

DIENSTVERLENING & WERKWIJZE

Een duidelijke beschrijving van onze werkwijze, tarieven en voorwaarden voor onze partners en opdrachtgevers.



Introductie en toelichting

Het ontwerpen, ontwikkelen en verbeteren van softwareproducten is op zichzelf een complexe opgave. Dit is de uitdaging die iedereen bij Label305 sinds 2010 elke dag aangaat. Het aanbod van onze dienstverlening en het inrichten van een goede werkwijze zijn beide erg belangrijk. Al vanaf 2012 werken we intern met Agile-methoden, en in 2014 hebben we deze methoden doorgetrokken naar ons dienstenaanbod. Onze opdrachtgevers werden vanaf dat moment direct bij onze Agile-processen betrokken. Hierdoor versterkten en verbeterden onze samenwerkingen, en kwamen we uiteindelijk tot nog betere eindproducten.

Nu, meer dan vier jaar later, merken we dat het nodig is om een volgende stap te maken. We horen dat er, zowel intern als vanuit onze opdrachtgevers, behoefte is aan een uniformer dienstenaanbod en een doortastender productontwikkelingsproces. Daarom introduceren wij met dit document evoluties op onze werkwijze en dienstverlening. We hebben als doel onze werkwijze glashelder toe te lichten, onze dienstverlening verder te verbeteren en zo opnieuw tot betere eindproducten te komen.

We blijven flexibel, benaderbaar en introduceren geen onnodige bureaucratie. De nieuwe werkwijze schept vooral meer duidelijkheid, en geeft oplossingen voor eventuele vervelende situaties.

De voornaamste wijzigingen en vernieuwingen zijn:

- Het verbeteren van onze servicedienstverlening door het aanbieden van twee typen serviceovereenkomsten: een SLA en een BESA;
- Het aanbieden van een expliciete garantieregeling voor sprintopleveringen;
- Het introduceren van prijsonderscheid tussen losse werkzaamheden en sprintwerkzaamheden, waarbij het tarief voor losse werkzaamheden stijgt;
- Het automatisch overdragen van het intellectueel eigendom voor nieuwe ontwikkeling; en
- Het aanbieden van een concreet ontwikkelingstransitieproces.

De werkwijze hier beschreven gaan we vanaf **1 april 2019** hanteren voor *al onze opdrachtgevers en samenwerkingen*. Voor nieuwe opdrachtgevers wordt deze werkwijze reeds aangehouden.

Dit document heeft als doel om in klare taal te beschrijven hoe we (gaan) werken en wat de voorwaarden zijn. Zo zijn er geen onduidelijkheden – niet voor ons en niet voor jou. Helaas betekent dit wel dat dit document een aardige omvang heeft. Toch willen we je vragen om het goed door te nemen. Het is geen saaie juridische tekst maar een duidelijke uitleg van allerlei aspecten van onze dienstverlening. We zoeken vaak de nuance op, en dat is nodig want softwareontwikkeling is een complexe uitdaging voor alle betrokken partijen.

Inhoudsopgave

1	Tarieven	1
1.1	Het hoge uurtarief: los werk, spoed en overig	1
1.2	Het lage uurtarief: sprint- en servicewerkzaamheden	1
1.3	Serviceovereenkomst	2
1.4	Ingekochte hosting en tooling	2
1.5	Overige aankopen en declaraties	3
1.6	Overheidsstimuleringen	3
1.7	Reizen en overnachten	4
1.8	Tariefswijzigingen	4
2	Uitgangspunten	5
2.1	De gebruiker voorop	5
2.2	Altijd inzicht in het plan	5
2.3	De rol van producteigenaar	6
2.3.1	Onze unieke kijk op producteigenaarschap	6
2.3.2	Een belangrijke verantwoordelijkheid	6
2.3.3	Overdracht producteigenaarschap	7
2.4	Voorkomen van vendor lock-in	7
2.5	Gezondheid van de codebase	8
2.5.1	Technical debt	8
2.5.2	Complexiteit en formaat van de codebase	9
2.5.3	Code optimaliseren	9
2.6	Belang en noodzaak van kennisuitbreiding	10
2.6.1	Intern maken we extra ruimte om verder te leren en te verdiepen	10
2.7	In rekening brengen van onze inspanningen	11
2.7.1	Acquisitie	11
2.7.2	Bespreken, vergaderen en kick-offmeetings	11
2.7.3	Consultancy-, ontwerp- en ontwikkelwerkzaamheden	12
2.7.4	Projectmanagement	12
2.7.5	Onderhoud	12
2.7.6	Ondersteuning	13
2.7.7	Senioriteit (juniores, werkstudenten en stagiairs)	13

2.8	Bewustzijn van verborgen kosten _____	14
2.8.1	Interne kosten gedurende het ontwikkelingsproces _____	14
2.8.2	Ondersteuning voor de eindgebruiker _____	14
2.8.3	Ontwikkelingsstagnatie _____	14
2.9	Het productrisico _____	15
3	Een agile ontwikkelproces: werken in sprints _____	17
3.1	Verwachtingen bij een agile softwareontwikkelingsproces _____	17
3.2	Sprints _____	18
3.2.1	Vaststellen sprintomvang _____	18
3.2.2	Vooraf sprintsamenstellingen via de backlog vaststellen _____	19
3.2.3	Formaat van een item in de backlog _____	19
3.2.4	Backlog onduidelijk voor aanvang sprint _____	20
3.3	Communicatie, voortgang en bijsturen tijdens de sprint _____	20
3.3.1	Keuze voor projectmanagementsoftware _____	20
3.3.2	Het wijzigen van de (sprint) backlog _____	21
3.3.3	Spoodwerkzaamheden tijdens de sprint _____	21
3.3.4	Gebruik van chatsoftware _____	21
3.3.5	Telefonisch overleg _____	22
3.3.6	Beperkte toegang tot versiebeheer- en CI-systemen _____	22
3.4	Opleveren aan het einde van de sprint _____	23
3.4.1	Verbeterpunten _____	23
3.4.2	Het verwerken van feedback _____	23
3.4.3	Publiceren _____	24
3.5	Sprintgarantie _____	24
3.5.1	Geldige redenen voor aanspraak op de sprintgarantie _____	24
3.5.2	Invulling van de compensatie _____	26
3.5.3	Verlopen mogelijkheid voor aanspraak op sprintgarantie _____	26
3.6	Coulance na de laatst ingeplande sprintoplevering _____	27
4	Losse werkzaamheden _____	28
4.1	Inschatten _____	28
4.1.1	Inschatting overschrijden _____	28
4.1.2	Inschatting aanpassen bij opleveren voor feedback _____	29

4.2	Opleveren _____	29
4.2.1	Feedback _____	29
4.2.2	Publiceren _____	30
4.3	Spoed _____	30
4.3.1	Wat wordt beschouwd als spoed? _____	31
4.4	Losse werkzaamheden en service _____	31
4.5	Coulance bij klein los werk _____	32
5	Doorlopende planning en dagindeling _____	33
5.1	Inschatten van nieuw productontwikkelingswerk _____	33
5.2	Milestones _____	33
5.2.1	Bijsturen _____	34
5.3	Sprints reserveren en inplannen _____	34
5.4	Akkoord gaan _____	35
5.5	Overwegingen en valkuilen _____	36
5.5.1	Sprintplanning voorbij akkoord _____	36
5.5.2	Volgepland? _____	36
5.5.3	Ingeplande losse werkzaamheden _____	36
5.6	Dagindeling en productiviteit _____	37
5.6.1	Flexibele kantoortijden _____	37
6	Quality assurance _____	38
6.1	Fundering van een kwalitatieve gebruikerservaring _____	38
6.1.1	Privacy en PbD _____	38
6.1.2	Veiligheid _____	39
6.1.3	Datamobilitieit _____	39
6.2	Kwaliteitsprincipes _____	40
6.2.1	Uitnodigend en moeiteloos _____	40
6.2.2	Vloeiend en vlot _____	41
6.2.3	Strak, gebalanceerd en duidelijk _____	41
6.2.4	Geen onnodige versieringen en functies _____	42
6.2.5	Geen onnodige technische complexiteit _____	42
6.2.6	Geen bugs en geen onverwacht gedrag _____	42
6.3	Methoden _____	42

6.3.1	Starten met weloverwogen ontwerpen	43
6.3.2	Automatische tests	43
6.3.3	Handmatig testen tijdens het ontwikkelen	44
6.3.4	Pair programming	44
6.3.5	Code review	45
6.3.6	Gebruiksreview	45
6.3.7	User testing	46
6.3.8	Monitoren en direct ingrijpen (bij een SLA)	47
6.4	Beperkingen	47
7	Privacy en veiligheid	49
7.1	Responsible disclosure	49
7.1.1	De beloning bij een correcte melding	49
7.1.2	Publiceren van het beleid	50
7.2	Security audits	50
7.3	Privacywetgeving	50
7.3.1	Verantwoordelijkheden van de controller	51
7.3.2	Verwerkers	52
7.3.3	Gegevensbeschermingseffectbeoordeling	53
7.3.4	Dataopvraag-, verwijder- en vergeetverzoeken	55
7.3.5	Cookies en tracking	55
7.3.6	Gegevens bij derde partijen	55
8	Service, onderhoud en ondersteuning	58
8.1	Definitie van servicewerk	60
8.1.1	Het beheren van applicatie-infrastructuur	61
8.1.2	Oplossen van acute gebruiksblokkerende bugs en problemen	63
8.1.3	Ondersteuning	63
8.1.4	Monitoren (SLA)	65
8.1.5	Administratieve werkzaamheden	66
8.1.6	Afhandelen van meldingen en verzoeken	66
8.2	Geen serviceovereenkomst	67
8.3	Best-effort Service Agreement (BESA)	67
8.3.1	Service-uren inkopen en opbouwen	68

8.3.2	Serviceverlening tijdens de sprint _____	68
8.4	Service Level Agreement (SLA) _____	68
8.4.1	Proactief servicewerk oppakken en het monitoren van de applicatie ___	69
8.4.2	Servicetegoed overschrijding voordeliger _____	70
8.4.3	SLA en doorlopende sprintwerkzaamheden _____	70
8.4.4	Applicatie-uptimegarantie _____	71
8.4.5	Gegevensintegriteitsgarantie _____	71
8.4.6	Garantie voor het tijdig oppakken van meldingen en problemen _____	72
8.4.7	Compensatie bij verzuim _____	72
9	Hosting en tooling _____	74
9.1	Maandelijks voorschot en kwartaalafrekening _____	74
9.2	Nieuwe diensten introduceren _____	75
9.3	Zelf diensten direct afnemen bij een derde partij _____	75
9.4	Hosting- en toolingkosten zonder serviceovereenkomst _____	76
9.5	Toegang tot beheeromgevingen en monitoringsystemen _____	76
10	Facturatie en betaling _____	77
10.1	Facturatie van sprints en losse werkzaamheden _____	77
10.1.1	Aanbetaling voor aanvang _____	77
10.1.2	Sprintgarantie, facturen en betalingen _____	78
10.2	Facturatie van service, hosting en tooling _____	78
10.2.1	Automatische incasso _____	78
10.3	Betalingsverzuim _____	79
10.3.1	Betalingsverzuim servicefacturen _____	80
11	Intellectueel eigendom _____	81
11.1	Waar het om gaat _____	81
11.2	Overdracht _____	81
11.2.1	Overdracht vindt plaats na betaling _____	82
11.2.2	Licentie terug naar ons _____	82
11.2.3	Geen automatische overdracht bij partnerschappen _____	82
11.3	Toelichting voor code _____	83
11.3.1	Praktische implicaties _____	84
11.3.2	Toegang tot de code _____	85

11.4	Toelichting voor audio, video en grafische elementen _____	85
12	Ontwikkelingstransitie en participatie _____	86
12.1	Voorwaarden ontwikkelingstransitie _____	86
12.2	Het proces voor een deelnemer _____	87
12.2.1	Selecteren en doorlichten _____	87
12.2.2	Kennis maken in de eerste weken _____	89
12.2.3	Opleiden en inwerken bij Label305 _____	89
12.2.4	Remote samenwerken met de deelnemer _____	90
12.3	Opbouwen van een volledig intern team _____	90
12.3.1	Overdracht van de procesverantwoordelijkheid _____	91
12.3.2	Advies en begeleiding na de overdracht _____	92
12.4	Vergoeding of participatie _____	92
12.4.1	Vergoeding voor ontwikkelingstransitie _____	92
12.4.2	Participatie _____	93
12.4.3	Samenwerkingsovereenkomst _____	94
13	Sunsetting _____	96
13.1	Sunsetprocedure _____	96
13.2	Sunsetting bij betalingsverzuim of faillissement _____	97
13.3	Opnieuw starten na sunseting _____	97

1 Tarieven

We beginnen met de basis: onze tarieven. Met deze tarieven kan een projectbegroting gemaakt worden, en kan je ons vergelijken met eventuele concurrenten. Wij concentreren ons op goede dienstverlening. Daarom is ons uurtarief de plek waar we onze marge maken.

De compensatie voor onze dienstverlening wordt berekend op basis van uren, en is altijd op basis van een **inspanningsverplichting**. Dienstverlening ingekocht in de vorm van sprints of een SLA geven bovenop de inspanningsverplichting extra garanties, later meer hierover.

Wij hanteren twee uurtarieven: hoog en laag. Het onderscheid wordt daarin niet gemaakt op basis van senioriteit of disciplines. Het verschil tussen de tarieven heeft daarentegen te maken met de wijze waarop werkzaamheden ingepland worden.

1.1 Het hoge uurtarief: los werk, spoed en overig

Het standaard uurtarief voor onze dienstverlening is als volgt:

Het hoge uurtarief: € [REDACTED]- per uur, exclusief btw

In principe worden werkzaamheden tegen dit hoge uurtarief in rekening gebracht, met uitzondering van sprints en specifieke servicewerkzaamheden.

Voorbeeld: We moeten een kleine functie toevoegen aan een systeem dat een jaar terug naar productie is gegaan, waarna geen (door)ontwikkeling plaats heeft gevonden. We zijn hier 8 uur mee bezig. Hiervoor rekenen we het hoge uurtarief.

1.2 Het lage uurtarief: sprint- en servicewerkzaamheden

Het merendeel van de uren die we werken wordt verrekend tegen het lage uurtarief:

Het lage uurtarief: € [REDACTED]- per uur, exclusief btw

Het ontwerpen en ontwikkelen van goede producten vergt planning en focus. Daarom plannen we het merendeel van onze werkzaamheden van tevoren in de vorm van sprints. De sprintprijs wordt berekend aan de hand van het lage uurtarief en de *minimale inspanningsverplichting*.

Voorbeeld: Gedurende één jaar zijn drie experts elke maand twee weken bezig met ontwerpen en doorontwikkelen van een softwareproduct. Dit wordt gedaan in sprints met een opleverfrequentie van één week. Hiermee gaat met elke sprint

een minimale inspanning gemoeid van 90 uur. Deze uren worden vervolgens doorgerekend tegen het lage uurtarief.

Naast het uitvoeren van ontwikkelwerkzaamheden verlenen we ook service. Wij willen hierop kunnen anticiperen. Vooraf en terugkerend ingekochte service-uren worden daarom berekend aan de hand van het lage uurtarief. Bovendien worden alle service-uren verrekend tegen het lage uurtarief indien een SLA is overeengekomen, dus ook extra service-uren.

1.3 Serviceovereenkomst

De producten die we ontwikkelen voor onze opdrachtgevers zijn vaak cruciaal voor bedrijfsprocessen, of het bedrijf zelve. Daarom is het monitoren en onderhouden van productiesystemen buitengewoon belangrijk. Bovendien is het fijn om garanties te hebben over het oppakken van acute problemen, het uitvoeren van regulier onderhoud, het beantwoorden van vragen en het proactief documenteren van verbeterpunten.

Het voldoen aan deze garanties vergt een continue paraatheid van onze experts. Deze continue beschikbaarheid en paraatheid leggen we vast in een SLA (*Service Level Agreement*) waarvoor het volgende tarief geldt:

SLA: vanaf € ■■■■.-* per maand, exclusief btw +
maandelijks ingekochte service-uren × lage uurtarief

** Deze kosten zijn gebaseerd op de continue paraatheid van minimaal een van onze specialisten voor een product met een normale gebruikersbelasting. Mochten we in de toekomst zien dat, om te voldoen aan de SLA, de continue paraatheid wordt geleverd van meer specialisten, dan kunnen we het tarief voor de SLA aanpassen.*

Service kan ook vooraf en terugkerend ingekocht worden, zonder dat we garanties over het serviceniveau afgeven. Dit kan middels een BESA (*Best Effort Service Agreement*). Hiervoor worden enkel service-uren in vooraf ingekocht.

BESA: maandelijks ingekochte service-uren × lage uurtarief

In het hoofdstuk 'Service, onderhoud en ondersteuning' gaan we verder in op de verschillende serviceovereenkomsten en bijbehorende werkwijzen.

1.4 Ingekochte hosting en tooling

Voor zowel het ontwikkelen van een softwareproduct, als het beschikbaar maken van dit product aan een eindgebruiker, worden verschillende hosting- en softwarediensten van derde partijen gebruikt. Door het gebruik hiervan kunnen we effectiever

ontwikkelen en service verlenen. De opdrachtgever bespaart op de initiële kosten en we kunnen het product goed klaarmaken voor groei.

Software- en hostingdiensten hebben vaak variërende periodieke kostenstructuren die afhankelijk zijn van het precieze gebruik per minuut, dag of maand. Bovendien worden sommige diensten enkel aangeboden in buitenlandse valuta.

We willen het gemakkelijk maken om profijt te hebben van deze dienstverlening, zonder de bijkomende administratieve rompslomp. Wij kunnen deze dienstverlening voor onze opdrachtgevers inkopen, zonder eindeloze communicatie. We maken daarbij vooraf een *inschatting* van de totale maandelijkse kosten en brengen deze in rekening bij de maandelijkse servicefacturatie, *zonder toegevoegde marge*. Vervolgens kijken we elk kwartaal of deze inschatting bijgesteld moet worden, en komen we aan het einde van een kwartaal met een **kwartaalafrekening**, waarbij wellicht iets extra's moet worden betaald of een deel wordt terugbetaald. Bij deze kwartaalafrekening kan een uitgebreide specificatie van alle afgenomen diensten en bijhorende kosten opgevraagd worden.

De inkoop van specifieke software, nodig voor het ontwikkelen en onderhouden van het product, wordt via deze constructie doorberekend. Diensten die worden gebruikt voor het inrichten en opereren van de applicatie-infrastructuur kunnen meestal direct afgenomen worden bij de derde partij, zoals een cloudprovider. Dan zitten wij hier niet tussen. Dit heeft meestal onze voorkeur.

1.5 Overige aankopen en declaraties

Af en toe is het noodzakelijk andere diensten of producten aan te schaffen voor het ontwikkelen van softwareproducten. Het heeft altijd onze voorkeur dat deze door onze opdrachtgever worden ingekocht. Het is ook geen enkel probleem als we dit moeten doen. In dat geval rekenen we de kosten door.

Voorbeeld: Een opdrachtgever wil dat we een applicatie ontwikkelen voor Google Glass, wij hebben zelf echter geen ontwikkelapparaat. We schaffen dan een Google Glass aan en rekenen de kosten hiervan door aan de opdrachtgever. We hebben het apparaat in gebruik gedurende de hele periode waarin we het product ontwikkelen en moeten blijven ondersteunen. Het apparaat is wel het eigendom van de opdrachtgever.

1.6 Overheidsstimuleringen

De overheid stimuleert R&D op verschillende manieren, zo ook via subsidies. Voor sommige werkzaamheden kunnen wij subsidie aanvragen, o.a. via de WBSO. Productontwikkeling is immers een belangrijk component van R&D. Het gebruik van de WBSO is reeds verwerkt in de vermelde uurtarieven.

Vanuit zowel de regionale, landelijke als Europese overheden zijn er ook mogelijkheden voor samenwerkingssubsidies. We staan ervoor open om samen met onze opdrachtgevers hier aanspraak op te maken.

1.7 Reizen en overnachten

Kosten voor reizen binnen Nederland worden niet doorgerekend, mits de reizen niet regelmatig en structureel moeten plaatsvinden. Kosten voor reizen naar het buitenland worden wel in rekening gebracht. Daarnaast worden ook alle overnachtingskosten in rekening gebracht. Reistijd wordt niet doorgerekend als gewerkte tijd.

Bij sommige bureaus zijn reiskosten meegenomen in het uurtarief. Dit is bij ons niet zo omdat veel reizen voor een project van ons zeker niet altijd vanzelfsprekend is.

Voorbeeld 1: In het kader van sprintwerkzaamheden moeten twee specialisten van Label305 voor een opdrachtgever elke twee weken één dag on-site zijn in Eindhoven gedurende een periode van zes maanden. Hiervoor worden reiskosten doorberekend. Gewerkte tijd on-site wordt vervolgens in rekening gebracht via de dan lopende sprint, en telt dus mee voor de minimale inspanningsverplichting.

Voorbeeld 2: Voor een andere opdrachtgever zullen drie specialisten van Label305 twee dagen aanwezig zijn op een beurs in Canada, dit gebeurt buiten een sprint om. Hiervoor worden reis- en overnachtingskosten in rekening gebracht. Daarbovenop worden 48 uren in rekening gebracht tegen het hoge uurtarief, voor drie mensen die twee volle dagen aan het werk zijn op de beurs.

1.8 Tariefswijzigingen

Wij voeren af en toe een tariefswijziging door. Dit doen wij bijvoorbeeld vanwege toegenomen schaarste of waarde van ons type dienstverlening. Ook corrigeren wij na verloop van tijd voor opgelopen inflatie.

Het verhogen doen wij maximaal éénmaal per kalenderjaar. Als een verhoging plaatsvindt, wordt dit gedaan in stappen van € 5, € 10, € 15 of € 20 op de verschillende uurtarieven. Wij verhogen ons hoge en lage uurtarief niet met meer dan € 20 per keer. Een verhoging wordt minimaal twee maanden van tevoren aangekondigd. Meestal voeren wij een eventuele verhoging door bij de overgang naar het nieuwe jaar, maar hier kunnen wij van afwijken.

Bij nieuwe opdrachtgevers laten we de eerste twaalf maanden na aanvang van de eerste werkzaamheden onze tarieven niet stijgen. Dit staat los van eventuele afspraken die genoemd zijn in de originele offerte, zoals een tijdelijke aanbieding.

Als reeds werk verder dan twee maanden vooruit is ingepland en wij kondigen een tariefswijziging aan, dan wordt ook dit werk tegen het nieuwe tarief in rekening gebracht.

2 Uitgangspunten

Voordat we ingaan op onze exacte werkwijze willen we een aantal uitgangspunten toelichten. Hiermee zitten de opdrachtgever en wijzelf altijd op een lijn als het gaat om belangrijke zaken.

2.1 De gebruiker voorop

Bij Label305 ontwerpen en ontwikkelen we softwareproducten met één overtuiging: 'de eindgebruiker staat voorop'. Wij geloven dat dit uiteindelijk leidt tot hogere kwaliteit, een betere gebruikerservaring en daarmee uiteindelijk tot betere producten.

'De eindgebruiker voorop' betekent ook dat de wensen van andere stakeholders zoals: onze projectmanager, de producteigenaar, de opdrachtgevende organisatie, en eventuele andere partijen niet altijd direct overgenomen worden. De eindgebruiker heeft vaak geen of een beperkte stem, met name in de vroege stadia van ontwerp en ontwikkeling. Daarom probeert iedereen bij Label305 de groep zo goed mogelijk te vertegenwoordigen. We zijn kritisch op alle productbeslissingen – niet alleen vanuit een technisch perspectief. Vooral het gebruikersperspectief staat voorop bij onze kritische kijk naar een plan en de uitvoering.

Fervente voorstanders van user-centered design zullen stellen dat de feedback van echte eindgebruikers tijdens alle fasen van het ontwikkelingsproces de enige echt correcte input is. Onze overtuiging ligt genuanceerder. Wij geloven zowel in de expertise en het inlevingsvermogen van onze specialisten als in de daadwerkelijke input van eindgebruikers.

We moeten gebruikers helpen met het verduidelijken en generaliseren van hun wensen. Zo weten eindgebruikers vaak niet precies wat ze echt willen zien, en moeten we de juiste productkeuzes maken en achteraf valideren. Een bekend voorbeeld is zogenaamde '*feature creep*', een behoefte om het eindproduct om in het eindproduct alle functies te verwerken die een gebruiker zich kan wensen. Vervolgens worden de samenhang en de kwaliteit van individuele functies over het hoofd gezien.

2.2 Altijd inzicht in het plan

Het is essentieel om continu inzicht te hebben in de visie en de plannen voor het softwareproduct. Hou ons ook daarom altijd op de hoogte van veranderingen in productfocus, plannen en visie. Vertel ons over de strakke budgetten en begrotingen, op zowel de korte als de lange termijn. Zo focussen we onze aandacht altijd op de belangrijkste zaken en houden we altijd beschikbare tijd en budgetten in ons achterhoofd terwijl we jou, de opdrachtgever, adviseren. Vervelende discussies achteraf moeten kosten wat kosten vermeden worden. Daarvoor dragen we een gezamenlijke verantwoordelijkheid.

2.3 De rol van producteigenaar

Tijdens het ontwikkelproces is een essentiële rol voor de opdrachtgever weggelegd, namelijk de rol van producteigenaar (*product owner*). De producteigenaar maakt alle beslissingen over de ontwikkelrichting en bepaalt aan welke functies gewerkt moet worden. Als meerdere personen vanuit de opdrachtgevende partij betrokken zijn bij de ontwikkeling, is het dus noodzakelijk om expliciet de rol van producteigenaar aan één persoon toe te wijzen.

2.3.1 Onze unieke kijk op producteigenaarschap

Bij veel grote internetbureaus en digital consultancies wordt vaak een producteigenaar aangewezen binnen de dienstverlener zelf. Deze persoon heeft de verantwoordelijkheid om de opdrachtgever binnen het team te vertegenwoordigen en de vertaalslag te maken van ontwikkelaar naar opdrachtgever. Sommige grote organisaties vinden het wellicht prettig om alles uit te besteden en zelf minder sturing over het project te hebben. Wij zien dat anders en hebben graag een meer hands-on aanpak met korte lijntjes.

Wij leggen expliciet de rol van producteigenaar neer bij de opdrachtgevende organisatie. Iemand uit eigen organisatie kan per slot van rekening beter de belangen van de organisatie vertegenwoordigen dan een externe producteigenaar bij Label305. Daarnaast scheelt het een extra communicatiestap en zorgt dit, in onze optiek, uiteindelijk voor een sneller en beter ontwikkelproces.

Let hierbij op dat de rol producteigenaar los staat van de rol van projectmanager, ook wel scrum master of productmanager genoemd. Bij Label305 begeleidt er altijd binnen het team iemand het ontwikkelproces: onze projectmanager. Deze persoon is de primaire contactpersoon van de producteigenaar.

2.3.2 Een belangrijke verantwoordelijkheid

Goed producteigenaarschap is essentieel voor een succesvol ontwikkelingsproces. Deze verantwoordelijke rol is dan ook meestal tijdsintensief. Zo moet bij aanvang van elke sprint een duidelijke geprioriteerde lijst van wensen aangeleverd worden, en is er continue communicatie tussen de producteigenaar en ons team.

Bij een oplevering is het zaak dat de producteigenaar goede inhoudelijke feedback geeft op het geleverde werk. Als meer betrokkenen feedback geven is het de taak van de producteigenaar om deze feedback te verzamelen en te verwerken in de lijst met wensen voor aankomende sprints. We adviseren de producteigenaar ook continu bij het opstellen en aanpassen van de wensenlijst.

De producteigenaar zorgt continu voor goede communicatie met onze projectmanager en ons hele team. Maar de producteigenaar is zelf geen direct onderdeel van ons

team. Hiermee blijft ons team zelfsturend. Het is voor ons belangrijk dat wij zelf bepalen hoe we onze werkzaamheden uitvoeren.

Omdat de producteigenaar vaak nauw betrokken is bij het ontwikkelingsproces, en soms zelf ook ervaring heeft in de ICT, kan het gebeuren dat door hem of haar suggesties gedaan worden over de aanpak van een probleem op detailniveau. We nemen deze input mee maar nemen uiteindelijk zelf een beslissing. Als we alles direct overnemen zou er een negatieve impact op onze effectiviteit kunnen zijn. Er wordt dan namelijk niet meer optimaal gebruik gemaakt wordt van onze ervaring en expertise.

Meer over het producteigenaarschap in het hoofdstuk: 'Een agile ontwikkelingsproces: werken in sprints'.

2.3.3 Overdracht producteigenaarschap

Het kan gebeuren dat de rol van het producteigenaarschap binnen de opdrachtgevende organisatie wordt overgedragen. Het is uitermate belangrijk dat dit zorgvuldig gebeurt en expliciet gemeld wordt aan alle betrokkenen. Als dit niet gebeurt, kan het fouten in de hand werken.

2.4 Voorkomen van vendor lock-in

We willen graag dat de producten die wij ontwikkelen succesvol worden. Zelfs zo succesvol dat onze opdrachtgevers op een gegeven moment niet meer van Label305 afhankelijk zijn voor doorontwikkeling, en de ruimte hebben om een eigen team op te bouwen*.

