemeld en nieuwe acties aangemeld.
<<zie het aparte document ActieEnBesluiten.doc>>

3. Besluitenlijst
Zie de aparte actie- en besluitenlijst voor de genomen beslissingen.
<<zie het aparte document ActieEnBesluiten.doc>>

4. Issuelog
Zie de aparte issuelog voor de issues welke nog openstaan.
Er zijn deze week geen nieuwe issues aangemeld.

5. Risicolog
Henk heeft een nieuw risico gesignaleerd dat we over het hoofd hebben gezien. Wat moeten we doen als onze leverancier van software failliet gaat? Dit risico is opgenomen in de risicoloos. Klaas gaat er over nadenken en kijken wat de contracten met de leverancier daarover zeggen.
Zie verder de risicolog.

6. Voortgang project

Partner 1
In de Java straat wordt hard gewerkt aan de realisatie van de software. In totaal zijn er nu drie engineers toegewezen aan het project
De testers zijn nu bezig met de voorbereiding van de testscripts. Henk heeft gevraagd aan Marie of hij dit heeft opgestuurd aan Floor. Kees geeft aan dat de detailtest plannen geen deliverables zijn. De testscripts wel. Indien de partner inzicht in de voor genoemde plannen wil. Dan kan dat uiteraard altijd.

Partner 2
Floor heeft aangegeven dat ze nog druk bezig zijn met een nieuwe naam voor het product. Henk zal alvast een wijzigingsvoorstel op zetten waardoor de impact op het project inzichtelijk wordt.
De offerte van Leverancier voor de Licenties is nu binnen.
Betreffende de rapportage aan de financier heeft Floor aangegeven dat ze begin oktober weer een rapportage willen doen aan de financier. We hebben afgesproken dat we vijf werkdagen na de afsluiting van September de rapportage aanleveren aan de partner.

Deelnemers afgelopen periode

Naam	Functie
Piet Pieterse	Uitvoeringsmanager
Jan Jansen	Projectleider
Enz.	Lead Engineer CLient PRODUCT
	Technisch Architect
	Lead Engineer Server PRODUCT
	Java Engineer
	Java Engineer
	Kwaliteits/Testmanager
	Test Coördinator

Deelnemers komende periode

Naam	Functie
Klaas Klaaszoon	Uitvoeringsmanager
Marie de Boer	Projectleider
Enz.	Lead Engineer CLient PRODUCT
	Technisch Architect
	Lead Engineer ServerPRODUCT
	Java Engineer
	Java Engineer
	Java Engineer
	Java Engineer
	Java Engineer
	Java Engineer
	Kwaliteits/Testmanager
	Test Coördinator
	Tester

7. Aanpassen planning
Hieronder staat de nieuwe planning waarvan de partners nu in het project hebben aangegeven dat hij realistisch is op dit moment.

Fase / Mijlpaal	Startdatum / Mijlpaaldatum	Einddatum	Wie
Voorbereiding	7-4-03	6-6-2003	Partner1
Ontwerp	7-4-03	6-6-2003	Partner1 + partner2
Besluitvorming	10-6-2003	13-6-2003	Partner1 + partner2
Accorderen ontwerp	13-6-2003	13-6-2003	Partner2
Realisatie/Test uitvoering	30-6-2003	28-11-2003	Partner1
Oplevering 1e versie	28-11-2003	28-11-2003	Partner1
Acceptatietest	1-12-2003	2-01-2004	Partner2
Ondersteuning bij acceptatietest	1-12-2003	2-01-2004	Partner1
Acceptatie	2-01-2004	2-01-2004	Partner2
Substantiële gebruikerstest	2-01-2004	25-06-2003	Partner2
Ondersteuning bij substantiële gebruikerstest	2-01-2004	25-06-2003	Partner1
Optimalisering	28-06-2003	27-08-2003	Partner1
Oplevering 2e versie	27-08-2003	27-08-2003	Partner1
Garantie	27-08-2003	26-11-2004	Partner1

8. Overleg met het management
Klaas heeft in zijn rol als projectleider overleg gehad met het management. Hij management wil dat het project binnen 1, uiterlijk 2 maanden afgerond is. Klaas heeft aangegeven dat hij denkt dat het niet haalbaar is, maar dat het team toch zijn best zal doen om het (on)mogelijke te doen.

