Software developer

Software developer

Hans van Rheenen

Boom beroepsonderwijs · Meppel · 2020

Boom beroepsonderwijs
info@boomberoepsonderwijs.nl
www.boomberoepsonderwijs.nl

Auteur: Hans van Rheenen
Redactie en opmaak: Henk Pel
Titel: Software developer
ISBN 978 90 372 5744 1
Illustraties: Erik Eshuis
Eerste druk / eerste oplage

Inhoud

Inleiding

Om een diploma te kunnen krijgen moet je aan een hele reeks voorwaarden voldoen. Deze zijn opgesteld in samenwerking met het bedrijfsleven en vastgesteld door de minister van Onderwijs, Cultuur en Wetenschap. Hieronder staan de kennis en vaardigheden waaraan je moet voldoen om als beginnend beroepsbeoefenaar aan de slag te kunnen als Software developer (Crebonr. 25604). Het volledige kwalificatiedossier is te vinden op de site van de Stichting Samenwerking Beroepsonderwijs Bedrijfsleven, oftewel S-BB (https://www.s-bb.nl).

Programmeren en coderen zijn de belangrijkste bezigheden van een developer. Dit boek gaat daar niet over. Dit boek gaat over alle zaken die om het maken van code heen van belang zijn. In weke omgeving je iets ontwikkelt zal sterk afhangen van het bedrijf en/of de toepassing. Op de site van Brinkman Uitgeverij kun je voor de verschillende omgevingen lesmateriaal vinden.

Het gaat daarbij om het werkproces *B1-K1-W3: Realiseert (onderdelen van) software* met de volgende eindtermen:

- heeft specialistische kennis van de principes van object oriented programming (OOP), waaronder encapsulation, modularity, inheritance, polymorphism;
- heeft specialistische kennis van één of meer programmeertalen (syntax en semantiek);
- kan diagrammen lezen, interpreteren en maken zoals UML;
- kan één of meerdere programmeertalen voor softwareontwikkeling toepassen (syntax en semantiek);
- kan één of meerdere softwareontwikkelingsmethodieken toepassen zoals Waterval, iteratief of incrementeel;
- kan één of meerdere softwareontwikkelingsprogramma's toepassen zoals IDE's;
- kan één of meerdere softwareontwikkelingstechnieken toepassen zoals objectgeoriënteerd programmeren, ECS, functioneel programmeren;
- kan gegevensverzamelingen omzetten in andere structuren;
- kan ontwerpeisen toepassen;
- kan technieken voor informatiebeveiliging toepassen.

B1-K1: Realiseert software

De beginnend beroepsbeoefenaar:

- Heeft brede kennis van cyber security en bedreigingen van netwerken en systemen.
- Heeft brede kennis van relevante zaken zoals wetgeving op het gebied van privacy, copyright en auteursrecht, computercriminaliteit.
- Heeft brede kennis van nieuwe ontwikkelingen op het gebied van software, zoals AI (artificial intelligence), machine learning en big data.

- Heeft kennis van ontwikkelingen op het vlak van ICT-infrastructuur en devices en welke consequenties deze op software development hebben.
- Heeft specialistische kennis van de IT-infrastructuur en/of ontwikkelingsplatforms waarop de software wordt toegepast.
- Heeft specialistische kennis van de principes van object oriented programming (OOP), waaronder encapsulation, modularity, inheritance, polymorphism.
- Heeft specialistische kennis van één of meer programmeertalen (syntax en semantiek).
- Heeft specialistische kennis van één of meerdere software-ontwikkelingsmethodieken zoals Waterval, iteratief of incrementeel.
- Heeft specialistische kennis van één of meerdere software-ontwikkelingsprogramma's zoals IDE's.
- Heeft specialistische kennis van één of meerdere testing tools en testtechnieken.
- Heeft specialistische kennis van één of meerdere software-ontwikkelingstechnieken zoals OOP, ECS of functioneel programmeren.
- Heeft specialistische kennis van licenties en gebruiksrechten.
- Kan diagrammen lezen, interpreteren en maken zoals UML.
- Kan één of meerdere programmeertalen voor software-ontwikkeling toepassen (syntax en semantiek).
- Kan één of meerdere software-ontwikkelingsmethodieken toepassen zoals Waterval, iteratief of incrementeel.
- Kan één of meerdere software-ontwikkelingsprogramma's toepassen zoals IDE's.
- Kan één of meerdere software-ontwikkelingstechnieken toepassen zoals objectgeoriënteerd programmeren, ECS, functioneel programmeren.
- Kan gegevensverzamelingen omzetten in andere structuren.
- Kan ontwerpeisen toepassen.
- Kan technieken voor informatiebeveiliging toepassen.
- Kan actief versiebeheer toepassen.
- Kan relevante wetgeving op het gebied van privacy, intellectueel eigendomsrecht, computercriminaliteit toepassen op software.
- Kan principes van Secure Software Development Life Cycle (SSDLC) toepassen.
- Kan controleren of een ontwerp van software voldoet aan gangbare beveiligingseisen en de bevindingen toelichten aan betrokkenen.

B1-K2: Werkt in een ontwikkelteam

De beginnend beroepsbeoefenaar:

- Heeft specialistische kennis van één of meerdere software-ontwikkelingsmethodieken zoals Waterval, iteratief of incrementeel.
- Heeft specialistische kennis van één of meerdere software-ontwikkelingsprogramma's zoals IDE's.
- Heeft specialistische kennis van één of meerdere testing tools en testtechnieken.
- Heeft specialistische kennis van één of meerdere software-ontwikkelingstechnieken zoals OOP, ECS of functioneel programmeren.
- Kan met betrokkenen communiceren over werkzaamheden.

- Kan gesprekstechnieken toepassen (zoals luisteren, samenvatten, doorvragen).
- Kan presentatietechnieken toepassen.
- Kan projectmatig werken.
- Kan relevante wetgeving op het gebied van privacy, intellectueel eigendomsrecht, computercriminaliteit toepassen op software.

1 Projectmatig werken

Als ontwikkelaar van software zul je bijna altijd aan projecten werken. Soms iets heel kleins, dat je alleen oplost, soms als onderdeel van een heel groot project, met veel deelprojecten. Maar meestal iets dat daartussenin zit qua omvang. Het overleg dat je moet voeren heeft dan ook vaak te maken met jouw rol binnen het project. Binnen een project zal er sprake zijn van een bepaalde vorm van projectorganisatie. Hoe en wat hangt van de gebruikte methodiek af. De meest voorkomende aspecten met betrekking tot een project worden hier behandeld.

Project

Anders dan een organisatie is een project eindig. Een project heeft een duidelijk eindpunt dat van tevoren is gedefinieerd. Een project is een eenmalige activiteit met een duidelijk eindresultaat. Het wordt meestal vastgelegd in een projectplan en geregeld door het projectmanagement. Binnen een project is de inzet van mensen geregeld en vastgelegd. Een van de definities van een project is:

> *een project is een unieke opgave, begrensd in tijd en middelen en afgesloten met een projectresultaat*

De kenmerken van een project zijn: tijd, middelen, product en management.

Tijd

Een project is een tijdelijk gegeven. Een project kan een week duren of een aantal jaren, maar zal nooit oneindig doorlopen. Een project heeft een van tevoren beoogd eindpunt. Het komt voor dat bij de eerste planning van een project de einddatum nog niet is vastgesteld. Dit is het geval als de duur van de tussenliggende stappen nog onduidelijk is.

Middelen

Binnen een project spreekt men van *resources*. Resources zijn alle middelen die binnen het project ingezet worden. Dit zijn zowel menselijke inspanningen als materialen. Beide vormen van resources brengen kosten met zich mee en bepalen de totale kostprijs van het project. Men streeft ernaar om de in te zetten resources bij aanvang van het project vast te leggen en in te schatten.

Product

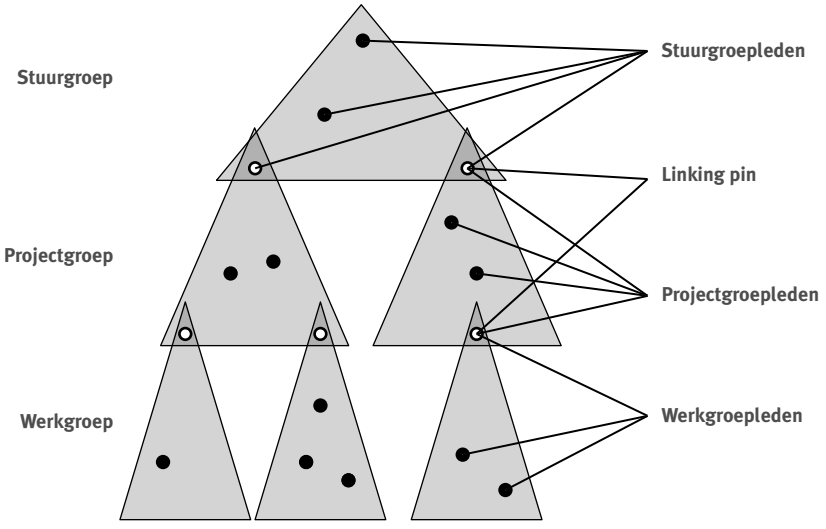
Op de einddatum moet het project uitmonden in een eindproduct. Er wordt een project opgezet om iets uitzonderlijks te realiseren. De normale organisatie gaat gewoon door.

Management

Een project heeft een eigen projectmanagement. Het projectmanagement moet ervoor zorgen dat het project met de beschikbare middelen binnen de afgesproken tijd tot een goed einde wordt gebracht.

Linking pin

Veelgebruikt is het linking-pin-principe. Er wordt een stuurgroep ingericht. Die stuurgroep richt één of meer projectgroepen in en die richten ieder weer één of meer werkgroepen in.



Er is een duidelijke taakverdeling binnen deze groepen.