** We spreken wel af dat we geen personeel van elkaar over nemen. Meer hierover in de algemene voorwaarden.*

Als we een softwareproduct ontwikkelen is het vrijwel onmogelijk om van het ene op het andere moment alle afhankelijkheid weg te nemen. Bij het ontwerpen en ontwikkelen wordt namelijk een enorme hoeveelheid projectspecifieke kennis opgebouwd. Daarom hebben we hier een plan voor.

Het voorkomen van vendor lock-in doen we op vier manieren:

1. Bij het ontwikkelen van een product gebruiken we populaire en robuuste tools, die bekend zijn in de ontwikkelgemeenschap. Meestal zijn deze tools open source.
2. We gebruiken gemakkelijk overdraagbare en losstaande applicatie-infrastructuur.
3. We hebben een duidelijke afspraak over intellectueel eigendom van de code, zodat dit niet in de weg zit.

4. We bieden een proces aan dat we *ontwikkelingstransitie* noemen, waarbij we mensen uit de opdrachtgevende organisatie inwerken en opleiden zodat ze uiteindelijk zelfstandig (mee) kunnen ontwikkelen aan het product.

2.5 Gezondheid van de codebase

De codebase is de verzameling van alle code waaruit het product bestaat. Om het ontwikkelproces soepel te laten verlopen, moeten alle partijen rekening houden met een aantal zaken die totaal onzichtbaar zijn voor de eindgebruiker.

2.5.1 Technical debt

Een belangrijke bezigheid tijdens het ontwikkelproces van softwareproducten is het in toom houden of verminderen van *technical debt*. Bij het introduceren van nieuwe functies aan een softwareproduct kan technical debt ontstaan. Zo creëren wij vaak eerst software zodat deze goed werkt en zodat de eindgebruikers de nieuwe functies kunnen gebruiken. Hierna kan vaak nog veel verbeterd worden aan de onderliggende structuur om zo de codebase compact en efficiënt te houden. Dit kan bijvoorbeeld door abstracties aan te brengen en bepaalde code samen te voegen. Een compacte en goed gestructureerde codebase met goed aangebrachte abstracties is buitengewoon belangrijk om effectief door te kunnen werken aan nieuwe functies. Het ontstaan van technical debt is eigenlijk onvermijdelijk, omdat toekomstige veranderingen volledig samenhangen met de huidige structuur van de code. Bij het schrijven van code kan nooit met alle mogelijke toekomstige aanpassingen rekening gehouden worden. We kunnen enkel werken aan code die een duidelijke structuur heeft en het wijzigen zo makkelijk mogelijk maakt. Het verminderen van technical debt is dan ook iets waar we ons regelmatig op moeten richten.

Wederom geldt dat wij graag op de hoogte worden gebracht van korte- en langetermijnplannen voor het product. Als er bijvoorbeeld op korte termijn een grote nieuwe functie aan het softwareproduct toegevoegd moet worden, gaan we ons daarvoor inzetten. Hierbij moeten we rekening houden dat na de oplevering van deze functies wellicht extra werk nodig is voor het verminderen van de ontstane technical debt. Het kan ook zijn dat we, voorafgaand aan het ontwikkelen van de nieuwe functie, eerst moeten werken aan betere structuren in de code – en eerst de huidige technical debt moeten verminderen.

De onzichtbare aard van technical debt en zakelijke belangen op de korte termijn gaan vaak niet goed samen. Soms zorgt dit ervoor dat wij samen beslissen om lange perioden bezig zijn met nieuwe functies, zonder de juiste aandacht te geven aan technical debt. Het behartigen van de kortetermijnbelangen kan op de lange termijn gevolgen hebben en bijvoorbeeld leiden tot meer onverwachte bugs en een stroever ontwikkelproces van nieuwe functies. Wel kunnen we op de korte termijn sneller veel waarde toevoegen aan het product. Het is niet vreemd om na een periode van intensieve ontwikkeling van nieuwe functies, een aantal sprints te besteden aan enkel

het beperken van technical debt. Bij een sprintoplevering wordt dan een nieuwe versie van het product opgeleverd zonder dat er iets veranderd lijkt te zijn. Toch is aandacht voor technical debt buitengewoon belangrijk voor de langetermijnontwikkeling van het product en voor de samenwerking.

2.5.2 Complexiteit en formaat van de codebase

Los van technical debt kan codebase van een softwareproduct naar verloop van tijd een enorm formaat hebben. Er zijn dan veel abstracties aangebracht en het product heeft veel functies met ingewikkelde kruisverbanden. Het is belangrijk om te realiseren dat een softwareproduct met veel ingewikkelde functies over het algemeen minder eenvoudig is om uit te breiden. Je zou kunnen stellen dat het introduceren van nieuwe (complexe) functies impact heeft op de ontwikkelsnelheid van toekomstige functies. Met het aanbrengen van goede abstracties en het verminderen van technical debt kunnen we dit effect tot een minimum beperken, maar nooit geheel wegnemen. Het is goed om in gedachten te houden bij het toevoegen van nieuwe (complexe) functies aan het softwareproduct.

Een goed voorbeeld van een nadelig effect van een grote codebase is het formaat van de zogenaamde testsuite. We schrijven veel automatische tests voor het softwareproduct zodat we met een druk op de knop kunnen testen of belangrijke functies in de applicatie, na veranderingen aangebracht te hebben, nog steeds goed werken. Des te meer functies het product heeft, des te langer deze automatische tests zullen duren, des te langer de specialist moet wachten op het resultaat van die tests, des te langzamer het ontwikkelproces gaat.

2.5.3 Code optimaliseren

Het optimaliseren van code wordt regelmatig in dezelfde adem genoemd als de term *technical debt*. Vaak pakken we deze problemen gelijktijdig aan, maar het is goed om te weten dat het twee verschillende concepten zijn. Bij het optimaliseren van code proberen we de software sneller maken. Bijvoorbeeld door een algoritme efficiënter te maken, of door cachingstrategieën te implementeren. Hiervoor introduceren we vaak veel nieuwe code.

Het is onverstandig om alle functies direct zo efficiënt mogelijk te schrijven. Hier is vaak meer code voor nodig en daardoor kan meer technical debt ontstaan. Dat terwijl optimalisatie de software effectief niet heel veel sneller hoeft te maken. Verstandiger is om af en toe goed te kijken welke onderdelen van de code het meest baat kunnen hebben bij grondige optimalisatie.

“(...) premature optimization is the root of all evil (or at least most of it) in programming.”
 – Donald Knuth (winnaar Turing Award)

2.6 Belang en noodzaak van kennisuitbreiding

Er zijn weinig vakgebieden zo dynamisch als softwareontwikkeling. Dit heeft zowel te maken met de exponentiële vooruitgang in informatietechnologie als met de relatief jonge leeftijd van het vakgebied. Dit is ook de reden dat onze specialisten zich in een continu proces van leren en ontplooiën bevinden. Het continu leren over nieuwe technieken, nieuwe tools, nieuwe platformarchitecturen en nieuwe werkwijzen is een essentieel onderdeel van het ontwerpen en ontwikkelen van softwareproducten.

Veranderingen gaan zo snel dat zelfs onze meest ervaren specialisten continu moeten blijven leren, ook terwijl ze aan producten werken. Af en toe stellen we voor om een compleet nieuwe techniek toe te passen in een product. We informeren hierbij vooraf de producteigenaar van ons gebrek aan ervaring en wegen de voordelen af tegen de risico's. Meestal staat dit goede productontwikkeling niet in de weg.

Als onze specialisten zichzelf niet blijven ontwikkelen kunnen zij op een gegeven moment onze opdrachtgevers niet meer optimaal van dienst zijn bij het creëren van innovatieve producten. Daarom is het continu uitbreiden van onze kennis voor opdrachtgevers noodzakelijk en voor Label305 cruciaal.

Wij houden een goed overzicht van de verschillende expertises en kijken altijd welke specialisten de beste aansluiting hebben bij de verschillende projecten. Daarnaast zetten we ons in om altijd meerdere specialisten te hebben met de specifieke expertises die nodig zijn voor een product. We laten zowel ervaren en minder ervaren specialisten aan een product werken.

Intern hanteren we een code reviewsysteem waarbij (ervaren) ontwikkelaars elkaar controleren en onderwijzen. Op deze manier kunnen we ook altijd onze opdrachtgevers zo goed mogelijk blijven bedienen.

Het is goed om te realiseren dat dit leerproces tijd in beslag neemt. Hierbij geldt dat tijd besteed aan het elkaar onderwijzen, op de korte termijn niet altijd direct zinvol lijkt, maar juist op de lange termijn erg belangrijk is voor het product. We zien continu leren dan ook als een integraal onderdeel van het ontwikkelingsproces.

2.6.1 Intern maken we extra ruimte om verder te leren en te verdiepen

Onze specialisten krijgen regelmatig extra ruimte om zichzelf verder te ontwikkelen buiten reguliere ontwikkel- en ondersteuningswerkzaamheden. Zo worden ze regelmatig een hele week *niet ingepland* voor werkzaamheden, om zich enkel te focussen op zelfontplooiing en interessante problemen die ze zelf willen aanpakken. Aan het einde van deze week geven ze regelmatig een interne presentatie van de hun bevindingen en nieuwe inzichten. Zo pikt iedereen een graantje mee.

2.7 In rekening brengen van onze inspanningen

Gewoon even heel eerlijk – bij Label305 draaien we op de uren die wij aan onze opdrachtgevers doorrekenen. Dit zijn uren waarin wij ons vol inzetten voor de opdrachtgever en hun product. Al onze expertise en diepgaande projectinhoudelijke kennis staat dan tot hun beschikking. Het is volgens ons dan ook niet meer dan logisch dat we in principe het overgrote deel van de tijd, waarin wij voor het product aan het werk zijn, doorrekenen. Bij sommige werkzaamheden maken we een uitzondering of geven we graag een extra toelichting.

2.7.1 Acquisitie

Bij een eerste ontmoeting en bij het initiële acquisitieproces brengen we vanzelfsprekend *niets* in rekening. Hieronder verstaan wij de eerste communicatie, kennismaking en offerte.

Besprekingen over doorontwikkeling zien we niet meer als acquisitie maar als projectmanagement. Voorbeelden zijn gesprekken die gaan over: het samenstellen, uitbreiden, inschatten en prioriteren van gewenste nieuwe functies. Dergelijke besprekingen zien we als onderdeel van het proces, waarin aanspraak gemaakt wordt op onze expertise en projectinhoudelijke kennis.

Het bespreken van (veel) kleine aanpassingen moet ook gezien worden als projectmanagement, of in andere gevallen als ondersteuning. Met name bij deze aanpassingen gaat de tijd primair zitten in het bespreken en communiceren van de (vele) kleine aanpassingen.

2.7.2 Bespreken, vergaderen en kick-offmeetings

Het is belangrijk om regelmatig samen om de tafel te gaan voorafgaand, gedurende en na een traject. Onze projectmanager bereidt elke bespreking goed voor om deze zo productief mogelijk in te richten. Tevens zullen we de conclusies uit de bespreking achteraf goed documenteren en naar de producteigenaar toesturen.

Het is goed om te realiseren dat wij de uren van iedere afzonderlijke vergaderingsdeelnemer doorberekenen, voor zowel de voorbereiding, de bespreking zelf, als de documentatie van de bespreking achteraf.

Als besprekingen onderdeel zijn van de huidige of aanstaande sprint worden de uren niet los in rekening gebracht, maar tellen deze mee voor de minimale inspanningsverplichting van de sprint.

Bij de aanvang van een productontwikkelingsproces organiseren we meestal een kick-offmeeting. Hiervoor doen we vooraf onderzoek en werken we achteraf plannen en details uit. Dit is een bespreking waarvoor we soms een uitzondering maken omdat we soms deze sessie zien als verlengstuk van het acquisitieproces. Daarom rekenen we

slechts een deel van de bestede uren door. Voor deze bespreking nemen we dan ook een vaste prijs op in de initiële offerte.

2.7.3 Consultancy-, ontwerp- en ontwikkelwerkzaamheden

Bij het uitvoeren van zogenaamde *productiewerkzaamheden* zoals: consultancy, ontwerp en ontwikkeling brengen we alle gewerkte uren in rekening. Het eigen maken van een benodigde nieuwe techniek of platformarchitectuur zien we als productiewerkzaamheden. Ook het reviewen van elkaars code en het inlezen in een codebase zien we als productiewerkzaamheden.

Als productiewerkzaamheden onderdeel zijn van de huidige sprint worden de uren niet los in rekening gebracht, maar tellen deze mee voor de minimale inspanningsverplichting van de huidige sprint. Dit betekent dat voor een sprint met een inspanningsverplichting van 30 uur wij enkel 30 uur in rekening brengen, zelfs wanneer er iets meer gewerkt is. Mocht er onverhoopt gebruik gemaakt worden van onze sprintgarantie dan wordt een beperkt aantal uren buiten de sprint voor extra productiewerkzaamheden niet in rekening gebracht. *Meer over sprints en sprintgarantie in een volgend hoofdstuk.*

2.7.4 Projectmanagement

Projectmanagementwerkzaamheden zoals het opstellen van gedetailleerde projectplannen, communiceren over voortgang en intern overleg is naast productiewerk onze belangrijkste tijdsbesteding. Deze uren staan dan ook qua verrekening op gelijke voet met productiewerkzaamheden.

Als projectmanagementwerkzaamheden onderdeel zijn van de aanstaande of de huidige sprint worden de uren niet los in rekening gebracht, maar tellen deze mee voor de minimale inspanningsverplichting van die sprint, ook als de sprint nog niet is aangevangen. We brengen dus voorafgaand aan een sprint geen losse werkzaamheden in rekening voor de voorbereiding van die sprint, maar zien dit werk als een onderdeel ervan. *Meer over sprints in een volgend hoofdstuk.*

2.7.5 Onderhoud

Uren waarin wij klein onderhoud uitvoeren aan de applicatie-infrastructuur of de codebase worden, indien mogelijk, verrekend via de op dat moment geldende serviceovereenkomst.

Voorbeeld klein onderhoud: Er is een softwarepatch beschikbaar gekomen voor een veiligheidsprobleem in de Linux-kernel die gebruikt wordt in de applicatie-infrastructuur. We duiken de infrastructuur in om deze softwarepatch overal toe te passen zonder (of met zo min mogelijk) downtime voor de applicatie.

Soms is het nodig om vanwege platformupdates, waar we afhankelijk van zijn, groot onderhoud uit te voeren. Als dit moment aanbreekt geven de projectmanagers dit aan en wordt er gepoogd groot onderhoud in te plannen in sprints.

Voorbeeld groot onderhoud: Een opdrachtgever heeft bij ons een native iPhone-applicatie laten ontwikkelen. Vervolgens brengt Apple zijn jaarlijkse update van de zogenaamde iOS software development kit (SDK) uit. Door deze update moeten grote wijzigingen in de codebase gedaan worden om prettig door te kunnen ontwikkelen met deze nieuwe SDK. Dit is iets dat we zien als groot onderhoud.

2.7.6 Ondersteuning

Uren waarin wij ondersteuning verlenen worden indien mogelijk verrekend via de dan lopende serviceovereenkomst.

Als medewerkers van de opdrachtgevende organisatie een probleem ondervinden, een vraag hebben of een suggestie willen doen staan we hier altijd voor open. Dit is typisch iets wat we zien als ondersteuning.

Een ander voorbeeld is het opvragen van een gegevensrapport of een gefilterde database dump. Dergelijke werkzaamheden zijn soms iets intensiever dan je zou denken, omdat we dit moeten doen op een manier die de privacy van de eindgebruiker waarborgt. Soms is het nodig om voor een eindgebruiker direct in de database een gegevensmutatie te doen, alhoewel dit zeker geen voorkeur heeft. Indien de tijdbesteding voor dergelijke werkzaamheden beperkt is, zien we deze als ondersteuning.

Meer over ondersteuning in het hoofdstuk: 'Service, onderhoud en ondersteuning'.

2.7.7 Senioriteit (juniores, werkstudenten en stagiairs)

Bij Label305 hebben we *geen* verschillende uurtarieven voor verschillende disciplines of senioriteitsniveaus. Wel is het zo dat wij bij uitzondering bepaalde uren niet doorrekenen op basis van senioriteitsniveau.

Natuurlijk hebben wij naast onze ervaren senioren en medioren ook junioren in dienst. Sommige van deze junioren werken parttime naast hun studie bij Label305. Als junioren meewerken aan een product en nog niet direct een volwaardige bijdrage leveren, dan brengen we niet altijd de volledige hoeveelheid gewerkte uren in rekening. Wij beoordelen dit zelf intern en doen dat naar redelijkheid.

Studenten aan hogescholen en universiteiten moeten verschillende stages doen tijdens hun opleiding. Voor Informatica, Multimedia Design en verwante opleidingen bieden we stages aan. Soms laten we in samenspraak met de opdrachtgever een stagiair werken aan een product. Dit gaat om een aanzienlijke periode: twee tot zes

maanden. Over het algemeen wordt slechts een zeer kleine fractie hiervan in rekening gebracht. Soms wordt hiervoor zelfs helemaal niets in rekening gebracht.

2.8 Bewustzijn van verborgen kosten

Aan het ontwikkelen en onderhouden van een softwareproduct zijn meer kosten verbonden dan alleen die voor het inhuren van een software development consultancy zoals Label305. Voor de hand liggend zijn de kosten voor het inrichten en serveren van applicatie-infrastructuur. Daarnaast zijn er ook nog een paar minder voor de hand liggende kosten. Een aantal hiervan lichten we hier dan ook graag nog even toe.

2.8.1 Interne kosten gedurende het ontwikkelingsproces

Hou er rekening mee dat ook vanuit de opdrachtgever een significante tijdsinvestering vereist is, bijvoorbeeld voor het invullen van de rol van producteigenaar. Continue heldere communicatie over de wensen en plannen is belangrijk – maar ook tijdrovend. Dit geldt ook voor het continu geven van feedback na de sprintoplevering. Hou hier dus rekening mee bij het maken van een eventuele begroting en een uiteindelijke beslissing om samen met ons een traject te starten.

2.8.2 Ondersteuning voor de eindgebruiker

Zodra een softwareproduct live gaat komen eindgebruikers vroeg of laat naar je toe met vragen en suggesties. Bij een succesvol product kan dit soms tientallen keren per dag gebeuren. Goede klantenservice is natuurlijk een belangrijk onderdeel van een goed product. Eindgebruikers verwachten tegenwoordig snel een goed antwoord te krijgen op hun vraag. Daarom is het goed te realiseren dat het leveren van directe ondersteuning aan eindgebruikers, *niet* iets is wat Label305 uit handen neemt. Hiervoor wordt een intern proces moeten opgetuigd. Dit brengt kosten met zich mee.

Zoals eerder genoemd is een terugkoppeling vanuit de klantenservice naar ons als productspecialisten belangrijk. Zo kunnen we technische problemen duiden en licht werpen op vreemde situaties. Wij staan hiervoor altijd paraat zoals beschreven in de serviceovereenkomst*. Met een onderzoekje of een inzicht van ons kan de klantenservice snel weer terug naar de eindgebruiker en antwoord geven.

** Communicatie over vragen of suggesties vanuit de eindgebruikers zien we als ondersteuning. Met een SLA garanderen we een snelle reactie. Met een BESA proberen we snel te reageren, maar geven we hiervoor geen garanties af. Als er geen onderhoudsovereenkomst is afgesloten, antwoorden wij en brengen achteraf de gewerkte uren in rekening tegen ons hoge uurtarief.*

2.8.3 Ontwikkelingsstagnatie

Over het algemeen is het voor de meeste softwareprojecten niet verstandig om de ontwikkeling lange tijd stil te leggen. In de meeste situaties is een softwareproduct eigenlijk nooit af. Er zijn altijd platformwijzigingen waarop ingespeeld moeten worden,

nieuwe wensen van eindgebruikers en verbeteringen aan te brengen. Ontwikkelingsbudgetten zijn vaak beperkt, dus daar moeten we rekening mee houden.

Volledige stagnatie is iets wat voorkomen moet worden. Dit zien we los van een eventuele serviceovereenkomst en hiermee bedoelen we dat er *niet* regelmatig doorontwikkeld wordt, bijvoorbeeld via een sprint. Stagnatie zorgt er namelijk voor dat groot onderhoud niet uitgevoerd wordt, maar ook dat een toegewijde specialist de mentale aansluiting met het project langzaam verliest. Een van de krachten van een goede specialist is dat hij/zij bezig is met het nadenken over de problemen en structuren van het softwareproduct, ook als hij/zij niet achter de computer zit. Bovendien zijn onze specialisten op de hoogte van alle veranderingen in het IT-landschap, daarbij is het fijn dat zij dan ook aan de impact op de ontwikkelde producten denken. Dit alles vereist dat een specialist helemaal is ingelezen. Als ontwikkeling stagneert, neemt logischerwijs de algemene betrokkenheid af. Bovendien kan het opnieuw inlezen na een lange periode van stagnatie een aanzienlijke hoeveelheid tijd in beslag nemen.

Wij raden af om ontwikkeling te stagneren en moedigen aan om een minimale ontwikkeling gaande te houden, ook als er minder budget is. Bijvoorbeeld één sprint per maand. Een cynicus zou misschien zeggen dat dit klinkt als het enkel behartigen van ons eigen belang, toch is dat juist niet de insteek van deze opmerkingen. De impact van stagnatie wordt helaas vaak onderschat.

Bij het omgaan met een ontwikkelingsbudget is het goed om te denken in termen van *runway*, oftewel de doorlooptijd van nu tot het opraken van het budget. Het is dus bijvoorbeeld in sommige gevallen verstandiger om ontwikkeling over een langere periode uit te smeren als er een beperkt budget is, en niet heel veel ontwikkeling te doen in een korte periode om daarna volledig te staken met ontwikkelen.

2.9 Het productrisico

Wij vinden het fantastisch om aan innovatieve en nieuwe softwareproducten te werken, bovendien zijn we er goed in. Een belangrijk aspect van dergelijke producten is het risico dat gemoeid gaat met het ontwikkelen van iets totaal nieuws, het vermarkten van het nieuwe product en het toepassen van een eventueel ongetest businessmodel. Over het algemeen is het zo dat: des te innovatiever het product, des te groter het risico. Omdat we graag aan verschillende innovatieve producten werken, bij deze een paar woorden over dit productrisico, waar elk project mee te maken heeft.

We hebben er samen baat bij dat het product succesvol is en zetten ons daar dus ook vol voor in. Desondanks is het goed om te realiseren dat de opdrachtgever *het volledige productrisico* op zich neemt. Je plukt per slot van rekening uiteindelijk alle vruchten van het succes van het product. We delen wel mee in het succes doordat je ons laat doen wat we fantastisch vinden: het product nog beter maken.

Het uitgangspunt is dat wij het productsucces helemaal willen faciliteren. Lees hierover meer in de hoofdstukken over: 'Ontwikkelingstransitie en participatie' en 'Intellectueel eigendom'.

Er moet gezegd worden dat het verrekenen van onze werkzaamheden en het aanhouden van onze werkwijze hier helemaal buiten wordt gehouden. Hiermee zorgen we er namelijk voor dat we altijd kunnen blijven doen wat we fantastisch vinden. Wij nemen wel verantwoordelijkheid als we iets helemaal verkeerd doen. Als we bij het naleven van de SLA of bij een sprintoplevering een fout maken, hebben we hiervoor een garantieregeling.

Soms doet een situatie verwant aan hetgeen hieronder beschreven zich voor.

Een opdrachtgever heeft weinig budget, maar ziet een succesvolle volgende ontwikkelslag als noodzakelijk voor een belangrijke volgende stap. Bijvoorbeeld: het binnenhalen van een grote investering, het betrekken van een nieuw partnerbedrijf, of het betrekken van een grote nieuwe groep eindgebruikers. Vervolgens worden we gevraagd om bijvoorbeeld: vooruit te werken, onze tarieven tijdelijk te verlagen, of van onze werkwijze af te wijken. Zo wordt soms gevraagd om een broodnodige ontwikkelslag op projectbasis te doen.

Wij blijven graag op de hoogte van alle plannen en beperkingen zodat we met de opdrachtgevende organisatie mee kunnen blijven denken. Wij zetten ons er helemaal voor in om onze opdrachtgever te helpen bij het verwezenlijken van zijn/haar doel. Maar zoals je zal begrijpen kunnen we niet ingaan op de bovengenoemde voorstellen. In deze voorstellen wordt namelijk (deels) het productrisico naar ons verlegd. Ondanks dat een dergelijke situatie zich soms voordoet en wij dan niet mee kunnen gaan in het voorstel, komen we vrijwel altijd tot een goede gezamenlijke oplossing.

3 Een agile ontwikkelproces: werken in sprints

Voor een langere periode in teamverband werken aan softwareproducten gaat via een gestructureerde en beproefde methode. Hierbij is het belangrijk om flexibel te blijven, om zodoende goed om te gaan met nieuwe inzichten en onverwachte tegenslagen. Bovendien willen we creativiteit maximaal tot haar recht te laten komen bij productontwikkeling en -verbetering. Dit is de reden dat wij ons ontwikkelproces op een *agile*-manier inrichten.

3.1 Verwachtingen bij een agile softwareontwikkelingsproces

In gesprekken over softwareontwikkeling wordt soms een metafoor gelegd naar ontwikkelingsprocessen in: de architectuur, machine ontwerp, civiele techniek of industrieel ontwerp. Deze metaforen kunnen leiden tot een verkeerd beeld van het procesverloop. Dit kan vervolgens, mede door onbegrip, een nadelig effect hebben op het proces zelf. Bij de genoemde projecttypen wordt er meestal vooraf een tot in de puntjes uitgewerkt plan gemaakt. Dit plan kan vervolgens soepel uitgevoerd worden, mede doordat er met allerlei factoren rekening is gehouden. Dit kan omdat er gewerkt wordt binnen een relatief langzaam veranderende omgeving. Het is meestal nadelig om in latere stadia wijzigingen aan te brengen in de scope of onderliggende structuren.

Softwareontwikkeling is anders – en met name agile softwareontwikkeling. Het in een later stadium aanbrengen van wijzigingen in de scope of onderliggende structuur, is juist bij softwareontwikkeling *we*/ mogelijk. Alhoewel de impact hiervan bij softwareprojecten van formaat nog steeds groot kan zijn, is dit nog steeds relatief kleiner dan bij andere typen projecten. Software zit flexibeler in elkaar. Daarentegen zijn wij bij softwareontwikkeling vaak onderhevig aan de snelle ontwikkelingen in het IT-landschap. Ook moeten we werken met zeer omvangrijke platformspecifieke API en externe koppelingen die continu veranderen. Dit is dan ook de reden voor onze agile-aanpak. Deze aanpak is dan ook bewust veel flexibeler en meer incrementeel dan de aanpak van projecten in andere industrieën. Dit is noodzakelijk om goed om te gaan met de totaal andere dynamiek die geldt voor in de wereld van software.

Wat betekent dit nu concreet? Bij een agile-aanpak zoeken we vooraf voldoende uit, om zo een goed overzicht te krijgen en risico's te minimaliseren. Alleen we werken niet altijd *alle* details tot in de puntjes uit voordat er gestart wordt met ontwikkelen. Een groot deel van de hieraan bestede tijd zou verspild kunnen zijn. Voortschrijdende inzichten, die voortkomen uit het gebruik en de ontwikkelingservaring, hebben meer waarde voor toekomstige ontwikkeling dan een groot gedetailleerd plan dat vooraf is opgesteld. Dit betekent natuurlijk niet dat er geen plannen gemaakt worden. Enkel dat dit incrementeel gebeurt. Ook is de detaillering hierin niet altijd van hetzelfde niveau als plannen bij projecten in andere industrieën. Dit is niet erg, maar juist wenselijk en leidt uiteindelijk tot een sneller en flexibeler ontwikkelproces.

3.2 Sprints

Een fundamenteel onderdeel van onze agile werkstructuur is de vaste opleverfrequentie. Elke oplevering gebeurt aan het einde van een zogenaamde **sprint**. In een sprint werkt een heel team in een vaste periode van één of twee weken toe naar één oplevering. Sprints zijn onze primaire werkeenheid. Projectwerkzaamheden worden bij ons voornamelijk afgenomen en doorberekend om de vorm van sprints.

Een sprintoplevering is *nooit* een oplevering naar een productieomgeving of naar een applicatiewinkel, maar altijd een oplevering met de vraag voor feedback of goedkeuring voor een lancering naar een productieomgeving.

Bepaalde momenten in het ontwikkelproces worden gemarkeerd als zogenaamde *milestones*. Deze milestones hebben een specifieke datum en een overkoepelend doel dat op die datum behaald moet zijn. Dit in tegenstelling tot sprints, die juist een doorlopend en iteratief karakter hebben. Voor een milestone werken wij bijvoorbeeld toe naar een grote lancering of naar een versie die getoond kan worden op een beurs. *Meer over milestones het hoofdstuk: 'Doorlopende planning en dagindeling'.*

3.2.1 Vaststellen sprintomvang

Concreet is een sprint een korte periode van één of twee weken waarin wij met een team aan een project werken. Binnen deze periode hebben wij een minimale inspanningsverplichting én een opleveringsverplichting.