9. Rondvraag
Marie geeft aan dat de vergadering nu alweer 2 uur heeft geduurd, terwijl de doelstelling was om dat te beperken tot maximaal 3 kwartier. Graag een ieders inzet om kort te vergaderen.

Henk geeft aan dat hij niet bij de volgende bijeenkomst kan zijn.

Volgende bijeenkomst(en)

Soort:	Project voortgangsoverleg.
Frequentie:	Wekelijks.
Dag:	Dinsdag 19 augustus 2006
Tijd:	13:00 uur
Locatie:	Partner 1 kantoor Rotterdam
Aanwezigen:	Klaas, Henk, Floor, Marie
Afwezig:	Henk

Voorlopige agenda:
1. Gespreksverslag vorige vergadering
2. Actielijst
3. Besluitenlijst
4. Afwijkingen van Specificatie
5. Voortgang project
6. Planning (mijlpalen/wijzigingen)
7. Meer/Minderwerk
8. Issuelog
9. Risico's en knelpunten.
10. Overige/Rondvraag

Bijlage(n)

Actie-/Besluitenlijst Project
Issuelog
Risicolog

Bijlage 10: Voorbeeld projectplan

<naam project>

Projectnaam: <vul hier de projectnaam in>
Datum: <vul hier de datum in van bijwerken>
Projectleider: <vul hier de naam in van de projectleider(s)>
Fase: <vul hier in: initiatiefase, definitiefase, ontwerpfase, voorbereidingsfase, realisatiefase of nazorgfase>

Voor akkoord: <naam + handtekening projectleider>

Datum:

Voor akkoord: <naam + handtekening opdrachtgever>

Datum:

Inleiding:

Dit document is een korte handleiding voor het opstellen van een projectplan. Het eerste projectplan wordt gemaakt na de initiatiefase en dient als goedkeuringsdocument voor het hele project. Na elke fase moet dit document bijgesteld worden. De eerstvolgende fase moet een plan voor worden opgesteld in detail. Voor de verder liggende fases is het projectplan meer globaal. Na elke fase vindt ook evaluatie van dit document plaats.

Algemene informatie over het project

1. Situatieschets en probleemstelling project

- Geef een beschrijving van de organisatie waar dit project in plaatsvindt
- Geef een beschrijving van de afdeling(en) waarbinnen dit project plaatsvindt
- Geef eventueel een beschrijving van relaties van dit project met andere projecten
- Geef een beschrijving voor de geschiedenis van dit project
- Geef een beschrijving van de aanleiding van dit project
- Geef aan wie de opdrachtgever(s) is/zijn voor dit project
- Geef aan wie de opdrachtnemer is voor dit project

2. Projectopdracht

- Leg uit waarom dit project gedaan wordt, doe dit zo SMART mogelijk (SMART=specifiek, meetbaar, aanwijsbaar, realistisch, tijdgebonden)
- Geef aan wat er opgeleverd gaat worden
- Geef de belangrijkste grenzen aan van het project (wat gebeurt er niet)

3. Risicoanalyse

- Geef aan welke risico's bekend zijn bij dit project
- Geef aan hoe omgegaan wordt met deze risico's (vermijden, bestrijden, verzekeren, accepteren)

4. Inrichting van het project

In dit deel van het projectplan wordt een beschrijving gegeven van de fases, de activiteiten per fase en de bijbehorende GOKIT factoren van het project. Het plan wordt van globaal voor de verder liggende fases uitgewerkt en concreet en specifiek voor de eerstvolgende fase. In dit voorbeeld is uitgegaan van het opstellen van het eerste projectplan dus na de initiatiefase. Na *elke* fase moet dit deel van het projectplan worden bijgesteld.