Stuurgroep

De stuurgroep heeft de leiding over het gehele project. De stuurgroepleden houden zich niet bezig met de dagelijkse werkzaamheden voor het project. Zij komen af en toe bijeen, meestal één- of tweemaal per maand. Aan de stuurgroep wordt gerapporteerd door de linking-pin-leden van de projectgroep over de voortgang van elk project. De stuurgroep schept de voorwaarden wat betreft tijd en geld om deze voortgang te realiseren.
De stuurgroep bestaat uit slechts een klein aantal personen. De samenstelling kan bestaan uit een directielid, de accountant, de ICT-manager en de projectleider.

Projectgroep

De projectgroep heeft de dagelijkse leiding over het project. De projectgroep komt vaak, minimaal eens per week, bij elkaar. De voorzitter van de projectgroep is meestal de projectleider en in die hoedanigheid ook lid van de stuurgroep. De andere projectgroepleden zijn verantwoordelijk voor de producten die het project moet opleveren.

Naast de projectleider kunnen in de projectgroep de volgende functionarissen zitten:

- applicatiebeheerders
- automatiseerders
- controlefunctionarissen
- materiedeskundigen (eindgebruikers)
- organisatieadviseurs
- personeelsfunctionarissen
- technische systeembeheerders

De samenstelling is afhankelijk van de aard van het project.

Werkgroep

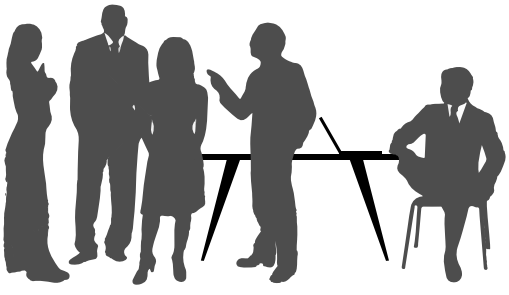
Om ervoor te zorgen dat ieders talenten goed tot hun recht komen en niet iedereen zich met alles gaat of hoeft te bemoeien, worden er werkgroepen ingericht. Een werkgroep heeft een specifieke taak binnen het project. Elke werkgroep is belast met één aspect van het project.

Deze werkgroepen zijn verantwoordelijk voor hun eigen resultaten. De werkgroepleider is de linking pin naar de projectgroep en is verantwoording verschuldigd aan de projectgroep.

Mogelijke werkgroepen:

- Administratieve organisatie
- Conversie
- Invoering
- Juridische aspecten (Wet persoonsregistratie)
- Nazorg
- Ontwikkeling van de verschillende deelsystemen
- Opleiding
- Testwerkgroep

Gebruikers zijn in de werkgroep een belangrijk onderdeel. Zij weten waar het over gaat en kunnen vanuit hun professie een belangrijke bijdrage leveren aan een werkgroep.



Waar je ook in een project wordt ingezet, vaak zal je deelname aan het project

maar een deel van je tijd beslaan. Daarom is het belangrijk dat er duidelijke afspraken worden gemaakt, ook over jouw inzet. Wanneer, waar en met welke bevoegdheid neem je deel aan het project? Zelfs als alles goed is afgesproken, kun je nog in moeilijke situaties terechtkomen. Je hebt namelijk te maken met twee verschillende leidinggevenden. Je hebt een functioneel leidinggevende in de vorm van je directe chef. Maar in de werkgroep heb je te maken met een operationeel leidinggevende in de vorm van de werkgroepleider of de projectgroepleider. Wat ga je doen als je baas zegt: 'We hebben een spoedklus, vandaag mag je even niet aan het project werken'?
Als je het goed doet, laat je dat door de twee leidinggevenden zelf uitzoeken. Breng ze met elkaar in contact en laat ze een oplossing bedenken. Wijs naar de afspraken zoals ze zijn gemaakt bij de inrichting van het project.

1.2 Het projectplan

Software kan op basis van veel verschillende methodieken worden ontwikkeld. Maar in bijna alle gevallen ligt er een vraag van een klant aan ten grondslag en op basis daarvan een projectplan. De vraag van de klant zal vaak resulteren in een Pakket van Eisen (PvE) waaraan de ontwikkelaar moet voldoen. Het projectplan zal in veel gevallen een Plan van Aanpak (PvA) bevatten. We onderscheiden drie 'hoofdmanieren' op het gebied van software-ontwikkeling: de Watervalmethode, prototyping of de iteratieve methode.
In alle gevallen zal er sprake zijn van de behoefte van een klant, waaraan voldaan moet worden. Een klant wil graag weten waar hij aan toe is: 'Wanneer beginnen we?', 'Wanneer is het klaar?', 'Wat gaat het kosten?', 'Wat wordt er van mij verwacht?', 'Wie zijn er nog meer bij betrokken?'– en mogelijk zijn er nog meer vragen. Door een projectplan te schrijven wordt het voor een klant tastbaarder en krijgt hij een belangrijk deel van de antwoorden op zijn vragen.

Wanneer met de klant overeengekomen is dat er iets gemaakt gaat worden moet er een plan gemaakt worden van de aanpak. Een projectplan wordt gebruikt om de uitvoering van het project te begeleiden. Het belangrijkste doel van het projectplan is de plannings-aannames en beslissingen te documenteren. Op basis van het projectplan wordt dan gemonitord of mijlpalen gehaald worden en of er moet worden bijgesteld. Het projectplan kan bestaan uit de volgende onderdelen:

- Inleiding
- Projectomschrijving
  - Aanleiding
  - Doelen
  - Resultaat
  - Afbakening
  - Risico's
  - Effecten
  - Randvoorwaarden

- Fasering en planning
- Projectbeheersing (van tijd, geld, kwaliteit, informatie en organisatie)

Taken en rollen

Taken

In een goed projectteam zijn de taken slim verdeeld. Het team bestaat uit specialisten met kennis en ervaring. Deze kennis en ervaring moeten zoveel mogelijk benut worden. De webdeveloper kan best een site-ontwerp maken, maar is er wellicht een mediadeveloper die het nog beter kan?

Naast het specialistische werk moeten in een project ook dingen gebeuren die niet zijn voorbehouden aan specialisten. De projectleider zal in de meeste gevallen de vergaderingen voorzitten. Maar wat als de projectleider een beroerde gespreksleider is? Dat doe je elkaar toch niet aan? Kijk eens hoe een ander het als gespreksleiderschap ervan af brengt. Zo zijn er veel zaken die moeten gebeuren. In alle gevallen is het belangrijk dat de persoon die het geschiktst is voor een taak deze ook krijgt toegewezen. Je moet hierbij denken aan:

- Bijhouden van het projectarchief
- Documenteren
- Eindredactie van rapporten
- Maken van de verslagen
- Onderhandelen met opdrachtgevers
- Uittesten van eerste versies

Je wilt graag in het projectteam spelvreugde en creativiteit ontwikkelen en ervoor zorgen dat iedereen goed functioneert.

Rollen

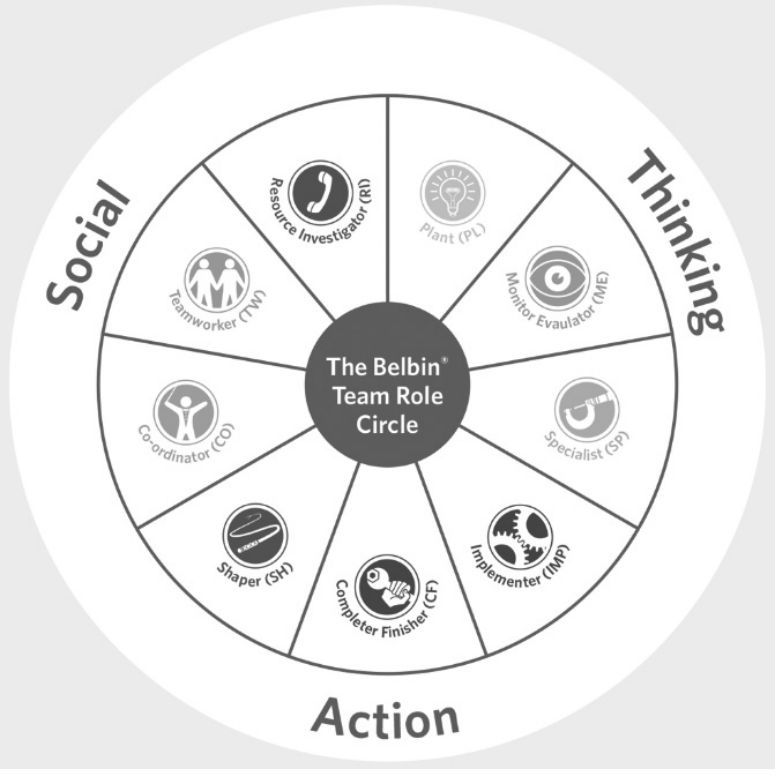
In een team zijn er niet alleen taken, maar ook rollen. Een rol is een afspraak over welk gedrag men per situatie vertoont. Het is niet het gedrag wat men altijd vertoont, maar gedrag dat is afgesproken. Je weet precies wat je aan elkaar hebt: 'Als jij een voorstel hebt om af te wijken van het PvA dan ga ik zeuren dat het toch niet kan. Samen zoeken we dan een haalbaar compromis.' 'Als Hans weer eens begint met 'Maar...' dan vraag jij hem niet alleen zijn bezwaar te noemen, maar ook een oplossing voor dat bezwaar.'

Een rolverdeling kan ook dodelijk zijn voor de creativiteit en het elan binnen het team. Toch is een afspraak over de rolverdeling belangrijk: duidelijkheid gaat boven alles! Mogelijk dat je afspraken kunt maken over verschillende rollen die iemand kan spelen.

