cegeka

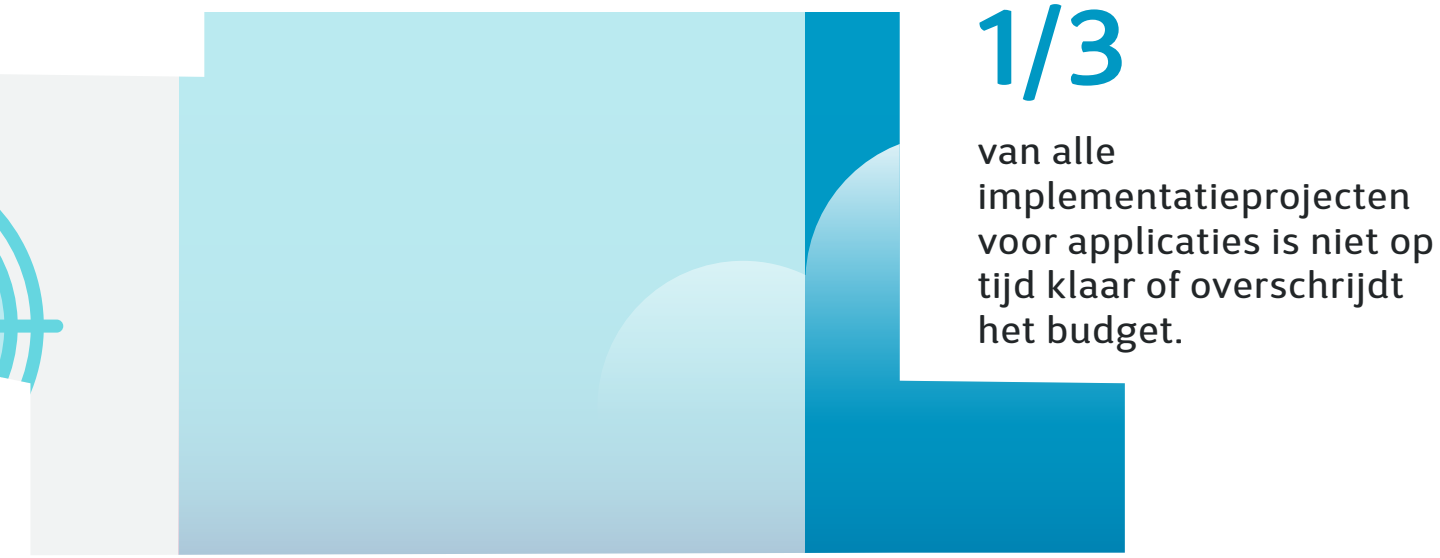
Zes tips om complexe applicatieontwikkelingsprojecten te laten slagen

Het aantal mislukte IT-projecten is verbazingwekkend. 1/3 van alle applicatie-implementatieprojecten wordt niet op tijd opgeleverd of overschrijdt het budget.*

Toch is IT nog nooit zo belangrijk geweest als in het dynamische digitale tijdperk waarin we nu leven.

Hoe komt het dat zoveel projecten worden gestart, maar later on hold gezet worden, of teleurstellende resultaten opleveren?

En is er een manier om deze projecten wél te laten slagen? Jazeker. Dit e-book geeft 6 tips om uw complexe applicatieprojecten op tijd en binnen budget te houden en ervoor te zorgen dat ze voldoen aan de behoeften van alle betrokkenen.



* Bron: Use These 10 Contracting Steps to Dramatically Reduce Your Application Implementation Project Overruns (Gartner, 2017)

INHOUDSOPGAVE

1

Waarom mislukken complexe IT-projecten?