4.1 Uitleg gebruikt projectmanagementmodel

Dit projectplan is gebaseerd op de watervalmethode/DANS-methode <kies hier voor een van de modellen> welke is beschreven in het handboek projectmanagement van DANS. Het handboek is bijgevoegd bij dit projectplan.

4.1.1 Initiatiefase

Het resultaat van de initiatiefase is dit projectplan. Deze fase hoeft niet verder uitvoerig beschreven te worden. Eventueel kan een opsomming gegeven worden van de activiteiten welke hebben plaatsgevonden in de voorbereiding.

4.1.2 Definitiefase

Geplande startdatum: <datum>
Geplande einddatum: <datum>

Beschrijving van het resultaat van de definitiefase:

In de definitiefase zal een eisen- en wensenlijst worden opgesteld over het te bereiken projectresultaat.

Belangrijkste mijlpalen: <(bijvoorbeeld)>

- Functionele eisen- en wensenlijst
- Onderzoek wettelijke eisen
- Eisen en wensen van eindgebruikers interviews
- Eisen en wensen van eindgebruikers testonderzoek
- Technische eisen en wensenrapport
- Goedkeuring door opdrachtgever van eisen- en wensenlijst

<Geef per mijlpaal aan wanneer die bereikt moet zijn en wie er verantwoordelijk voor is. Geef aan welke kwaliteit een bepaalde mijlpaal (= tussenproduct) moet hebben.>

Activiteiten in de definitiefase:
Geef een lijst van activiteiten die moeten plaatsvinden om de mijlpalen te bereiken. Geef aan wie ze uitvoeren, wanneer en door wie de resultaten van de activiteiten worden goedgekeurd (eindverantwoordelijke).

Tijdpad:
Zet hier de lijst van activiteiten uit in de tijd, bijvoorbeeld door gebruik te maken van een balkenschema. Markeer de mijlpalen duidelijk in de tijd. Vergeet niet om marges in te bouwen.

Begroting:
Geef hier een begroting van de kosten per activiteit in deze fase. Geef daarnaast de kosten voor materiaal en materieel en eventuele overige kosten aan. Verwijs eventueel naar een extern document met daarin de kosten (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

4.1.3 Ontwerpfase
Geplande startdatum: <datum>
Geplande einddatum: <datum>

Beschrijving van het resultaat van de ontwerpfase:
In de ontwerpfase zal een ontwerp / aantal ontwerpen gemaakt worden voor het te bereiken projectresultaat.

Belangrijkste mijlpalen: <(bijvoorbeeld)>
Een of meerdere dummy's
Schermontwerpen
Foto-impressies/schetsen

<Geef per mijlpaal aan wanneer die bereikt moet zijn en wie er verantwoordelijk voor is. Geef aan welke kwaliteit een bepaalde mijlpaal (= tussenproduct) moet hebben.>

Activiteiten in de ontwerpfase:
Geef een lijst van activiteiten die moeten plaatsvinden om de mijlpalen te bereiken. Geef aan wie ze uitvoeren, wanneer en door wie de resultaten van de activiteiten worden goedgekeurd (eindverantwoordelijke).

Tijdpad:
Zet hier de lijst van activiteiten uit in de tijd, bijvoorbeeld door gebruik te maken van een balkenschema. Markeer de mijlpalen duidelijk in het overzicht. Vergeet niet om marges in te bouwen.

Begroting kosten:
Geef hier een begroting van de kosten per activiteit in deze fase. Geef daarnaast de kosten voor materiaal en materieel en eventuele overige kosten aan. Vergeet niet een post 'onvoorzien' op te nemen en de kosten voor het projectmanagement zelf. Verwijs eventueel naar een extern document met daarin de kosten (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages tijdens en na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

Na de ontwerpfase gaat de watervalmethode verder met de voorbereidingsfase, de DANS-methode voor software ontwikkeling gaat verder met het cyclische deel. De verschillende mogelijkheden worden hieronder na elkaar beschreven. Voor het schrijven van een projectplan moet een van de twee mogelijkheden gekozen worden.