De belangrijkste teamrollen zijn door de Britse onderzoeker Meredith Belbin onderzocht en als volgt omschreven:

- **Afronder (Completer Finisher)**
  - Consciëntieus
  - Regelt het planmatig verloop
  - Werkt achter de schermen
  - Zorgelijk en bezorgd
- **Brononderzoeker (Resource Investigator)**
  - Communicatief
  - Enthousiast
  - Extravert
  - Gaat op zoek naar ideeën
  - Nieuwsgierig
- **Coördinator (Coordinator)**
  - Doelgericht
  - Kalm
  - Laat ieder teamlid tot zijn recht komen
  - Realistisch en nuchter
  - Zelfvertrouwen
- **Groepswerker (Teamworker)**
  - Gevoelig
  - Kan goed luisteren
  - Mild
  - Sociaal gericht
  - Stimuleert en integreert
- **Monitor (Monitor Evaluator)**
  - Analyseert en bekritiseert
  - Nuchter
  - Weinig emoties
  - Zakelijk
- **Plant (Plant)**
  - Fantasie
  - Individualistisch
  - Intellect, kennis
  - Onorthodoxe ideeën
- **Specialist (Specialist)**
  - Plichtsgetrouw
  - Praktisch
  - Zelfdiscipline
  - Zet beslissingen om in werkzaamheden
- **Uitvoerder (Implementer)**
  - Ordelijk
  - Plichtsgetrouw
  - Taakgericht
  - Weinig flexibel
- **Vormer (Shaper)**

- Dynamisch
- Energiek
- Extravert
- Ongeduldig



Als een team uit een groot aantal personen bestaat, moet om te beginnen worden bepaald welke rollen er van nature aanwezig zijn. De sterkte en/of zwakte van een team kun je met behulp van de Belbin Team Circle in beeld brengen. Als je een grote groep hebt, deel je deelnemers in 'teams' van ongeveer vier tot zes in. Als je met een kleinere groep werkt, hoef je de groep natuurlijk niet op te splitsen.

1 Vraag elke deelnemer zichzelf drie rollen toe te bedelen die het dichtst bij hem in de buurt zitten. Laat dit ook door twee collega's doen. Vergelijk de uitkomst en bepaal samen in welke volgorde de rollen het beste bij die persoon passen.
2 Elk team tekent een Belbin-cirkel met negen secties, een voor elk van de teamrollen van Belbin. Van elk teamlid worden de namen in de segmenten gezet die overeenkomen met hun bovenste twee rollen.
3 Laat het team vijf hoofdgebieden benoemen waarvan zij denken dat daar hun sterke en zwakke punten liggen.
4 Het team moet drie actiepunten bedenken op basis van de bevindingen. Dit moeten actiepunten zijn die ervoor zorgen dat het team beter gaat presteren.

Fasering en planning

De fasering en planning is het belangrijkste onderdeel uit het projectplan. De projectleider zal dit continu gebruiken om de voortgang te monitoren en daar waar nodig actie te ondernemen en een en ander bij te stellen.

Planning

Een planning is 'het voor de toekomst systematisch voorbereiden'. Dit doe je door het op elkaar afstemmen en het nemen van besluiten die noodzakelijk zijn om een bepaald doel te bereiken.
De complexiteit van een planning heeft nauwe samenhang met de omvang en de complexiteit van het project. Of je ergens in je eentje aan werkt of dat je met een team aan een opdracht werkt heeft een directe relatie tot hoe de planning moet worden opgezet.
Er is een veelheid aan planningssoftware beschikbaar, zowel gratis als commerciële software. De keuze voor het gebruik van bepaalde software zal afhangen van geld en ervaring, maar vooral van de aard van het project. Vaak wordt er gebruik gemaakt van een spreadsheet en bouwt men een eigen planningsomgeving.
Vaak wordt er gebruik gemaakt van een strokenplanning. Aan de hand van wat voorbeelden werken we deze uit.

Voorbeeld 1

	Voorbeeld 1	**Week**																						
Fase	**Mijlpalen/activiteiten**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Pakket van eisen																								
2	Plan van aanpak																								
3	Ontwerp																								
4	Realiseren																								
5	Implementeren																								

Wanneer je alleen werkt voor een klant en bijvoorbeeld een app moet gaan maken, is een eenvoudige strokenplanning vaak genoeg. In de planning staat wat je wanneer gaat opleveren. Voor jou en je klant is dit waarschijnlijk toereikend.
In het schema is te zien dat de voorbereidingen voor de implementatie al beginnen voordat de realisatie is afgesloten.

Voorbeeld 2

	Voorbeeld 2			Week																						
Fase	**Mijlpalen/activiteiten**	**Actoren**	**Verantwoordelijke**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Pakket van eisen	Mieke/Jesse/Klant	Mieke																								
2	Plan van aanpak	Mieke	Mieke																								
3	Ontwerp																										
3.1	Functioneel ontwerp	Mieke/Jesse/Klant	Mieke																								
3.2	Technisch ontwerp	Ahmed/Jesse	Ahmed																								
4	Realiseren	Ahmed/Jesse	Ahmed																								
5	Implementeren	Mieke/Ahmed/Klant	Mieke																								

Het kan zijn dat er wat meer detaillering nodig is in de planning. Als voorbeeld de mijlpaal Ontwerp, die uit twee subdelen bestaan, de mijlpaal Functioneel ontwerp en de mijlpaal Technisch ontwerp.
Ook kan het gebeuren dat er meerdere personen bij het project betrokken zijn die ieder hun eigen verantwoordelijkheden hebben. Mieke weet dat zij in week 4 het Functioneel ontwerp moet opleveren wil Ahmad in week 5 kunnen starten met het Technisch ontwerp. Per fase is er duidelijk wie er een rol heeft (actoren).

Voorbeeld 3

	Voorbeeld 3			Week																							
Fase	**Mijlpalen/activiteiten**	**Actoren**	**Verantwoordelijke**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Pakket van eisen	Mieke/Jesse/Klant	Mieke																								
2	Plan van aanpak	Mieke	Mieke																								
3	Ontwerp																										
3.1	Functioneel ontwerp																										
3.1.1	Vaststellen requirements	Mieke/Jesse/Klant	Mieke																								
3.1.2	Prioriteit requirements	Jesse/Klant	Mieke																								
3.1.3	Maken use case	Jesse	Mieke																								
3.1.4	Opleveren functioneel ontwerp	Mieke/Klant	Mieke																								
3.2	Technisch ontwerp	Ahmed/Jesse	Ahmed																								
4	Realiseren	Ahmed/Jesse	Ahmed																								
5	Implementeren	Mieke/Ahmed/Klant	Mieke																								

Hoe groter een project wordt hoe nauwkeuriger de planning moet worden uitgewerkt. In Voorbeeld 3 zie je een voorbeeld van de uitwerking van de mijlpaal Functioneel ontwerp. Deze bestaat uit de activiteiten: Vaststellen requirements, Prioriteit requirements, Maken use case en Opleveren functioneel ontwerp.
Een nog verdere uitwerking van de planning zou kunnen bestaan uit het aantal uren dat nodig is. Voor de voortgang zou er bijgehouden kunnen worden hoeveel procent van een bepaalde activiteit of mijlpaal al gerealiseerd is.

Projectbeheersing

Onder projectbeheersing zie je voor een deel principes van PRINCE2 terugkomen in combinatie met de punten die zijn behandeld bij de projectomschrijving en de projectplanning. PRINCE2 is de opvolger van de methode PRINCE, die in de jaren tachtig is ontwikkeld voor ICT-projecten. Het bundelen van praktijkervaringen, zogenaamde best practices, is hierbij de kern. PRINCE is door de Britse semi-overheidsinstelling de Office of Government Commerce (OGC) ontwikkeld en wordt in de versie PRINCE2 in veel bedrijven in Europese landen gebruikt. Het gaat om zaken als:

- beheersing van het bestaansrecht
- beheersing projectscope
- beheersing tijd en geld
- beheersing kwaliteit
- beheersing overige randvoorwaarden
- beheersing van de projectrisico's
- beheersing van de projectaanpak

Van belang is dat je hier aangeeft hoe, door wie en wanneer de projectvoortgang in de gaten wordt gehouden. Het zou heel goed kunnen zijn dat de oplevering van elke mijlpaal of activiteit een moment is om even te kijken of en hoe de volgende stap moet worden gezet.

1.3 Opgaven

1. Wat is de belangrijkste input voor het projectplan?
2. Wat zijn de belangrijkste taken in een projectteam software-ontwikkeling?
3. Wat is een rol?
4. Over welke eigenschappen moet een Groepswerker beschikken?
5. Over welke eigenschappen moet een Uitvoerder beschikken?
6. Vul met een team van 4 tot 6 personen de Belbin Team Circle in. Bepaal eerst over welke eigenschappen elk teamlid beschikt. Volg de stappen zoals die zijn beschreven in paragraaf 1.2.
7. Uit welke onderdelen kan een projectplan bestaan?
8. Wat is het belangrijkste dat moet worden beschreven bij het onderdeel Resultaat van een projectomschrijving?
9. Wat is het belang van het benoemen van risico's in de projectomschrijving?
10. Geef een definitie van planning.
11. Wat is het belang van projectbeheersing?
12. Maak de planning in Voorbeeld 2 verder af. Gebruik hierbij de volgende gegevens.

- Stappen Technisch ontwerp:
  - Vertaal het Functioneel ontwerp naar technische specificaties.
  - Ontwerp een grafisch ontwerp van de user interface.
  - Ontwerp een relationeel datamodel.
  - Lever het Technisch ontwerp.

- Stappen Inrichten ontwikkelomgeving:
  - Inventariseer de benodigde onderdelen voor de realisatie.
  - Installeer en configureer de ontwikkelomgeving.
  - Test de ontwikkelomgeving.
  - Documenteer de instellingen en wijzigingen.

- Stappen Realisatie:
  - Opleveren deelproduct 1.
  - Opleveren deelproduct 2.
  - Opleveren deelproduct 3.
  - Opleveren deelproduct 4.

- Stappen Implementatie:
  - Stel een acceptatietest op voor de applicatie.
  - Begeleiden van de acceptatietest.

– Beschrijf eventuele aanpassingen.
– Werk de documentatie van de applicatie bij.
– Evalueer het product met de betrokkenen.
– Evalueer het proces met de betrokkenen.
– Leg dit vast in een evaluatieverslag.
– Laat het evaluatieverslag accorderen.

1.4 Projectomschrijving projectplan

De projectomschrijving kan bestaan uit de volgende onderdelen:

- aanleiding
- doelen
- resultaat
- afbakening
- planning
- risico's
- randvoorwaarden

Aanleiding

Onder dit kopje geef je aan wat de aanleiding is om het project te starten. Vaak is dat een probleem dat opgelost moet worden.