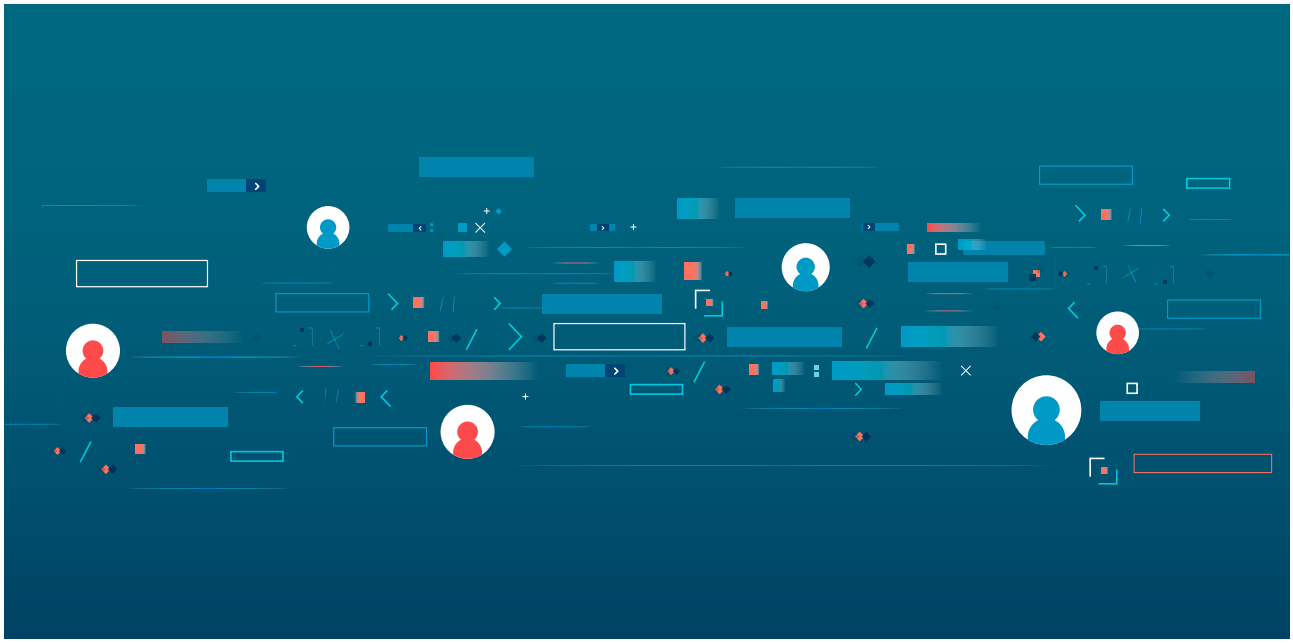
Waarom mislukken complexe IT-projecten?

Grootschalige en complexe ontwikkelingsprojecten voor applicaties lijken bijna per definitie vaker fout te gaan. Ze duren immers vaak erg lang, waardoor de kans stijgt dat de vereisten en scope in de loop van het project zullen veranderen. Bovendien zijn er veel mensen betrokken bij deze projecten, en blijft de samenstelling van de teams vaak niet gelijk. Dat maakt het moeilijk om dezelfde focus te behouden en een consistente aanpak te hanteren.

Er zijn veel factoren die IT-projecten kunnen laten mislukken, zoals deadlines die niet gehaald worden, een budget dat opraakt en verwachtingen van gebruikers die niet worden ingelost. **Mogelijke factoren die een IT-project kunnen laten mislukken**:

- slechte initiële definitie van de scope;
- veranderende prioriteiten en scope;
- gebrek aan focus op het budget;
- gebrekkige technische expertise of technieken op het gebied van softwareontwikkeling.

Lees verder en ontdek **6 tips om van uw nieuwe applicatie een succes te maken.**



2

6 tips om van uw applicatie een succes te maken

6 tips om van uw applicatie een succes te maken

Tip 1: ken uw gebruikers en de business drivers

Een van de belangrijkste redenen voor mislukte softwareprojecten heeft niets met technologie te maken, maar met een gebrek aan inzicht in de behoeften van de gebruiker en het bedrijf. Een diepgaande analyse kan u helpen om inzicht te krijgen in de business drivers en doelstellingen van het project, zodat u de scope correct kunt bepalen.

Gebruikers willen software die is afgestemd op hun specifieke behoeften en doelstellingen. Hoe goed uw applicatie er ook uitziet of hoe vlot elke functie ook werkt, als ze niet aan de verwachtingen van de gebruiker voldoet, zal ze geen succes worden.