4.1.4 Voorbereidingsfase <(alleen waterval)>
Geplande startdatum: <datum>
Geplande einddatum: <datum>

Beschrijving van het resultaat van de voorbereidingsfase:
In de voorbereidingsfase zal een draaiboek opgesteld worden ter voorbereiding van de realisatiefase. <NB. Deze fase is niet altijd nodig bij projecten en kan dus bij met name wat kleinere projecten eventueel komen te vervallen>

Belangrijkste mijlpalen: <(bijvoorbeeld)>
Draaiboek deel 1
Draaiboek deel 2
enz.

<Geef per mijlpaal aan wanneer die bereikt moet zijn en wie er verantwoordelijk voor is. Geef aan welke kwaliteit een bepaalde mijlpaal (= tussenproduct) moet hebben.>

Activiteiten in de voorbereidingsfase:
Geef een lijst van activiteiten die moeten plaatsvinden om de mijlpalen te bereiken. Geef aan wie ze uitvoeren, wanneer en door wie de resultaten van de activiteiten worden goedgekeurd (eindverantwoordelijke).

Tijdpad:
Zet hier de lijst van activiteiten uit in de tijd, bijvoorbeeld door gebruik te maken van een balkenschema. Markeer de mijlpalen duidelijk in de tijd. Vergeet niet om marges in te bouwen.

Begroting kosten:
Geef hier een begroting van de kosten per activiteit in deze fase. Verwijs eventueel naar een extern document met daarin de kosten. Vergeet niet een post 'onvoorzien'

op te nemen en de kosten voor het projectmanagement zelf (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages tijdens en na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

4.1.5 Realisatiefase <(alleen waterval)>
Geplande startdatum: <datum>
Geplande einddatum: <datum>

Beschrijving van het resultaat van de realisatiefase:
In de realisatiefase zal het projectresultaat gebouwd worden.

Belangrijkste mijlpalen: <(bijvoorbeeld)>
Element 1 van de realisatie
Element 2 van de realisatie
enz.

<Geef per mijlpaal aan wanneer die bereikt moet zijn en wie er verantwoordelijk voor is. Geef aan welke kwaliteit een bepaalde mijlpaal (= tussenproduct) moet hebben.>

Activiteiten in de realisatiefase:
Geef een lijst van activiteiten die moeten plaatsvinden om de mijlpalen te bereiken. Geef aan wie ze uitvoeren, wanneer en door wie de resultaten van de activiteiten worden goedgekeurd (eindverantwoordelijke).

Tijdpad:
Zet hier de lijst van activiteiten uit in de tijd, bijvoorbeeld door gebruik te maken van een balkenschema. Markeer de mijlpalen duidelijk in de tijd. Vergeet niet om marges in te bouwen.

Begroting kosten:
Geef hier een begroting van de kosten per activiteit in deze fase. Geef daarnaast de kosten voor materiaal en materieel en eventuele overige kosten aan. Vergeet niet een post ‘onvoorzien’ op te nemen en de kosten voor het projectmanagement zelf. Verwijs eventueel naar een extern document met daarin de kosten (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages tijdens en na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

4.1.6 Cyclische fase <(alleen DANS-methode voor software ontwikkeling)>
Geplande startdatum: <datum>
Geplande einddatum: <datum>

Beschrijving van het resultaat van de realisatiefase:
In de cyclische fase zal het projectresultaat nader onderzocht, gespecificeerd en gebouwd worden.

Beschikbaar aantal uren voor de cyclische fase:
Geef hier aan hoeveel uur er beschikbaar is voor de cyclische fase, eventueel verdeeld over de verschillende teamleden, als die verdeling niet gelijkmatig is.