Doelen

Hier geef je aan wat het doel is dat met de uitvoering van het project moet worden bereikt. Omdat een project meestal bestaat uit een aantal elkaar opvolgende fasen zul je per fase de doelen omschrijven. Dit worden ook wel de mijlpalen genoemd.

Resultaat

Bij het resultaat moet exact worden aangegeven wat er opgeleverd moet worden. Hierbij moet je zo concreet mogelijk elk op te leveren product benoemen. Benoem de cruciale succesfactoren, de onderdelen waarop geconcludeerd kan worden of het project is geslaagd of juist mislukt. Het is verstandig om een duidelijke afbakening te maken door ook de zaken te benoemen die niet door het project worden opgeleverd of door de beperkingen te noemen. Dit noemen we ook wel de mijlpaalproducten.

Afbakening

Heel belangrijk is het managen van verwachtingen. Dat doe je door duidelijk aan te geven wat er wel en wat er niet wordt gedaan. Hoe duidelijker je dit doet, hoe minder 'gezeur' je hoeft te verwachten. De afbakening kan je halen uit de MoSCoW-methode (zie paragraaf 4.5) die in het Programma van eisen is opgesteld.

Planning

Een planning is het voor de toekomst systematisch voorbereiden. Dit doe je door het op elkaar afstemmen en het nemen van besluiten die noodzakelijk zijn om een bepaald doel te bereiken. De detaillering zal afhangen van de aard en de omvang van het project. Het toepassen van een strokenplanning is meestal behulpzaam.

Risico's

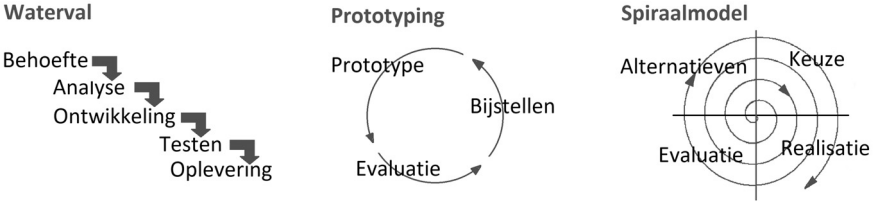
Aan elk project zijn risico's te koppelen die een succesvol afsluiten van het project in de weg kunnen staan. Vaak zijn die al in een vroeg stadium zichtbaar. Door ze duidelijk te benoemen, zijn ze goed in beeld en kan erop gestuurd worden om ze zoveel mogelijk te vermijden.
Een risico is de kans dat iets fout gaat. Maar dat is niet exact waar het hierom gaat. Het risico dat er een taalfout op de website staat is van een heel andere orde dan dat de website niet op tijd afkomt. Wat heb je aan een website met de aankondiging van een evenement als dat evenement al voorbij is? Risico's moeten ook gewogen worden. De mate van impact moet worden meegenomen. Daar kan de projectleider dan op sturen.

Randvoorwaarden

Met het benoemen van de randvoorwaarden of ook wel de uitgangspunten waaronder het project uitgevoerd moet worden, kun je een deel van de risico's verkleinen. De beschikbaarheid van mensen en middelen is hierbij een belangrijk onderdeel, zoals ook wie verantwoordelijk is voor de realisatie. Dit maakt het de projectleider makkelijk om mensen aan te spreken als er problemen dreigen te ontstaan.

2 Ontwikkelmethode

Er zijn veel verschillende methoden om software te ontwikkelen. Je kunt natuurlijk gewoon aan het werk gaan en iets maken, maar de kans dat dat iets bruikbaars oplevert is klein. De bedoeling van het gebruik van een software-ontwikkelmethodiek is om het software-ontwikkeltraject binnen de vastgestelde tijd en het budget af te ronden. Al vanaf het eerste begin is men op zoek naar de ideale methode voor de ontwikkeling van software. In de praktijk blijkt dat die niet bestaat. De omvang en de aard van een traject, de mensen en beschikbare middelen zijn allemaal factoren die meespelen. In jouw praktijk zul je met verschillende methoden te maken krijgen. Daarom is het belangrijk dat je op de hoogte bent van de belangrijkste kenmerken.



2.1 Watervalmethode

De Watervalmethode vraagt veel ervaring. Zij lijkt erg op wat er in de bouw gebeurt en is zo genoemd omdat het ontwikkelproject in een aantal duidelijke fasen als een waterval vloeiend naar beneden loopt. Men laat om te beginnen onderzoek

doen naar de behoefte. Dit is meteen de belangrijkste fase. Als het lukt om duidelijk vast te leggen wat men wil, dan is de kans reëel dat het gaat lukken. Vervolgens komt er een traject van ontwerp, realisatie, implementatie en gebruik en beheer. De bekendste Watervalmethode is SDM.

SDM

SDM kent zeven fasen die in een vastgestelde volgorde worden afgewerkt. Het einde van een fase wordt een mijlpaalproduct genoemd. Voor elke fase worden dan ook mijlpaalproducten beschreven, de zaken die opgeleverd worden aan het einde van deze fase. Je kunt pas met de volgende fase beginnen als de mijlpaalproducten zijn goedgekeurd door de opdrachtgever. Wanneer een mijlpaalproduct is goedgekeurd, is dit vastgesteld en kan het niet meer worden aangepast.
De belangrijkste voordelen hiervan zijn dat de voortgang goed kan worden gevolgd. Omdat er een planning met deadlines wordt gebruikt, kan gecontroleerd worden of de voortgang van het project op schema ligt.

Fasen van SDM

Fase 0 – Informatieplanning

In de eerste fase wordt een analyse gemaakt van de huidige situatie en van mogelijke oplossingen voor het probleem. De mijlpaalproducten zijn:

- de Situatieanalyse, oftewel de beschrijving van de huidige situatie;
- het Informatieplan, waarin het totale systeem wordt beschreven, inclusief afspraken over de toekomstige informatiebehoefte, de verschillende mogelijkheden van automatisering en de consequenties hiervan.

De informatiebehoefte is bij alles wat er verder wordt gedaan het uitgangspunt. Dat is wat de klant nodig heeft.

Fase 1 – Definitiestudie

In deze fase wordt onderzocht of het realiseren van het project haalbaar is. Het gaat om zaken als: kan het, is het betaalbaar, levert het meer op dan het kost? De mijlpaalproducten zijn:

- het Rapport definitiestudie: een omschrijving van de systeemeisen;
- het Systeemconcept: een omschrijving van de hoofdfuncties van het systeem;
- Kosten-batenanalyse: hierin wordt beschreven hoe men het systeem gaat bouwen en welke voorwaarden en kosten aan het systeem verbonden zijn.

Fase 2 – Basisontwerp

Het Basisontwerp beschrijft wat het systeem zal doen. De mijlpaalproducten zijn:

- de Basisgegevensstructuur: met welke gegevens wordt in het nieuwe systeem gewerkt en hoe worden de gegevens van het oude naar het nieuwe systeem geconverteerd?
- de Basisfunctiestructuur: de functies van het systeem;
- de Technische systeemstructuur: welke apparatuur, programmatuur en bestandsstructuur zijn nodig?

Fase 3 – Detailontwerp

Het Basisontwerp wordt verder in detail uitgewerkt door specifieker te beschrijven wat elk onderdeel van het systeem doet. Het resultaat hiervan is het Functioneel ontwerp. De mijlpaalproducten zijn:

- het Rapport functioneel ontwerp: hierin staat de beschrijving van de functies, van de gegevensstructuur en van de interface;
- het Rapport technisch ontwerp: hierin staan de formulieren en procedures, interfaces, opslagstructuur en welke hardwarecomponenten nodig zijn;
- Plan voor systeemtest: dit is de beschrijving van de uit te voeren test;
- Plan voor acceptatietest, oftewel de test voor de gebruiker; de opdrachtgever kan op basis van deze test het systeem wel of niet accepteren.

Fase 4 – Realisatie

In deze fase wordt het systeem gebouwd op basis van het Functioneel ontwerp en het Technisch ontwerp. De mijlpaalproducten zijn:

- het Rapport systeemtest, oftewel de uitslag van de Systeemtest;
- het Rapport acceptatietest, oftewel de uitslag van de Acceptatietest.

Fase 5 – Invoering

Het installeren van het systeem en het begeleiden van de gebruikers bij het gebruik, eventueel door het verzorgen van trainingen. De mijlpaalproducten zijn:

- conversie en invoeringsplan hoe een en ander moet plaatsvinden;
- afgesloten projectdocumentatie: alle documentatie wordt nagekeken en bijgewerkt opgeleverd aan de beheerder;
- overdrachtsrapport: dit is het eindrapport waarmee het systeem overgedragen wordt aan de organisatie.

Fase 6 – Gebruik en beheer

Belangrijk zijn de regels die ervoor zorgen dat het systeem blijft voldoen aan de gestelde eisen. Verder wordt in deze fase aangegeven hoe defecten en storingen verholpen moeten worden. Zolang het systeem wordt gebruikt zal deze fase niet eindigen. De mijlpaalproducten zijn:

- organisatie van gebruik en onderhoud;
- verschillende gebruiks- en beheersplannen;
- systeembeschrijving;
- periodiek beoordelingsrapport.

Nadelen van de Watervalmethode

De Watervalmethode is een lange tijd de meest gebruikte methode geweest. De omvang van de software en de snelheid waarmee software moet worden ontwikkeld zijn twee van de veranderingen die hebben plaatsgevonden. Met deze veranderingen is ook de behoefte ontstaan aan een andere ontwikkelmethodiek dan de Watervalmethode. Voor de ontwikkeling van software zijn de belangrijkste bezwaren:

- de kans op mislukking is groot;
- deze methode kost veel tijd en dus geld;
- in de definitiestudie moet het systeem al volledig worden beschreven;
- er is weinig gebruikersparticipatie;
- het is een inefficiënte manier van werken.

2.2 Spiraalmethode

Agile

Agile werken kom je in steeds meer organisaties tegen. Was het in het begin nog voorbehouden aan automatiseerders, op dit moment kom je Agile werken in allerlei organisaties tegen.
Als je Agile vertaalt uit het Engels dan krijg je in het Nederlands: behendig, lenig, rap en vlug. Dat is dan ook meteen waar het naar verwijst.