Daarom beginnen succesvolle IT-projecten met een grondige analyse. Hierbij moeten veel vragen worden gesteld: niet alleen over de gewenste applicatie, maar ook over de visie, strategie, manier van werken en toekomstplannen van het bedrijf en/of het team. Door **uw gebruikers en hun bedrijfscontext goed te leren kennen**, kunt u de business drivers en dus de beste oplossing voor de gebruikers bepalen en hiermee rekening houden bij elke stap van het ontwikkelingsproces.

> *Hoe goed uw applicatie er ook uitziet of hoe vlot elke functie ook werkt, als ze niet aan de verwachtingen van de gebruiker voldoet, zal ze geen succes worden.*

WORKSHOPS ORGANISEREN OM DE BUSINESS DRIVERS VAN UW KLANT TE BEPALEN

Wat is nu precies de scope van een nieuw IT-project? Zijn alle stakeholders het eens over de doelstellingen en de vereisten? Om er zeker van te zijn dat iedereen op dezelfde golflengte zit, is een **uitgebreid vooronderzoek** cruciaal.

Organiseer **workshops met de klant** om hun organisatie en de uitdagingen die ze willen oplossen in kaart te brengen en betrek hierbij zoveel mogelijk stakeholders. Op basis van die input kunt u vervolgens de **business drivers** (doel, actoren, impact en deliverables) en de **high-level projectscope** (applicatiecontext, bedrijfsprocessen, conceptueel model en story mapping) bepalen en visualiseren. Tijdens een tweede workshop kunt u feedback verzamelen over uw voorstel. Ten slotte kunt u in nauwe samenwerking met alle stakeholders de **technologische behoeften in kaart brengen**, inclusief de functionele en niet-functionele vereisten, **de risico's en het budget**.

TIP 1

Neem de tijd om **de behoeften van de organisatie en de gebruikers** te analyseren en wees geduldig als uw development partner veel vragen stelt. Een grondige bedrijfsanalyse vooraf is de beste manier om van uw project een succes te maken!

Tip 2: ga stap voor stap te werk

Hoe precies u de initiële scope ook heeft bepaald, de vereisten en prioriteiten van een ontwikkelingsproject zullen gaandeweg toch veranderen. Door in iteraties te werken en regelmatig feedbacksessies te organiseren, blijft u op de hoogte van de behoeften van de gebruikers en kunt u snel en efficiënt reageren op wijzigingen.

Bij softwareontwikkeling is verandering de enige constante. Nieuwe wetgeving, een nieuwe CEO, nieuwe vereisten van de klant... Er zijn veel factoren die de scope en prioriteiten van uw project kunnen beïnvloeden. De mogelijkheid om u aan te passen aan veranderingen, en daarover te communiceren, is cruciaal om uw project de hoogste slaagkansen te geven.

Bepaalde benaderingen voor softwareontwikkeling die vandaag de dag worden toegepast, zoals de agile- en DevSecOps-methodologie, helpen teams om in te spelen op veranderende omstandigheden en zorgen voor een constante stroom van feedback. Door de applicatie in **kleine, incrementele stappen** te verdelen met **korte feedbacklussen** tussen het development-team en de **business** (agile) of **operations** (DevSecOps), ontdekt u snel inconsistenties of problemen en kunt u ze oplossen voordat u aan het volgende stukje begint te werken.

Bovendien is het zinvol om **risicovolle ontwikkelingstaken al vroeg in de levenscyclus van het project in te plannen.** Een alternatieve oplossing vinden is immers gemakkelijker in de beginfase dan in een later stadium.

AGILE EN DEVSECOPS: WAT IS HET?

Zowel Agile als DevSecOps zijn bedoeld om de samenwerking en het teamwerk te bevorderen en de oplevering te versnellen. Beide benaderingen gaan echter over verschillende aspecten van het deliveryproces:

- **Agile** creëert korte feedbacklussen tussen de **development-teams en de business**. Door agile technieken te gebruiken, kan de applicatie snel worden geëvalueerd en aangepast aan de vereisten van de business en de eindgebruikers.