De sprintomvang wordt vastgesteld op basis van teamgrootte en opleverfrequentie. Sommige van onze opdrachtgevers vinden het prettig elke week een oplevering te zien en continu feedback te geven. Andere opdrachtgevers vinden het juist fijn om het iets meer los te laten en elke twee weken een oplevering te bekijken. De sprintomvang kan daarom per project verschillen en gedurende een project veranderen. Samen met de producteigenaar kijken we welke specifieke omvang past bij de aanstaande werkzaamheden. Als een sprintomvang is vastgesteld heeft een sprint een vaste prijs zodat onze opdrachtgever weet waar hij/zij aan toe is.

Wij werken niet met kortere opleverfrequenties dan een week, en niet met langere opleverfrequenties dan twee weken. Dit doen wij om onze werkstructuur optimaal te houden, communicatiefouten te voorkomen en uitloop te vermijden.

De sprintprijs die wij in rekening brengen wordt bepaald met het **lage uurtarief** maal het aantal uren van de minimale inspanningsverplichting. Als we een uurtje of twee extra werken in de sprint brengen we die niet in rekening. We zorgen er wel voor dat we sowieso voldoen aan de minimale inspanningsverplichting. Achteraf kan een specificatie van de uren opgevraagd worden.

Team-grootte	Opleverfrequentie in weken	Inspanningsverplichting*
1	1	30 Uren
1	2	60 Uren
2	1	60 Uren
2	2	120 Uren
3	1	90 Uren
3	2	180 Uren
4	1	120 Uren
4	2	240 Uren
5	1	150 Uren
5	2	300 Uren
6	1	180 Uren
6	2	360 Uren

** Als een sprint samenvalt met een feestdag, wordt de minimale inspanning in verlaagd met 6 uur per persoon, per feestdag. Hiermee dalen dan ook de kosten voor die sprint.*

3.2.2 Vooraf sprintsamenstellingen via de backlog vaststellen

Voorafgaand aan een sprint worden door de producteigenaar, in samenspraak met onze projectmanager, de werkzaamheden bepaald en op prioriteit gesorteerd. Deze lijst noemen wij de *backlog*. De geprioriteerde lijst met concrete werkzaamheden wordt overeengekomen voor de dag dat de sprint aanvangt. Het bovenste deel van de backlog, dat we verwachten in de sprint te doen, noemen we de *sprint backlog*. De backlog kan veel meer taken bevatten dan gedaan kunnen worden in één of twee sprints. Soms heeft het verwerken van feedback op eerder opgeleverde functies prioriteit. Op een ander moment starten we met het optimaliseren van delen van het product, of we beginnen juist met het ontwikkelen van nieuwe onderdelen van het product.

Als er geen backlogwijzigingen zijn doorgegeven voor aanvang van de sprint, gaan wij ervan uit dat niets is veranderd aan de prioriteiten en gaan wij verder met de eerder vastgestelde lijst. Ervaring leert dat er voor vrijwel iedere sprint gekeken wordt naar de backlog, omdat er vaak feedback terugkomt van eerdere opleveringen.

3.2.3 Formaat van een item in de backlog

Er wordt ons vaak gevraagd om voor verschillende taken in de backlog een ureninschatting te maken. Dit kunnen wij doen. Hiervoor kijken wij naar inschattingen uit het verleden en de daadwerkelijk gewerkte uren voor eerder uitgevoerde taken.

Een ureninschatting voor een specifieke taak valt soms zo hoog uit dat de taak meer tijd kost dan überhaupt in één sprint gewerkt kan worden. Dit is een goede indicatie dat we de taak op moeten splitsen in subtaken die afzonderlijk in de backlog moeten worden opgenomen.

Ervaring leert dat het uitermate lastig is om een exact accurate ureninschatting te maken per taak, daarom kunnen ook geen rechten worden ontleend aan inschattingen per taak. Wel kunnen we deze informatie in combinatie met de daadwerkelijk gewerkte uren gebruiken om zogenaamde sprint velocity te bepalen. Met een sprint velocity kunnen we uiteindelijk beter inschatten hoeveel werk wij gemiddeld per sprint kunnen doen. Ook geeft een inschatting de producteigenaar een ruw idee waaraan gedacht moet worden.

Het is vaak effectiever om te werken met milestones, en hierbinnen ons te focussen op de taken met de hoogste prioriteit. Een eventuele sprint velocity kan gebruikt worden om goede datums te bepalen voor specifieke milestones.

3.2.4 Backlog onduidelijk voor aanvang sprint

Is een sprint ingepland maar is de (sprint) backlog nog niet vastgesteld? Dan kunnen we eigenlijk niet starten met de sprint.

Onze projectmanager draagt samen met de producteigenaar de verantwoordelijkheid om de lijst goed en duidelijk te hebben vastgesteld voor de aanvang van een sprint. Als een sprint aanvangt, maar we geen backlog kunnen vaststellen dan ontstaat een vervelende situatie. Dit kan bijvoorbeeld gebeuren doordat de producteigenaar niet beschikbaar is. We proberen dergelijke situaties zo snel mogelijk op te lossen.

Mochten we onverhoopt niet een backlog kunnen vaststellen door toedoen van de opdrachtgever, dan moeten we alsnog de sprint in rekening brengen. We doen dan ons best de sprint naar eigen inzicht zo goed mogelijk in te vullen. Echter, vervalt de opleveringsverplichting en de mogelijkheid voor aanspraak op garantie in deze situatie.

3.3 Communicatie, voortgang en bijsturen tijdens de sprint

We houden graag tijdens de sprint contact over ontwikkelingen die invloed hebben op de huidige sprint, en ook over ontwikkelingen die van invloed kunnen zijn op volgende sprints. Dit doen wij bij voorkeur niet via e-mail, telefoon of chat maar juist via projectmanagementsoftware. Projectmanagementsoftware geeft voor iedereen altijd een goed overzicht van alle communicatie, en iedereen kan altijd makkelijk alles terugvinden. Contact per e-mail, telefoon of chat is ook mogelijk.

3.3.1 Keuze voor projectmanagementsoftware

Voordat we een agile ontwikkelproces starten kijken wij welke projectmanagementsoftware geschikt is voor de samenwerking. Het aanhouden van de werkwijze, het begrip hiervoor en goede communicatie, zijn allemaal veel belangrijker dan de keuze voor een specifiek softwarepakket. Toch is één centrale plek waar alle afspraken worden vastgelegd vaak buitengewoon handig. Bij veel projecten gebruiken wij hiervoor de projectmanagementtool Basecamp.

3.3.2 Het wijzigen van de (sprint) backlog

Gedurende de sprint hebben wij liever niet dat volledig nieuwe punten aan de sprint backlog worden toegevoegd. Dit is het deel van de backlog waaraan gewerkt wordt in de huidige sprint. Het wijzigen van dit deel van de backlog kan namelijk de huidige werkzaamheden verstoren. Voor urgente zaken kan het verstoren van de werkzaamheden acceptabel zijn, en dus willen we graag flexibel zijn.

Het is geen probleem om feedback van vorige sprints direct op te pakken in de huidige sprint zodra dit binnenkomt.

Nieuwe taken kunnen worden ingediend met verschillende prioriteitsaanduidingen:

- **Spoed of onmiddellijk oppakken**, bij de hoogste prioriteit laten wij alles vallen om het nieuwe punt meteen op te pakken;
- **Oppakken als volgende taak**, we maken het huidige punt af en daarna pakken deze taak op, dit is wat ons vaak wordt gevraagd bij feedback;
- **Oppakken in de volgende sprint**, we ronden de huidige sprint af en pakken dit punt meteen de eerstvolgende sprint op; en
- **Ooit oppakken**, we plaatsen deze wens onderaan de lijst met prioriteiten en wachten totdat deze opnieuw ingedeeld wordt.

Soms lopen wij tegen een probleem aan waarbij input van de producteigenaar nodig is. Ook in dit geval is het vanzelfsprekend dat veel communicatie nodig is gedurende de sprint. Als we wachten op input gaan we in de tussentijd verder met het volgende punt in de backlog.

3.3.3 Spoedwerkzaamheden tijdens de sprint

Wanneer tijdens een sprint spoedwerkzaamheden uitgevoerd moeten worden, gaan we hier direct mee aan de slag. Buiten de sprint worden spoedwerkzaamheden direct opgepakt als losse werkzaamheden*. Het kan voorkomen dat door spoedwerkzaamheden binnen de sprint de eerder gestelde sprintdoelen niet gehaald worden. Meestal doen we alsnog een sprintoplevering als er spoedwerkzaamheden tussen de sprint doorkwamen, maar soms zorgen de spoedwerkzaamheden ervoor dat er te weinig tijd over is voor een goede sprintoplevering. Om deze redenen vervalt de opleveringsverplichting zodra spoedwerkzaamheden in de sprint moeten plaatsvinden.

** Spoedwerkzaamheden buiten de sprint worden niet ingeschat en worden achteraf in rekening gebracht tegen het **hoge uurtarief**. Meer over spoedwerkzaamheden buiten de sprint in een volgend hoofdstuk: 'Losse werkzaamheden'.*

3.3.4 Gebruik van chatsoftware

Wij maken gebruik van chatsoftware, dit doen wij voornamelijk voor communicatie binnen het team. Daarnaast gebruiken we chatsoftware om automatische meldingen te

ontvangen vanuit monitoringsystemen. Personen buiten het team hebben meestal geen of beperkte toegang tot deze interne chatsoftware.

Om ervoor te zorgen dat ieder teamlid gefocust kan blijven werken, is het voor ons belangrijk dat sommige teamleden juist niet altijd beschikbaar zijn en hun aandacht onverdeeld kunnen besteden aan ontwikkelwerkzaamheden. Hiervoor is vaak diepe focus vereist.

3.3.5 Telefonisch overleg

Regelmatig telefonisch overleg tussen de producteigenaar en ons team tijdens de sprint is vaak goed. Overleg kan plaatsvinden met zowel met onze projectmanager als met ieder afzonderlijk teamlid. Het kan gaan over globale overkoepelende ideeën of er kan juist de details ingedoken worden om snel iets af te stemmen. Hierbij moet rekening gehouden worden met de impact die veel telefonische communicatie kan hebben op de focus van onze teamleden.

We willen vragen om telefonisch overleg over nieuwe functies met de teamleden alleen te doen tijdens een lopende sprint. Zo zorgen we ervoor dat het team zich kan concentreren op het product waar ze op dat moment mee bezig zijn. Dit staat helemaal los van het bespreken van de planning met onze projectmanager, dit kan op elk gewenst moment. Mocht er toch telefonisch overleg zijn met teamleden buiten de sprint, dan zien we dit als ondersteuning. Ook kan het team altijd telefonisch benaderd worden voor andere ondersteuning, zoals we hebben vastgelegd in de serviceovereenkomst.

3.3.6 Beperkte toegang tot versiebeheer- en CI-systemen

Tijdens ontwikkeling maken wij gebruik van versiebeheerssoftware voor het beheren van de codebase, en continuous integration software voor het automatisch controleren van wijzigingen. Af en toe wordt ons gevraagd toegang te verlenen aan een van onze opdrachtgevers zodat zij een oogje in het zeil kunnen houden, met name als zij zelf technisch onderlegd zijn. Dit begrijpen wij. Toch hebben wij dit liever niet, vooral omdat dit het zelfsturende karakter van het team ondermijnt en onnodig communicatie over de invulling van de werkwijze tot gevolg kan hebben. Het is beter als de producteigenaar zich bezighoudt met het geven van feedback op onze oplevering, alsmede het samenstellen van de (sprint) backlog, zodat ons team helemaal zelfsturend kan opereren. Als wij samenwerken met een intern software-ontwikkelingsteam bij de opdrachtgever is dit niet van toepassing.

Dit alles staat verder los van het eigendom van de ontwerpen en de code. Hierover meer in het hoofdstuk: 'Intellectueel eigendom'.

3.4 Opleveren aan het einde van de sprint

Aan het einde van de sprint doen wij een oplevering. Dit kan concreet betekenen: een testversie publiceren op een stagingomgeving, een testversie insturen bij een app store of ontwerpen aanleveren. Bij een dergelijke oplevering sturen wij altijd een bijgaand bericht met een beschrijving van de voortgang die gemaakt is. Daarbij geven we aan welke meevallers en tegenslagen er waren, wat allemaal af is gekomen, en waar extra werk nog nodig is.

3.4.1 Verbeterpunten

Tijdens een complex en omvangrijk ontwikkelingsproces is het normaal dat niet alles vlekkeloos verloopt. Het verzamelen van feedback na een oplevering is daarom juist onderdeel van het proces. Daarbij kan het gebeuren dat, ondanks onze automatische en manuele kwaliteitscontrole, een fout zit in een oplevering van een functie. Kleine spelfouten in de teksten (*copy*), bugs gerelateerd aan randgevallen (*edge cases*) of compatibiliteitsproblemen zijn soms onvermijdelijk als we snel en flexibel willen blijven ontwikkelen. Het opmerken en rapporteren van deze problemen zien we dan ook als input voor een volgende ontwikkelslag. *Meer over wat wij doen om te voorkomen dat dergelijke fouten onder de neus van echte gebruikers terechtkomen in een volgend hoofdstuk over 'Quality assurance'.*

3.4.2 Het verwerken van feedback

Bij feedback op de oplevering of het constateren van kleine problemen worden deze punten verzameld en opgenomen in de backlog. We kunnen de feedback vervolgens verwerken in een volgende sprint. Het is hierbij wederom belangrijk een prioriteitsaanduiding mee te geven en de lijst met punten te herprioriteren.

Door dit feedbackmechanisme kan soms aan meerdere dingen tegelijk gewerkt worden. Een veel voorkomende flow is:

1. In sprint 1 wordt feature A afgemaakt en aan feature B begonnen;
2. Sprint 1 wordt opgeleverd;
3. In sprint 2 wordt doorgewerkt aan feature B;
4. Wij ontvangen feedback op feature A en hier wordt verder mee gegaan in sprint 2;
5. Sprint 2 wordt opgeleverd, met verwerkte feedback van feature A en een eerste versie van feature B;
6. Et cetera...

Een feature is eigenlijk zelden na één sprint, zonder feedbackverwerking, gereed voor publicatie naar een productieomgeving.

3.4.3 Publiceren

Wij zien een publicatie naar een productieomgeving (live) **niet** als een sprintoplevering maar als losse taak, die binnen een volgende sprint uitgevoerd kan worden. Voordat een deel van de functies naar een productieomgeving kan willen we juist graag alle feedback verwerkt hebben, terwijl bij een sprintoplevering we juist vragen om opnieuw feedback te geven.

Door gebruik te maken van een goede versiebeheerworkflow is het geen probleem om eerder afgemaakte functies te publiceren, terwijl op dat moment gewerkt wordt aan nieuwe functies. Wij kunnen functies die afgemaakt zijn publiceren, zonder functies die nog in ontwikkeling zijn mee te nemen.

Na het publiceren van een nieuwe versie naar een live-omgeving is het belangrijk dat we foutlogs en crashrapportages in de gaten houden. Door onze uitgebreide kwaliteitswaarborging en feedbackrondes proberen we zo foutloos mogelijk op te leveren. Het is echter, zeker bij een veelgebruikte applicatie, onmogelijk om 100% van de gebruiksgevallen te testen. Daarom zorgt onze SLA* ervoor dat als een foutje onopgemerkt gebleven is voorafgaand aan publicatie, we deze alsnog snel kunnen opmerken na publicatie.

** Bij systemen die in productie draaien is het noodzakelijk om een serviceovereenkomst te hebben afgesloten. Dat kan zowel een BESA als een SLA zijn bij jonge softwareproducten. Bij bedrijfskritische softwareproducten adviseren wij eigenlijk enkel een SLA. Meer over serviceovereenkomsten in het hoofdstuk: 'Service, onderhoud en ondersteuning'.*

3.5 Sprintgarantie

Het komt zelden voor, maar soms is na een oplevering iets ernstigers aan de hand dan het opmerken van foutjes of het noemen van verbeterpunten. Het ontwikkelen van software is mensenwerk en soms gaat door een fout van ons wel degelijk iets mis bij een oplevering. Dat moet niet gebeuren en daarom hebben we voor deze gevallen een *sprintgarantie*. Om rekening te houden met de rol van producteigenaar in het proces, hebben we regels en leidraden over het gebruik van de sprintgarantie.

3.5.1 Geldige redenen voor aanspraak op de sprintgarantie

Als opdrachtgever heb je de mogelijkheid om aanspraak te maken op de sprintgarantie in verschillende situaties, bijvoorbeeld:

- Er is niet opgeleverd;
- Er is niet voldaan aan de overeengekomen minimale inspanning;
- Er is in de verkeerde volgorde of aan iets totaal anders gewerkt dan was overeengekomen in de sprintbacklog;
- Er is een totaal onbruikbare functie opgeleverd, ondanks dat door ons gemeld werd dat deze helemaal klaar voor gebruik was.

Hieronder volgt een korte toelichting op elk van deze situaties.

Er is niet opgeleverd

Het is kwalijk wanneer er helemaal niet is opgeleverd. Natuurlijk kan er dan aanspraak gemaakt worden op onze sprintgarantie. Let hierbij wel op dat dit anders is dan een oplevering zonder zichtbare veranderingen aan het product zoals bij het enkel beperken van technical debt. In een dergelijk geval doen we een oplevering enkel in de vorm van een rapportage over de werkzaamheden.

Zoals eerder genoemd kan het zo zijn dat spoedwerkzaamheden ervoor zorgen dat we geen oplevering kunnen doen. In dat geval kan er geen aanspraak gemaakt worden op de sprintgarantie.

Er is niet voldaan aan de minimale inspanning

Dit komt bijna niet voor, maar als uit de opleverrapportage blijkt dat wij niet hebben voldaan aan de minimale inspanning kan aanspraak gemaakt worden op sprintgarantie.

Het zou kunnen gebeuren dat dit kwam doordat feestdagen per administratieve fout niet waren meegenomen bij het verminderen van de minimale inspanningsverplichting. Als dit gebeurt, kan er geen gebruik worden gemaakt van de sprintgarantie, maar corrigeren we de rekening naar een lagere sprintprijs.

Er is in de verkeerde volgorde of aan iets totaal anders gewerkt dan was overeengekomen in de sprintbacklog

De afspraak is dat wij aan taken werken in een, door de producteigenaar bepaalde, volgorde. Wij geven hierover wel advies, maar uiteindelijk heeft de producteigenaar het laatste woord. Mochten wij duidelijk deze afspraak schenden dan kan aanspraak gemaakt worden op de sprintgarantie.

Er is een totaal onbruikbare functie opgeleverd, ondanks dat door ons gemeld werd dat deze helemaal klaar voor gebruik was

Dit is de lastigste reden voor aanspraak op sprintgarantie. Soms kan het zo zijn dat wij, door een randgeval of een compatibiliteitsprobleem, niet hebben opgemerkt dat in een bepaalde situatie een functie onbruikbaar is. Meestal zien wij dergelijke foutjes als verbeterpunten die opgepakt kunnen worden in de volgende sprint.

Wanneer wij bij een oplevering melden dat een functie klaar is voor gebruik, en vervolgens blijkt tijdens controle van de producteigenaar dat deze functie helemaal niet klaar voor gebruik is, dan kan aanspraak gemaakt worden op de sprintgarantie. Hierbij zien wij een functie als totaal onbruikbaar indien:

- De fout consistent reproduceerbaar is met een eenvoudig stappenplan;

- De fout hardware- en platformonafhankelijk is, en daarmee dus ook reproduceerbaar is op verschillende typen hardware en verschillende platformen;
- De fout duidelijk gaat over iets dat elk gebruik van de functie treft;
- De fout niets te maken heeft met performanceproblemen; en
- De fout te maken heeft met een functie waaraan gewerkt is in de sprint waarvoor aanspraak wordt gemaakt op garantie.

3.5.2 Invulling van de compensatie

Als geground aanspraak gemaakt wordt op de sprintgarantie werken wij **kosteloos** door om het probleem te verhelpen. Dit doen we op het eerstvolgende vrije moment van de betrokken teamleden. Het zou kunnen dat hier een paar dagen of paar weken tussen zit in verband met reeds ingeplande werkzaamheden voor andere opdrachtgevers. Het team lost eerst de openstaande problemen voor de sprintgarantie op, voordat ze verder werken met volgende geplande sprints voor het betreffende product.

Mocht het verhelpen van de problemen meer tijd in beslag nemen, dan kunnen wij over gaan tot het aanbieden van een gratis extra *garantiesprint*. Deze wordt apart ingepland, of in de plaats gezet van de eerstvolgende sprint voor het project.

Kosteloos doorwerken doen wij maximaal tot 100% van de geldende minimale inspanningsverplichting van de sprint waarvoor aanspraak wordt gemaakt op garantie. Bij een sprint met een minimale inspanningsverplichting van 30 uur spannen wij, bij een gegronde aanspraak op garantie, ons maximaal 30 uur extra kosteloos in. Is het maximumaantal uren van de kosteloze inspanning ingevuld? Dan brengen wij alle verdere werkzaamheden voor het oplossen van de problemen los in rekening tegen ons **lage uurtarief**.

Het is niet mogelijk om naar aanleiding van een *garantiesprint* of garantiewerkzaamheden opnieuw aanspraak te maken op garantie met kosteloze inspanning.

3.5.3 Verlopen mogelijkheid voor aanspraak op sprintgarantie

Aanspraak maken op sprintgarantie kan enkel voor de meest recent opgeleverde sprint, en uitsluitend binnen drie weken na de sprintoplevering. De mogelijkheid voor aanspraak op sprintgarantie voor een eerdere sprint vervalt zodra wij een nieuwe sprint opleveren.

De mogelijkheid op garantie vervalt ook als door de producteigenaar opdracht wordt gegeven tot het publiceren of live zetten van (een deel van) het werk uit de sprint, bijvoorbeeld naar een productieomgeving of applicatiewinkel.

3.6 Couulance na de laatst ingeplande sprintoplevering

Zoals besproken in een eerder hoofdstuk, is het vaak verstandig om altijd doorontwikkeling in te plannen. Zo stagneert de ontwikkeling niet. Soms zijn er juist heel logische redenen om dit niet te doen, bijvoorbeeld door een oprakend budget. Voor deze gevallen zijn we coulant bij extra los werk dat nog gedaan moet worden na de laatst ingeplande sprintoplevering, bijvoorbeeld om even de laatste feedback te verwerken en te publiceren.

Als er na de laatste oplevering nog een paar kleine wijzigingen los gedaan moeten worden om iets af te maken, dan brengen wij deze extra losse uren in rekening tegen het **lage uurtarief**. Dit geldt alleen voor kleine werkzaamheden die te maken hebben met de inhoud van de vorige sprint. Bovendien is dit alleen van toepassing als verder geen sprints ingepland zijn, en er tevens geen plan is om extra sprints in te plannen.

Is het extra werk gedaan in minder dan 30 minuten? Dan brengen we het extra werk helemaal niet in rekening.

Deze couulance geldt niet voor los werk dat niets met inhoud van de vorige sprints te maken heeft. Deze werkzaamheden worden gewoon zoals gebruikelijk verrekend tegen het **hoge uurtarief**.

4 Losse werkzaamheden

Is in de huidige en aankomende periode geen sprint ingepland? Wil je als opdrachtgever dat we iets voor je doen maar is dit niet meteen genoeg werk voor een hele sprint? Dan kunnen wij alsnog aan de slag gaan. Dit werk, dat niet gedaan wordt in een sprint noch voldoet aan onze definitie van servicewerk, noemen we: **los werk of losse werkzaamheden**.

Deze werkzaamheden worden per uur in rekening gebracht en zijn, net als bij sprints, op basis van een inspanningsverplichting. Er geldt voor losse werkzaamheden *geen* opleveringsverplichting en ook kan *geen* aanspraak gemaakt worden op garantie zoals dat kan bij sprints.

Losse werkzaamheden zijn lastiger in te plannen en daarom brengen we dit werk in rekening tegen het **hoge uurtarief**. Er wordt op de factuur afgerond naar hele uren.

4.1 Inschatten

Losse werkzaamheden worden van tevoren ingeschat. We specificeren de uren voor verschillende afzonderlijke taken niet uit, maar schatten deze in als een totaalaantal uren. We kunnen achteraf een gedetailleerde specificatie van de gemaakte uren op verzoek geven.

Bij een inschatting geven we aan hoeveel uren we verwachten dat de werkzaamheden gaan duren. Daarbij geven we ook aan hoeveel uren we eventueel kunnen uitlopen. Bij het akkoord gaan met de inschatting, gaat de opdrachtgever ook akkoord met het invullen van de aangegeven mogelijke uitloop.

Voorbeeld: Wij geven een inschatting voor het toevoegen van een nieuwe berekening aan een algoritme. We schatten dat dit waarschijnlijk rond de 6 uur zal duren, maar dat het goed is om rekening te houden met maximaal 8 uur. We werken uiteindelijk 6 uur en 52 minuten aan het probleem. We factureren vervolgens 7 uur.

4.1.1 Inschatting overschrijden

Op het moment dat we de inschatting en de uitloopinschatting overschrijden omdat we tegen een probleem zijn aangelopen, stoppen we de werkzaamheden en nemen we contact op met de producteigenaar. We leggen uit wat het probleem is en geven een nieuwe inschatting voor het afronden van de werkzaamheden. Dit doen we bij voorkeur telefonisch zodat we snel verder kunnen.

Als de uitbreiding van de inschatting afgewezen wordt, stoppen we met de ontwikkeling en doen we geen oplevering. Het uitgevoerde werk wordt dan in rekening gebracht. Het werk kan later weer opgepakt worden.

4.1.2 Inschatting aanpassen bij opleveren voor feedback

We geven bij de oplevering aan hoeveel uren van de originele inschatting gebruikt zijn. Het kan gebeuren dat we klaar zijn en om feedback vragen, terwijl alle ingeschatte uren al (bijna) helemaal zijn ingevuld. We geven dan aan extra tijd nodig te hebben voor het verwerken van de eventuele feedback. We stemmen dit meestal snel af over de telefoon, zodat de opdrachtgever en wijzelf niet op elkaar wachten.

Zelfs als geen feedback terugkomt, kan het zijn dat er nog extra tijd nodig is voor een publicatie van de functies naar een productieomgeving. Ook als dit nodig is laten we dat weten.

4.2 Opleveren

Een opleveringen van los werk is in veel opzichten hetzelfde als een sprintoplevering. Een van de verschillen is dat de oplevering niet aan het einde van de sprint plaatsvindt, maar juist midden in de week.

We doen eerst een oplevering waarbij we vragen om feedback. Dit betekent concreet dat bijvoorbeeld een nieuwe versie van het softwareproduct op een stagingomgeving gezet wordt, een drafrapport verstuurd wordt, een testversie van een app gedistribueerd wordt of een concept ontwerp gepresenteerd wordt.

Wanneer we na (eventueel meerdere) feedbackcycli de opdracht krijgen om te publiceren, zetten we het werk online.

4.2.1 Feedback

Net als bij sprints, is ook bij los werk de producteigenaar de persoon die de samenstelling en de kwaliteit van het product bewaakt. Daarom vragen we de producteigenaar om feedback bij een oplevering.

Soms moet er nog iets aangepast worden na een oplevering. Daarbij kan het gebeuren dat, ondanks onze automatische en manuele kwaliteitscontrole, een fout zit in een oplevering van een functie. Kleine spelfouten in de teksten (*copy*), bugs gerelateerd aan randgevallen (*edge cases*) of compatibiliteitsproblemen zijn soms onvermijdelijk als we snel en flexibel willen blijven ontwikkelen. Zodra we de feedback binnen krijgen, gaan we dit verwerken. Het kan zijn dat we pas het juiste resultaat hebben na meerdere cycli.

Zoals aangegeven in het hoofdstuk '*Doorlopende planning en dagindeling*', is het juist bij los werk belangrijk om tijdig feedback te geven. Het zou kunnen dat we de verwerking door moeten schuiven wanneer we de feedback te laat ontvangen, vaak in verband met ander ingepland werk.

4.2.2 Publiceren

Nadat we samen besluiten dat het losse werk is afgerond, gaan we over tot publicatie naar een live-omgeving of een applicatiewinkel. In tegenstelling tot sprintopleveringen, zien we deze taak als onderdeel van het losse werk.

Bij omvangrijke systemen kan het live zetten in sommige gevallen extra tijd in beslag nemen. Als er weinig ruimte is binnen de laatste inschatting, geven we aan extra tijd nodig te hebben.

Na het publiceren van een nieuwe versie is het belangrijk dat we foutlogs en crashrapportages in de gaten houden. Middels onze uitgebreide kwaliteitswaarborging en feedbackrondes proberen we zo foutloos mogelijk op te leveren. Het is alleen, zeker bij een veelgebruikte applicatie, onmogelijk om 100% van de gebruikgevallen te testen. Daarom zorgt onze SLA ervoor dat onopgemerkte foutjes voor de publicatie, alsnog snel opgemerkt worden na publicatie via monitoring*.

Als spoedwerkzaamheden voortkomen uit een publicatie naar productie gelden deze als nieuwe losse spoedwerkzaamheden. In de volgende paragraaf meer over losse spoedwerkzaamheden.