Eerste inschatting de het aantal cycli en hun producten <(bijvoorbeeld)>
cyclus 1: basisarchitectuur
cyclus 2: interactie tussen servers
cyclus 3: interactie met klant
enz.

Dit deel is voornamelijk van belang bij het opstellen van het eerste projectplan. Naarmate de cyclische fase dichterbij komt, wordt overgestapt op het planningssysteem met storycards (zie handboek DANS projectmanagement).

Deelnemers in de cyclische fase:
Geef een lijst van de deelnemers in de cyclische fase en hun verantwoordelijkheden:
<bijvoorbeeld>

Jan Jansen: programmeur
Piet Pietersen: programmeur
Marie Pedro Del Mar: programmeur
Kees Keeszoon: Ontwerper (interactie en grafisch)
Klant: procesinformatie en testen
enz.

Begroting kosten:
Geef hier een begroting van de kosten van de cyclische fase. Geef daarnaast de kosten voor materiaal en materieel en eventuele overige kosten aan. Vergeet niet een post ‘onvoorzien’ op te nemen en de kosten voor het projectmanagement zelf. Verwijs eventueel naar een extern document met daarin de kosten (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie. <Er zal met storycards gewerkt worden, evt. Aangevuld met een projectlog, CVS systeem, bugtracker en tools zoals Xplanner>.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages tijdens en na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

4.1.7 Nazorgfase <(waterval en DANS-methode voor softwareontwikkeling)>
Geplande startdatum: <datum>
Geplande einddatum: <datum>

NB: geef duidelijk aan wanneer de nazorg stopt vanuit het projectteam.

Beschrijving van het resultaat van de nazorgfase:
In de nazorgfase wordt het project afgerond. Geef hier aan wat er wel en niet onder de afronding valt.

Belangrijkste mijlpalen: <(bijvoorbeeld)>
Projectverslag
Overdrachtsdocument naar beheersorganisatie
Projectwebsite met gebruikersinformatie
enz.

<Geef per mijlpaal aan wanneer die bereikt moet zijn en wie er verantwoordelijk voor is. Geef aan welke kwaliteit een bepaalde mijlpaal (= tussenproduct) moet hebben.>

Activiteiten in de nazorgfase:
Geef een lijst van activiteiten die moeten plaatsvinden om de mijlpalen te bereiken. Geef aan wie ze uitvoeren, wanneer en door wie de resultaten van de activiteiten worden goedgekeurd (eindverantwoordelijke).

Tijdpad:
Zet hier de lijst van activiteiten uit in de tijd, bijvoorbeeld door gebruik te maken van een balkenschema. Markeer de mijlpalen duidelijk in de tijd. Vergeet niet om marges in te bouwen.

Begroting kosten:
Geef hier een begroting van de kosten per activiteit in deze fase. Geef daarnaast de kosten voor materiaal en materieel en eventuele overige kosten aan. Vergeet niet een post ‘onvoorzien’ op te nemen en de kosten voor het projectmanagement zelf. Verwijs eventueel naar een extern document met daarin de kosten (zie ook het aparte model voor begroten).

Informatie Intern:
Geef aan hoe de informatie uit deze fase geregistreerd en gearchiveerd wordt. Geef aan welke hulpmiddelen eventueel gebruikt worden en indien nodig wie er wel of niet de beschikking heeft over de informatie.

Informatie Extern:
Een belangrijk informatiemoment is de goedkeuring van deze fase door de opdrachtgever/klant. Geef aan welke rapportages tijdens en na deze fase opgeleverd worden aan de opdrachtgever, klant en of extern management.

Overzicht

In het laatste deel van het project plan staat een overzicht van de kosten en tijdpad van het hele project.