- **DevSecOps** verbetert de samenwerking tussen de **development-, operations- en securityteams.** Het richt zich vooral op de frequentie van de deliveries, waarbij silo's zo veel mogelijk worden vermeden om snelle integratie, tests en implementatie van softwarewijzigingen in de productieomgeving mogelijk te maken. Dit verbetert de planning, het ontwerp en de releaseprocessen.

TIP 2

Ontwikkel de applicatie stap voor stap (incrementeel) en krijg snel feedback, zodat u indien nodig van koers kunt veranderen zonder grote nadelige gevolgen.

Tip 3: houd het budget goed in de gaten

35%* van de complexe softwareprojecten krijgt te maken met budgetoverschrijdingen. Dat is eigenlijk niet zo verwonderlijk, aangezien de budgetten al vóór de start van het project worden vastgelegd, terwijl er dan nog veel onzekerheden zijn. Bovendien is het zeer waarschijnlijk dat de scope nog zal veranderen in de loop van het project. Vanaf de afbakening van de scope tot de oplevering van het project moet u de kosten goed in de gaten houden om budgetoverschrijdingen te vermijden.

Een **duidelijke scope** die voldoende rekening houdt met de vereisten van het project is natuurlijk een must om het budget correct te kunnen inschatten en te vermijden dat het te snel is uitgeput. **Bereid u voor op verrassingen** wanneer u de kosten inschat, want bij grote projecten wordt u vaak geconfronteerd met onvoorziene omstandigheden die een impact hebben op het budget.

Denk groot, maar begin klein bij de start van uw project. Houd rekening met de projectvereisten en bouw eerst een **minimum viable product** dat zich richt op de **'must-haves'**. Dit zijn de features die de organisatie het meest nodig heeft en die echte business value leveren. 'Nice-to-haves' kunnen in een later stadium worden toegevoegd als er nog budget over is.

Om het budget onder controle te houden, moet u **het goed opvolgen en er te allen tijde over communiceren** met alle betrokkenen. Een iteratief, stapsgewijs ontwikkelingsproces met regelmatige feedbacksessies is hierbij een belangrijk hulpmiddel. Tijdens tweewekelijkse vergaderingen kunt u de applicatie en de voortgang van het project opvolgen, maar ook de daadwerkelijke uitgaven afzetten tegenover het budget. Indien nodig kunt u dan de prioriteiten bijsturen om te voorkomen dat het budget wordt overschreden.

Bereid u voor op verrassingen wanneer u de kosten inschat, want bij grote projecten wordt u vaak geconfronteerd met onvoorziene omstandigheden die een impact hebben op het budget.

TIP 3

Denk groot, maar begin klein als u een applicatie ontwikkelt. Focus op de 'must-haves' en voeg alleen 'nice-to-haves' toe als u zeker weet dat er voldoende budget is.

* Bron: Use These 10 Contracting Steps to Dramatically Reduce Your Application Implementation Project Overruns (Gartner, 2017)

Tip 4: ontwikkel software als een professional

IT-projecten waarbij grote applicaties worden ontwikkeld zijn complexe ondernemingen. Dit verhoogt het risico op mislukkingen door technische problemen. Zorg voor een duidelijke softwarearchitectuur, gebruik hands-on ontwikkelmethoden en zorg ervoor dat elke developer de best practices voor software engineering strikt naleeft.

Slimme softwareontwikkeling begint met het juiste team: een groep **toegewijde experts** die de skills hebben om de doelstellingen van het project te realiseren en die een goed inzicht hebben in de visie, doelstellingen en timeline van het project.

Succesvolle softwaredevelopers schrijven **cleane, herbruikbare code** die gemakkelijk te testen en te lezen is. Er zijn verschillende **best practices** die helpen bij het bouwen van kwaliteitsvolle code die niet te complex is. Als het hele development-team een **uniforme aanpak** volgt qua design- en coderingsstandaarden, kan de code eenvoudig met collega's en toekomstige developers worden gedeeld. Hierdoor blijft de continuïteit van uw project gewaarborgd. Bovendien leidt deze aanpak tot een ontwerp van hoge kwaliteit en applicaties en software die 'first-time-right' zijn.