Er zijn verschillende Agile-methoden. Agile staat voor software-ontwikkeling in korte overzichtelijke periodes die zich herhalen, zogenoemde iteraties. Elke periode is een project op zich en levert een werkend product op. Een periode kan een dag, een week maar zeker niet meer dan een maand duren. Het behendige, lenige, rappe en vlugge van Agile zit hem in het feit dat elke iteratie (herhaling) een werkend product oplevert. Dit wordt getest, getoond en geëvalueerd, zodat men weet of men op de goede weg zit. Omdat men steeds een werkend product ziet wordt alles heel concreet. Op basis van de uitkomst kan men een nieuwe cyclus afspreken waarin nieuwe ideeën worden gerealiseerd.

In een Agile-organisatie is er sprake van zelforganiserende teams die de klantwaarde altijd als uitgangspunt hebben. Het werk wordt gedaan op een herhalende

manier (iteratie) waarbij met regelmaat sprake is van interactie met toekomstige gebruikers. De organisatie kan hierdoor het te ontwikkelen product voor iedere gebruiker continu verbeteren.

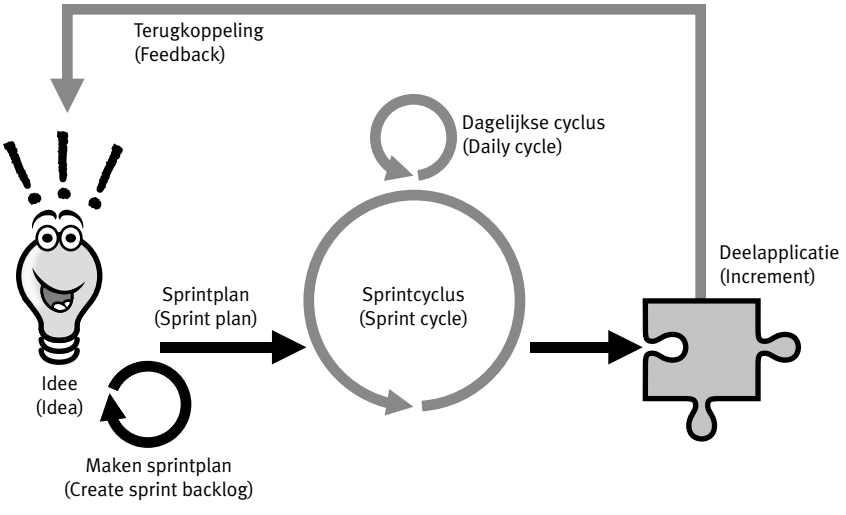
Agile-methoden zijn sterk gericht op communicatie. Men streeft naar persoonlijk contact in plaats van geschreven verslaglegging. Er wordt dan ook weinig tijd besteed aan geschreven documentatie. Een Agile-team bevat verschillende specialisten en een vertegenwoordiging van de opdrachtgever. Bij voorkeur werken zij gedurende het project in één ruimte. Deze ruimte wordt ook wel de *bullpen* genoemd. De voortgang wordt gemeten in prototypen en een werkend product. Agile-ontwikkelaars zetten zich vaak af tegen het Waterval-ontwikkelmodel. De Waterval-methoden zijn te log en te bureaucratisch (veel papierwerk!). Maar dat is dan ook meteen de zwakke plek van de Agile-methode. Plannen, telkens veranderen en weinig of niet documenteren kan leiden tot chaos. Daar staat tegenover dat Agile vaak kostenbesparend is en veel eerder werkende producten oplevert.

De belangrijkste redenen om Agile toe te passen zijn:
- Agile is vaak kostenbesparend.
- Er zijn snel eerste resultaten gewenst.
- Er is onduidelijk wat men precies wil.
- De eisen wijzigen steeds.

Maar dan moet men wel in het volgende kunnen voorzien:
- De continue beschikbaarheid van klant of gebruikers.
- Goed overweg kunnen met veranderingen.

Scrum

Een voorbeeld van de Agile-methode is Scrum.



Hoe Scrum werkt – het begin

1. Het begint met een idee.
2. De ideeën moeten via discussie worden vertaald naar een verhaal en dat wordt in stukjes opgedeeld. Elk stukje noemen we een sprint. Het Scrum-team is verantwoordelijk voor het resultaat en daarmee voor de planning, de werkverdeling en de voortgangsbewaking.
3. Elk van die stukjes is een stukje applicatie (sprint) dat wordt uitgewerkt. Er wordt een Scrum-master aangesteld, die zorgt dat de teamleden zich aan de Scrum-regels houden, maar... is vooral bezig met het ondersteunen van het team, ook als daarvoor van de regels moet worden afgeweken.
4. Er wordt een sprint-plan gemaakt waarin alle vereisten in user story's worden uitgewerkt en er wordt een prioriteit bepaald. Het opdelen van het werk in kleine, overzichtelijke taken is van belang om overzicht te houden en de voortgang goed in beeld te hebben. Het is de Product Owner die de beslissingen neemt over welke sprints het belangrijkst zijn.

Professionals in een Scrum

De belangrijkste succesfactor van Scrum is het vermogen van professionals om zelforganiserend te zijn. Daarvoor is het noodzakelijk dat men elkaar in de kracht zet: 'Laat mensen doen waar ze goed in zijn.'

Het Scrum-team

Het Scrum-team is zelf grotendeels verantwoordelijk voor het realiseren van projectdoelstellingen, ook als het even tegen zit. Bij grote obstakels springt de Scrummaster in.
De teamleden moeten zelf werk uitkiezen wat bij ze past en dit uitsplitsen tot taakniveau. Het Scrum-team is ook verantwoordelijk voor de planning, de werkverdeling en de voortgangsbewaking. De dagelijkse werkverdeling wordt door de teamleden bepaald op basis van prioriteiten en competenties

Op een Scrum-bord worden de sprints bijgehouden door het Scrum-team:

Project		**nr.**	**Teamnaam**	**Teamleden**	
Product backlog	**Product backlog**		**To do**	**Busy** bespreken	**Done**
Def. of done	**Def. of fun**	**Burn down chart**		**Stand-up**	**Retrospective**
				A. Wat heb je gisteren gedaan? B. Wat ga je vandaag doen? C. Zijn er ergens problemen ontstaan? D. Invullen Burn down chart E. Invullen proceslogboek	A. Wat ging er goed? B. Wat kan beter? C. Wat is het actiepunt in de volgende sprint?

De Scrum-master

Een Scrum-team wordt begeleid door een Scrum-master. Dat is vooral **niet** een hiërarchische projectleider. De Scrum-master zet het team in zijn kracht. Niet iedereen is overal even goed in, maar iedereen heeft wel iets waar hij goed in is. Het is aan de Scrum-master om zaken zo te verdelen dat de teamleden kunnen doen waar ze goed in zijn. Natuurlijk moet de Scrum-master ervoor zorgen dat alles wordt gedaan. De Scrum-master rapporteert niet over de voortgang, dat doet het ontwikkelteam in de review meeting. De Scrum-master faciliteert wel de voortgang door Scrum-meetings goed te laten verlopen en daar waar mogelijk obstakels weg te nemen. Door te faciliteren en coachen helpt de Scrum-master het team om succesvol te zijn.
De Scrum-master zorgt ervoor dat teamleden zelf werk uitkiezen en zorgt ervoor dat er een realistische *workload* is.

Product owner

De Product owner is verantwoordelijk voor iteratief projectmanagement en wordt hierbij ondersteund door de Scrum-master. Dit houdt in dat de scope incompleet kan zijn en voortdurend kan wijzigen.
De verantwoordelijkheid voor het budget, de scope en de tijdlijn ligt dan ook bij de Product owner.

De Product owner is altijd één persoon, bij voorkeur de klant of de opdrachtgever. De belangrijkste taak is de belangen van de klant goed te vertegenwoordigen. De Product owner moet in de juiste frequentie contact onderhouden met alle betrokkenen.

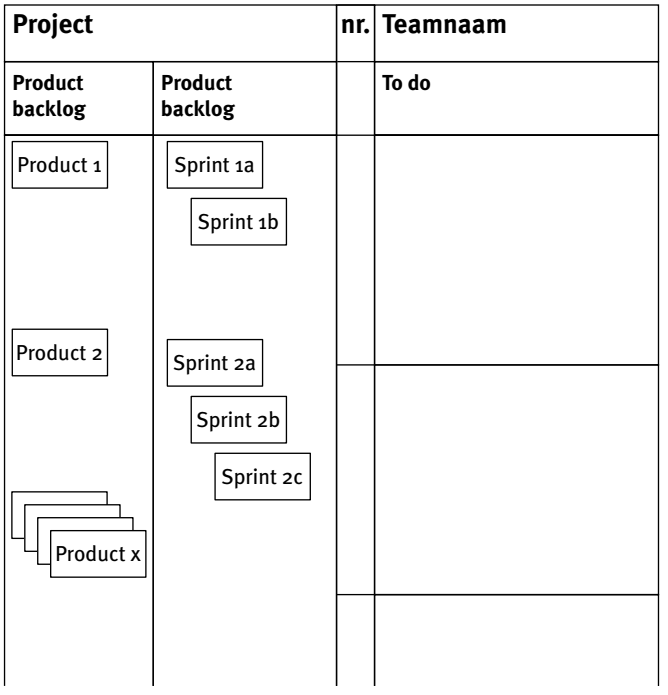
In de sprint-planning zal de Product owner belangrijke inzichten met betrekking tot het product delen met de rest van het Scrum-team. De samenwerking en communicatie met de Scrum-master en het Scrum-team is van groot belang. De Product owner bepaalt *wat* er moet worden gedaan, maar laat het ontwikkelteam volledig vrij in *hoe* dit wordt gerealiseerd. De Product owner kan het best onderdeel zijn van het Scrum-team (development team).