Fase	Verwachte startdatum	Verwachte einddatum	Totale kostenschatting
Definitie			
Ontwerp			
Voorbereiding			
Realisatie			
Nazorg			
		Totaal:	

Bijlage 11: Voorbeeld begroting

Begroting
<Projectnaam>
<fase van het project: initiatiefase - nazorg>
<Naam opsteller dit document>
<datum>

bedragen in Euro

Initiatiefase	*wie?*	*uren*	*tarief*	*totaal*
Onderzoek concept	projectleider	20	75	1500
Overleg partners	projectleider	10	75	750
Projectaanvraag schrijven	hoofd afdeling	40	80	3200
Vormgeving aanvraag	grafisch design	10	50	500
Inkoop kennis			1000	1000
...				
		subtotaal initiatiefase:		6950
Definitiefase				
Overleg eindgebruikers	projectleider	10	75	750
	functioneel ontwerper	10	65	650
Overleg experts	projectleider	20	75	1500
	functioneel ontwerper	30	65	1950
	interne experts	30	100	3000
Inkoop kennis				2500
Catering bijeenkomsten				250
Projectmanagement		25	75	250
		subtotaal definitiefase:		10850
Ontwerpfase				
Reiskosten	teamleden			1000
Ontwerpschetsen	ontwerper	60	60	3600
Presentatie ontwerpen	ontwerper	20	60	1200
Materialen				2000
Webpresentatie				
content	tekstschrijver	20	60	1200
html	programmeur	20	65	1300
vormgeving	ontwerper	30	60	1800
Projectmanagement		15	75	1125
		subtotaal ontwerpfase:		13225

Voorbereidingsfase

Reiskosten				2000
Materiaalkosten				15000
Uitwerken draaiboeken	projectleider	40	75	3000
Projectmanagement		15	75	1125
			subtotaal voorbereidingfase:	22125

Realisatiefase

Reiskosten personeel				2000
Kosten vervoer materiaal				3000
Uitvoering eigen personeel	timmermannen	160	45	7200
	loodgieters	160	50	8000
	vormgevers	80	65	5200
Uitzendkrachten				25000
Gas, water, licht				3000
Projectmanagement		80	75	6000
			subtotaal realisatiefase:	59400

Nazorgfase

Reiskosten personeel				2000
Afvoer materialen				3000
Opruimen materialen	projectteam	40	60	2400
Eindrapportage	projectleider	40	75	3000
Accountantsverklaring				1000
Projectmanagement		40	75	3000
			subtotaal nazorg:	15900

Resumé

subtotaal initiatiefase:		6950	
subtotaal definitiefase:		10850	
subtotaal ontwerpfase:		13225	
subtotaal voorbereidingfase:		22125	
subtotaal realisatiefase:		59400	
subtotaal nazorg:		15900	
	subtotaal:		128450
onvoorzien 10%			13495
	totaal:		141945

Deze begroting is de globale begroting van het gehele project <projectnaam>. Aan het begin

van elke nieuwe fase zal een gedetailleerde begroting aangeleverd worden van die fase.

Bijlage 12: Voorbeeld financiële verantwoording

Financiële rapportage
<Projectnaam>
<Naam opsteller dit document>
<datum>

bedragen in Euro

Initiatiefase	*wie?*	*uren begroot*	*uren gerealiseerd*	*tarief*	*totaal begroot*	*totaal gerealiseerd*
Onderzoek concept	projectleider	20	23	75	1500	1725
Overleg partners	projectleider	10	12	75	750	900
Projectaanvraag schrijven	hoofd afdeling	40	35	80	3200	2800
Vormgeving aanvraag	grafisch design	10	20	50	500	1000
Inkoop kennis				1000	1000	980
...						
Onvoorzien						861
		subtotaal initiatiefase:			6950	7405
Definitiefase						
Overleg eindgebruikers	projectleider	10	10	75	750	750
	functioneel ontwerper	10	10	65	650	650
Overleg experts	projectleider	20	18	75	1500	1350
	functioneel ontwerper	30	35	65	1950	2275
	interne experts	30	40	100	3000	4000
Inkoop kennis					2500	2700
Catering bijeenkomsten					250	247
Projectmanagement		25	41	75	250	3075
Onvoorzien						120
		subtotaal definitiefase:			10850	15167
Ontwerpfase						
Reiskosten	teamleden				1000	1221
Ontwerpschetsen	ontwerper	60	63	60	3600	3780
Presentatie ontwerpen	ontwerper	20	19	60	1200	1140
Materialen					2000	1873
Webpresentatie						
content	tekstschrijver	20	18	60	1200	1080
html	programmeur	20	32	65	1300	2080
vormgeving	ontwerper	30	18	60	1800	1080
Projectmanagement		15	12	75	1125	900
Onvoorzien						1245
		subtotaal ontwerpfase:			13225	14399