** Bij systemen die in productie draaien raden wij met klem een serviceovereenkomst af te sluiten. Dat kan zowel een BESA als een SLA zijn bij jonge softwareproducten. Bij bedrijfskritische producten raden we enkel een SLA aan. Meer over serviceovereenkomsten in een volgend hoofdstuk.*

4.3 Spoed

Is er iets dringends aan de hand? Geen probleem, laat het ons meteen weten en wij kunnen het snel oppakken. Dit geldt voor al het spoedwerk, dus ook voor spoedwerk dat niet onder een eventueel afgesloten SLA valt. Bij spoedwerkzaamheden maken we *geen inschattingen* maar **houden we de gewerkte tijd bij**, deze brengen we achteraf in rekening. Als een sprint loopt, kunnen we spoedwerkzaamheden oppakken binnen de sprint.

Sommige spoedwerkzaamheden vallen onder onze SLA en kunnen gezien worden als servicewerk. Indien een SLA is afgesloten wordt dergelijk werk verrekend met het openstaande servicetegoed. Staat er geen tegoed meer open? Dan wordt het werk in rekening gebracht tegen het **lage uurtarief**. Spoedwerk buiten de sprint en buiten het servicewerk van een SLA wordt in rekening gebracht tegen het **hoge uurtarief**.

Wij geven geen expliciete garantie op het oppakken van spoedwerk buiten de werkzaamheden die vallen onder onze SLA. Wel doen we altijd ons uiterste best spoedwerk snel en adequaat op te pakken – wat de situatie ook is.

4.3.1 Wat wordt beschouwd als spoed?

Indien wij een e-mail, telefoontje of bericht ontvangen waarin spoedwerk van ons verlangd wordt, gaan we dit meteen oppakken. Dit kan bijvoorbeeld door expliciet te melden dat het om spoed gaat, of dat impliciet de hoge urgentie duidelijk is in het bericht. We bevestigen dan dat we de werkzaamheden met spoed oppakken.

Soms merken wij zelf problemen op een productieomgeving, hierbij kunnen wij ook zelf besluiten om spoedwerk te starten. Dit doen wij eigenlijk alleen als een SLA is overeengekomen.

Een voorbeeld: Wij zien dat servers met DDOS worden aangevallen. We sturen de producteigenaar dan een kort berichtje waarin we uitleggen dat we problemen zien en met spoed aan de slag gaan. We wachten daarna niet op bevestiging en starten meteen.

4.4 Losse werkzaamheden en service

Om softwareproducten in productie van goede ondersteuning te voorzien, is het nodig een serviceovereenkomst te hebben. Daarbij wordt servicewerk binnen vooraf ingekochte service-uren ingevuld, en bij overschrijding achteraf verrekend. Dit servicewerk behandelen wij anders dan normaal los werk.

Het is goed om te weten dat niet al het werk als service ingevuld kan worden. Zo wordt het ontwikkelen van nieuwe functies **niet** binnen service-uren ingevuld. Dit geldt ook voor kleine wijzigingen die niet direct iets te maken hebben met een gebruiksblokkerende bug. Ook groot onderhoud en grote infrastructuurwijzigingen zijn uitgesloten. Voor al het werk dat niet onder de definitie van servicewerk valt komen we dus terug bij los werk of sprints.

Als een opdrachtgever expliciet kiest om *geen* serviceovereenkomst af te sluiten, ondanks dat er toch een softwareproduct in productie is, doen wij ons servicewerk reactief. We houden gewerkte tijd bij en deze uren worden achteraf in rekening gebracht tegen het **hoge uurtarief**. Voor klein servicewerk zoals het beantwoorden van een vraag, of het aanpassen van gegevens in onze systemen sturen we geen inschatting, maar houden we de tijd bij en gaan we direct aan de slag. Groter servicewerk behandelen we in dit geval wel als normaal los werk. Hiervoor komen we met een inschatting.

Ook als er geen serviceovereenkomst is wordt een maandelijkse factuur verstuurd voor bijgehouden losse servicewerkzaamheden en de inkoop van diensten zoals hosting en tooling.

Meer over servicewerkzaamheden en de exacte definitie van servicewerk in een volgend hoofdstuk: 'Service, onderhoud en ondersteuning'.

4.5 Cou lance bij klein los werk

Moet er klein los werk uitgevoerd worden en kan dit werk niet gezien worden als servicewerk? Dan zijn we coulant als dit losse werk echt klein werk is. We zien los werk als klein als we er minder dan 30 minuten mee bezig verwachten te zijn. We geven dit aan zonder uitgebreide inschatting, gaan meteen aan de slag en houden de tijd bij.

We brengen klein los werk niet in rekening als minder dan 30 minuten aan los werk is uitgevoerd in een facturatieperiode. We willen voorkomen dat bij een paar kleine verzoekjes direct een factuur gestuurd wordt. Mocht er meer dan 30 minuten gewerkt zijn, dan wordt ook klein los werk in rekening gebracht tegen het **hoge uurtarief**.

5 Doorlopende planning en dagindeling

Om een succesvol product te kunnen ontwikkelen moet er constant een duidelijke korte- en langetermijnplanning zijn. Door de complexe aard van softwareontwikkeling verandert de planningsinhoud voortdurend. Dit is juist goed, omdat we zo continu inspelen op nieuwe technische en zakelijke inzichten.

Een planning heeft twee aspecten: 'Wie werkt wanneer?' en 'Wat gaan we doen?'. In het hoofdstuk '*een agile ontwikkelproces: werken in sprints*' zijn sprints en de sprintbacklog behandeld, die de kortetermijnplanning bepalen. In dit hoofdstuk gaan we in op overkoepelende milestones, die invulling geven aan de langetermijnplanning. Ook vertellen we alles over inplannen en gaan we in op de vraag: 'Wie werkt wanneer?'. Tot slot bespreken we hoe onze specialisten dagelijks te werk gaan en hoe zij productontwikkeling goed kunnen combineren met het verlenen van service.

5.1 Inschatten van nieuw productontwikkelingswerk

Bij het inschatten van nieuw productontwikkelingswerk doen wij vooraf een voorstel. Bij nieuwe opdrachtgevers is dit vaak in de vorm van een eerste offerte, later doen we dit meestal in de vorm van sprintakkoorden.

Na het eerste contact en één of meerdere gesprekken komen wij met een voorstel in een eerste offerte. Deze offerte omvat meestal een kick-offmeeting en een grove inschatting voor het aantal sprints nodig voor het behalen van een aantal eerste milestones. De laatste van deze milestones is vaak de eerste lancering van het product. Het is belangrijk dat er ook na lancering met behulp van gebruikersinput doorontwikkeld wordt zodat een product goed aansluit bij de wensen van de eindgebruikers.

Bij het bepalen van een volgende productontwikkelingsstap werken we vaak met sprintakkoorden. Dit doen we voor zowel een groot aantal sprints per keer of slechts voor enkele sprints. Bij een groot aantal sprints definiëren we meteen een nieuwe milestone. Bij een klein aantal werken we meestal door aan de backlog.

Bij het opstellen van een sprintakkoord bekijken we vooraf samen de huidige backlog en passen deze eventueel aan. Vervolgens kijken we welke specialisten en hoeveel sprints hiervoor nodig zijn.

5.2 Milestones

Een milestone is een overkoepelend doel dat wij willen behalen voor een vastgestelde datum. Bij doorlopende ontwikkeling definiëren wij meestal één milestone per **vier tot tien weken**. Hierbinnen wordt in sprints gewerkt. De sprints hebben, in tegenstelling tot milestones, een doorlopend en iteratief karakter. Bij het samenstellen van de (sprint) backlog is het doel van de milestone leidend.

Voorbeelden van milestonedoelen: het product gereed maken voor een eerste presentatie op een beurs of een grote lancering voor het nieuwe seizoen.

Het doel van een milestone hangt samen met het werk dat we in de sprints uit gaan voeren. Naast het overkoepelende doel prioriteren en beschrijven we ook de verschillende aspecten en productonderdelen waaraan wordt gewerkt. We koppelen de milestone echter niet direct aan kleine individuele functies, omdat daar juist vaak in geschoven moet worden. Een beslissing om te schuiven in de planning wordt meestal genomen op basis van het milestonedoel. Het behalen van het uiteindelijke milestonedoel is daarbij belangrijker dan de toevoeging van die ene laatste extra functie.

5.2.1 Bijsturen

Als het milestonedoel door omstandigheden in gevaar komt moeten we bijsturen. Dat kan gebeuren als we denken niet alle noodzakelijke wensen te kunnen verwezenlijken in de nog ingeplande sprints. In dat geval kunnen drie maatregelen nemen.

We kunnen accepteren dat bepaalde onderdelen niet meer afgerond worden binnen de milestone. We slanken het doel dan af. Dit is vaak een goede oplossing. Meestal is het behalen van een het overkoepelende doel niet afhankelijk van individuele backlogpunten. Aangezien we door ons proces eerst aan de belangrijkste items werken is het vaak minder erg om minder belangrijke items door te schuiven.

Soms kiezen we ervoor om het team tijdelijk uit te breiden. We kunnen zo bij een tegenslag nog steeds het volledige milestonedoel behalen. Dit brengt wel extra kosten met zich mee. Bij sprints wordt de omvang van een sprint uitgebreid en daarmee ook de minimale inspanningsverplichting en de prijs.

Als laatste kunnen we de datum van de milestone doorschuiven en in extra sprints doorwerken om zo het doel alsnog te behalen. Ook dit brengt extra kosten met zich mee. Als besloten wordt de datum door te schuiven, worden logischerwijs toekomstige milestones eveneens doorgeschoven.

5.3 Sprints reserveren en inplannen

In de praktijk doorlopen sprints in het planningsproces drie stadia voordat ze zijn ingepland: mogelijkheid, reservering en definitief ingepland. Continue communicatie is belangrijk zodat wij de juiste specialisten voor het softwareproduct beschikbaar hebben op de momenten dat de opdrachtgever ze nodig heeft. Daarbij is het dus voor zowel onze opdrachtgevers als onszelf belangrijk tijdig akkoorden te ontvangen.

A. Sprintmogelijkheid (onzeker)

Wanneer wij samen de opdrachtgever plannen bespreken gaan wij intern rekening houden met eventuele sprintwerkzaamheden die daaruit voortvloeien. We kijken welke

specialisten hiervoor ingezet kunnen worden en in welke aanstaande weken hier ruimte voor is.

Deze mogelijkheden geven wij dan globaal aan de opdrachtgever door en we markeren ze intern als sprintmogelijkheid. Dit kan gedaan worden voor een specifieke specialist in een bepaalde week of voor een langere periode waarin een aantal sprints gedaan kunnen worden door verschillende specialisten.

Je kan als opdrachtgever er niet van uitgaan dat een sprintmogelijkheid ook leidt tot een ingeplande sprint. Als we niets horen, plannen we namelijk ander werk over de mogelijkheden heen. Als je als opdrachtgever geïnteresseerd bent, laat dit dan meteen weten. We zetten de mogelijkheden dan om in een reservering.

B. Sprintreservering (vrij zeker)

Wij reserveren werkweken van specialisten als een opdrachtgever aangeeft serieus te overwegen sprints uit te laten voeren. Een reservering is nog geen definitief akkoord maar wel een intentieverklaring.

We kunnen nog een week schuiven tussen onze verschillende specialisten. Als de week nadert en er is nog geen definitief akkoord **zes weken** voordat de sprint zou aanvangen, behouden wij het recht om ander werk over de reservering heen te plannen en eventueel de originele planning naar achter te schuiven.

C. Sprint is ingepland (definitief)

Als de opdrachtgever expliciet akkoord geeft op een plan, bijvoorbeeld door een offerte of een sprintakkoord te ondertekenen, plannen wij onze specialisten definitief in. Er wordt dan niet meer geschoven zonder overleg. Met het ondertekenen van een dergelijk akkoord stemt de opdrachtgever in principe ook meteen in met de uiteindelijke betaling.

Het is na het ondertekenen van een akkoord soms in overleg mogelijk om de planning te verschuiven. Het is soms ook nog mogelijk om geplande sprints te annuleren mits ze **minimaal zes weken** in de toekomst liggen en niet eerder verschoven zijn.

5.4 Akkoord gaan

Bij de eerste offerte vragen wij vaak of de opdrachtgever een handtekening wil zetten op een offerte. Een akkoord moet altijd gegeven worden door een persoon die volledige tekenbevoegdheid heeft.

Met de handtekening wordt direct akkoord gegaan met onze voorwaarden en met de werkwijze beschreven in dit document. Wij mogen wijzigingen aanbrengen in onze algemene voorwaarden en onze werkwijze. Dit kondigen we dan ruim van tevoren aan. Bij het wijzigen van onze voorwaarden heeft de opdrachtgever altijd de mogelijkheid om alle overeenkomsten per ingaan van de wijziging te ontbinden.

Nadat een opdrachtgever akkoord is gegaan met de initiële offerte is het voor ons voldoende dat akkoord gegeven wordt op een klein aantal sprints via eenvoudige e-mails. Daarmee is een simpele mail met: “Akkoord” vaak prima. Een e-mail wordt dan gezien als officieel akkoord. Bij een initieel akkoord of een akkoord voor een grote hoeveelheid sprints vragen we daarnaast soms ook om een gescand document met handtekening.

5.5 Overwegingen en valkuilen

Het is goed om altijd een paar zaken in het achterhoofd te houden als het gaat om sprintplanning. We willen opdrachtgevers met de volgende suggesties graag behoeden voor een paar bekende valkuilen.

5.5.1 Sprintplanning voorbij akkoord

Zoals we eerder noemden in het hoofdstuk ‘*Uitgangspunten*’ is het voor ons belangrijk om op de hoogte te zijn van alle plannen betreffende het product dat wij ontwikkelen. Wij willen hier bijvoorbeeld graag rekening mee houden als wij onze overkoepelende werkplanning indelen.

Als er bijvoorbeeld werk ingepland is om toe te werken naar een milestone, is het goed om alvast werkweken te reserveren voor een daaropvolgende milestone. Het gebeurt af en toe dat overleg hierover afgehouden wordt tot na de oplevering van een milestone. Dit kan leiden tot onnodige vertraging.

Zelfs vage plannen die pas gaan spelen over een jaar zijn goed om te bespreken. Zo kunnen wij alvast gaan kijken naar sprintmogelijkheden op die termijn. Natuurlijk kan een opdrachtgever sprints reserveren voor de lange termijn zonder dat hij/zij daarbij direct grote risico’s loopt – er kan immers geannuleerd worden tot 6 weken voor aanvang van de sprints.

5.5.2 Volgepland?

Het gebeurt regelmatig dat we de aankomende **zes tot twaalf weken** helemaal volgepland zijn voor sprintwerkzaamheden. Onze projectmanagers proberen opdrachtgevers proactief te informeren over een volle planning. Toch is het goed om **twaalf weken** in overweging te nemen als je nadenkt over de termijn waarop nieuw werk kan plaatsvinden.

5.5.3 Ingeplande losse werkzaamheden

Meestal werkt een specialist van Label305 aan los werk voor verschillende opdrachtgevers in één week. Losse werkzaamheden plannen wij over het algemeen niet in naast sprintwerkzaamheden. Het is verstandig om hier rekening mee te houden bij het geven van feedback, zodat wij los werk tijdig kunnen afronden in dezelfde week.

Het is meestal zo dat de opvolgende week namelijk weer gebruikt wordt voor sprintwerkzaamheden van andere opdrachtgevers.

5.6 Dagindeling en productiviteit

Wij kiezen ervoor om onze specialisten wekelijks 30 uur aan productontwikkeling te laten werken. Het is voor al onze opdrachtgevers belangrijk dat alle projectinhoudelijke kennis van onze specialisten beschikbaar is voor bijvoorbeeld ondersteuning en het oplossen van acute problemen. Daarom zorgen wij ervoor dat iedereen een periode per dag tijd heeft voor service en ondersteuning. We stimuleren dat onze specialisten dit altijd op in één vast tijdsblok in de dag oppakken. Zo worden de projectwerkzaamheden niet verstoord.

Ook stimuleren wij dat iedereen voor elkaar klaarstaat en elkaar helpt, juist bij productontwikkeling. Zo kunnen opdrachtgevers profiteren van alle ervaring die bij ons aanwezig is, ook van specialisten die niet direct bij hun project betrokken zijn. Een belangrijk onderdeel hiervan is het reviewen van elkaars code. Hier neemt iedereen dagelijks kort de tijd voor*, zodat iedereen continu leert van elkaar. Ook hiervoor geldt weer dat we stimuleren dit te doen op een vast moment en in één tijdsblok.

Naast dit alles moeten wij ook tijd vrijhouden voor interne besprekingen en activiteiten. Zo blijft uiteindelijk gemiddeld dagelijks 6 uur over voor echt inhoudelijk projectwerk.

** Tijd die andere specialisten nemen om het projectteam bij te staan, door bijvoorbeeld code te reviewen of samen naar een probleem te kijken, telt in principe mee voor de minimale inspanningsverplichting bij sprints, en wordt in principe op gebruikelijke wijze in rekening gebracht bij los werk en service.*

5.6.1 Flexibele kantoortijden

Label305 hanteert flexibele kantoortijden. Alle werkende specialisten zijn sowieso aanwezig tussen 10:00 en 16:00 en zij zijn in principe elke dag 8,5 uur aanwezig, waarvan 30 minuten beschikbaar is voor lunchpauze. Het kantoor opent op zijn vroegst om 7:30 en sluit op zijn laatst om 18:30.

Hiermee proberen we iedereen bij Label305 flexibiliteit te geven terwijl toch als team bij elkaar gewerkt wordt. We leggen de verantwoordelijkheid voor het goed indelen van een werkdag bij iedereen zelf neer – want iedereen is anders. Vanzelfsprekend zijn al onze specialisten op werkdagen beschikbaar voor afspraken met opdrachtgevers en anderen, ook als dit eerder is dan 10:00 of later is dan 16:00.

Onze kantoren zijn gesloten in het weekend, op alle officiële feestdagen en op oudjaarsdag. Daarnaast kan ons kantoor gesloten zijn op goede vrijdag en op bevrijdingsdag.

6 Quality assurance

Bij Label305 staat de eindgebruiker voorop. De applicaties die wij maken moeten allereerst kwaliteit hebben in de ogen van de eindgebruiker. Ook technische kwaliteit is belangrijk. Dit is iets dat de gebruiker niet altijd meteen kan zien. Technische kwaliteit zorgt ervoor dat we in de toekomst fijne en soepele doorontwikkeling kunnen garanderen. Natuurlijk is 'kwaliteit' een diffuse en ongreepbare term op zich. Daarom hier eerst een duidelijke definitie wat kwaliteit voor ons betekent en hoe we kwaliteit verwezenlijken in de producten die we ontwikkelen.

6.1 Fundering van een kwalitatieve gebruikerservaring

Voordat gesproken wordt over de functies en de look-and-feel van een softwareproduct moeten we aandacht besteden aan goede kwaliteitsfundering. Een softwareproduct moet veilig zijn en de privacy van zijn gebruikers waarborgen. Dit is niet alleen fatsoenlijk maar staat ook in de Nederlandse en Europese wet verankerd.

6.1.1 Privacy en PbD

Om de privacy van onze gebruikers te waarborgen ontwerpen wij softwareproducten zo dat we alleen de data verzamelen die nodig is voor het gebruik. Bij elk extra stukje data dat we willen opslaan denken we na: "Hebben we dit stukje data echt nodig voor een beter product?" Dit geldt voor data die gebruikers invoeren, en ook voor gegevens die we genereren tijdens het gebruik van het softwareproduct.

We houden ook rekening met het betrekken van derde partijen voor het verbeteren van het softwareproduct. We hebben een gezonde scepsis naar deze partijen. Zo hebben we bijvoorbeeld geen enkel probleem met het gebruik van cloudproviders, aangezien de wijze van dataverwerking ervoor zorgt dat deze providers zelf niets met de gegevens van onze gebruikers zouden kunnen. We blijven weg van gratis dienstverlening van derde partijen waarbij duidelijk betaald wordt met gebruikersdata.

Om dit alles mee te nemen vanaf de eerste stap houden we in het hele proces de ontwerpfilosofie *privacy by design* (PbD) aan. Het aanhouden van PbD is niet alleen goed voor eindgebruikers maar ook **verplicht** volgens Europese privacywetgeving. Dit raamwerk stelt dat onze opdrachtgever en wijzelf de volgende beginselen moeten volgen bij alle ontwerpbeslissingen:

1. Privacy moet proactief zijn, niet reactief en moet anticiperen op privacyproblemen voordat deze de gebruiker bereiken. Privacy moet ook preventief zijn en niet corrigerend.
2. Privacy moet de standaardinstelling zijn. De gebruiker hoeft geen actie te ondernemen om zijn privacy te beveiligen, en toestemming van de gebruiker voor het delen van gegevens moet niet worden aangenomen.
3. Privacy moet worden ingebed in het ontwerp. Het moet een kernfunctie van het product of de dienst zijn, geen toevoeging achteraf.

4. Privacy moet een win-winsituatie zijn en dichotomieën vermijden. PbD ziet bijvoorbeeld een haalbare balans tussen privacy en beveiliging, niet een zero-sum spel van privacy of veiligheid.
5. Privacy moet end-to-end lifecycle-bescherming van gebruikersgegevens bieden. Dit houdt in dat we ons bezighouden met correcte gegevensminimalisatie, retentie en verwijderingsprocessen.
6. Privacystandaarden moeten zichtbaar, transparant, open, gedocumenteerd en onafhankelijk controleerbaar zijn. Processen moeten, met andere woorden, bestand zijn tegen externe controle.
7. Privacy moet user-centric zijn. Dit betekent dat gebruikers granulaire privacy-opties, gemaximaliseerde privacy-standaardinstellingen, gedetailleerde privacy-informatieberichten, gebruiksvriendelijke opties en duidelijke kennisgeving van wijzigingen krijgen.

6.1.2 Veiligheid

Gebruikers moeten de veiligheid van een product kunnen vertrouwen. Concreet betekent dit dat we goed na moeten denken over toegangslagen in de applicatie-infrastructuur, over de toepassing van encryptie op verschillende niveaus, over de wijze van authenticatie en sessiebehoud, et cetera. Bovendien moeten we voorkomen dat veiligheidslekken ontstaan en dat alle software die we gebruiken up-to-date is. We denken tijdens het ontwerpen met de gebruiker mee, zodat het product goede, veilige beslissingen van gebruikers aanspoort en in de hand werkt.

6.1.3 Datamobiliteit

Gegevens die gebruikers invoeren en genereren terwijl ze een softwareproduct gebruiken, behoren toe aan die gebruikers. Wij geloven in producten die gebruikt blijven worden vanwege hun superieure gebruikerservaring, en niet omdat de data er nou eenmaal in vast zit. We willen user lock-in voorkomen en onze gebruikers alle vrijheid geven. Daarom geven we gebruikers de mogelijkheid hun gegevens te exporteren of te verwijderen.

Wij moeten al rekening houden met datamobiliteit bij de eerste ontwikkelfases. De wijze waarop we data structureren en opslaan moet zo ingericht worden dat deze data vanaf één plek toegankelijk is, en bovendien gemakkelijk te verwijderen is zonder dat ergens verwijzingen achterblijven. Dit betekent niet altijd dat vanaf het eerste moment een automatisch exportfunctie of verwijderfunctie in het product hoeft te zitten, maar wel dat de applicatie-architectuur deze functies volledig mogelijk maakt.

Meer over de methode voor het waarborgen van een goede kwaliteitsfundering in het hoofdstuk: 'Privacy en veiligheid'.

6.2 Kwaliteitsprincipes

Wij streven bij het ontwikkelen van softwareproducten onze kwaliteitsprincipes na. Dit is een belangrijke reden voor de kwaliteit van onze softwareproducten. We hebben ook een aantal methoden waarmee we kwaliteitscontrole uitvoeren en deze principes toetsen. Bovenop een stevige *fundering voor een kwalitatieve gebruikerservaring* vinden we dat product van hoge kwaliteit:

- Uitnodigend en moeiteloos is;
- Vloeiend en vlot aanvoelt;
- Strak, gebalanceerd en duidelijk oogt;
- Geen onnodige versieringen en functies heeft;
- Geen onnodige technische complexiteit bevat; en
- Geen bugs en onverwacht gedrag vertoont.

Hieronder gaan we in op elk van deze principes, en brengen nuance aan waar nodig.

6.2.1 Uitnodigend en moeiteloos

Een softwareproduct moet uitnodigend en vriendelijk zijn. Zo moet een gebruiker niet afgeschrikt worden door een overvloed aan knopjes en functies. Eenvoudige functies van het product moeten voor zichzelf spreken. Er zou hiervoor geen documentatie nodig moeten zijn. Bij ingewikkeldere krachtige functies kunnen wij deze geleidelijk introduceren aan gebruikers bijvoorbeeld via een introducerende presentatie direct in de interface (*onboarding*).

Ook een power user moet gemakkelijk verschillende acties in het systeem uit kunnen voeren zonder te veel moeite. Zo kan hij/zij snel voor elkaar krijgen wat er moet gebeuren. Voorbeelden van het faciliteren van power users zijn: het verminderen van het aantal klikken dat nodig is voor een actie, het faciliteren van snelle acties met sneltoetsen of speciale gestures, maar ook het mogelijk maken om een actie uit te voeren op verschillende elementen in één keer. Alle gegevens en functies moeten gemakkelijk te vinden zijn, bijvoorbeeld met eenvoudige zoekfuncties.

De werking van functies in een softwareproduct moet niet afwijken van wat gebruikers verwachten op het platform dat ze gebruiken. Zo is het goed om platformidiomen te gebruiken en over te nemen bij het ontwerpen van nieuwe interfaces en functies.

Concepten en datastructuren die een applicatie moet introduceren aan de gebruikers moeten zo eenvoudig mogelijk zijn. Het gebruik van de juiste metaforen is hierbij uitermate belangrijk. Een goed gegevensmodel is zo eenvoudig dat het aan elke gebruiker uit te leggen is.

Voorbeeld: Zo is het voor iedereen duidelijk dat een e-mailapplicatie verschillende folders kan hebben. Elke folder kan verschillende e-mailberichten bevatten. Een e-mailbericht heeft altijd één afzender, maar kan verschillende ontvangers hebben.

Dit zijn eigenlijk allemaal gemakkelijke metaforen voor fysieke brieven in de echte wereld, deze metaforen zijn door nieuwe gebruikers meteen goed te begrijpen.

6.2.2 Vloeiend en vlot

Een softwareproduct moet vloeiend en vlot aanvoelen. Zo zou de gebruiker niet onnodig hoeven te wachten op het laden van gegevens of het afspelen van een lange onnodige animatie. Acties in het systeem moeten efficiënt genoeg zijn dat power users snel kunnen werken, zelfs met grote hoeveelheden gegevens.

Een vloeiende interface schokt niet en gebruikt de juiste animaties op plekken waar deze wat toevoegen, de structuur verduidelijken en bijdragen aan het ontsluiten van informatie.

Gedrag van de applicatie-interface moet acties van gebruikers bevestigen en ze geruuststellen als een actie even de tijd nodig heeft.

Voorbeeld: Een bekend voorbeeld is het versturen van een chat-bericht op een smartphone. Hier toont de app een bericht als verstuurd direct na het indrukken van 'Verstuur'. In 99% van de situaties wordt het bericht ook echt verstuurd binnen een paar seconden. Alleen als dit niet lukt, door bijvoorbeeld een slechte verbinding, krijgt de gebruiker een melding. Zo kan de gebruiker direct na het sturen van het bericht zijn telefoon weggelaten zonder zich zorgen te maken.

6.2.3 Strak, gebalanceerd en duidelijk

Een applicatie-interface moet aantrekkelijk zijn. Door functies goed in te delen, elementen goed uit te lijnen, witruimte juist toe te passen en typografie te balanceren kan voor elk softwareproduct een prachtige interface gemaakt worden.

Het ontwerp en de look-and-feel van een applicatie moeten het doel van de applicatie onderschrijven, en niet enkel esthetiek nastreven. Gebruik van kleur, micro-copy (tekst), grafische elementen en raamwerken moeten allemaal in dienst staan van het verduidelijken van de werking van het product.

Een applicatie mag een exclusieve look-and-feel hebben. Zo kan de stijl met het product samenvallen en de interface onderscheidend, uniek en herkenbaar gemaakt worden.

Onderdelen van de applicatie-interface moeten herkenbaar zijn. Zo moeten interface-elementen en -idiomen die terugkeren op verschillende plekken op dezelfde wijze functioneren. Hierdoor kunnen gebruikers iets dat zij leren in een hoek van het product toepassen op alle andere plaatsen.

Termen die gebruikt worden om concepten te introduceren aan de gebruiker moeten goed uitgedacht zijn. Deze termen moeten vervolgens juist en consequent worden toegepast zodat alles herkenbaar is en blijft.

6.2.4 Geen onnodige versieringen en functies

Soms is het verleidelijk om aan een applicatie wat extra franjes toe te voegen zonder dat dit echt een positief effect heeft op de gebruikerservaring. Deze versieringen kunnen onnodige visuele elementen of animaties zijn. Soms gaat het over functies die eigenlijk niet echt gebruikt worden. Elk stukje code en elk interface-element voegt extra complexiteit toe aan het softwareproduct, daarom moeten we te allen tijde onnodige zaken proberen te vermijden.