De Product owner bepaalt de volgorde waarin de items moeten worden opgepakt en prioriteert de product backlog. De belangrijkste wensen staan bovenaan, omdat deze de grootste waarde opleveren voor de uiteindelijke gebruikers van het eindproduct. Ook het duidelijk omschrijven van de onderdelen is een belangrijke taak. Een goede omschrijving van de items op de product backlog zorgt ervoor dat het projectteam ze begrijpt en kan oppakken. Hiervoor worden user story's geschreven. Een user story is een beschrijving van hoe een gebruiker een product gaat gebruiken. Voor het schrijven van user story's zijn regels, deze komen in paragraaf 5.1 aan de orde.

Het is de Product owner die de beslissingen neemt. Hierdoor is er meer focus en raakt men niet verloren in eindeloze discussies over wie wat belangrijk vindt en wat de vervolgkeuzes zijn. Hiervoor zou men een MoSCoW-analyse kunnen gebruiken. In paragraaf 4.5 komt het maken van een MoSCoW-analyse aan de orde.

Hoe Scrum werkt – het vervolg

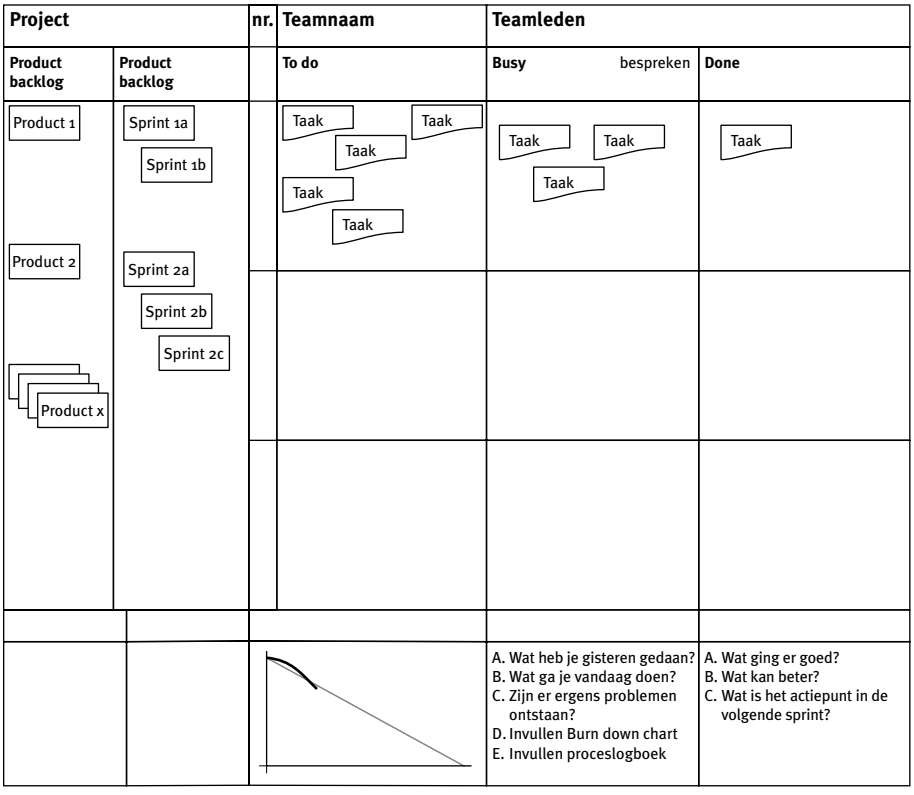
5 De belangrijkste user story's staan bovenaan en worden door een Scrum-team in een sprint backlog opgenomen (sprint backlog is een beschrijving van wat men gaat doen in de sprint). In de sprint planning zal de Product owner de belangrijke inzichten met betrekking tot het product delen met de rest van het Scrum-team. Op het Scrum-bord wordt nu door het team de product backlog in de juiste volgorde gezet.



6 User story's worden door het Scrum-team omgezet naar een kant-en-klaar product, inclusief test en documentatie.

7 Dagelijks (daily cycle) wordt er door de Scrum-master gestuurd op de voortgang aan de hand van de volgende drie vragen:
   a Wat heb je gisteren gedaan?
   b Wat ga je vandaag doen?
   c Zijn er ergens problemen ontstaan? (En hoe kunnen we je helpen?)

   Tijdens een sprint staan alle taken duidelijk zichtbaar op het Scrum-board en bepalen de teamleden zelf welke taken op een bepaald moment uitgevoerd worden en wie dit doet.
   Iedere sprint wordt afgesloten met een evaluatie en vastlegging van leerpunten (*retrospective*) zodat het team de volgende sprint nog beter kan uitvoeren.



Het opdelen van het werk in kleine work items, *to do's*, geeft de ontwikkelaars zelf de mogelijkheid om de voortgang te bewaken. Hiervoor wordt de Burn down chart ingevuld. Dit geeft elke Stand up meeting (zie verderop) voldoening over bereikte resultaten!

8 De nieuwe functionaliteit wordt getoond.
De review meeting aan het einde van elke sprint is erg belangrijk voor de Product owner, omdat daar wordt getoond wat er is gemaakt en waarop feedback moet worden gegeven. Tijdens de review meeting werkt het Scrum-team samen met alle betrokkenen. Er wordt feedback verzameld op het opgeleverde werk en men kijkt vooruit naar wat er komen gaat.

9 Er wordt bepaald welke sprint er gaat volgen.
Het Scrum-team is zelf grotendeels verantwoordelijk voor het realiseren van projectdoelstellingen, maar het is de Product owner die de beslissingen neemt.

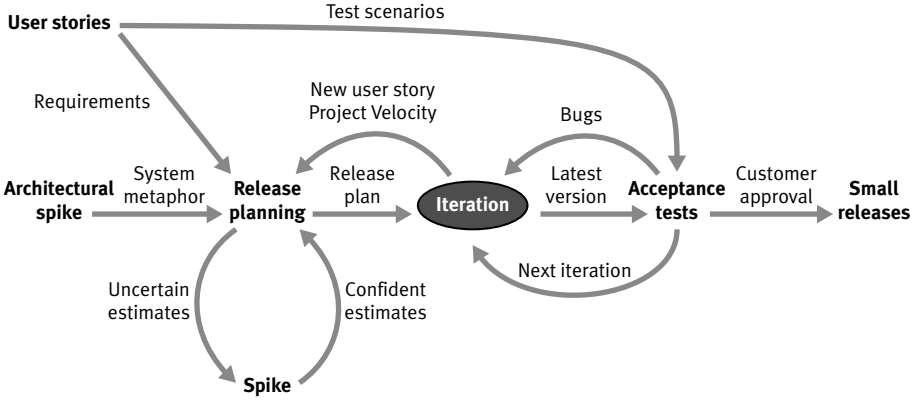
Er zijn ook digitale omgevingen waarin men een Scrum-bord bij kan houden, een voorbeeld daarvan is Trello. Toch kiezen veel Scrum-teams ervoor om een analoog Scrum-bord aan de muur te hangen. Iedereen kan in één oogopslag zien wie waarmee bezig is en wat de actuele stand van zaken is.

eXtreme Programming (XP)

eXtreme Programming is een 'humanistische discipline van software-ontwikkeling, gebaseerd op principes van eenvoud, communicatie, feedback en moed'. Het succes van eXtreme Programming hangt af van de feedback, een continu proces, gedeelde kennis en het welzijn van de ontwikkelaars.
eXtreme Programming is vooral geschikt voor projecten waarbij de exacte applicatie-eisen niet bij voorbaat vastliggen.

Bij eXtreme Programming wordt de software-ontwikkeling in tweetallen uitgevoerd – twee ontwikkelaars achter één systeem. Belangrijk is dat de teams regelmatig van samenstelling wisselen. Hierdoor ontstaat er een collectief 'eigendomsgevoel' en worden peer-review en code-inspectie als van nature onderdeel van het normale software-ontwikkelproces. De code van verschillende onderdelen zal beter en meer op elkaar zijn afgestemd en beter onderhoudbaar zijn.

Andere voordelen van deze manier van werken is dat er altijd minstens twee mensen ieder stuk code volledig doorgronden. Het inwerken van nieuwe mensen gaat als vanzelf omdat er een voortdurende *training-on-the-job* plaatsvindt.



Alles begint bij de planning die wekelijks plaatsvindt en bestaat uit de release planning en de iteratieplanning.

Release-planning

Om een release-plan te maken is er een release-planningsbijeenkomst waarin het totale project wordt bekeken. Het release-plan dient als basis voor de iteratieplannen. Het is belangrijk dat technische mensen de technische beslissingen en zakenmensen de zakelijke beslissingen nemen. De release-planning, die uit drie fasen bestaat, bepaald welke functionaliteit in welke release gerealiseerd wordt. De essentie van de release-planningsbijeenkomst is dat het ontwikkelteam voor elke user story inschat hoeveel programmeerweken nodig zijn. De klant bepaalt vervolgens welke user story het belangrijkste is en de hoogste prioriteit krijgt.

Exploration phase

In de onderzoeksfase maken de gebruikers user story's waarin de belangrijkste eisen voor het nieuwe systeem zijn verwerkt. Deze moeten kort en overzichtelijk zijn.

Commitment phase

In de conceptbesluitvorming wordt besloten in welke volgorde de user story's worden uitgewerkt.

Steering phase

In de wijzigingsfase kan het plan nog worden aangepast. Er kunnen nieuwe user story's worden toegevoegd en bestaande user story's worden verwijderd.

User story

User story's worden gebruikt om tijdschattingen te maken voor de release-planning. User story's worden geschreven door de klant en/of eindgebruiker als dingen die het systeem voor hen moet doen. Ze bestaan uit een paar zinnen tekst geschreven door de klant in de terminologie van de klant zonder technische informatie. De user story wordt ook gebruikt voor het maken van de acceptatietests. Er moeten één of meer geautomatiseerde acceptatietests worden gemaakt om te controleren of de user story, 'het gebruikersverhaal', correct is geïmplementeerd. De user story's mogen niet te veel in detail gaan, maar ze moeten wel voldoende details bevatten om een redelijke schatting te kunnen maken van de tijd die het gaat kosten om de user story te implementeren. Tijdens de implementatie krijgen de ontwikkelaars een gedetailleerde beschrijving van de vereisten van de klant.
De ontwikkelaars schatten per user story in hoe lang het zal duren om deze te implementeren. De ideale ontwikkeltijd is 1, 2 of 3 weken. Langer dan 3 weken wil zeggen dat de user story verder moet worden opgedeeld. Bij minder dan 1 week is men bezig op een te gedetailleerd niveau. In dat geval moeten user story's gecombineerd worden.