Voorbereidingsfase

Reiskosten					2000	1200
Materiaalkosten					15000	13981
Uitwerken draaiboeken	projectleider	40	45	75	3000	3375
Projectmanagement		15	22	75	1125	1650
Onvoorzien						11003
		subtotaal voorbereidingfase:			22125	31209

Realisatiefase

Reiskosten personeel					2000	2100
Kosten vervoer materiaal					3000	2850
Uitvoering eigen personeel	timmermannen	160	200	45	7200	9000
	loodgieters	160	165	50	8000	8250
	vormgevers	80	75	65	5200	4875
Uitzendkrachten					25000	24560
Gas, water, licht					3000	3800
Projectmanagement		80	81	75	6000	1650
Onvoorzien						231
		subtotaal realisatiefase:			59400	57316

Nazorgfase

Reiskosten personeel					2000	1642
Afvoer materialen					3000	3500
Opruimen materialen	projectteam	40	45	60	2400	2700
Eindrapportage	projectleider	40	40	75	3000	3000
Accountantsverklaring					1000	981
Projectmanagement		40	62	75	3000	4650
Onvoorzien						861
		subtotaal nazorg:			15900	17334

Resumé

	begroot	
subtotaal initiatiefase:	6950	7405
subtotaal definitiefase:	10850	15167
subtotaal ontwerpfase:	13225	14399
subtotaal voorbereidingfase:	22125	31209
subtotaal realisatiefase:	59400	57316
subtotaal nazorg:	15900	17334
onvoorzien 10%	13495	14321
totaal:	141945	157151

Bij deze financiële rapportage hoort een korte analyse van de grootste verschillen tussen begroting en realisatie.

Literatuur

Chromatic and Diaz, T. Apandi (Ed.) (2003). *Extreme Programming Pocket Guide.* Sebastopol, CA: O'Reilly & Associates.

Goldratt, E.M. (2002). *De zwakste schakel.* Utrecht: Het Spectrum.

Kor, R. and Wijnen, G. (2002). 50 checklisten voor project- en programmamanagement. Deventer: Kluwer.

Kroll, P. and Kruchten, Ph. (2004). The Rational Unified Process Made Easy: A practitioner's Guide to the RUP. Boston: Addison Wesley.

McConnell, S. (1996). *Rapid Development: Taming Wild Software Schedules.* Redmond, Washington: Microsoft Press.

Stapleton, J. (2002). DSDM Dynamic Systems Development Method: De methode in de praktijk. Schoonhoven: Academic Service.

Wijnen, G., Renes, W. and Storm, P. (2004). *Projectmatig werken.* Utrecht: Het Spectrum.

Internet bronnen

[i] Spolsky, J. (2000). Painless Functional Specifications - Part 1: Why Bother? Geraadpleegd 31 mei 2006 via http://www.joelonsoftware.com/articles/fog0000000036.html

[ii] Why power your site with XP? (z.j.). Geraadpleegd 31 mei 2006 via http://www.XP.com/

[iii] Extreme Programming: A gentle introduction (Last modified February 17, 2006). Geraadpleegd 31 mei 2006 via http://www.extremeprogramming.org/

[iv] Spolsky, J. (2000). WhatTimeIsIt.com. Functional Specification. Geraadpleegd 31 mei 2006 via http://www.joelonsoftware.com/articles/WhatTimeIsIt.html