6.2.5 Geen onnodige technische complexiteit

Bij het ontwikkelen van software gebeurt het soms onbedoeld dat onnodige technische complexiteit geïntroduceerd wordt. Het maakt problemen soms interessanter, of biedt de mogelijkheid nieuwe technieken toe te passen. We zijn waakzaam om het introduceren van dergelijke technische complexiteit te voorkomen. Alle onnodige code heeft uiteindelijk een impact op de flexibiliteit van de codebase. Elk stukje code kan bugs bevatten, elk stukje code moet onderhouden worden en elk stukje code moet perfect begrepen worden door ieder teamlid die het aanraakt. Code zonder onnodige technische complexiteit is daarom belangrijk.

Let hierbij wel op dat meer code niet altijd meer complexiteit betekent. Soms is het juist verstandig om iets meer eenvoudige en correct losgekoppelde code te schrijven dan code met een hoge dichtheid en veel kruisverbanden.

6.2.6 Geen bugs en geen onverwacht gedrag

Het klinkt vanzelfsprekend dat een softwareproduct geen bugs moet bevatten. Dit principe gaat echter dieper dan dat. We moeten namelijk code zo ontwerpen dat we voorkomen dat bugs ontstaan. Bovendien kunnen we in code zogenaamde onmogelijke state (*illegal state*) detecteren en dit op een juiste manier melden. Zo kunnen we gemakkelijk en snel bugs opsporen en vervolgens in de kiem smoren.

Hiernaast is het belangrijk dat we ervoor zorgen dat alle functies van een applicatie doen wat de gebruiker verwacht dat ze doen. Zo ziet een gebruiker niet (een onderdeel van) een functie voor een bug aan, of andersom.

6.3 Methoden

De zojuist genoemde principes proberen wij altijd zo goed mogelijk na te leven bij het ontwikkelen van een product. Om ervoor te zorgen dat dit ook echt gebeurt, hebben we een aantal methoden waarmee we nieuwe wijzigingen of het totale product toetsen aan onze principes. Methoden op zichzelf zijn geen garantie voor perfecte kwaliteit,

maar we zijn ervan overtuigd dat ze samen met onze principes leiden tot producten van hoge kwaliteit.

6.3.1 Starten met weloverwogen ontwerpen

Goede ontwerpen zijn uitermate belangrijk voor een kwalitatief product. Daarom hebben wij een ontwerpproces waarbij we de juiste aandacht geven aan elk ontwerpaspect van het te creëren product.

Bij het ontwerpen van datamodellen en interactieschema's voor een gloednieuw product nemen we vroeg in het proces de tijd om een zo eenvoudig mogelijk ontwerp te maken. Hierbij zijn we kritisch op de initiële wensenlijst voor functies. We proberen deze lijst zo ver mogelijk terug te brengen tot de kern van het product. Het terugbrengen tot de kern voorkomt veel onnodig werk later in het proces.

Ook bij het toevoegen van nieuwe functies aan een product nemen we de tijd om datamodellen en interactieschema's goed uit te werken. Dit doen we met het gehele team. Datamodellen en interactieschema's zien we vaak als een tussenproduct voor een sprintoplevering. We vragen daarna vaak feedback hierop aan de producteigenaar.

Bij het ontwerpen van applicatie-interfaces beginnen we meestal met een zogenaamd wireframe waarin interacties op detailniveau al worden ontworpen. Hierbij wordt al nagedacht over plaatsing en de verschillende toestanden van een interface.

Voorbeeld: Een foutmelding is niet altijd zichtbaar, maar er moet wel zijn nagedacht hoe en waar een foutmelding getoond wordt in een interface.

Uiteindelijk maken wij bij belangrijke interfaces een volledig uitgewerkte grafische weergave. Hier kan soms middels een interactieve mock-up doorheen geklikt worden. Zo kunnen we voorafgaand aan de implementatie al aanvoelen hoe een applicatie werkt voor de gebruiker.

6.3.2 Automatische tests

Bij het introduceren van elke functie schrijven we automatische tests. Deze tests kunnen we met een druk op de knop uitvoeren. Zo controleren we of alle functies van het systeem blijven werken na het introduceren van een verandering of toevoeging. Voordat we een nieuw stuk code accepteren worden deze tests automatisch uitgevoerd door middel van een *continuous integration* systeem. Dit zorgt ervoor dat iemand bij code review automatisch te zien krijgt of het softwareproduct nog helemaal werkt volgens de eerder geschreven tests.

Tests verdelen we onder in verschillende categorieën. Zo zijn er *unit tests* die individuele stukken code testen zonder afhankelijkheden mee te nemen. Daarnaast zijn er verschillende typen *integratietests*. Dit zijn tests die alle afhankelijkheden en

componenten van de applicatie meenemen en top-down worden uitgevoerd. Er zijn integratietests die de werking van data-ophaalcomponenten controleren en daarmee meteen kijken of de database onderwater goed wordt aangesproken. Er zijn ook high-level interfacetests waarbij daadwerkelijk als een gebruiker door de applicatie heen wordt geklikt om verschillende acties uit te voeren en te kijken of deze correct werken.

Bij het introduceren van nieuwe functies controleren we of alle tests geschreven zijn, zodat we de correcte werking van de functie in de toekomst kunnen blijven waarborgen. Hierbij moet erop gelet worden dat we de juiste tests schrijven voor de juiste functies, en niet te veel tijd kwijt zijn aan het schrijven van tests die wellicht helemaal niet belangrijk zijn. Vanzelfsprekend heeft een ingewikkelde functie meer tests nodig dan een eenvoudige functie. Het schrijven van voldoende tests kan met name bij ingewikkelde functies veel tijd in beslag nemen. Het zal altijd zo zijn dat bepaalde randgevallen niet worden afgevangen met onze automatische tests. Met tests verminderen we de kans op bugs, maar we kunnen ze niet helemaal uitsluiten.

Het ontbreken van (voldoende) tests voor een functie kan worden gezien als een belangrijke vorm van technical debt. Soms zorgen nieuwe inzichten ervoor dat tests die eerder niet nodig werden geacht, alsnog handig blijken te zijn. Ook kunnen we door het vooropstellen van kortetermijnbelangen samen kiezen om minder aandacht te besteden aan het schrijven van automatische tests. Het is verstandig om regelmatig te kijken welke tests nog toegevoegd kunnen worden zodat de technical debt verminderd en beperkt blijft.

6.3.3 Handmatig testen tijdens het ontwikkelen

Terwijl we een nieuwe functie aan ontwikkelen zijn proberen we gedurende de ontwikkeling continu de functie handmatig uit. Daarbij nemen we regelmatig een moment om de creativiteit de vrije loop te laten en onwaarschijnlijke randgevallen te bedenken. Hierbij stellen we ons sceptisch op tegenover onze eigen implementatie en proberen deze 'kapot' te maken. Elke keer dat dit lukt, lossen we het probleem op en kunnen we voorkomen dat een eindgebruiker hier tegenaan loopt. Voordat we een nieuwe functie of een codewijziging insturen voor review, testen we de functie dus zelf alsof we een echte gebruiker zijn.

6.3.4 Pair programming

Bij het implementeren van belangrijke ingewikkelde structuren, componenten en algoritmen zitten soms twee specialisten samen achter één computer om problemen samen op detailniveau op te lossen. Dit noemen we *pair programming*.

Het is bekend dat pair programming, met name bij ingewikkelde problemen, vaak sneller leidt tot een betere oplossing. Dit is vergeleken met de situatie waarin twee specialisten apart aan verschillende onderdelen werken achter hun eigen computer. We zijn ons er bewust van dat pair programming niet bij elk probleem de gewenste

snelheids- en kwaliteitswinst oplevert, en soms juist vertragend kan werken. Wij maken daarom altijd een goede afweging.

We gebruiken pair programming ook om specialisten te introduceren bij een project met een grote codebase waar ze nog niet eerder aan gewerkt hebben. Zo kan de specialist die bekend is met het project goed alle structuren uitleggen en toelichten, terwijl er toch gewerkt wordt aan een nieuwe functie of een verbetering.

Wij kunnen vanzelfsprekend alleen pair programming toepassen als er meerdere specialisten met vergelijkbare expertises aan het project werken in dezelfde sprint. Dit is soms planmatig niet mogelijk.

6.3.5 Code review

Als een specialist een wijziging of toevoeging aan het product heeft gedaan, wordt deze ingediend voor code review. De code wordt dan gelezen, gecontroleerd en verbeterd samen met andere specialisten. De reviewers toetsen de code op onze kwaliteitsprincipes. Ze letten hierbij bijvoorbeeld op fouten en onnodige complexe code. Ze doen suggesties voor een betere oplossing – zo kunnen bijvoorbeeld betere algoritmen, datastructuren en abstracties worden voorgesteld. Tot slot wordt gekeken of er voldoende automatische tests zijn toegevoegd om de werking van de functie in de toekomst te blijven garanderen.

Schrijver en reviewer zijn samen verantwoordelijk voor de aanpassingen aan de code en een uiteindelijk akkoord. We moedigen al onze specialisten dan ook aan om niet alleen kritiek te uiten, maar vooral samen toe te werken naar nog betere oplossing voordat deze geaccepteerd wordt. Het gebeurt regelmatig dat reviewer en schrijver samen na een review door de code heenlopen en direct samen bezig gaan met het verbeteren van de code.

Bij spoedprocedures wordt soepeler met code review omgegaan dan bij de ontwikkeling van nieuwe functies. Er is altijd een moment waar de code nog een keer helemaal doorgenomen wordt voordat deze samengevoegd wordt met de rest van de code van het softwareproduct. Ook worden altijd alle automatische tests gedraaid.

Bij een spoedprocedure wordt soms de code review overgeslagen om zo vertraging te minimaliseren.

6.3.6 Gebruiksreview

Bij het reviewen van een nieuwe toevoeging of wijziging wordt meestal niet alleen gekeken naar de code. Een reviewer draait ook de code op zijn eigen computer of testapparaat om te kijken hoe een echte gebruiker de nieuwe toevoeging of wijziging zal ervaren.

Ten eerste wordt hier gecontroleerd op fouten. Wederom wordt een moment genomen om creatief na te denken over onwaarschijnlijke randgevallen. Vaak leidt dit ertoe dat we toch fouten vinden die niet altijd vanzelfsprekend boven water komen tijdens een code review. Het is nog belangrijker dat een reviewer onze principes toetst als een echte gebruiker. Er worden vaak suggesties gedaan voor verbeteringen, die vervolgens samen met de schrijver uitgewerkt worden. Een frisse blik zorgt voor nieuwe inzichten. Het gebeurt regelmatig dat we met deze inzichten kleine aanpassingen doen, die uiteindelijk een groot effect hebben op de kwaliteit.

Sommige codewijzigingen raken geen functies voor de gebruiker. Daarnaast is ook de ingestuurde wijziging soms klein. In deze gevallen wordt er niet altijd een gebruiksreview gedaan.

Gebruiksreview, feedback en acceptatie door de producteigenaar

Naast onze reviewer voert ook de producteigenaar een gebruiksreview uit voor het accepteren van de wijzigingen. Vaak wordt dit gedaan als de wijzigingen al door onze code review en gebruiksreview heen zijn. Zo gaat dit vaak om code die gepubliceerd is op een stagingomgeving, zodat de producteigenaar hier gemakkelijk de code kan testen. Meestal is de gebruiksreview van de producteigenaar de laatste stap voordat we nieuwe functies en aanpassingen naar een productieomgeving publiceren.

Meestal komt uit een gebruiksreview door de producteigenaar feedback. Hierbij is het altijd aan de producteigenaar om af te wegen of hij/zij wil dat deze feedback eerst verwerkt wordt, of dat een publicatie kan plaatsvinden en de feedback later in een volgende ontwikkelslag meegenomen wordt.

6.3.7 User testing

Als nieuwe functies gepubliceerd zijn op een productieomgeving (of een stagingomgeving) kunnen wij deze gaan testen met echte gebruikers. We vragen de opdrachtgever om ons, zo af en toe, in contact te brengen met eindgebruikers. Wij kunnen dan user testing-sessies met hen in plannen. Hierbij vragen wij naar hun ervaringen met het softwareproduct en nemen hun suggesties mee voor mogelijke verbeteringen. Bij deze sessies willen wij het liefst spreken met zowel power users als beginnende gebruikers.

Bij user testing kijken we vooral hoe eindgebruikers het product gebruiken. We vragen ze om verschillende taken uit te voeren, om zo te zien waar knelpunten zitten. Ook vragen we, na het uitvoeren van de taken, naar suggesties voor verbeteringen.

De indrukken die onze specialisten opdoen bij deze user testing-sessies gebruiken we voor de verdere ontwikkeling van het product. Meestal gaan we na een user testing-sessie terug naar de tekentafel om de reeds ontwikkelde functies van het product verder te verfijnen.

De positieve impact van user testing-sessies op het product kan groot zijn en moet zeker niet onderschat worden. Wel zijn de sessies organisatorisch ingewikkeld en tijdrovend. Bovendien is er vaak een grote rol voor de producteigenaar weggelegd om deze sessies tot stand te laten komen.

6.3.8 Monitoren en direct ingrijpen (bij een SLA)

Mocht onverhoopt een bug op productie terechtkomen dan kunnen wij meteen ingrijpen. Indien een SLA is overeengekomen houden wij met monitoringsoftware continu de gezondheid van het product in de gaten. Als een grote bug boven komt drijven, kunnen we meteen ingrijpen en snel het probleem verhelpen zodat de impact voor de meeste gebruikers beperkt is.

Meer over monitoring in een volgend hoofdstuk: 'Service, onderhoud en ondersteuning'.

6.4 Beperkingen

Elk softwareproduct is anders en heeft andere gebruikers, daarom betekent kwaliteit ook bij elk product net iets anders. Wij proberen altijd een goed kwaliteitsniveau behalen middels onze principes en methoden. Hierbij moet zeker worden meegenomen dat de opdrachtgever en producteigenaar ook invloed hebben op de uiteindelijke kwaliteit. Zo is logischerwijs de grootte van het projectteam en de doorlooptijd van de milestones van invloed – in het verlengde hiervan dus ook de gestelde deadlines en het beschikbare budget. Wij proberen altijd het best mogelijke resultaat te leveren binnen deze beperkingen.

De producteigenaar bepaalt waaraan gewerkt moet worden in een sprint. We vragen ons advies in deze beslissing mee te nemen. Het streven naar hoge kwaliteit kan soms recht tegenover zakelijke belangen op de korte termijn staan. Als producteigenaar moet je deze zaken goed afwegen, terwijl wij je adviseren en behoeden voor het veronachtzamen van kwaliteit. Perfectionisme ligt ook wel eens op de loer. Wij waken hier zelf voor, maar het is ook goed om als producteigenaar hiervoor uit te kijken. Er bestaat altijd iets als 'te perfectionistisch', vooral bij zaken die wellicht uiteindelijk voor de eindgebruiker minder belangrijk zijn.

Indien het snel maken van veel nieuwe functies belangrijk is voor een beslistmoment op de korte termijn, dan zetten wij ons hiervoor in. Soms moeten we hiervoor genoegen nemen met een lagere kwaliteit. Het is altijd goed om dit in het achterhoofd te houden. Zoals we al aangaven in het hoofdstuk over 'uitgangspunten' is het in dergelijke gevallen goed om achteraf alsnog extra aandacht te besteden aan het verhogen van de kwaliteit. Dit geldt zowel voor kwaliteit die de gebruiker ervaart maar ook voor technische kwaliteit. Zo moet er regelmatig aandacht zijn voor verminderen van technical debt. Ook is het goed om regelmatig nog een keer naar de bestaande functies te kijken en ze verder te verbeteren.

Het is niet altijd een goed idee om functies die nog niet ontwikkeld zijn te verkopen aan (potentiele) klanten. Het opnemen van onontwikkelde functies in het salesproces leidt er vaak toe dat deze functies slecht doordacht en gehaast toegevoegd moeten worden aan een product in een krappe periode. Deze beloftes zijn dus soms gevaarlijk. Het is goed om hier alert op te zijn en altijd een goed beeld te hebben van de impact van het verkoopproces op de productkwaliteit. Het is wel een goed idee om in het verkoopproces input te krijgen van de wensen van (potentiele) klanten, ook voorafgaand aan de lancering van het product. Wees hierbij alert op het feit dat de inkoper en de eindgebruiker soms andere mensen zijn.

Tot slot is het belangrijkste om als producteigenaar en opdrachtgever ook goed nagedacht te hebben over wat kwaliteit voor het betreffende softwareproduct betekent. Vertel ons hier uitgebreid over zodat we altijd goed op een lijn liggen.

7 Privacy en veiligheid

In het vorige hoofdstuk spraken we al over privacy en veiligheid. Label305 zet de gebruiker voorop en dat betekent allereerst dat we de privacy en veiligheid van de gebruikers belangrijk vinden. Dit zie je terug in onze werkwijze omtrent kwaliteitswaarborging. Er zijn ook andere zaken die onze opdrachtgevers en wijzelf mee kunnen nemen voor het verhogen van de veiligheid en privacy. Daarover geven we hier in dit hoofdstuk graag een toelichting.

7.1 Responsible disclosure

Om ervoor te zorgen dat alle systemen en softwareproducten zo veilig mogelijk zijn, moedigen wij veiligheidsonderzoekers en ethische hackers aan om in te breken, mits ze vervolgens verantwoordelijk omgaan met het melden van een eventueel veiligheidsprobleem. Daarom heeft Label305 zelf een zogenaamde “Responsible disclosure policy” of ontsluitingsbeleid, en we vragen onze opdrachtgevers om dit beleid over te nemen voor hun eigen softwareproducten.

Soms worden veiligheidsonderzoekers en ethische hackers afgeschrikt door mogelijke beschuldigingen en rechtszaken. Bij afwezigheid van een responsible disclosure policy gaan ze er misschien vanuit dat ze aangeklaagd worden na het melden van een veiligheidslek. Daarom is het extra belangrijk dat we aangeven dit niet te doen, mits ze de melding netjes en volgens ons beleid doen.

Vanzelfsprekend doen wij er alles aan om de software die wij schrijven veilig te houden. Echter kunnen we de aanwezigheid van een lek nooit helemaal uitsluiten. Input van veiligheidsonderzoekers en ethische hackers kan daarom heel waardevol zijn.

7.1.1 De beloning bij een correcte melding

Het is gebruikelijk dat in een ontsluitingsbeleid ook een minimale beloning vastgesteld staat. Meestal moet je hierbij denken aan een cadeaubon van € 50,- bij kleine en relatief ongevaarlijke problemen zoals een XSS-mogelijkheid (*cross site scripting*) op een plek in de applicatie waar dit een klein gevaar vormt. Bij grotere lekken gedacht worden aan een geldbedrag van enkele honderden of duizenden euro's, denk bijvoorbeeld aan situaties waarbij een ethische hacker een weg heeft gevonden naar directe databasetoegang.

Wij kunnen helpen bij het inzicht geven in de omvang van een melding en bij het bepalen van een passende beloning bij het afhandelen van de melding. Hierbij willen we de melder goed behandelen, ze hebben ons per slot van rekening een belangrijke dienst bewezen. Het is goed om ervoor te zorgen dat we snel overgaan tot het belonen om ook zo onze waardering te uiten.

7.1.2 Publiceren van het beleid

Er zijn verschillende websites te vinden waarop we een ontsluitingsbeleid kunnen publiceren en onder de aandacht kunnen brengen. Deze websites worden gebruikt door onderzoekers om doelwitten te vinden die vriendelijk omgaan met meldingen van veiligheidslekken. We moedigen onze opdrachtgevers dan ook aan om het ontsluitingsbeleid van hun softwareproducten hierop te publiceren. We hebben vertrouwen in de veiligheid van de ontwikkelde software. Daarnaast willen we ook goed en snel gewezen worden op eventuele veiligheidsproblemen – zonder dat ervoor het melden onnodige barrières zijn opgeworpen.

7.2 Security audits

Computerveiligheidsexperts kunnen door hun vindingrijkheid en creativiteit soms langs onze veiligheidsmaatregelen komen op een wijze die we niet voor mogelijk hielden. Naast responsible disclosure is het dus ook goed om af en toe veiligheidsexperts direct in te huren en hen de veiligheid van het softwareproduct te laten controleren. Dit kan gedaan worden middels een zogenaamde *security audit*.

Het laten uitvoeren van een security audit is een goed idee vlak voor de initiële lancering, maar ook na een grote ontwikkelslag. Een goede vuistregel is om dit in ieder geval één keer per jaar te doen.

Security audits doen wij niet zelf. We hebben verschillende partners die we hiervoor kunnen aanspreken. Het is natuurlijk ook mogelijk om zelf een partij te zoeken en deze aan te dragen. We verwachten na een security audit een uitgebreid en gedetailleerd rapport waarmee wij de veiligheidsproblemen kunnen reproduceren. Voor het laten uitvoeren van een security audit moet er gedacht worden aan kosten van € 1.500 tot € 6.000 *per audit*.

Met de resultaten van een goed bevonden security audit in de hand is het vaak makkelijker om een softwareproduct te verkopen aan grote organisaties zoals overheden of beursgenoteerde bedrijven.

7.3 Privacywetgeving

In de Europese Unie is sinds mei 2018 nieuwe privacywetgeving van kracht geworden – de algemene verordening gegevensbescherming (AVG) of general data protection regulation (GDPR). Als ontwikkelaar van softwareproducten zijn wij nauw betrokken bij de naleving van deze en andere privacywetgeving. Hiernaast hebben vooral ook de onze opdrachtgevers veel verantwoordelijkheden onder de huidige wet, als aanbieder van hun softwareproduct. Hier lichten we kort een aantal aspecten toe.

7.3.1 Verantwoordelijkheden van de controller

Onze opdrachtgevers zijn in termen van de GDPR de *controller* van persoonsgegevens. De controller heeft de uiteindelijke verantwoordelijkheid over de gegevens die zijn/haar product verzamelt van eindgebruikers. Ook is de controller het directe aanspreekpunt voor eindgebruikers. Hieronder lichten we een paar verantwoordelijkheden toe. Wij zijn bij Label305 geen juridische experts dus we raden aan om je hiernaast ook goed te informeren.

Toestemming vragen gegevensverwerking

Allereerst is het noodzakelijk om alle gebruikers toestemming te vragen voor de gegevensverwerking. Het moet meteen duidelijk zijn welke gegevens verwerkt worden en waarom dit gebeurt. Zo is het belangrijk om dit te doen bij registratie van een gebruiker, maar ook als gegevens ingevoerd worden in een beheeromgeving.

Privacyverklaring

Op de website van het softwareproduct moet een privacyverklaring beschikbaar zijn waarin staat hoe de privacy van de eindgebruiker wordt gewaarborgd. Ten eerste moet de privacyverklaring beschrijven welke persoonsgegevens worden verzameld, en wat ermee gedaan wordt. In deze privacyverklaring is het ook belangrijk om alle (sub)verwerkers van persoonsgegevens te noemen, en aan te geven welke gegevens bij welke (sub)verwerkers terecht komen. Tot slot is het noodzakelijk om te melden wat precies de procedures zijn voor inkijk-, verander- en vergeetverzoeken.

Datalek

Als wij een datalek constateren in een softwareproduct nemen we meteen contact met de producteigenaar op. Het is zaak voor onze opdrachtgever om vervolgens binnen **72 uur** alle getroffen gebruikers op de hoogte te stellen. Dit is ook het geval als er een lek is dat een van de (sub)verwerkers treft. In alle gevallen moeten alle getroffen eindgebruikers zo snel en adequaat mogelijk op de hoogte worden gesteld. De volgende informatie moet worden gedeeld bij het melden van een gegevenslek:

- De kenmerken van het incident, zoals: datum en tijdstip constatering, een samenvatting van het incident, en de aard van het incident;
- Indien bekend, de oorzaak van het lek;
- De maatregelen die zijn genomen om verdere schade te voorkomen;
- De omvang van de groep betrokkenen; en
- Het soort gegevens dat door het incident wordt getroffen.

Een gegevenslek kan ook bij de opdrachtgever zelf ontstaan, doordat bijvoorbeeld onzorgvuldig met gebruikersgegevens omgegaan wordt. Dit kan bijvoorbeeld gebeuren als een spreadsheet-bestand met alle e-mailadressen van gebruikers op een onbeveiligde USB-stick staat en vervolgens deze USB-stick wordt gestolen. Het is goed om hier waakzaam voor te zijn en veiligheidsprocedures in te voeren.

7.3.2 Verwerkers

Naast de controller heeft elke softwareproduct verschillende zogenaamde *verwerkers* – ook wel *processors* genoemd. Verwerkers hebben wel sommige persoonsgegevens in handen, maar hebben geen directe relatie met de eindgebruikers, denk bijvoorbeeld aan een hostingprovider of aan ons als ontwikkelingsconsultancy. Verwerkers kunnen op hun beurt ook verwerkers hebben, deze noemen we *subverwerkers*. Het is zaak dat elke verwerker van persoonsgegevens een zogenaamde verwerkersovereenkomst heeft met de controller. De controller is hier verantwoordelijk voor.

Verwerkersovereenkomst met ons

Wij sluiten met elke opdrachtgever een verwerkersovereenkomst af zodat onze inzage in gebruikersdata geen probleem is voor de privacy en goed contractueel is afgedekt. In deze verwerkersovereenkomst staat duidelijk wat wij met de gebruikersdata kunnen doen en hoe wij de veiligheid van die gebruikersdata waarborgen.

Bovendien verwijzen wij in de overeenkomst naar onze verwerkers en subverwerkers en houden hier ook altijd een lijst van bij op onze website. Onze verwerkers zijn tools die wij intern gebruiken voor communicatie en gegevensuitwisseling. Deze tools kunnen gegevens over de projecten en opdrachtgevers verwerken, maar niet de gegevens van eindgebruikers. Denk bijvoorbeeld aan projectmanagement-, bestandsynchronisatie- en e-mailsoftware. Onze subverwerkers zijn voornamelijk tools die ook applicatiedata en gegevens van eindgebruikers verwerken, voorbeelden zijn logaggregatie- en crashrapportagesoftware. Voor de meeste projecten kan je ervan uitgaan dat wij subverwerkers beperkt inzetten, maar voor specifieke gevallen kunnen we hier aparte afspraken over maken.

De relatie met hosting- & toolingproviders

Het belangrijk om te noemen dat de meeste hosting- & toolingdienstverlening die wij direct voor onze opdrachtgevers inkopen niet vallen onder de verwerkersovereenkomst met ons.

We vragen namelijk onze opdrachtgevers om zelf met alle hostingproviders en sommige toolingproviders een verwerkersovereenkomst af te sluiten. Het heeft dan ook onze voorkeur dat onze opdrachtgevers een volledige directe klantrelatie hebben met de meeste hosting- & toolingproviders. Ook als wij de onze opdrachtgever helpen met het aanschaffen van deze diensten tijdens de ontwikkeling en kort na lancering, heeft de opdrachtgever nog steeds een directe verwerkersovereenkomst met al deze partijen nodig om alle verwerking van data goed juridisch af te dekken. De opdrachtgever draagt dan ook zelf de verantwoordelijkheid voor de communicatie met deze providers als het gaat om privacyzaken.

Typische verwerkers van persoonsgegevens, waarbij onze opdrachtgevers zelf een verwerkersovereenkomst moeten overeenkomen, zijn:

- Providers van webhosting en webservers;
- Providers van databasservers en cachingservers;
- Providers van blobopslag en CDNs;
- Providers van DDOS-beveiliging en proxies die werken met ontsleutelde data;
- Providers van voorspellende modellen en andere API die gebruikersdata consumeren,
- Providers van logbestandaggregatie- en crashrapportagesoftware (*dit wordt soms afgedekt door de verwerkersovereenkomst met Label305*);
- Providers van transactionele e-mail- en nieuwsbriefsoftware;
- Interne helpdesk en e-mailsoftware; en
- Analytics- en advertentieattributiesoftware.

7.3.3 Gegevensbeschermingseffectbeoordeling

Om ervoor te zorgen dat er altijd een goed overzicht is van alle zaken die impact hebben op de privacy van gebruikers moet een zogenaamde gegevensbeschermings-effectbeoordeling (DPIA) bijgehouden worden. Onze opdrachtgever is hier verantwoordelijk voor. Wij kunnen helpen met het beantwoorden van verschillende vragen in de DPIA. Het hebben van een up-to-date DPIA is verplicht onder de GDPR. Bij veel klanten doen we dit in onze projectmanagementomgeving, bijvoorbeeld Basecamp. Vaak is de producteigenaar ook de persoon die de DPIA bij- en actueel houdt. Hier volgen een aantal vragen die kunnen worden beantwoord in de DPIA. Deze lijst is niet per se uitputtend, en het is dus noodzakelijk om je volledig in te lezen in de GDPR zodat er een goede DPIA gemaakt kan worden voor jouw specifieke situatie.

Dataverzameling en bewaring

- Welke persoonsgegevens worden verwerkt?
- Hoe worden deze data verzameld en vastgehouden?
- Worden de gegevens lokaal opgeslagen, of ook op servers?
- Hoe lang blijven de data opgeslagen, en wanneer worden data verwijderd?
- Is het verzamelen en verwerken van de data gespecificeerd, expliciet en legitiem?
- Wat voor proces bestaat er voor het toestemming vragen om de gegevens te verwerken? Is deze toestemming expliciet en verifieerbaar?
- Wat is de basis voor de toestemming voor het verwerken van de data?
- Als geen toestemming nodig is, wat is de wettelijke rechtvaardiging voor het verwerken van de gegevens?
- Zijn alle gegevens geminimaliseerd tot wat expliciet noodzakelijk is?
- Is de data accuraat en up-to-date?
- Hoe worden eindgebruikers geïnformeerd over de gegevensverwerking?