Architectual spike en Spike

Het idee hiervan is 'beperk je tot het probleem dat nu speelt'. Ontwerp deel-oplossingen om antwoorden te vinden op lastige technische of ontwerpproblemen.
Een Spike-oplossing is een heel eenvoudig programma om mogelijke oplossingen te verkennen. Kern hiervan is: bouw de oplossing om alleen het probleem aan te pakken dat wordt onderzocht en negeer alle andere zorgen. De meeste Spikes zijn niet goed genoeg om te behouden en zullen worden weggegooid. Het doel van Spike-oplossingen is het risico van een technisch probleem te verminderen of de betrouwbaarheid van een user story te vergroten.

Acceptance test (Acceptatietests)

Acceptatietests worden gemaakt op basis van de user story's. Tijdens een iteratie worden de user story's die tijdens de iteratieplanningsbijeenkomst zijn geselecteerd vertaald in acceptatietests. De klant levert de scenario's om te testen. Een user

story kan één of meerdere acceptatietests hebben om ervoor te zorgen dat de functionaliteit werkt.

Acceptatietests zijn black-box-systeemtests, dit komt in hoofdstuk 7 'Testen van software' nog aan de orde. Elke acceptatietest vertegenwoordigt een verwacht resultaat van het systeem. Klanten zijn verantwoordelijk voor het verifiëren van de juistheid van de acceptatietests en het beoordelen van testscores om te beslissen welke mislukte tests de hoogste prioriteit hebben. Een user story is niet voltooid totdat het de acceptatietests heeft doorstaan.

Kwaliteitswaarborging (quality assurance, QA) is een essentieel onderdeel van het proces. Het kan zijn dat de kwaliteit wordt gewaarborgd door een afzonderlijke groep. Zo moeten acceptatietests worden geautomatiseerd, zodat ze vaak kunnen worden uitgevoerd. De acceptatietestscore wordt gepubliceerd aan het team. Het is de verantwoordelijkheid van het team om ook tijd te plannen binnen elke iteratie om mislukte tests te verhelpen.
Vaak worden deze tests functionele tests genoemd. Dit geeft beter de intentie weer, namelijk garanderen dat aan de vereisten van een klant wordt voldaan en het systeem aanvaardbaar is.

Iteration

Een iteration is een herhaling. In dit geval gaat het om de herhaling onderzoeken, verdelen, ontwikkelen, onderzoeken, verdelen, ontwikkelen, onderzoeken enzovoort.
Iteratief ontwikkelen maakt het ontwikkelingsproces flexibel. Het is belangrijk de lengte van de iteratie constant te houden gedurende het hele project. Dit is de hartslag van het project. Hierdoor wordt het meten van voortgang en planning eenvoudig en betrouwbaar. De programmeerwerkzaamheden worden niet van tevoren gepland. In een iteratieplanningsbijeenkomst aan het begin van elke iteratie is het zaak duidelijk te krijgen wat er gaat gebeuren. Deze just-in-time-planning is een eenvoudige manier om op de hoogte te blijven van veranderende gebruikersvereisten.
De inspanningen moeten gericht zijn op het voltooien van de belangrijkste taken zoals die door de klant zijn aangegeven.

De iteration planning bestaat uit drie fasen. Door de ontwikkelaars worden uit de user story's, die in de sprint zijn opgenomen, de taken gehaald.

Exploration phase

In de onderzoeksfase worden de user story's naar taken vertaald en op taakkaarten geschreven.

Commitment phase

In de toewijzingsfase wordt van elk van deze taken ingeschat hoelang het duurt om deze te realiseren, om vervolgens aan de ontwikkelaars, bij voorkeur paren, toegewezen.

Steering phase

In de ontwikkelfase worden de taken uitgevoerd. Het resultaat wordt vergeleken met de originele tijdsplanning van de user story.

Stand up meeting

De Stand up meeting is een dagelijks gebeuren waarin alle betrokkenen overleggen over de uit te voeren zaken. De klant en/of gebruiker is onderdeel van het ontwikkelteam en is daarom continu voor vragen beschikbaar. eXtreme Programming gaat ervan uit dat de code klantspecifiek is en dat daarom de klant zo dicht mogelijk bij het ontwikkeltraject moet staan. Een toekomstige gebruiker moet aanwezig zijn om ervoor te zorgen dat de applicatie wordt wat de klant wenst. Daarom werkt eXtreme Programming met korte ontwikkelcycli waarin steeds een aantal geselecteerde testcases wordt geïmplementeerd tot een werkend systeem. Aan het eind van iedere cyclus kan de klant het systeem beoordelen en zo nodig de ontwikkeling bijsturen.

2.3 Prototyping

Er zijn verschillende methoden van prototyping en deze methodiek wordt ook vaak toegepast buiten de software-ontwikkeling. Denk maar aan een prototype voor een auto, een model dat nog helemaal niet in productie is. Waarschijnlijk ontbreken er essentiële onderdelen, maar van buiten ziet het er al heel echt uit.
In de software-ontwikkeling is het werken met prototyping niet anders. Er wordt een model van een applicatie gemaakt waarin bepaalde functionaliteit nog helemaal niet werkt of gesimuleerd wordt. Toch kan dit de klant/opdrachtgever al een duidelijk beeld geven van hoe een en ander eruit gaat zien. Hierop kan dan feedback worden gegeven en deze kan in het uiteindelijke product worden verwerkt.

Waarom prototyping?

Een prototype maak je om een goed beeld te kunnen krijgen van een product. In de meeste gevallen zal dat zijn:

Gebruiker

Voor de uiteindelijke gebruiker van het systeem is een duidelijk beeld van wat hij gaat krijgen erg belangrijk. Als je met de uiteindelijke gebruiker van het systeem in gesprek gaat over wat deze nodig heeft aan de hand van een prototype, is de kans op goede feedback het grootst. Je hoeft dan nog niet het hele product te bouwen, maar krijgt wel alle informatie die je nodig hebt.

Opdrachtgever en ontwikkelteam

Als er veel verschillende mensen betrokken zijn bij de ontwikkeling van een systeem zijn er ook veel interpretaties. Iedereen heeft zijn eigen werkelijkheid. Als je tegen een groep vrienden zegt: 'Ik heb een heel mooie boom in mijn tuin geplant,' aan welke boom denken ze dan?

Voor het ontwikkelteam is het van belang dat iedere betrokkene een zelfde beeld heeft. Het prototype kan een belangrijke bijdrage leveren aan de specificatie van een systeem.

Jezelf

Door te werken met prototyping kan ook jijzelf een goed beeld krijgen van de schermopbouw of van de beoogde navigatie door het systeem. Is alles logisch, vallen de juiste zaken op? Daarom moet je ook proberen in de natuurlijke context te ontwerpen. Is het bedoeld voor een smartphone, zorg dan ook dat je een prototype maakt voor de smartphone. Is het bedoeld voor een scherm, ontwerp dan ook op een scherm.

Wanneer prototyping?

Prototyping kun je al toepassen op het moment dat de klant zijn vraag nog niet duidelijk heeft. Je kunt het onder andere gebruiken om de vraag duidelijker te formuleren, helemaal als je het over ontwerpen hebt die nogal moeilijk zijn voor te stellen. Wel beperk je je tot de hoognodige details. Dit om te voorkomen dat het maken van het prototype meer werk is dan het uiteindelijke product.

Hoe prototyping?

Bij voorkeur gebruik je tools waarmee je al overweg kunt. Je kunt gebruikmaken van:

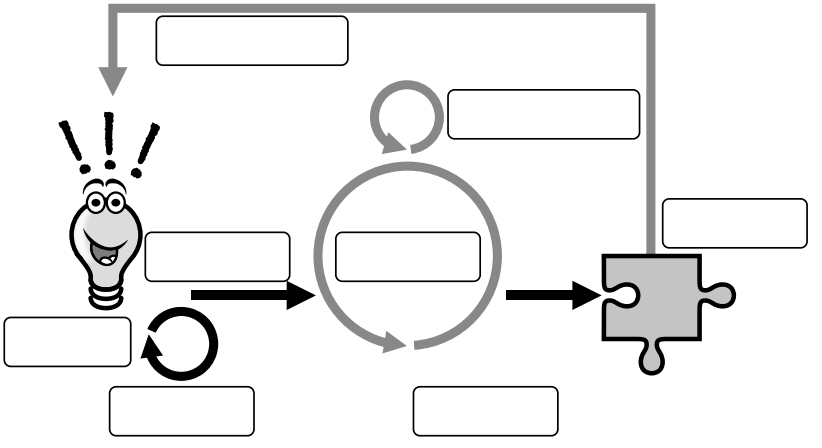
- moodboard, om met de opdrachtgever kleur en uiterlijk te bepalen;
- animaties, om te laten zien hoe een en ander op elkaar volgt;
- interactie-flows, als er sprake is van matrixnavigatie waar gebruikers snel van pagina kunnen wisselen.

De tools die je kiest moeten je vooral niet beperken bij het komen tot mooie oplossingen.

2.4 Opgaven

1. Wat is het belangrijkste argument om SDM toe te passen?
2. Uit welke fasen bestaat SDM?
3. Wat zijn mijlpaalproducten bij SDM?
4. Wat zijn de mijlpaalproducten van de fase Informatieplanning bij SDM?