[v] Spolsky, J. (2002). Fire And Motion. Geraadpleegd 31 mei 2006 via http://www.joelonsoftware.com/articles/fog0000000339.html

[vi] Pair Programming successes and failures (2001-2003). Geraadpleegd 31 mei 2006 via http://discuss.fogcreek.com/joelonsoftware/default.asp?cmd=show&ixPost=2058

Voorwoord bij versie 1.1

Wie ooit heeft meegewerkt aan een project zal beamen dat het niet eenvoudig is om een project tot een succes te maken. De knelpunten vertalen zich onder andere in: het (sterk) uitlopen van de planning, het (sterk) overschrijden van het budget, onvoldoende resultaat van het project, ontevreden klanten en/of veel stress in het projectteam. Hoe ontstaan al die problemen?

Projecten kenmerken zich door vier aspecten: er is een groep mensen, er is een doel, er is altijd een beperkte hoeveelheid tijd en geld en tenslotte is er meer of minder onzekerheid of het allemaal wel gaat lukken. Een projectmanager heeft met al deze vier aspecten te maken. Het leiden en sturen van projecten is daarom niet eenvoudig.

Projecten komen steeds vaker voor. Projectmatig werken heeft ook zijn weg gevonden binnen non-profit organisaties als DANS[1]. Bij non-profit organisaties zijn de spelregels voor een project anders dan bij commerciële organisaties. Bij non-profit organisaties spelen politieke factoren veelal de belangrijkste rol spelen. Hierdoor zijn projecten nog moeilijker succesvol te laten worden, dan projecten waar commerciële factoren een rol spelen. Het is goed als de projectleider zich dat realiseert en ook het politieke spel weet te spelen.

Na een aantal jaar ervaring met ICT-projecten is het de makers van dit handboek steeds meer opgevallen hoe anders ICT-projecten zijn dan 'gewone' projecten. ICT-projecten blijken vooral dynamischer en dynamiek kent zo zijn voor- en nadelen. Wij hebben vastgesteld dat ICT-projecten in ieder geval gedeeltelijk een andere aanpak vereisen dan bijvoorbeeld bouw- of reorganisatieprojecten.

Dit handboek is geschreven voor projecten die uitgevoerd worden bij DANS. Het eerste deel beschrijft een werkwijze die gevolgd kan worden bij 'traditionele' projecten. In het tweede deel wordt de werkwijze voor ICT-projecten beschreven, met name softwareontwikkeling.

In dit handboek wordt een praktisch model gegeven waarmee projectleden, projectleiders, projectmanagers, algemene managers, programmamanagers, klanten en projectpartners hun rol in de projecten van DANS beter kunnen spelen.

Als het over het vak projectmanagement gaat, raakt een mens nooit uitgeleerd. De ontwikkelingen van de theorie en de ervaringen in de praktijk leiden steeds weer tot nieuwe inzichten. Dit handboek is dus niet af en zal daarom meegroeien met de vernieuwde ontwikkelingen op het gebied van projectmanagement. Om dit mogelijk te maken, is er voor gekozen de tekst van dit handboek onder een creative commons licentie uit te geven. Dat betekent dat iedereen deze tekst vrij mag gebruiken, delen mag kopiëren of veranderen[2]. Het betekent vooral ook dat een ieder die denkt iets te moeten aanvullen of verbeteren aan deze tekst dat vooral niet moet laten!

Henk Harmsen
Adjunct directeur
DANS
Den Haag, mei 2006

[1] DANS (Data Archiving and Network Services) is een gezamenlijk initiatief van KNAW (Koninklijke Nederlandse Akademie van Wetenschappen) en NWO (Nederlandse Organisatie voor Wetenschappelijk Onderzoek) ter verbetering van de wetenschappelijke data-infrastructuur in Nederland

[2] Zie Bijlage 4 voor de exacte bepalingen van de licentie.