Tot slot moet u zorgen dat uw software engineers **steeds op de hoogte zijn van de nieuwste trends en technologieën**, zoals nieuwe programmeertalen, frameworks, methoden, enzovoort. Bovendien moeten ze ook de kans krijgen om deze meteen in de praktijk te brengen.

EEN SELECTIE VAN BEST PRACTICES VOOR SOFTWAREONTWIKKELING

Elke best practice is belangrijk om futureproof software van hoge kwaliteit te bouwen. In de praktijk kiezen softwareteams echter vaak welke best practices ze zullen toepassen in het licht van de context en behoeften van een project. Dit zijn enkele voorbeelden:

Leesbare code
Leesbare code is essentieel voor software van hoge kwaliteit. Alle code moet door alle developers in het team kunnen worden begrepen en 'voor zich spreken', zodat ook mensen die er later bijkomen ze kunnen lezen.

Pair programming
Om kwaliteitsproblemen te vermijden, kunt u twee developers laten samenwerken aan een taak, waarbij de ene de 'driver' is en de andere de 'navigator'. De 'driver' schrijft de code en legt tegelijkertijd uit wat hij aan het doen is en waarom. De navigator denkt na over de volgende stappen en mogelijke valkuilen en anticipeert op problemen voordat deze zich voordoen. Door hun ervaring te combineren bedenken developers vaak oplossingen die ze nooit alleen hadden gevonden.

Collective code ownership
Collective code ownership betekent dat de code de verantwoordelijkheid is van het hele team. Dit houdt in dat elke developer stukjes code kan bewerken en met de volgende iteratie kan starten. Developers worden aangemoedigd, en zelfs verondersteld, om de code waar nodig te wijzigen.

Domain-driven design
Als de woorden die in de software worden gebruikt niet precies overeenkomen met de woorden die door de business worden gebruikt, kan dit tot allerlei problemen leiden. Een domain-driven design verbindt de implementatie met een model van het domein waarin de software zal worden gebruikt. Hierbij wordt een taal gehanteerd die wordt gedeeld door het team en de klant/ gebruikers.

Re-factoring
Bij re-factoring zorgt u ervoor dat de interne codestructuur zo 'clean' mogelijk wordt gehouden en duidelijk wordt afgestemd op de architectuur en frameworks. Hierdoor vermijdt u dat er later in het project dure rework-werkzaamheden moeten worden uitgevoerd.

DRIE MANIEREN OM UW DEVELOPERS GEMOTIVEERD TE HOUDEN

U zorgt er al voor dat uw developers de juiste skills bezitten. Maar hoe zorgt u ervoor dat ze betrokken, tevreden en productief blijven? Hieronder vindt u drie manieren om uw developers tevreden te houden (spoiler alert: een schandalig hoog salaris hoort er niet bij!):

1. Geef hen een doel
Uw medewerkers, en zeker de millennials onder hen, zijn meer dan ooit op zoek naar een baan die bijdraagt aan de maatschappij of waarin ze de kans krijgen om andere mensen te helpen. Als u dus het beste in uw mensen naar boven wilt halen, geef ze dan een duidelijk doel.

2. Zorg voor toegang tot de nieuwste technologieën
Software engineers vinden het leuk om de nieuwste innovaties uit te proberen. Zorg ervoor dat ze de nieuwste en beste tools kunnen gebruiken en dat ze nieuwe technologie kunnen uittesten. Het is ook belangrijk om developers die bestaande applicaties updaten of onderhouden de kans te geven om dit werk te combineren met boeiende nieuwe projecten.

3. Geef hen de kans om bij te leren en te groeien

Technische medewerkers willen de kans krijgen om nieuwe dingen te leren en te groeien. Geef hen wat ze willen en daag ze uit om zich op steeds grotere schaal te bewijzen.

TIP 4

Zorg ervoor dat uw development-team zich houdt aan de best practices en een uniforme aanpak volgt qua design- en coderingsstandaarden. Dit is cruciaal om de kwaliteit en de continuïteit van complexe applicatieprojecten te kunnen waarborgen.