5 Wat is het belang van de fase Definitiestudie bij SDM?
6 Wat zijn de mijlpaalproducten van de fase Definitiestudie bij SDM?
7 Wat gebeurt er in de fase Basisontwerp bij SDM?
8 Wat zijn de mijlpaalproducten van de fase Basisontwerp bij SDM?
9 Wat gebeurt er in de fase Detailontwerp bij SDM?
10 Wat zijn de mijlpaalproducten van de fase Detailontwerp bij SDM?
11 Wat gebeurt er in de fase Realisatie bij SDM?
12 Wat zijn de mijlpaalproducten van de fase Realisatie bij SDM?
13 Wat gebeurt er in de fase Invoering bij SDM?
14 Wat zijn de mijlpaalproducten van de fase Invoering bij SDM?
15 Waarom is fase 6, Gebruik en beheer, feitelijk geen fase bij SDM?
16 Wat zijn de nadelen van de Watervalmethode?
17 Met welke Nederlandse woorden zou je Agile kunnen vertalen?
18 Waar staat Agile voor?
19 Wat zijn de belangrijkste redenen om Agile toe te passen?
20 Waarin moet worden voorzien om Agile succesvol toe te passen?
21 Zet de begrippen in de onderstaande figuur op de juiste plaats:
- Dagelijkse cyclus
- Deelapplicatie
- Idee
- Maken sprintplan
- Sprintcyclus
- Sprintplan
- Terugkoppeling



22 Zet de Engelse benaming achter de volgende Scrum-begrippen op de juiste plaats in de figuur op de vorige bladzijde:
- Dagelijkse cyclus
- Deelapplicatie
- Idee
- Maken sprintplan
- Sprintcyclus
- Sprintplan
- Terugkoppeling

23 Dagelijks wordt er door de Scrum-master gestuurd op de voortgang. Aan de hand van welke vragen?

24 Wat is de belangrijke succesfactor van Scrum?

25 Wat is de Scrum-master vooral **niet**?

26 Wat of wie is de Product owner?

27 Op welke principes is eXtreme Programming gebaseerd?

28 Wanneer is eXtreme Programming een geschikte ontwikkelmethodiek?

29 Wat is Architectual spike en Spike?

30 Uit welke drie fasen bestaat de Iteration planning?

31 Wat is een stand up meeting?

32 Waarom zou je prototyping gebruiken?

33 Wat voor baat heeft een gebruiker/klant bij prototyping?

34 Voor wie, anders dan de gebruiker, kan prototyping van belang zijn?

35 Wanneer kun je prototyping toepassen?

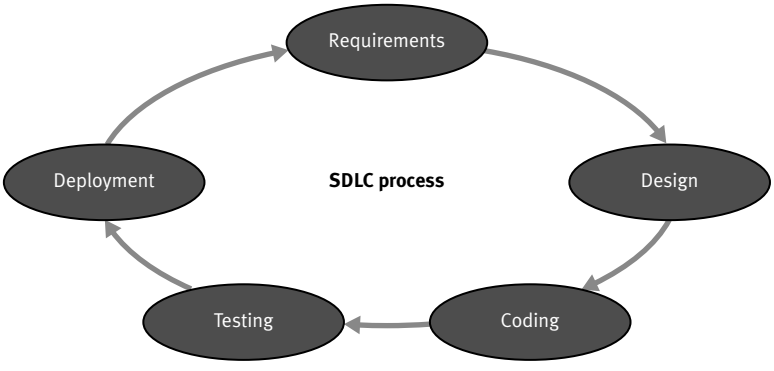
36 Welke tools moet je gebruiken bij prototyping?

3 Veilig en betrouwbaar software ontwikkelen

Wanneer je onderdeel van een team bent dat software gaat ontwikkelen dan zullen er afspraken zijn over de te volgen methodiek. Secure Software Development Life Cycle (vanaf nu S-SDLC) heeft steeds meer aan populariteit gewonnen. In veel organisaties is het een standaard die gebruikt moet worden. S-SDLD is het logisch vervolg op SDLC (Software Development Life Cycle). In dit hoofdstuk worden de belangrijkste onderdelen van SDLC en S-SDLC behandeld. In de hoofdstukken die hierop volgen komen dan verschillende methoden en technieken aan de orde die je nodig hebt om een goede applicatie te ontwikkelen.

3.1 Software Development Life Cycle (SDLC)

SDLC is opgebouwd uit vijf fasen die elkaar continu opvolgen.



Wanneer je een applicatie gaat ontwikkelen dan ligt het voor de hand dat je SDLC toepast. Het is een gestructureerde manier om softwaretoepassingen te bouwen die door veel organisaties wordt gebruikt voor het ontwikkelen van software. Soms heeft men het proces wat aangepast, maar de basis blijft gelijk. De levenscyclus van software-ontwikkeling, de vijf fasen, wordt hier verder uitgewerkt.

Requirements

Een Pakket van Eisen (Software Requirement Specification of SRS) is een document waarin staat wat men verwacht van het systeem of de software die moet worden ontwikkeld. De MoSCoW-methode kan hier als hulpmiddel dienen. Het ontwikkelingsteam verzamelt input van verschillende stakeholders (belanghebben-

den), zoals klanten, sales, interne en externe experts en ontwikkelaars, om het Pakket van Eisen op te stellen. Het team stelt vast wat er nodig is om aan die vereisten te voldoen en berekent de te verwachten kosten. Belangrijk is dat het team niet alleen beschrijft wat ze in de software willen, maar ook wat ze niet willen. Wanneer het gaat over een upgrade of vervanging van bestaande software is dit zo mogelijk nog belangrijker dan wanneer het om nieuwe software gaat.

Design

Het software-ontwerp is de blauwdruk die aan ontwikkelaars wordt gegeven voor het maken van de code. Om een software-ontwerp op hoog niveau te kunnen maken moeten de ontwerpers rekening houden met de eerder opgestelde vereisten (PvE). Het te bouwen systeem en de software zullen mede afhangen van de te gebruiken technologieën, de mogelijkheden van het team en projectbeperkingen als tijd en geld. Het ontworpen softwaresysteem wordt dan vertaald in softwaremodules, functies, bibliotheken enzovoort. Het ontwerp wordt meestal beschreven in een ontwerpspecificatiedocument.

Coding

De coderingsfase wordt ook wel de implementatiefase genoemd. Tijdens deze fase wordt de blauwdruk van de software omgezet in code. De broncode van de hele applicatie ontwikkelen neemt meestal de meeste tijd in beslag. De tijd die nodig is om de ontwikkeling te voltooien, is afhankelijk van de grootte van de applicatie en het aantal programmeurs.

Testing

Nadat de applicatie is gebouwd moet deze worden getest om te beoordelen of deze aan het Pakket van Eisen voldoet. De testers voeren de afgesproken testen uit. Als ze een fout ontdekken dan informeren ze de ontwikkelaars. De ontwikkelaar kijkt of er echt sprake is van een defect. Als er echt sprake is van een defect moet dat opgelost worden. Er komt dan een nieuwe versie van de software, die opnieuw getest moet worden. Dit wordt voor elk van de onderdelen gedaan en net zolang herhaald totdat het ontwikkelingsteam alle ontdekte defecten heeft verholpen en de software gereed is voor implementatie in de productieomgeving.

Deployment

Deployment wordt ook wel Release-fase of Implementatiefase genoemd. Het komt er in deze fase op neer dat de applicatie klaar is om live te gaan.
Zodra de applicatie klaar is om live te gaan, wordt deze in deze fase op een productieserver geïmplementeerd. Als deze is ontwikkeld voor een client, vindt de implementatie plaats op een client-locatie of datacenter waar de client de applicatie wil installeren. Het vrijgeven van de voltooide software in de productieomgeving maakt het uitvoeren van post-productie-activiteiten zoals monitoring mogelijk. In

de productieomgeving worden nog extra testen uitgevoerd en de definitieve implementatie vindt plaats nadat alle bugs zijn ontdekt en opgelost.

3.2 Informatiebeveiliging

De realiteit van vandaag de dag is dat er mensen (of organisaties) zijn die als enig doel hebben in te breken in systemen. Vaak zijn dit hackers die op zoek zijn naar roem door erover op te scheppen op internet. Maar het kan ook een groep georganiseerde criminelen zijn die in alle stilte informatie stelen. Dit kan een enorm verlies voor de organisatie in kwestie betekenen.
In mei 2018 verscheen het volgende artikel in het *Algemeen Dagblad*:

Verander nu je wachtwoord: 3,3 miljoen Nederlandse wachtwoorden eenvoudig te vinden

VIDEO'S | **Een grote hoeveelheid e-mailadressen en 3,3 miljoen wachtwoorden van Nederlanders liggen op straat. Ze blijken sinds eind vorig jaar betrekkelijk eenvoudig in te zien in lijsten die online circuleren. Een hacker maakt ze vandaag via een speciale zoekmachine zelfs nog toegankelijker: een 'Google' voor wachtwoorden.**

Thomas Boeschoten en Cyril Rosman 30-03-18, 06:00 **Laatste update:** 01-02-19, 16:52

f 648 53

Uit onderzoek van deze krant blijkt dat medewerkers van veel grote Nederlandse (overheids)organisaties en bedrijven massaal in de lijsten voorkomen, waaronder organisaties die vitale functies vervullen. Ook staan er parlementariërs en bekende Nederlanders in de lijst. ,,Schrikken, want deze wachtwoorden gebruikte ik voor andere accounts nog steeds", zegt ex-Tweede Kamerlid Sharon Gesthuizen.

De gegevens zijn waarschijnlijk afkomstig uit tientallen grote datalekken van de afgelopen jaren. Onder meer van LinkedIn, Dropbox, Playstation, Uber (bij Uber ging het niet om wachtwoorden) en eBay is bekend dat gegevens zijn gelekt, en daarover is al veel gepubliceerd. Maar waar die enorme lijsten eerst vooral in de krochten van het internet en tegen (forse) betaling beschikbaar waren, worden ze nu steeds makkelijker en openbaar beschikbaar.

De criminelen of hackers kunnen op verschillende manieren inbreken. Naast gelekte wachtwoorden is een bekende route via de applicatiehost. Als de applicaties die op een applicatieserver staan kwetsbaar zijn, kan dit ernstige gevolgen hebben. Er zijn bedrijven hierdoor in problemen gekomen. De slechte berichten in de pers kunnen een aandelencrash tot gevolg hebben. Alle, maar vooral financiële organisaties zoals banken, lopen gevaar. Bedrijven en particulieren kunnen te maken