- Welke controlemiddelen zijn er voor eindgebruikers waarmee ze zelf invloed hebben op de gegevensverwerking en -retentie?

Technische en veiligheidsmaatregelen

- Is data versleuteld? Zo ja, welke data?
- Is data geanonimiseerd of gepseudonimiseerd? Zo ja, welke data?
- Is er een back-up van de data? Wat is de retentie van deze back-up?
- Wat zijn de technische en veiligheidsmaatregelen op de fysieke hostinglocaties?

Personeel

- Wie heeft toegang tot de data?
- Welke dataveiligheidstrainingen hebben deze personen gehad?
- Welke veiligheidsmaatregelen nemen deze personen?
- Welke datalekmelding en waarschuwingsprocedures zijn er?
- Wat voor procedures zijn er voor overheidsaanvragen?

Rechten van de eindgebruiker

- Hoe kunnen eindgebruikers hun toegangsrechten uitoefenen?
- Hoe kunnen eindgebruikers hun rechten voor mobiele data uitoefenen?
- Hoe kunnen eindgebruikers hun rechten voor het verwijderen en vergeten van data uitoefenen?
- Hoe kunnen eindgebruikers hun rechten voor bezwaar en beperking uitoefenen?

Juridisch

- Zijn de verplichtingen van alle gegevensverwerkers, inclusief de subverwerkers, gedekt door een verwerkersovereenkomst?
- Worden gegevens buiten de Europese Unie verwerkt? Wat zijn de veiligheidsmaatregelen als dit gebeurt?

Risico's

- Wat voor risico's zijn er voor de eindgebruiker als hun gegevens worden misbruikt, gelekt of als er onrechtmatige toegang is tot hun gegevens?
- Wat zijn de risico's voor de eindgebruiker als data worden bewerkt?
- Wat zijn de risico's voor eindgebruikers als data worden verloren?
- Wat zijn de belangrijkste risicobronnen?
- Welke maatregelen zijn er om al deze risico's te verkleinen?

7.3.4 Dataopvraag-, verwijder- en vergeetverzoeken

Onder de GDPR hebben alle eindgebruikers het recht om hun gegevens op te vragen, te laten verwijderen en te worden vergeten. Niet altijd zijn hier volledige automatische procedures voor. Het is verplicht om gehoor te geven aan verzoeken, maar de procedures hoeven echter niet per se volledig geautomatiseerd te zijn. Komt een dergelijk verzoek binnen? Dan kan dit verzoek naar ons worden doorgezet. Wij zorgen er vervolgens voor dat alle technische onderdelen van het verzoek correct worden afgehandeld. Uiteindelijk is onze opdrachtgever wel verantwoordelijk voor alle communicatie met de eindgebruiker, daarmee ook de uiteindelijke terugkoppeling.

7.3.5 Cookies en tracking

In de context van privacywetgeving wordt vaak groots gesproken over cookies, ondanks dat dit slechts een klein component is van privacywaarborging. Vaak moet op een website aangegeven worden of men akkoord gaat met het gebruik van cookies, vanwege de specifieke Nederlandse en Europese privacywetgeving. Het is hierbij belangrijk om te realiseren dat dit slechts geldt voor specifieke cookies. Zo hoeft voor het plaatsen van cookies die het technisch gebruik van de website faciliteren geen toestemming gevraagd worden.

Het is belangrijk om te noemen in de privacyverklaring welke cookies geplaatst worden en voor welk doel elke cookie dient. We kunnen helpen met het opstellen van deze lijst met cookies. Cookies waarvoor in de regel toestemming gevraagd moet worden zijn: cookies voor analytics, advertentieattributie en cookies gebruikt voor andere marketingdoeleinden.

Verwijzingen naar derde partijen in de frontend

De cookiewetgeving bestaat om iedereen te beschermen tegen tracking over het internet. Nu zijn cookies niet de enige methode waarmee derde partijen dit kunnen doen, zo kan dit bijvoorbeeld ook door het gebruik van verwijzingen naar derde partijen in de frontend van een softwareproduct. Bijvoorbeeld het gebruik van een CDN voor het aanbieden van fonts of een specifieke JavaScript-bibliotheek. We proberen dan ook dergelijke verwijzingen zo veel mogelijk te vermijden, in de frontendcode die wij schrijven.

7.3.6 Gegevens bij derde partijen

Veel van de softwareproducten die wij ontwikkelen verwerken uiteindelijk gebruikersgegevens bij derde partijen. Dit doen we met name omdat alles in-house halen enorme kosten met zich meebrengt. Bovendien zou er een aanzienlijke impact zijn op onze ontwikkelflexibiliteit. Hieronder lichten we een paar veelvoorkomende situaties toe zodat je een idee hebt waar je aan moet denken bij gebruikersdata die verwerkt wordt door derde partijen.

DDoS-beveiliging, proxies en load balancers

Voordat een gebruiker een webapplicatie benadert, of via een mobiele app een backend aanspreekt, kan de aanvraag door verschillende machines heengaan. Dit kunnen bijvoorbeeld proxies zijn voor DDoS-bescherming, maar ook load balancers. Daarnaast wordt elk pakketje verwerkt door alle tussenliggende machines. Om gebruikersdata veilig te stellen is het belangrijk dat alle aanvragen end-to-end zijn versleuteld met een TLS-verbinding. Als we hiervoor zorgen, voorkomen we dat gebruikersgegevens verwerkt kunnen worden op plekken voordat ze de webserver ingaan.

Soms is het nodig dat een proxy bepaalde gegevens ontsleutelt. In een dergelijk geval zorgen we ervoor dat gegevens niet opgeslagen worden door een proxy en dat de proxy enkel als doorgeefluik wordt gebruikt. Het is in een dergelijk geval noodzakelijk om met deze proxydienstverlener een verwerkersovereenkomst af te sluiten.

Hosting en opslag

Diensten voor hosting en opslag zijn de meest voor de hand liggende gegevensverwerkers. Zij draaien immers de machines waarop gegevens uiteindelijk binnenkomt en wordt opgeslagen. Het is goed om een overzicht te hebben van alle onderdelen van de applicatie-infrastructuur waar de data verwerkt worden. Zo staan de meeste data vaak in een relationele database. Bijna altijd worden op de webserver caches en sessies bijgehouden van meer kortstondige data. Daarnaast worden de data uit een relationele database vaak periodiek opgeslagen in zogenaamde blobstorage-buckets, hier staat vaak een kopie van alle gegevens. Andere blobstorage zoals afbeeldingen en documenten worden ook vaak hier opgeslagen, deze bestanden kunnen persoonsgegevens bevatten.

Bij het voldoen aan vergeetverzoeken is het extra belangrijk dat gegevens niet alleen uit de database verwijderd worden, maar ook uit de back-ups en dat andere gerelateerde gegevens ook correct verwijderd worden uit blobopslag. Bij back-ups is het aanhouden van een korte retentietijd vaak een goede oplossing om dit probleem af te vangen. Mocht een back-up toch hersteld moeten worden, moeten vergeetverzoeken opnieuw verwerkt worden. De opdrachtgever is naar de gebruiker direct verantwoordelijk voor het correct verwerken van vergeetverzoeken. Als wij de opdracht krijgen een vergeetverzoek handmatig uit te voeren en binnen twee weken een back-up herstellen, voeren wij recente vergeetverzoeken wel opnieuw uit.

Analytics en advertentieattributie

Voor marketing en UX-doeleinden is het vaak een goed idee om te weten hoe een applicatie precies gebruikt wordt. Hiervoor worden vaak externe gratis tools gebruikt zoals Google Analytics. Het is belangrijk om te realiseren dat Google Analytics niets voor niets een gratis tool is. Er kan in Google Analytics afzonderlijk aangegeven worden of gegevens gedeeld worden met Google voor andere doeleinden. Daarnaast

geeft Google Analytics de mogelijkheid om gebruikers te tracken. Indien deze optie aangezet wordt, is het noodzakelijk om een cookiemelding op de website te plaatsen.

Onlineadvertenties, als die van Facebook en van Google, hebben vaak een 'pixel' voor advertentieattributie. Voor het gebruik van deze pixels is het ook noodzakelijk dat eerst toestemming is gevraagd aan de gebruiker via een cookiemelding.

Logs en crashrapportages

Voor mobiele apps, frontendonderdelen en backendonderdelen is het goed idee om centrale logs en crashrapportages bij te houden. Op die manier kunnen wij snel problemen diagnosticeren en monitoren voor plotselinge fouten in productiesystemen. We proberen te voorkomen dat in deze logs persoonsgegevens zoals IP-adressen, gebruikersnamen en e-mailadressen terecht komen. Soms kiezen we er wel voor om pseudoniemen, in de vorm van nummers, mee te sturen. Deze nummers kunnen verwijzen naar een gebruiker in onze database. We gebruiken pseudoniemen zodat wij individuele ondersteuningsvragen goed kunnen beantwoorden.

Het is onvermijdelijk dat bepaalde persoonsgegevens in een log of crashrapportage terechtkomen, zelfs als we dit proberen te minimaliseren. Daarom is het ook voor deze logs en rapportages goed om een retentietijd in te stellen, en logs en rapporten slechts vast te houden voor een beperkt aantal dagen.

Nieuwsbrieven, transactionele e-mails en pushberichten

De meeste softwareproducten kunnen e-mails naar gebruikers sturen. Bijvoorbeeld voor het ondersteunen van een 'wachtwoord vergeten'-functie. Het betrouwbaar versturen van e-mails, zodat deze aankomen in alle inboxen van gebruikers zonder te belanden in spamfolders, is een ingewikkeld proces. Daarom schakelen wij meestal transactionele e-maildiensten in om e-mails te versturen. Deze diensten verwerken de persoonsgegevens die in de e-mail staan, zoals het e-mailadres zelf, maar ook bijvoorbeeld naam en andere details in de e-mails. Om vergeetverzoeken correct te verwerken, is het ook belangrijk dat deze services een beperkte dataretentie hebben.

Nieuwsbriefsoftware is een apart geval. Deze software wordt vaak niet direct gekoppeld aan de database van het softwareproduct, maar houdt zelf e-mailadressen bij. Het is dan ook belangrijk om bij vergeetverzoeken een e-mailadres te verwijderen uit de gebruikte nieuwsbriefsoftware.

Voor het versturen van pushberichten naar mobiele applicaties gebruiken wij de diensten die Google en Apple zelf aanbieden. Deze diensten zitten zo in elkaar dat zowel Google als Apple geen gegevens verzamelen over de inhoud van de berichten. Wel is het zo dat zowel Google als Apple de inhoud van pushberichten kan lezen, en daarmee ook persoonsgegevens verwerkt.

8 Service, onderhoud en ondersteuning

Naast continue doorontwikkeling van een softwareproduct is ook het verlenen van goede service belangrijk. Wij hebben drie opties voor het afnemen van service: geen overeenkomst, een *best-effort service agreement* (BESA) en een *service level agreement* (SLA).

Serviceovereenkomst	<i>Geen</i>	BESA	SLA
Maandelijks nieuwe service-uren inkopen, met opbouwend tegoed	Geen	Min. 1 uur	Min. 4 uur
Elk kwartaal de maandelijks inkoop van service-uren bijstellen	N.v.t.	Ja	Ja
Enkel klein servicewerk, zoals het geven van ondersteuning, maandelijks bijhouden en achteraf in rekening brengen	Ja* hoge uurtarief	N.v.t.	N.v.t.
Garantie dat er servicewerk uitgevoerd kan worden	Nee	Ja tegoed	Ja tegoed & meer
Maandelijks voorschot op hosting en tooling kosten doorrekenen	Ja	Ja	Ja
Elk kwartaal het maandelijks voorschot op hosting en tooling bijstellen en een verrekening doen voor de gemaakte kosten afgelopen kwartaal	Ja	Ja	Ja
Proactief uitvoeren van klein serveronderhoud, zoals veiligheidsupdates	Nee	Ja	Ja
Monitoren van de gezondheid van de applicatie-infrastructuur, en proactief handelen bij impactvolle problemen	Nee	Nee	Ja
Monitoren van crashrapportages en de gezondheid van client-side applicaties; en hierop proactief handelen bij impactvolle problemen	Nee	Nee	Ja
Proactief kleine verbeteringen aanbrengen in het softwareproduct zelf	Nee	Nee	Ja
Proactief bijschalen van de serverinfrastructuur	Nee	Nee	Ja

Serviceovereenkomst	<i>Geen</i>	BESA	SLA
Applicatie-uptimegarantie	Nee	Nee	99.5%
Gegevensgarantie	Nee	Nee	Ja
Spoedwerk oppakken	Best effort	Best effort	Binnen 1 werkdag
Ondersteuning via de telefoon	Terugbellen	Terugbellen	Ja**
Reactie ondersteuning via e-mail of projectmanagementsysteem	Best effort	Best effort	Binnen 1 werkdag
Oppakken van een responsible disclosure melding	Best effort	Best effort	Binnen 1 werkdag
Oppakken van aanvragen tot inzage, verwijdering of aanpassing van (persoons)gegevens	Best effort	Best effort	Binnen 1 werkdag
Beschikbaarheidsgarantie van een ingelezen specialist	Nee	Nee	Ja***
Monitoren van aanvallen zoals pogingen tot verkrijgen onwenselijke applicatietoegang	Nee	Nee	Ja
Onmiddellijke melding bij een geconstateerd gegevenslek	Ja	Ja	Ja
Expliciete garantieregeling voor het naleven van de overeenkomst	Nee	Nee	Ja
Uurtarief service-uren vooraf inkopen	N.v.t.	Laag	Laag
Uurtarief extra service-uren bij onvoldoende tegoed, dit brengen we achteraf in rekening	Hoog	Hoog	Laag
Uurtarief losse werkzaamheden die <i>niet</i> voldoen aan de definitie van servicewerk	Hoog	Hoog	Hoog
Vast maandelijks tarief, naast inkoop service-tegoed, hosting en tooling	Nee	Nee	Ja

** Indien er gekozen wordt om geen serviceovereenkomst af te sluiten, houden we klein servicewerk bij en rekenen we het achteraf door tegen het hoge uurtarief. Groter servicewerk zien we als los werk en daarvoor maken we eerst een inschatting.*

*** Wanneer een SLA is afgesloten kan de opdrachtgever ons bellen tijdens kantooruren en staan we meteen klaar. Bij spoedwerk garanderen we dat we het werk snel oppakken. Als gebeld wordt voor iets dat minder haast heeft, en onze specialist is niet beschikbaar, dan bellen we later terug.*

**** Afhankelijk van de omvang van de applicatie waarvoor de SLA is opgesteld gaat dit om één specialist of meerdere. Bij een SLA met een beperkte omvang wordt de beschikbaarheids-garantie niet afgegeven tijdens vakantieperiodes van de betrokken specialisten, hier wordt de producteigenaar vooraf over geïnformeerd.*

Bij aanvang van een softwareontwikkelingstraject nemen we opties voor de verschillende serviceovereenkomsten op in de eerste offerte. Het is prima om geen actieve serviceovereenkomst te hebben voordat lancering van de eerste publieke testversie heeft plaatsgevonden. Wij nemen in offertes op dat, zodra dit gebeurt, we aanvangen met een **BESA**. Tijdens een publieke testfase of vroege productiefase zijn de garanties van de SLA soms nog niet nodig.

We raden aan om op een **SLA** over te stappen zodra een applicatie kritiek is geworden voor de bedrijfsvoering of voor de productpropositie. Bijvoorbeeld als een aanzienlijk deel van de omzet afhankelijk is van een soepel draaiende applicatie, of dat een verrijkende applicatie essentieel is geworden voor de verkoop van het bijhorende (fysieke) product. Het afnemen van een serviceovereenkomst staat los van het belang van continue doorontwikkeling, wat ook essentieel is in een dergelijke situatie.

8.1 Definitie van servicewerk

Om ervoor te zorgen dat ons ontwikkelproces soepel verloopt, is het belangrijk dat we tijdens het ontwikkelen onze agile werkwijze aanhouden. Daarnaast moet er ruimte zijn voor het verlenen van ondersteuning en klein onderhoud. Om ervoor te zorgen dat we gefocust aan ontwikkeltaken kunnen werken in sprints, en toch doorlopend goede service kunnen leveren, hebben we beperkt wat voor werk we binnen de serviceovereenkomst kunnen oppakken.

We kiezen ervoor om het werken aan nieuwe functies of het doen van kleine niet-acute veranderingen aan softwareproducten **niet** op te pakken binnen de serviceovereenkomst. Dergelijke veranderingen kunnen gedaan worden in de aankomende sprint, of als los werk. Dit doen we vooral omdat we hiervoor onze agile werkwijze en feedbackafspraken willen aanhouden.

Wat valt wel onder de definitie van servicewerk? In ieder geval het volgende:

- Het beheren van applicatie-infrastructuur;
- Het oplossen van gebruiksblokkerende bugs en problemen die plotseling ontstaan zijn op een productieomgeving;

- Ondersteuningswerk zoals het beantwoorden van vragen, zowel per mail als via de telefoon;
- Monitoren van applicatiegezondheid en crashrapportages, en op basis hiervan kleine verbeteringen doorvoeren en verbeteringsvoorstellen toevoegen aan de backlog (*alleen met een SLA*);
- Administratieve werkzaamheden voor het inrichten en faciliteren van softwareontwikkeling en -distributie;
- Afhandelen van responsible disclosure meldingen en gegevensverzoeken;

In de volgende paragrafen geven we voor elk van deze onderdelen een verdere uitleg.

8.1.1 Het beheren van applicatie-infrastructuur

Het beheren van applicatie-infrastructuur is een belangrijke onderhoudstaak. Hiermee bedoelen we het opzetten en beheren van servers en diensten waarop de applicatie draait of waar functies in de applicatie afhankelijk van zijn. Dit gaat vrijwel altijd om een aantal servers die via een beveiligd virtueel netwerk aan elkaar geknoopt zijn. Daarbij wordt ook meestal gebruik gemaakt van verschillende additionele hosting- en clouddienstverlening.

Voorbeelden van applicatie-infrastructuur:

- Domeinnamen en TLS-certificaten;
- Webservers en load balancers;
- Blobstorage-dienst voor opslag van grote bestanden;
- Relationale database servers;
- Snelle in-memory cachingservers;
- Dienst voor transactionele e-mail;
- DNS-server.

Productie-, staging- en development-omgevingen

Een belangrijk voorwaarde voor onze werkwijze is het opzetten van losstaande applicatie-infrastructuur voor twee of drie verschillende situaties. Zo zetten we een productieomgeving op voor het serveren van de applicatie aan daadwerkelijke eindgebruikers. De productieomgeving is vaak uitgerust met krachtigere servers zodat veel verkeer tegelijk verwerkt kan worden.

Naast de productieomgeving hebben we ook een stagingomgeving nodig. De omgeving waarop we software testen en presenteren aan de opdrachtgever. Af en toe schalen we deze naar hetzelfde niveau als de productieomgeving om bijvoorbeeld een load-test uit te voeren. Los daarvan is de omgeving meestal minder krachtig uitgerust. Om ervoor te zorgen dat we er zeker van zijn dat de stagingomgeving niet de productieomgeving kan beïnvloeden, hebben we hiervoor ook vaak een aparte load balancer, databaseserver, webserver(s) en andere benodigde losse diensten.

Als we een ingewikkelde applicatie-infrastructuur opzetten, hebben we ook vaak een losse omgeving nodig om de ontwikkeling van de applicatie-infrastructuur zelf op te doen. Deze omgeving noemen we de developmentomgeving.

Het is meestal zo dat we bij aanvang van de ontwikkeling al een stagingomgeving en developmentomgeving opzetten, voordat we voor het eerst lanceren. De kosten hiervoor zie je terug onder de maandelijkse kosten voor hosting en tooling.

Deze omgevingen moeten worden beheerd en onderhouden. Eigenlijk combineren we het beheer en onderhoud van de twee of drie omgevingen altijd zodat we dit zo effectief mogelijk doen.

Uitvoeren van software-updates

Om ervoor te zorgen dat applicatie-infrastructuur veilig en optimaal blijft functioneren moeten we regelmatig software-updates uitvoeren. Soms kan dit tijdelijk tot serviceonderbrekingen leiden. Afhankelijk van de omvang van de applicatie-infrastructuur kunnen we de impact hiervan beperken. Het is vaak zo dat we met een uitgebreide infrastructuur serviceonderbrekingen voor software-updates zoveel mogelijk kunnen beperken.

Opschalen van applicatie-infrastructuur

Bij een verhoogde belasting is het nodig om servers op te schalen zodat het softwareproduct soepel blijft draaien. Een verhoogde belasting kan komen door meer gebruikers, maar ook door meer gegevens of een nieuwe functie die meer computer-resources vereist. Met het opschalen nemen ook de maandelijkse hostingkosten toe. Als servers opgeschaald worden tot ware supercomputers kunnen deze bedragen aanzienlijk zijn. Bij een SLA houden we de belasting van de servers in de gaten en schalen proactief op als dit nodig is.

Oppakken van taken vanuit de providers

Het gebeurt regelmatig dat onze providers onderliggende diensten of infrastructuur, waarvan het softwareproduct gebruik maakt, upgraden of veranderen. Meestal gebeurt dit zonder dat we hier actieve keuzes in kunnen en hoeven te maken, maar af en toe is dit nodig. Zodra dit gebeurt pakken we dit op om ervoor te zorgen dat de applicatie-infrastructuur goed blijft draaien.

Ook als er geen serviceovereenkomst is afgesloten pakken we taken vanuit de providers op. Het zou anders serviceonderbreking voor het softwareproduct tot gevolg kunnen hebben. Net als bij ander servicewerk zonder overeenkomst, brengen we het werk aan het einde van de maand in rekening.

Het inrichten van infrastructuur of aanbrengen van structurele wijzigingen

Bij aanvang van de ontwikkeling richten we de applicatie-infrastructuur in tijdens onze sprintwerkzaamheden. Het is af en toe nodig om grote structurele wijzigingen aan te brengen aan de applicatie-infrastructuur. Zo is het meestal verstandig om na een paar jaar te migreren naar een compleet nieuw ingerichte infrastructuur. Elk jaar zijn er nieuwe ontwikkelingen die applicatie-infrastructuur nog beter, gemakkelijker en veiliger maken. Deze grote werkzaamheden zien we niet als servicewerk en deze kunnen dus worden uitgevoerd in sprints of als los werk.

8.1.2 Oplossen van acute gebruiksblokkerende bugs en problemen

Wanneer een acute bug ontstaat die een aanzienlijk deel van de eindgebruikers treft, kunnen we het probleem oppakken en oplossen als servicewerk. Dit geldt ook voor serviceonderbrekingen door bijvoorbeeld overbelasting.

Hoe kan een bug ontstaan? Dit kan gebeuren door bijvoorbeeld het uitkomen van een nieuwe versie van een browser of besturingssysteem, waarin onze code anders geïnterpreteerd wordt. Een software-update op de server kan wijzigingen introduceren die de applicatiewerking in de weg zitten. Of een dienst van een derde partij kan problemen ondervinden waardoor een bepaalde functie in het softwareproduct niet meer goed werkt.

We grijpen in als delen van het softwareproduct niet meer juist werken door overbelasting. Dat kan in de applicatie-infrastructuur, door servers op te schalen, maar we kunnen ook aanpassingen aanbrengen in de code om bijvoorbeeld de efficiëntie te verhogen.

We kunnen ervoor kiezen om een snelle oplossing te introduceren, ondanks dat er op lange termijn een betere oplossing nodig is. Het introduceren van de betere oplossing wordt dan opgenomen in de backlog om op te pakken in een toekomstige sprint.

Het gebeurt ook wel eens dat een acuut probleem ontstaat na de publicatie van een nieuwe versie naar de productieomgeving. Bijvoorbeeld omdat we bij de ontwikkeling en kwaliteitswaarborging een randgeval over het hoofd hebben gezien. Als we met een kleine wijziging het probleem op kunnen lossen doen we dit als servicewerk. Een andere optie is om terug te vallen op een eerdere versie van het softwareproduct.

Bugs of kleine fouten die niet direct het gebruik van een belangrijke functie in het softwareproduct beperken, nemen we op in de backlog om te verwerken in een volgende sprint. Voorbeelden hiervan zijn: het verbeteren van een tekstuele fout, of het aanpassen van een validatieregel.

8.1.3 Ondersteuning

We helpen onze opdrachtgevers met ondersteuning voor het ontwikkelde softwareproduct. Zo zijn er wellicht bepaalde power user functies niet altijd duidelijk,

of zijn er vragen over de ontwikkeling. Op andere momenten moet de opdrachtgever eindgebruikers verder helpen met hun vragen, maar is hiervoor informatie van ons nodig. Zoals eerder genoemd in het hoofdstuk over 'uitgangspunten' levert Label305 niet zelf direct ondersteuning aan eindgebruikers. We helpen onze opdrachtgevers bij het beantwoorden van hun vragen. Bij ingewikkelde ondersteuningsvragen is soms diep onderzoek nodig. Dit onderzoek zien we ook als onderdeel van het servicewerk.

Als een SLA is afgesloten zullen we binnen 1 werkdag reageren. Dit geldt voor zowel e-mails als berichten geplaatst in een projectmanagementsysteem.

Aanpassingen aan het product voor ondersteuning

Ondersteuningsvragen kunnen leiden tot nieuwe wensen voor aanpassingen aan het softwareproduct. Deze pakken we *niet* op binnen het servicewerk, en nemen we op in de backlog. We doen aanpassingen aan het softwareproduct in de sprint of tijdens los werk.

Telefonische ondersteuning

Is er een vraag of een probleem dat meteen besproken moet worden? Dan kan gebeld worden naar onze kantoren. Meestal krijg je meteen iemand te spreken die je kan helpen, maar we geven hier niet in alle gevallen garantie op.

Als een SLA is afgesloten zorgen we ervoor dat we snel kunnen helpen. Wordt er gebeld over spoedwerk? Dan staan we direct klaar om het probleem te analyseren en op te pakken. Worden we gebeld voor iets minder dringends, dan proberen we desalniettemin direct te helpen. Het zou kunnen dat de specialist die het beste kan assisteren niet direct beschikbaar is. In dat geval vragen we of we terug kunnen bellen. Kan dit niet? Dan helpen onze andere specialisten je alsnog meteen.

Als geen SLA is afgesloten, bellen we bij ondersteuningsvragen vaak later op de dag terug. We kunnen niet altijd direct helpen. We bellen terug op de momenten waarop onze specialisten tijd vrijmaken voor het beantwoorden van ondersteuningse-mails. Meer hierover in het hoofdstuk over 'dagindeling'. We kunnen bij een SLA sneller assisteren omdat wij met de een deel van de maandelijkse SLA-tarief specialisten beschikbaar kunnen houden voor deze directe ondersteuningswerkzaamheden.

Tijdens de sprint kan gebeld worden met alle betrokken teamleden, los van de serviceovereenkomst. Er kan ook telefonisch gesproken worden met onze projectmanager over planningszaken, los van een lopende sprint of een serviceovereenkomst.

Samenwerken met interne klantenservice

Een goeddraaiend softwareproduct met veel eindgebruikers heeft dagelijks te maken met tientallen vragen en suggesties. De juiste klantenservice is een belangrijk onderdeel van een goed product. De eigenaar van het softwareproduct moet zelf

bezig gaan met het inrichten van deze klantenservice. Het kan gebeuren dat deze klantenservice niet direct antwoord kan geven omdat bijvoorbeeld niet alle informatie beschikbaar is.

We kunnen functies toevoegen aan het softwareproduct om het gemakkelijker te maken om de eindgebruikers goed te helpen. Bijvoorbeeld door het ontwikkelen van een koppeling met een ticketsysteem, het bouwen van een impersonatiefunctie en het maken van een auditlog van gebruikersacties.

Daarnaast helpen wij bij het beantwoorden van vragen. Goede en prettige terugkoppeling vanuit de interne klantenservice naar ons als productspecialisten is belangrijk. Zo kunnen wij technische problemen duiden en licht werpen op vreemde situaties. Met een onderzoek of een inzicht van ons kan de klantenservice weer terug naar de eindgebruiker en antwoord geven.

Wij worden benaderd door de eindgebruiker

Het gebeurt wel eens dat eindgebruikers ons direct benaderen omdat ze een applicatie zien in ons portfolio. In een dergelijk geval reageren we alsnog en verwijzen ze door naar de interne klantenservice van onze opdrachtgever voor directe ondersteuning. Het snel en adequaat reageren op deze eindgebruikers is belangrijk voor om de productervaring hoog te houden. We zien ook dit als servicewerk.

8.1.4 Monitoren (SLA)

Als een SLA is afgesloten houden we tijdens kantoortijden de applicatiegezondheid continu in de gaten met de hulp van verschillende tools. We controleren of de applicatie goed draait en beschikbaar is voor alle gebruikers. Mocht een applicatie onverhoopt offline gaan of problemen ondervinden, dan krijgen wij meteen een melding zowel tijdens als buiten kantoortijden.

We houden verschillende belastingsniveaus in de gaten. Dit gaat onder andere om: de bezetting van de processoren, het beschikbare interne geheugen, de doorstroom van het netwerkverkeer en de interacties met de opslag. Als een indicator boven een kritieke waarden komt geeft dit vaak aan dat er iets aan de hand is, en gaan we op onderzoek uit.