Tip 5: testen, testen en nog eens testen

Hoewel best practices voor software engineering een goed begin vormen voor de ontwikkeling van hoogwaardige software, is een goed beheerd testproces net zo belangrijk. Door op regelmatige tijdstippen geautomatiseerde tests uit te voeren kunt u vanaf het begin van de developmentfase bugs opsporen. Zo voorkomt u dat uw applicatie het laat afweten wanneer ze uiteindelijk in gebruik wordt genomen.

De afgelopen jaren is er veel te doen geweest over het gebruik van automatische tests om het softwareontwikkelingsproces te versnellen. De beste manier om complexe applicaties van de hoogste kwaliteit te bouwen is echter een c**ombinatie van handmatige en geautomatiseerde tests**. Beide types hebben immers hun voor- en nadelen.

Bij het bepalen van het ontwerp of de bruikbaarheid van uw applicatie is menselijk inzicht bijvoorbeeld cruciaal. Daarom zijn **handmatig tests**, zowel door developers als door gebruikers, in dit geval de beste optie. Bij PEN testing is bijvoorbeeld menselijke creativiteit vereist om te beoordelen of uw applicatie en infrastructuur voldoen aan de veiligheidsnormen en ze te beschermen tegen bedreigingen van buitenaf. **Geautomatiseerde tests** zijn dan weer ideaal om grote applicaties met veel variabelen en platformen te testen. Ze zijn bijvoorbeeld erg nuttig wanneer de tests snel en nauwkeurig moeten worden uitgevoerd. Ze worden gebruikt om te controleren of een applicatie nog correct werkt en of ze aan de niet-functionele vereisten, zoals de prestatiebehoeften, voldoet.

> *Automatisering is ideaal als er snel en accuraat getest moet worden en bij grote applicaties met veel variabelen en platformen.*

TIP 5

Geautomatiseerd testen is zeker de investering waard, vooral bij grote projecten voor applicatieontwikkeling. Hoewel de bouw en het onderhoud van een geautomatiseerde testsuite veel tijd in beslag nemen, zal u dit op lange termijn veel tijd besparen.

Tip 6: non-stop samenwerken en communiceren

Als er één tip is die u zeker niet mag vergeten, dan is het deze wel: vergeet nooit de menselijke factor!

We hebben het in dit e-book al verschillende keren gezegd: alles staat en valt met communicatie. Praat met de klant en de gebruikers en stel hen alle mogelijke vragen om **de juiste scope te bepalen**. **Vraag regelmatig feedback** om op koers te blijven, misverstanden te vermijden en uw applicatie snel aan te passen waar nodig, zonder veel tijd te verliezen. **Praat openlijk over het budget.** Werk nauw samen met uw collega-developers en de klant om software van hoge kwaliteit te ontwikkelen en te testen.

> *Een IT-project is gedoemd te mislukken als de samenwerking en communicatie tekortschieten.*

TIP 6

Softwareprojecten gaan over mensen. Een goede samenwerking en communicatie kunnen het verschil maken tussen slagen en falen.

Conclusie

Grote, complexe projecten voor applicatieontwikkeling zijn niet per definitie gedoemd om te mislukken. Er zijn verschillende manieren om ze te laten slagen. Als u rekening houdt met onze zes tips, dan zit uw project alvast op de goede weg:

1. Ken uw klanten;
2. Ga stap voor stap te werk;
3. Houd het budget goed in de gaten;
4. Ontwikkel software als een professional;
5. Testen, testen en nog eens testen;
6. Non-stop samenwerken en communiceren.

Op zoek naar meer tips?

Bent u op zoek naar nog meer tips om van uw groot, complex project voor applicatieontwikkeling een succes te maken? Bent u op zoek naar een partner die zijn kwaliteiten heeft bewezen in talloze complexe IT-projecten?

Neem contact op en ontdek hoe wij u kunnen helpen.

Henrik Oskam
Business Development Executive Applications
henrik.oskam@cegeka.com

WWW.CEGEKA.COM