We monitoren nieuw binnenkomende crashrapporten voor zowel frontend, backend, mobiele apps en andere productonderdelen. Als iets fout gaat op applicatieniveau heeft de gebruiker de mogelijkheid een crashrapport op te sturen. Indien we in een korte periode vaker dezelfde rapporten binnen zien komen gaan we direct de melding onderzoeken. Soms zien we dat dit gebeurt na een besturingssysteem- of browserupdate, dan is het goed om meteen op de hoogte te zijn van deze problemen en te gaan werken aan een oplossing. Het in de gaten houden van crashrapportages geeft ons de mogelijkheid om extra inzicht te krijgen zodra een gebruiker de helpdesk

benadert met een klacht. We kunnen terugzoeken waar en waarom het exact fout is gegaan.

8.1.5 Administratieve werkzaamheden

Af en toe moeten wij administratief werk verrichten voor een softwareproduct dat we ondersteunen. Denk hierbij aan administratief werk voor het valideren van onze opdrachtgevers bij cloudproviders, domeinnaaminstanties of bij de beheerders van applicatiewinkels.

Zo heeft Apple bijvoorbeeld een streng beleid voor toegang tot de App Store. Daarbij heeft Apple een speciaal programma voor hardwarefabrikanten. Zowel voor het hardwarprogramma als App Store-toegang moeten we administratief werk verrichten.

Het opstellen van de kwartaalafrekening en overzichten voor hosting- en toolingkosten zien we als administratieve werkzaamheden. We maken geen marge op deze dienstverlening zelf, maar brengen de tijd waarin we bezig zijn met administratie in rekening.

8.1.6 Afhandelen van meldingen en verzoeken

Verzoeken tot het verkrijgen, veranderen of verwijderen van persoonsgegevens moeten wettelijk worden voldaan. Een verzoek kan aan ons worden doorgezet. Wij handelen dan het technische aspect af. Dit geldt als servicewerk. We sturen het resultaat uiteindelijk naar de interne klantenservice terug. Daar ligt de uiteindelijke verantwoordelijkheid voor het versturen van de gegevens/bevestiging naar de eindgebruiker. Daarbij is de klantenservice van het product verantwoordelijk voor de verificatie van de gebruiker.

Iedereen in de Europese Unie heeft het recht om vergeten te worden. In sommige softwareproducten is dit lastiger dan je in eerste instantie zou denken. We moeten de eindgebruikers hiervoor de mogelijkheid geven. Ze kunnen een aanvraag indienen per e-mail, of via een functie die speciaal is ingebouwd in de applicatie om dit soort verzoeken af te handelen. Als we geen speciale functie hebben kunnen we een verzoek handmatig oppakken. Het kan zo zijn dat we gegevens uit een systeem halen in verband met de Amerikaanse DMCA of voor een gerechtelijk bevel. Ook hiervoor is meestal handmatig werk nodig.

Bij een melding die voortkomt uit het responsible disclosure-beleid nemen wij het contact over. We onderzoeken het veiligheidsprobleem en stellen de onderzoeker en de producteigenaar op de hoogte van onze bevindingen. Tevens verhelpen we het veiligheidsprobleem, indien nodig. We geven tevens de producteigenaar een aanbeveling voor een beloning. Uiteindelijk is het aan hem/haar om de beloning te bepalen en uit te keren.

8.2 Geen serviceovereenkomst

Tijdens de initiële ontwikkelfase is het hebben van een serviceovereenkomst niet noodzakelijk. Omdat in de initiële ontwikkelfase meestal doorlopend sprints ingepland zijn, kan het servicewerk daarin gedaan worden. Daarnaast is er voorafgaand aan de lancering vaak geen noodzaak voor serveronderhoud of het helpen bij het beantwoorden van gebruikersvragen. Dit alles verandert zodra het product lanceert.

Wij raden **ten zeerste af** om geen serviceovereenkomst af te sluiten zodra het softwareproduct in productie draait. Sporadisch wordt hier toch voor gekozen omdat bijvoorbeeld de aard van het product zo eenvoudig is dat bijna geen service nodig is. We verlenen in een dergelijk geval voornamelijk reactief service. Het uitvoeren van serveronderhoud schuiven we door naar toekomstige sprints.

Omdat wij zonder serviceovereenkomst minder goed kunnen anticiperen op servicewerk rekenen wij het **hoge uurtarief** als toch servicewerk nodig is. Zoals eerder beschreven in het hoofdstuk 'uitgangspunten' geldt dit ook voor: het beantwoorden van ondersteuningsmails, het uitvoeren van kleine beheertaken voor de applicatie infrastructuur, en ander klein servicewerk. Voor de meeste van deze werkzaamheden vragen we niet apart om een akkoord. We factureren de gewerkte tijd aan het einde van de maand.

Coulance bij kleine hoeveelheden serviceverlening

Als er gedurende een gehele maand minder dan 30 minuten service geleverd is en er is geen serviceovereenkomst afgesloten, dan brengen we deze minuten niet in rekening. We willen voorkomen dat bij een paar e-mailtjes en een telefoontje van 5 minuten direct een servicefactuur gestuurd wordt.

8.3 Best-effort Service Agreement (BESA)

Draait een softwareproduct in productie? Dan is het verstandig om een serviceovereenkomst af te sluiten. Hierbij kan gekozen worden tussen onze *best-effort service agreement* (BESA) en onze *service level agreement* (SLA). Als het softwareproduct nog niet essentieel is voor het bedrijf, dan is niet altijd nodig om onze uitgebreide SLA af te sluiten. Voor deze softwareproducten hebben we de BESA. Er kan van een BESA over worden gestapt naar een SLA als toch meer garanties nodig blijken.

Bij een BESA bepalen we vooraf hoeveel servicewerk elke maand beschikbaar is, zodat wij hierop kunnen anticiperen. Daarmee voeren we onder andere proactief klein serveronderhoud uit zodat het softwareproduct draait op up-to-date applicatie-infrastructuur.

We geven geen garanties af over het serviceniveau, de uptime, de beschikbaarheid en het tijdig oppakken van problemen. We monitoren de applicatie niet. Ook bij

telefonisch contact kan het gebeuren dat we niet direct kunnen helpen. Wel garanderen we dat de vooraf ingekochte uren ingevuld kunnen worden. We doen eveneens altijd ons uiterste best om goed en snel te helpen. De overeenkomst die meer geeft dan enkel 'best effort' is onze SLA, daarover verderop meer.

8.3.1 Service-uren inkopen en opbouwen

Met de BESA koopt de opdrachtgever elke maand vooraf service-uren in voor het **lage uurtarief**. In deze uren voeren wij servicewerk uit. Worden er minder service-uren gebruikt dan er beschikbaar zijn? Dan sparen ze op om in te vullen in een volgende maand. Uren waarvoor betaald wordt verlopen dus niet.

Worden meer service-uren gebruikt dan er vooraf zijn ingekocht? Dan rekenen we het **hoge uurtarief** voor deze extra service-uren, en zetten ze op de volgende servicefactuur. Blijkt er aan het einde van de maand dat er meer service is gebruikt dan vooraf was gereserveerd, maar minder dan 30 minuten extra? Dan brengen we het extra servicewerk niet in rekening. Dit geldt zowel voor de SLA als voor de BESA.

Elk kwartaal kijken we hoeveel uren we het afgelopen kwartaal ingevuld hebben en aan de hand hiervan stellen we de maandelijkse inkoop bij. Hierbij houden we rekening met openstaand tegoed. Voor een geldige BESA dient **minimaal één service-uur** per maand ingekocht te worden.

Voor het invullen van service-uren vragen we niet vooraf een akkoord. Bij het overschrijden van de vooraf ingekochte service-uren zijn we niet verplicht dit te melden. We proberen dit wel te doen.

8.3.2 Serviceverlening tijdens de sprint

Als we met een sprint bezig zijn en er moet servicewerk uitgevoerd worden, dan proberen we dit te doen binnen de sprint. De specialisten die het beste de service kunnen verlenen zitten meestal ook in het sprintteam. In een dergelijk geval wordt de tijd verrekend met de minimale inspanningsverplichting voor de sprint, in plaats van het servicetegoed. Als opdrachtgever, kan je aangeven dat je dit liever los van elkaar ziet.

Zijn continu doorlopend sprints ingepland? En kan al het nodige servicewerk uitgevoerd worden tijdens de sprints? Is er dan nog wel een serviceovereenkomst nodig? Bij een draaiende productieomgeving raden we alsnog aan om een overeenkomst af te sluiten. We kunnen de hoeveelheid maandelijks ingekochte uren beperken tot het minimum van één uur.

8.4 Service Level Agreement (SLA)

De service level agreement is een uitgebreide serviceovereenkomst waarbij wij klaar staan om onze beste service te verlenen. Wij raden aan om een SLA af te sluiten zodra een applicatie kritiek is geworden voor de bedrijfsvoering of voor de productpropositie.

Meestal is dit bij de publieke lancering van het softwareproduct, of een paar maanden hierna.

Net als bij een BESA schatten we vooraf in hoeveel servicewerk elke maand beschikbaar moet zijn zodat wij hierop kunnen anticiperen. In tegenstelling tot de BESA houden we naast het uitvoeren van serveronderhoud ook continu alles in de gaten. We monitoren de gezondheid van de applicatie en de applicatie-infrastructuur. Detecteren we een probleem? Dan pakken we dit meteen zelf op zonder dat de producteigenaar ernaar om hoeft te kijken, en voordat de eindgebruiker een bugmelding hoeft te sturen. Is het iets drukker? Dan schalen we de applicatie-infrastructuur meteen op om zo goed om te kunnen gaan met het extra verkeer.

Bij een SLA geven we bovendien concrete garanties af over uptime, gegevensintegriteit, het tijdig oppakken van problemen en de beschikbaarheid van ingelezen specialisten. We zetten ons hier volledig voor in. Daarom nemen we dus ook met vol vertrouwen concrete consequenties op in de overeenkomst voor het geval dat we verzuimen te voldoen aan de garanties.

Voor deze uitgebreide service wordt naast de maandelijks ingekochte service-uren ook een vast maandelijks tarief gerekend. Dit tarief wordt bepaald op basis van het formaat van de applicatie en de noodzakelijke hoeveelheid ingelezen specialisten.

Zijn er meerdere softwareproducten met afzonderlijke productieomgevingen? Dan kan een SLA afgesloten worden voor een deel van deze omgevingen/producten of voor allemaal. Het basistarief kan verschillen als meerdere omgevingen en applicaties opgenomen moeten worden in de SLA.

8.4.1 Proactief servicewerk oppakken en het monitoren van de applicatie

Bij de BESA voeren we proactief software-updates uit op de applicatie-infrastructuur. Bij een SLA zijn we nog intensiever bezig met een zo soepel mogelijk draaiende productieomgeving of applicatie. We monitoren voor problemen op verschillende niveaus en lossen problemen proactief op zodra we deze detecteren.

Ook brengen we zelf kleine verbeteringen of optimalisaties aan als we zien dat we hiermee de applicatieprestatie kunnen verbeteren. Een betere prestatie (*performance*) zorgt er weer voor een betere gebruikerservaring, maar ook voor lagere hostingkosten.

Bij een overbelaste infrastructuur schalen we op, zodat de producteigenaar hier geen omkijken naar heeft. We stellen hem/haar hiervan wel op de hoogte. Bij het opschalen van de infrastructuur moet rekening gehouden worden met afrekening en verhoging van de hostingkosten aan het einde van het kwartaal.

8.4.2 Servicetegoed overschrijding voordeliger

Het vooraf inkopen van service-uren werkt hetzelfde als bij de BESA, met een paar uitzonderingen. Gaan we over het servicetegoed heen? Dan wordt voor de extra service-uren ook het **lage uurtarief** betaald, en niet het hoge uurtarief zoals bij de BESA. Bij een SLA voeren we verschillende periodiek taken uit. Daarom geldt de minimale maandelijkse inkoop van servicewerk van **vier service-uren**.

Ook bij de SLA stellen we elk kwartaal de maandelijkse reservering van het aantal service-uren bij op basis van het gebruik het afgelopen kwartaal. Ongebruikte service-uren worden opgespaard in een tegoed. Ze kunnen ingevuld worden in een volgende maand. Het tegoed verloopt niet.

Voor het invullen van service-uren vragen we niet vooraf om een akkoord. Bij het overschrijden van de vooraf ingekochte service-uren zijn we niet verplicht dit te melden. We proberen dit wel altijd te doen. Net als bij de BESA hebben we een coulance bij een kleine overschrijding van minder dan 30 minuten.

Als wij door de producteigenaar of de opdrachtgever enigszins beperkt worden in het invullen van de servicewerkzaamheden, kan geen aanspraak gemaakt worden op de SLA-garantieregeling.

8.4.3 SLA en doorlopende sprintwerkzaamheden

Bij een softwareproduct is continue doorontwikkeling en de juiste service voor de productie-applicatie allebei belangrijk. Is een sprint bezig? Dan is er automatisch de gegarandeerde beschikbaarheid van een ingelezen specialist. Het afnemen van een SLA geeft voor die week niet direct de garantie op een extra ingelezen specialist naast de leden van het sprintteam. We monitoren de applicatie-gezondheid en voeren proactief servicewerk uit naast de sprint. Effectief besteden de specialisten ook veel extra tijd aan het product – van de gemiddeld tien uur die zij te besteden hebben buiten de sprint. Het product profiteert van de uptime-garantie en de gegevensintegriteitsgarantie met een SLA naast sprintwerkzaamheden, net als buiten de sprint.

Wij passen niet ons SLA-aanbod aan tijdens maanden waarin veel sprintwerk gedaan wordt. Wel kan de vooraf ingekochte service-tijd naar beneden gaan omdat een deel van het werk ingevuld wordt in de sprints.

Plannen we samen voor meer dan vijftig weken doorlopend sprintwerk in – zonder interruptie? Dan is het mogelijk om een tijdelijke korting te krijgen op het SLA-basistarief. Informeer hierover als je van plan bent meer dan vijftig weken doorlopend sprints af te nemen. Als je het uitvoeren van doorlopende sprints staakt, vervalt de korting.

8.4.4 Applicatie-uptimegarantie

Wij garanderen een applicatie-uptime van **99.5% voor de afgelopen 365 dagen**. Hierbij tellen wij dagen waarop er geen geldige SLA was als dagen met 100% uptime. Deze uptime-garantie geldt alleen voor onverwachte downtime op de productieomgeving.

Applicatie-uptime is gedefinieerd als: de applicatie wordt geserveerd over HTTP(s) met een geldig certificaat, daarbij zijn de webserver dus beschikbaar, daarnaast is ook een juiste en goede verbinding tussen de databaseserver, de cachingserver en de webserver. Applicatie-uptime wordt bepaald vanuit, de door ons geselecteerde, monitoringtools.

Onze definitie van applicatie-uptime is een brede definitie als je deze vergelijkt met andere SLA-contracten. Dit is de reden dat we niet meer dan 99.5% kunnen garanderen. Sommige hostingproviders leveren SLA-contracten met 99.99% uptime-garantie, alleen hierbij wordt vaak vermeld in de kleine lettertjes dat het enkel gaat over het beantwoorden van een simpele ping door de webserver, en niet het serveren van een volledige applicatie. Wij kiezen voor een brede definitie omdat onze opdrachtgevers alleen wat hebben aan een werkende applicatie.

Deze uptime-garantie heeft als voorwaarden dat wij het volledige beheer over de applicatie-infrastructuur hebben en dat onze adviezen over groot onderhoud en infrastructuur-wijzigingen in sprints of bij losse werkzaamheden worden opgevolgd. De aanwezigheid van een bug op de productieomgeving geldt op zichzelf niet als downtime. Wel kunnen we met een SLA een acute bug meteen oppakken.

Soms is downtime noodzakelijk voor het upgraden van de database of voor het overstappen op een nieuwe infrastructuur. Ingepland onderhoud tellen wij niet mee als onverwachte downtime.

Buiten kantoortijden blijven wij de applicatie monitoren voor downtime. Mochten grote problemen ontstaan buiten kantoortijden dan worden wij op de hoogte gesteld middels een melding. Wij verhelpen dan vaak de problemen, zelfs buiten kantoortijden.

8.4.5 Gegevensintegriteitsgarantie

Wij garanderen de gegevensintegriteit van de applicatiedata voor de inhoud van relationele databases en voor blobopslag met maximaal **60 minuten** verloren data bij een gegevenscorruptie-incident. Ook garanderen we het terughalen van een databasekopie tot **twee weken**, zodat de data te herstellen zijn als we corruptie binnen twee weken opmerken. We bewaren databasekopieën niet langer dan twee weken in verband met de GDPR.

Net als de uptime-garantie, heeft de gegevensgarantie als voorwaarden dat wij het volledige beheer over de applicatie-infrastructuur hebben en dat onze adviezen over

groot onderhoud en infrastructuurwijzigingen in sprints of bij losse werkzaamheden worden opgevolgd.

8.4.6 Garantie voor het tijdig oppakken van meldingen en problemen

Wordt een probleem met spoed gemeld? Dan garanderen we dat dit probleem wordt opgepakt binnen **één werkdag**. Een voorbeeld: als er op dinsdag om 11:00 uur iets gemeld wordt, pakken we dit sowieso op voor woensdag om 11:00 uur. Dit is geen streeftijd maar een maximum, dus in de praktijk pakken we problemen nog veel sneller op, eigenlijk bijna altijd dezelfde werkdag binnen één of twee uur.

Deze garantie geldt voor al het gewenste spoedwerk. Dus voor zowel urgent servicewerk als voor ander los spoedwerk. Dit is mogelijk omdat bij een SLA een ingelezen specialist toch al voor het product beschikbaar wordt gehouden. We willen hier graag flexibel in zijn. Wel beperken we de invulling van spoedwerk met deze garantie tot het dubbele aantal vooraf ingekochte service-uren, om te vermijden dat werk constant met een spoedmelding ingeschoten wordt. Een spoedmelding is wel echt bedoeld voor spoed.

Bij kleine projecten is het aantal ingelezen specialisten meestal beperkt. Als door een vakantie tijdelijk niemand beschikbaar is, kondigen we dit van tevoren aan. We helpen dan uiteraard nog steeds bij spoedsituaties, maar omdat de personen die de problemen oppakken dan niet ingelezen zijn, kan het oplossen langer duren.

Moeten altijd meerdere specialisten beschikbaar zijn? Dat kan, dan breiden we de SLA uit met een hoger basistarief, en betrekken meerdere specialisten bij de doorlopende ontwikkeling.

Het tijdig oppakken van problemen geldt voor spoedmeldingen maar ook voor schriftelijke ondersteuning, het oppakken van responsible disclosure-meldingen en het voldoen aan inzage-, verwijder- en vergeetverzoeken. Hierbij is het altijd goed om aan te geven dat iets urgent is. We garanderen dan de zaken binnen **één werkdag** worden opgepakt. Ander servicewerk dat minder urgent is, proberen we met een SLA ook binnen één werkdag op te pakken, hier geven we alleen geen garantie voor af.

8.4.7 Compensatie bij verzuim

Gaat iets fout, en verzuimen wij te voldoen aan de garanties die we geven in de SLA? Dan kan aanspraak gemaakt worden op onze SLA-garantieregeling. Indien dit terecht is, vindt compensatie plaats. Bij compensatie wordt per incident de **totale som** van het vaste maandelijks SLA-basistarief **gecrediteerd**, bovendien proberen we de problemen alsnog kosteloos te verhelpen. Dit doen we door 30 extra service-uren beschikbaar te stellen waarin we het probleem gaan verhelpen. We werken per incident **maximaal 30 uur kosteloos** door, daarna rekenen we de service-uren weer

op normale wijze door. Aanspraak maken op de SLA-garantieregeling kan niet vaker dan drie keer per kwartaal.

Overmacht

Het aanspraak maken op de SLA-garantieregeling kan alleen als er geen sprake is van overmacht. Dit betekent onder andere dat de ontstane problemen niet direct aan onze nalatigheid toe te schrijven zijn. Voorbeelden waarbij er sprake is van overmacht:

- De applicatie is onbereikbaar door een storing bij een derde partij waarbij wij hosting afnemen;
- Wij kunnen geen nieuwe versie live zetten door een storing bij een derde partij waarbij wij hosting- of toolingdiensten afnemen;
- Onze ingelezen specialist is ziek en is daardoor niet beschikbaar voor het geven van ondersteuning;
- Vervoersproblemen waardoor een specialist niet op tijd op locatie is;
- Een elektriciteitsstoring of brand bij een kantoor van Label305; *enzovoort...*

9 Hosting en tooling

Voor het ontwikkelen van een softwareproduct en voor het serveren van dit softwareproduct aan eindgebruikers wordt gebruik gemaakt van verschillende hostingoplossingen en assisterende software (tooling).

Hosting is de dienstverlening noodzakelijk voor het draaien van jouw applicatie. Al deze diensten gezamenlijk zijn de infrastructuur waarop de applicatie draait. Het gaat om de huur van servercomputers maar ook andere diensten. Enkele voorbeelden zijn:

- Het registreren en vernieuwen van domeinnamen en TLS-certificaten;
- Webhosting en load balancers via een reguliere hosting- of een cloudprovider;
- Blobstorage voor opslag van grote bestanden;
- Relationale database servers via Database as a Service;
- Snelle in-memory cachingservers;
- Software voor het versturen van transactionele e-mailberichten

Voor de applicatie-infrastructuur hebben we meestal nog één of twee omgevingen nodig naast de productieomgeving die moeten draaien – de stagingomgeving en de developmentomgeving.

Naast hosting gebruiken wij gereedschappen (tooling) die belangrijk zijn voor de ontwikkeling en het verlenen van goede service. Enkele voorbeelden zijn:

- Software voor logbestandaggregatie en crashrapportage;
- Software voor het monitoren en analyseren van de gezondheid van de applicatie-infrastructuur;
- Software voor het faciliteren van onze ontwikkelingsworkflow, bijvoorbeeld versiebeheerssoftware en continuous integration software;

9.1 Maandelijks voorschot en kwartaalafrekening

Veel van de hostingoplossingen en tools van derde partijen worden op basis van gebruik in rekening gebracht. Dit gebruik kan variëren. We kunnen daarom kosten voor diensten die wij voor het softwareproduct inzetten eens per maand doorrekenen. Dit geldt voor alle tools die we gebruiken voor het ontwikkelen van het product, maar ook in sommige gevallen voor hostingdiensten die we gebruiken voor het inrichten en draaien van de applicatie-infrastructuur. Wij rekenen kosten voor hosting en tooling door **zonder marge**.

Omdat we vooraf niet precies weten hoeveel moet worden uitgegeven aan hosting en tooling, wordt het doorrekenen gedaan middels een maandelijks voorschot. Dit maandelijks voorschot stellen we elk kwartaal bij op basis van de werkelijke kosten van het afgelopen kwartaal en de verwachting voor het volgende kwartaal.

We komen aan het einde van een kwartaal met een **kwartaalafrekening**, waarbij de opdrachtgever wellicht iets extra's moet betalen of iets terugkrijgt. Voor deze kwartaalafrekening kan een uitgebreide specificatie opgevraagd worden van alle afgenomen diensten en bijhorende kosten. We zijn elk kwartaal even bezig met het opstellen van de kwartaalafrekening, dit werk wordt via de service-uren verrekend.

Als een opdrachtgever liever elk kwartaal één factuur heeft is dit ook mogelijk. Dan sturen we elk kwartaal één factuur met een afrekening van het vorige kwartaal en een inschatting voor het nieuwe kwartaal. We werken niet met enkel achteraf doorrekenen.

9.2 Nieuwe diensten introduceren

Om het zelfsturende karakter van ons team te waarborgen willen wij graag zelf beslissen over het gebruik van specifieke hostingdiensten en tools. Het introduceren van nieuwe diensten doen wij omdat we verwachten hiermee in het vervolg beter en effectiever kunnen werken. We blijven diensten enkel gebruiken als ze leiden tot een beter product, meer zekerheid of een effectievere werkwijze.

Bij het beslissen over de introductie van nieuwe diensten houden wij de kosten voor de opdrachtgever en de impact op de privacy van gebruikers in het achterhoofd. Het staat ons altijd vrij om zonder contact of expliciete toestemming van de opdrachtgever diensten te introduceren, zolang deze diensten geen gegevens verwerken. Onze opdrachtgever kan specifieke randvoorwaarden hebben voor diensten die toegang hebben tot gegevens van gebruikers. Deze voorwaarden respecteren we en nemen we mee in de overweging een nieuwe gegevensverwerkende dienst te introduceren.

We evalueren het nut van de gebruikte diensten regelmatig. Zo stoppen we soms ook met het gebruik van specifieke diensten als we zien dat ze niet meer de verwachte toegevoegde waarde leveren.

9.3 Zelf diensten direct afnemen bij een derde partij

Soms is het wenselijk dat de opdrachtgever zelf direct hostingdiensten afneemt. Sterker nog, voor het een groot deel van de diensten heeft dit vaak onze voorkeur. Vooral voor de verantwoordelijkheidsketen van de gegevensverwerking in een productieomgeving is dit belangrijk.

Het is vaak mogelijk om verschillende hostingdiensten zelf in beheer te nemen en ons alle toegang te verschaffen, in dat geval kan de dienst direct worden afgenomen bij de derde partij. Dit is met name bij cloudhosting voor de productieomgeving een goed idee.

We dragen normaal gesproken het beheer over de versiebeheeromgeving, continuous integrationsystemen en andere ontwikkelaarstooling niet over, hiervoor rekenen we de kosten door.

9.4 Hosting- en toolingkosten zonder serviceovereenkomst

Als geen overeenkomst voor servicewerk is afgesloten, is het vaak toch nodig om periodiek hosting- en toolingkosten in rekening te brengen. Zolang we moeten kunnen ontwikkelen aan de applicatie is het hebben van een juist ingerichte stagingomgeving, versiebeheerssoftware en continuous integrationsystemen noodzakelijk.

Zo is in de vroege stadia van ontwikkeling vaak geen serviceovereenkomst maar zijn er bijvoorbeeld wel hostingkosten voor een stagingomgeving en eventuele developmentomgeving. Ook als de applicatie nog niet gelanceerd is.

Ook als expliciet gekozen is om geen serviceovereenkomst af te sluiten is het beschikbaar hebben van de juiste hostingdiensten en tools noodzakelijk. Deze worden ook zelfs zonder serviceovereenkomst volgens onze reguliere wijze doorgerekend.

Het stoppen van periodieke kosten bij ontwikkelingsstagnatie

Als weinig ontwikkeling plaatsvindt beperken we de periodieke kosten voor zover dat mogelijk is. Een product met een productieomgeving houdt hiervoor de kosten, maar als er geen productieomgeving nodig is en geen ontwikkeling plaatsvindt, zijn de maandelijkse kosten beperkt.

Moet de periodieke factuur toch helemaal gestaakt worden? Wellicht omdat de ontwikkeling lang op een laag pitje zal staan of helemaal stopt, dan kan sunsetten worden overwogen. *Meer over het sunsetten van een softwareproduct in een volgend hoofdstuk.*

9.5 Toegang tot beheeromgevingen en monitoringsystemen

Soms wordt ons gevraagd om toegang te verschaffen tot de verschillende hosting- en toolingomgevingen aan bijvoorbeeld de producteigenaar. Dit kan nodig zijn om bijvoorbeeld de afrekening van cloudhostingkosten over te nemen.

In de meeste gevallen willen we graag de toegang beperkt houden, omdat dit bijdraagt aan het zelfsturende karakter van ons team. Daarbij is het beperken van de toegang goed voor de algehele veiligheid van de applicatie. Sommige meldingen uit monitoringssystemen zijn onschuldig, en het is niet noodzakelijk dat alle meldingen met spoed opgepakt moeten worden. Beperkte toegang voorkomt zo bijvoorbeeld onnodige spoedcommunicatie over onschuldige meldingen.

Als het blijkt dat iemand via de opdrachtgevende organisatie verstorende beheertaken heeft uitgevoerd in de applicatie-infrastructuur vervalt de mogelijkheid om aanspraak te maken op een garantieregeling. Mocht het toch gebeuren dat er beheertaken gedaan worden door personen buiten Label305, laat ons dan weten wat er precies gebeurd is zodat wij op de hoogte zijn en goede service kunnen blijven leveren.

10 Facturatie en betaling

Vertrouwen is de basis van onze relatie met opdrachtgevers. Zij vertrouwen erop dat wij ons uiterste best doen om een goed softwareproduct neer te zetten.

Daartegenover staat dat wij de opdrachtgever onder andere vertrouwen de uiteindelijke betaling volgens afspraak te voldoen. Wij brengen onze werkzaamheden, inspanningen, ingekochte diensten en serviceverlening in rekening middels facturen. Hierover geven we graag wat extra toelichting zodat er geen onduidelijkheid over is.

10.1 Facturatie van sprints en losse werkzaamheden

Wij factureren normaal gesproken onze projectwerkzaamheden **achteraf** en sturen facturen altijd met een ruime betalingstermijn van **30 dagen**. Bij doorlopende ontwikkeling sturen we **elke twee weken** een factuur voor de dan opgeleverde sprint(s) en uitgevoerde losse werkzaamheden. Bij enkele sprints of enkele losse werkzaamheden sturen we in de week na het uitvoeren een factuur.

10.1.1 Aanbetaling voor aanvang

Om het initiële vertrouwen te vestigen werken we met een aanbetaling voor aanvang van de eerste ontwikkelwerkzaamheden. Dit doen we bij aanvang van een nieuw project voor een nieuwe opdrachtgever. Hierbij gaat het om **de eerste vier weken aan werkzaamheden**. Als werk is ingepland in éénweekse sprints gaat dit om de eerste vier sprints. De aanbetaling moet voldaan zijn voor aanvang van de eerste werkzaamheden.

Bij de aanvang van een productontwikkelingsproces organiseren wij vaak een kick-offmeeting. Deze kick-offmeeting hoeft niet aanbetaald te worden. We brengen deze kick-offmeeting achteraf in rekening. Meestal volgen de eerste sprints of losse werkzaamheden, die met een aanbetaling voldaan moeten worden, een paar weken na deze kick-offmeeting. We sturen de factuur voor aanbetaling kort na de kick-offmeeting. Op deze factuur wordt vaak meteen de kick-offmeeting meegenomen.

Als de aanbetaling niet volledig voldaan is voor de maandag waarop de eerste sprint zou aanvangen, wordt niet gestart met de werkzaamheden. Het volledig opschuiven van de planning is hierbij niet vanzelfsprekend, aangezien vaak andere werkzaamheden voor andere opdrachtgevers zijn ingepland na de sprints die moesten worden aanbetaald.

Bij het uitblijven van de aanbetaling vervalt voor de eerste sprint onze opleverings- en inspanningsverplichting, maar moet uiteindelijk alsnog voor de eerste sprint betaald worden. We halen in overleg verdere ingeplande sprints uit de planning. Mocht besloten worden alsnog te starten bij binnenkomst van de aanbetaling, maar is onverhoopt nog niet betaald tegen de tijd dat de volgende sprint zou aanvangen, dan geldt voor volgende sprints hetzelfde als voor de eerste. Wederom vervalt de

opleverings- en inspanningsverplichting en moet uiteindelijk alsnog voor de sprint betaald worden. Zodra we andere sprints, die aanbetaald zouden worden, uit de planning halen hoeft voor deze sprints niet betaald te worden.

De aanbetalingsperiode geldt alleen voor nieuwe opdrachtgevers. De eerste werkzaamheden voor een nieuw project van een bestaande opdrachtgever behoeven geen aanbetaling.

Na de aanbetalingsperiode worden opgeleverde sprints en uitgevoerd los werk elke twee weken achteraf gefactureerd.

10.1.2 Sprintgarantie, facturen en betalingen

Indien geldig aanspraak gemaakt wordt op de sprintgarantie wachten wij met het sturen van de factuur voor de sprint waarover het gaat. Indien de factuur reeds verstuurd is, hoeft deze pas betaald te worden zodra de problemen zijn opgelost of alle kosteloze compensatiewerkzaamheden zijn uitgevoerd.

10.2 Facturatie van service, hosting en tooling

Voor onze serviceovereenkomsten wordt gewerkt met vooraf ingekochte service-uren en een inschatting voor andere kosten. Achteraf worden de daadwerkelijke uren en kosten vervolgens verrekend in een volgende factuur. Die volgende factuur bevat dan ook tevens de inschattingen voor de volgende maand.

Naast servicewerk wordt ook een voorschot voor hosting- en toolingkosten gefactureerd aan het begin van de maand. Aan het begin van het kwartaal wordt een afrekening meegenomen van het vorige kwartaal en het maandelijkse voorschot bijgesteld.

De facturatie van service, hosting en tooling vindt gezamenlijk plaats op onze **servicefacturen**, die meestal **aan het begin van elke maand** verstuurd worden. Er kan ook gekozen worden voor servicefacturen per kwartaal in plaats van per maand.

10.2.1 Automatische incasso

Normaal gesproken worden facturen voor sprints en losse werkzaamheden handmatig overgeboekt naar onze rekening. Omdat servicefacturen maandelijks betaald moeten worden en relatief klein zijn vergeleken bij onze andere facturen willen we hiervoor de administratieve last laag houden. Daarom incasseren we deze facturen vrijwel altijd automatisch. Dit gebeurt vaak een paar dagen nadat de factuur verstuurd is.

Wij incasseren in principe standaard via SEPA Core-incasso's, dit betekent dat er niet speciaal in de internetbankieromgeving van de opdrachtgever een incassocontract geregistreerd hoeft te worden. Ook kan bij SEPA Core-incasso's tot 8 weken na de incasso zonder opgaaf van reden gestorneerd worden. Indien gewenst kunnen wij ook

een SEPA B2B-incassocontract opstellen, hiervoor moet een speciale registratie gedaan worden in de internetbankieromgeving van de opdrachtgever.

Bij de eerste offerte vragen we ook meteen om een SEPA-incassomachtiging te tekenen. Hiermee worden wij geautoriseerd om **alleen servicefacturen** maandelijks of per kwartaal automatisch te incasseren. Facturen voor sprints en losse werkzaamheden moeten handmatig overgeboekt worden.

Bij uitzondering is het mogelijk om elke servicefactuur handmatig te betalen. Bijvoorbeeld als de opdrachtgevende organisatie een inkoopbeleid heeft dat geen SEPA-incasso betalingen accepteert. Het handmatig betalen van servicefacturen heeft niet onze voorkeur.

Mocht een SEPA-incasso mislukken of worden gestorneerd, dan vragen we alsnog het verschuldigde bedrag van de servicefactuur handmatig over te boeken.

10.3 Betalingsverzuim

Als na het verlopen van de betalingstermijn van 30 dagen niet betaald is, zitten wij in een vervelende situatie waar we snel uit moeten komen. Mocht iets onduidelijk zijn, klopt iets niet, of ben je als opdrachtgever ergens ontevreden over, laat dit dan meteen weten. We proberen dan snel tot een goede oplossing te komen.

Afgezien van onze garantieregelingen, gaan voorstellen vanuit de opdrachtgever om pas te betalen nadat extra werk gedaan is in tegen onze manier van werken. Wij hebben er vertrouwen in dat alle betrokkenen op de hoogte zijn van onze werkwijze.

We sturen na het verlopen van de betalingstermijn een eerste herinneringse-mail waarin gevraagd wordt binnen 7 dagen te betalen. Mocht er niet binnen 7 dagen betaald zijn, dan sturen we opnieuw een e-mail. Daarnaast nemen we ook telefonisch contact op met de producteigenaar om te vragen binnen 7 dagen de factuur te voldoen.

Als een factuur **14 dagen** na het verlopen van de betalingstermijn (44 dagen na facturatie) nog niet betaald is, halen wij alle reeds ingeplande werkzaamheden uit de planning en plannen we ook geen nieuw werk meer in. Aangevragen sprints worden nog wel uitgevoerd.

Als een factuur **45 dagen** na het verlopen van de betalingstermijn (75 dagen na facturatie) nog niet betaald is stellen we de debiteur **in gebreke**, brengen we de wettelijke **incassokosten** in rekening, gaan we de wettelijke **handelsrente** heffen over het openstaande bedrag en schakelen we een **incassobureau** in. Daarnaast brengen we per direct alle nog niet in rekening gebrachte uitgevoerde losse werkzaamheden, servicewerkzaamheden en aangevragen sprints in rekening. Wij worden hierbij ook direct gevrijwaard van de inspannings- en opleveringsverplichting voor deze sprints.

10.3.1 Betalingsverzuim servicefacturen

Servicefacturen keren maandelijks terug, daarom maken we het betalen van deze facturen gemakkelijk met automatische incasso's. Mocht het voorkomen dat deze facturen niet op tijd betaald worden, dan zijn er nog een paar extra zaken waarmee rekening gehouden moet worden, bovenop de zaken genoemd in de vorige paragraaf.

Als **14 dagen** na het verlopen van een servicefactuur (44 dagen na facturatie) nog niet betaald is kan geen aanspraak meer gemaakt worden op SLA-garantie en wordt er geen servicewerk meer uitgevoerd.

Servicewerk en de inkoop van hosting en tooling moet uitgevoerd worden om het softwareproduct beschikbaar te houden voor eindgebruikers. Als verzuimd wordt te betalen, staken wij op een gegeven moment met de inkoop van de diensten bij derde partijen. Om dit netjes af te handelen moet het betreffende softwareproduct gesunset worden, hierbij worden alle gegevens veilig bewaard zodat het softwareproduct eventueel later opnieuw gestart kan worden. *Meer over sunsetten in een volgend hoofdstuk.*

Als **45 dagen** na het verlopen van een servicefactuur (75 dagen na facturatie) nog niet betaald is, stellen we de debiteur **in gebreke**, brengen we de wettelijke **incassokosten** in rekening, gaan we de wettelijke **handelsrente** heffen over het openstaande bedrag en schakelen we een **incassobureau** in. We worden gevrijwaard van de inspanningsverplichting voor de ingekochte service-uren en het openstaande servicetegoed. Hiernaast starten we met het **sunsetten** van het softwareproduct en met de applicatie-infrastructuur offline te halen. Hierdoor zou het kunnen dat (delen van) het product onbruikbaar worden voor eindgebruikers. De werkzaamheden die noodzakelijk zijn voor het sunsetten van de applicatie worden in rekening gebracht tegen het **hoge uurtarief** en ook meteen gefactureerd. Ook factureren we alle nog niet in rekening gebrachte (deels) uitgevoerde werkzaamheden. Wij worden hierbij ook meteen gevrijwaard van de inspannings- en opleveringsverplichting voor reeds aangevangen sprints.

11 Intellectueel eigendom

Wij willen het succes van onze opdrachtgevers faciliteren. Dit betekent onder andere dat onze opdrachtgever de eigenaar moet kunnen zijn van de ontwerpdocumenten, de UI-ontwerpen en de programmacode in het softwareproduct. Dit is vooral belangrijk bij het vinden van investeerders, het aangaan van partnerschappen, het in laten kopen van nieuwe deelnemers en het volledig verkopen van het product. In deze gevallen vindt *due diligence* plaats waarbij het bezit van het intellectueel eigendom een uitermate belangrijk onderdeel is.

11.1 Waar het om gaat

Onder het intellectueel eigendom (IP) verstaan wij: het auteursrecht, het domeinnaamrecht, het tekeningenrecht en het databankenrecht. En het 'werk' waarvoor het intellectueel eigendom kan worden overgedragen aan onze opdrachtgever is het volgende:

- Een deel van de geschreven code in het codedepot (repository) van het softwareproduct, hierover straks meer toelichting;
- Architectuur en ontwerpdocumenten die exclusief voor het product zijn uitgewerkt en zijn opgeleverd;
- Opgeleverde grafische ontwerpen van de interface van het product;
- De opgeleverde digitale huisstijl van het product, inclusief ontworpen merkuitingen en domeinnaamregistraties;
- Ontwikkelde databanken met vergaarde gegevens;
- Geschreven teksten in de opgeleverde applicatie; en
- Opgeleverde documentatie specifiek geschreven voor het product.

Voor alle andere zaken dragen we het intellectueel eigendom *niet* over.

11.2 Overdracht

Onze opdrachtgevers ontvangen automatisch het intellectueel eigendom van het werk dat wij creëren. Dit betekent dat als wij ontwerpdocumenten schrijven, UI-ontwerpen schetsen of software ontwikkelen, de opdrachtgever hier uiteindelijk de volledige eigenaar van wordt.

Dit doen wij op een manier waarbij de opdrachtgever alle vrijheid krijgt maar onze werkwijze niet onnodig belemmerd wordt. Deze balans is lastig om te vinden. Voor zover wij weten zijn wij als bureau uitzonderlijk in het aanbieden van een dergelijke automatische overdrachtsregeling zonder onnodige belemmeringen.

De overdracht vloeit voort uit onze algemene voorwaarden. Hierin staat dan ook genoemd wanneer overdracht voor afzonderlijke onderdelen van het 'werk' plaatsvindt. Er hoeft geen speciaal document te worden ondertekend. Mocht het wenselijk zijn alsnog een expliciete overeenkomst te bezitten, dan kunnen wij alsnog een

overdrachtsovereenkomst opstellen onder exact dezelfde voorwaarden en gezamenlijk ondertekenen.

11.2.1 Overdracht vindt plaats na betaling

Het intellectueel eigendom kan pas als overgedragen worden beschouwd zodra alle financiële verplichtingen aan ons zijn voldaan. Simpel gezegd heeft de overdracht pas plaatsgevonden zodra alle rekeningen betaald zijn.

Dit betekent dat het intellectueel eigendom in delen wordt overgedragen bij doorontwikkeling. Bij nieuw werk of nieuwe aanpassingen aan het oude werk dragen we het intellectueel eigendom hiervan over zodra alle rekeningen hiervoor betaald zijn.

Wij dragen in principe het eigendom over aan de organisatie waarnaar wij de factuur sturen. Moeten wij de factuur sturen naar een holding of aparte stichting? Dan wordt het intellectueel eigendom hiernaar overgedragen.

11.2.2 Licentie terug naar ons

De overdracht van het intellectueel eigendom gebeurt altijd onder voorwaarden dat wij bij de overdracht direct een brede overdraagbare sublicentieerbare en eeuwigdurende licentie op het gebruik van het werk terugontvangen. Hiervoor hoeven wij niets te betalen. Daarnaast krijgen wij expliciet toestemming om het werk op te nemen in ons portfolio. Het afgeven van een licentie vloeit net als de overdracht voort uit onze algemene voorwaarden.

11.2.3 Geen automatische overdracht bij partnerschappen

We hebben een belangrijke uitzondering waarbij wij het intellectueel eigendom *niet* automatisch overdragen.

Wij maken bij nieuwe opdrachtgevers geen speciale afspraken over deelneming of diep partnerschap. Soms ontstaat er een partnerschap met directer belang na een lange periode van samenwerken. Bij een samenwerking waarbij wij direct belang hebben in het succes van het door ons ontwikkelde product maken we aparte afspraken over het de overdracht van het (nieuwe) intellectueel eigendom.

We bedoelen hier partnerschappen anders dan een normale klantrelatie, zoals bijvoorbeeld:

- Wij bezitten aandelen of opties in de opdrachtgevende organisatie;
- Wij hebben een afspraak over omzet- of winstdeling; of
- Wij hebben samen aanspraak gemaakt op een investering, grant of subsidie.

Als hetgeen hierboven beschreven is nog niet heeft plaatsgevonden maar wel redelijkerwijs in het verschiet zou kunnen liggen, dragen we het intellectueel eigendom voor nieuw werk ook niet automatisch over.

Bij een partnerschap maken we altijd aparte afspraken over de eventuele (toekomstige) overdracht. Als die afspraken niet gemaakt zijn blijft het intellectueel eigendom bij ons liggen tot deze gemaakt worden. Als eerder al wel een deel van het intellectueel eigendom is overgedragen en de partnerschapssituatie ontstaat, dan blijft enkel het intellectueel eigendom van nieuw werk en nieuwe aanpassingen aan het oude werk bij ons liggen totdat een concrete afspraak is gemaakt.

11.3 Toelichting voor code

De rechten die beschikbaar zijn voor overdracht gelden voor een deel van de code in repositories specifiek voor het project. Soms kunnen dit er meerdere zijn, zo kan het zijn dat er een repository is voor de webapplicatie en een andere repository voor de Androidapplicatie.

Wij kunnen niet alle rechten overdragen omdat simpelweg niet alle intellectuele eigendommen van alle stukken code in die repositories bij ons liggen. Zo kunnen wij er soms selectief voor kiezen om code te gebruiken die wij onder licentie hebben, dit kunnen zowel vrije open source licenties als beperktere licenties zijn. Daarnaast zijn er delen van de code zijn waar wij wel het zelf intellectueel eigendom van hebben en willen behouden. Zo zijn sommige delen van de code door ons uitgegeven onder een open source licentie. Het intellectueel eigendom hierop dragen wij niet over, omdat de open source licentie de opdrachtgever ook alle noodzakelijke vrijheid geeft.

Bij Label305 staan wij achter open source – dit is een van onze kernwaarden. Wij willen de mogelijkheid hebben om interessante of deelbare code uit te geven onder een open source licentie en publiek beschikbaar te maken. Dit doen wij ook terwijl we deze code voor onze opdrachtgevers aan het schrijven zijn, en dus voor onze opdrachtgevers aan het werk zijn. Het is aan ons om te beslissen met welke code wij dit doen. Met het uitgeven van code onder een open source licentie kunnen wij de code gebruiken in andere projecten en kan de wereldwijde programmeergemeenschap er hun voordeel mee doen.

We begrijpen dat code die wij open source beschikbaar stellen geen bedrijfskritische code moet zijn, hier waken wij dan ook altijd voor. De rechten voor de code zijn dus beschikbaar voor overdracht of de code wordt uitgegeven onder een open source licentie. We geven zelf geen code aan opdrachtgevers uit onder een beperkte propriëtaire licentie.

Naast code in repositories specifiek voor een project worden ook zogenaamde afhankelijkheden (dependencies) naar softwarebibliotheken (libraries) in andere repositories opgenomen. Deze afhankelijkheden stellen ons in staat om significant sneller software te ontwikkelen. Wij bouwen namelijk op een enorme berg kennis en abstracties die al ontwikkeld zijn door anderen in de programmeergemeenschap. Het product kan zonder deze afhankelijkheden niet functioneren.

De afhankelijkheden van het softwareproduct kunnen verwijzen naar bibliotheken die geschreven of aangepast zijn door Label305, ook deze bibliotheken zijn open source beschikbaar. Verreweg de meeste afhankelijkheden zijn geschreven door anderen in de programmeergemeenschap, en de auteurs daarvan hebben dan ook zelf de controle over het intellectueel eigendom en de licentie.

Soms kiezen we ervoor om code uit een projectrepository los te halen naar een aparte bibliotheek voor eventuele toekomstige open source publicatie. Mochten we toch kiezen om een library niet (meteen) te publiceren dan stellen we deze nog wel voor de opdrachtgever onder een open source licentie beschikbaar.

11.3.1 Praktische implicaties

Terwijl wij bezig zijn met het ontwerpen en creëren van het softwareproduct willen wij ons niet onnodig continu bezig houden met vragen omtrent intellectueel eigendom en licenties. Dit zou wegnemen van de focus en onnodige vertraging opleveren.

Wij volgen een werkwijze die gebruikelijk is in de softwareontwikkelingsindustrie waarbij wij bruikbare open source code gebruiken en integreren. Wij vragen in principe geen expliciete toestemming voor het toepassen van open source code in een project voor een opdrachtgever.

Wij vinden het belangrijk dat we melding maken naar gebruikte open source projecten en bijbehorende licenties, zowel in de code als in de applicatie zelf. Een dergelijke vermelding wordt ook wel een attributie genoemd. We moeten bij ontwerpen en het ontwikkelen van de applicatie-interface hiermee rekening houden. Onze opdrachtgevers moeten rekening houden met de extra tijd noodzakelijk voor het bijhouden van deze attributies.

Het kan voorkomen dat code die wij onder licentie hebben/nemen of zelf open source uitgeven niet altijd correct geannoteerd is in de codebase. Natuurlijk pogen wij dit wel altijd correct en volledig te doen.

Je kan als opdrachtgever ons vragen of wij een versie van de codebase beschikbaar kunnen maken waarbij alle code expliciet correct geannoteerd is. Wij gaan dan hiermee bezig. Ook kunnen wij zorgvuldig controleren of alle afhankelijkheden nog steeds een juiste en vrije open source licentie hebben. Het zorgvuldig controleren van alle code en afhankelijkheden kost natuurlijk tijd. Er kunnen alleen vlak na een dergelijke controle rechten ontleend worden aan de codebase.

Als kosten verbonden zijn aan het aanschaffen van een noodzakelijke licentie worden deze aan de opdrachtgever doorgerekend. Dit kan bijvoorbeeld gebeuren bij een specifieke library. Hiervoor plegen we eerst overleg.

11.3.2 Toegang tot de code

Wij geven geen continue toegang tot onze versiebeheeromgeving aan personen die niet direct onderdeel zijn van het team. Dit doen wij om ervoor te zorgen dat ons team zelfsturend kan opereren. De opdrachtgever moet als intellectueel eigendomshouder toegang kunnen hebben tot de code. Daarom kunnen wij code bij elke oplevering publiceren naar een repository in het beheer van de opdrachtgever. Het is aan hem/haar om dit te organiseren, maar wij kunnen hierbij assisteren. Wij behouden het recht om in dergelijke situaties de code pas beschikbaar te stellen zodra het intellectueel eigendom daadwerkelijk is overgedragen.

11.4 Toelichting voor audio, video en grafische elementen

Bij het ontwerpen van een geweldige gebruikerservaring gebruiken we vaak audio, video, lettertypen, iconen, foto's, tekeningen en andere grafische elementen waar wij zelf niet de intellectuele eigendomsrechten van hebben. Hierbij controleren we altijd de licentie en zorgen we ervoor dat we de elementen mogen gebruiken.

Als een licentie moet worden aangeschaft rekenen we deze aan de opdrachtgever door. Dit kan bijvoorbeeld gebeuren als wij stockfoto's of lettertypen aanschaffen. We vragen in principe geen expliciete toestemming voor het toepassen van grafische elementen uitgegeven onder een *creative commons* licentie of vergelijkbare licentie.

Ook voor grafische elementen geldt dat de opdrachtgever aan ons kan vragen om een gedetailleerd verslag waarin staat voor welke onderdelen de intellectuele eigendommen bij de opdrachtgever liggen, en voor welke onderdelen gebruik wordt gemaakt van een licentie. Wij zoeken dit dan grondig gaan uit, wederom kan dit enige tijd in beslag nemen.

12 Ontwikkelingstransitie en participatie

Naast het automatisch overdragen van intellectueel eigendom doen wij nog meer om het succes van onze opdrachtgevers te faciliteren. We willen dat het succes van het product kan blijven groeien, en dat betekent ook dat het product ons zou moeten kunnen ontgroeien.

Bij aanzienlijke groei is op een gegeven moment een groter specialistisch productteam nodig dan dat wij kunnen bieden. In die situatie helpen we door het starten van een *ontwikkelingstransitieproces*. Dit is een proces waarbij onze opdrachtgever een eigen productteam gaat opbouwen. Wij assisteren de opdrachtgever met het doorlichten van kandidaten. Na goedkeuring werken zij tijdelijk bij ons intern mee aan de productontwikkeling. Zij leren daarbij alles over onze werkwijze en krijgen alle benodigde projectinhoudelijke kennis. Nadat een ontwikkelaar geruime tijd met ons intern heeft meegedraaid kunnen we op afstand blijven samenwerken. Naar verloop van tijd en met een groot genoeg productteam kan de verdere toekomstige ontwikkeling geheel intern worden georganiseerd en uitgebouwd.

12.1 Voorwaarden ontwikkelingstransitie

Net als productontwikkeling is ook ontwikkelingstransitie een ingewikkeld proces waarbij duidelijkheid moet zijn over de verantwoordelijkheid van alle betrokken partijen. Wanneer je als opdrachtgever nadenkt over een ontwikkelingstransitietraject neem dan contact met ons op. Zo kunnen we hier samen een passend plan voor maken. Wij hebben wel een paar vereisten voordat we kunnen starten met een traject:

- Een ontwikkelingstransitietraject kan alleen plaatsvinden gedurende een periode van continue doorontwikkeling;
- Daarbij moet een eerste versie van het product gelanceerd zijn en moet een aanzienlijke gebruikersgroep aanwezig zijn, zodat wij alle aspecten van productontwikkeling kunnen behandelen bij het ontwikkelingstransitietraject;
- Deelnemers moeten door ons worden doorgelicht zodat wij kunnen zien of zij goede productspecialisten zijn, of dat kunnen worden;
- Deelnemers moeten volledig beschikbaar zijn en mee kunnen draaien in ons team op ons kantoor gedurende het gehele proces; en
- Het doel moet zijn om een eigen intern productontwikkelingsteam op te starten dat uiteindelijk de volledige procesverantwoordelijkheid op zich neemt, hiervoor zijn vaak meerdere specialisten nodig.

Er moet ruimte bij ons zijn om een ontwikkelingstransitietraject uit te voeren. Het zou goed kunnen dat dit niet direct kan omdat we andere trajecten hebben lopen die we eerst dienen af te ronden.

12.2 Het proces voor een deelnemer

Voor de ontwikkelingstransitie doorlopen we met elke kandidaat voor het nieuwe interne team een programma. Als kandidaten na acceptatie kunnen beginnen starten ze als deelnemer aan het programma. De deelnemers zijn in dienst van onze opdrachtgever, wij verzorgen enkel de begeleiding tijdens het programma. Vooral bij de eerste deelnemer die het proces doorloopt is het extra belangrijk dat alles soepel gaat, omdat dit meestal de persoon is die een leidende rol krijgt in het interne team van onze opdrachtgever.

12.2.1 Selecteren en doorlichten

Het proces begint bij het selecteren van een kandidaat. Bij het selecteren is de opdrachtgever verantwoordelijk voor het aandragen van goede kandidaten. Dit kan bijvoorbeeld door deze kandidaten zelf te werven, of door ze te werven via een recruitmentbureau. Wij kunnen assisteren bij het organiseren van de werving van goede kandidaten. Wij geven aan welke capaciteiten een kandidaat moet hebben, hierbij gaat het onder andere om technische en ook sociale vaardigheden, zodat een goede vacature opgesteld kan worden. Over de jaren heen hebben wij ervaring opgedaan met werving en daarom kunnen wij verschillende handige tips geven tijdens dit proces.

Het selecteren van goede kandidaten voor een team blijft een subjectief proces. Daarom blijft het lastig om 100% zeker te zijn van een kandidaat, bovendien is het mogelijk dat we de verkeerde beslissing maken om een kandidaat wel of niet aan te nemen. Uiteindelijk maakt onze opdrachtgever de beslissing. Wij proberen te helpen bij het zo goed mogelijk selecteren en doorlichten.

Cv's selecteren

Het doorlichten van kandidaten begint bij het goed- of afkeuren van cv's, dit doen wij samen met de opdrachtgever. Als kandidaten niet geschikt zijn, laten we dat weten en geven daarbij een goede onderbouwing. We zeggen nooit op basis van enkel een cv "ja" op een kandidaat, een cv zegt immers niet alles.

Zodra we het cv van een kandidaat goedkeuren is het belangrijk dat we in alle vervolgstappen helder zijn over het proces en de voorwaarden. Daarom is het goed om bij het eerste contact na acceptatie al het volledige proces uit te leggen aan een kandidaat. Waar moet een kandidaat zich op voorbereiden? Wat is de inhoud en indeling van de sollicitatiegesprekken? Wat voor arbeidsvoorwaarden kan een kandidaat verwachten? Het vinden van goede ontwikkelaars is uitdagend, daarom is het belangrijk dat we de juiste aandacht besteden aan potentiële kandidaten.

Sollicitatiegesprek

Afhankelijk van wat wij afspreken met de opdrachtgever voeren wij ook een groot deel van één van de sollicitatiegespreken. Dit kan in het eerste of in het tweede sollicitatiegesprek. We vertellen de kandidaten over Label305 en hun verantwoordelijkheden tijdens en na de ontwikkelingstransitie.

Met name bij de eerste kandidaat kijken we naar leiderschapsvaardigheden. Heeft de kandidaat het in zich om overkoepelende verantwoordelijkheid over code, ontwerp en het team op zich te nemen? Kan hij/zij zelf met nieuwe en doordachte ideeën komen?

Hierna vragen we de kandidaat naar verschillende aspecten van UX-ontwerp en softwareontwikkeling. We kijken hiermee of voldoende kennis aanwezig is. Tot slot nemen we een korte programmeertest af bij de kandidaat. Deze programmeertest geeft ons inzicht in het probleemoplossend vermogen en de werkwijze van een kandidaat. Meestal duurt dit alles bij elkaar, tussen 60 en 90 minuten.

Uiteindelijk geven wij na het sollicitatiegesprek aan of een kandidaat geschikt is. Indien wij een kandidaat echt ongeschikt vinden, zullen we ook niet het programma starten met deze kandidaat. De uiteindelijke beslissing over het aanbieden van een baan ligt bij de opdrachtgever.

Huidige medewerkers

Als er al medewerkers zijn die eventueel zouden kunnen deelnemen aan een ontwikkelingstransitietraject, dan lichten we deze personen, net zoals bij nieuwe kandidaten, door in een gesprek met behulp van dezelfde tests. Ook als we in dit geval iemand ongeschikt achten, geven we dit aan. We zullen dan niet starten met deze persoon aan het programma.

Aanbod

Uiteindelijk is het aan onze opdrachtgever om een kandidaat een aanbod te doen. Meestal bevat dit aanbod een goed salaris en goede secundaire voorwaarden. Soms gebeurt het dat de eerste paar productontwikkelingsmedewerkers opties krijgen op aandelen. Deze worden dan omgezet zodra ze een bepaalde periode in dienst zijn.

Kosten

Onze hulp bij het selecteren en doorlichten van kandidaten wordt meestal in rekening gebracht als losse werkzaamheden tegen het **hoge uurtarief**. We houden tijdens het gehele proces de uren bij, zowel voor het advies als voor het deelnemen aan sollicitatiegesprekken. In het geval van een participatie- of samenwerkingsovereenkomst kan een andere afspraak gemaakt worden.

12.2.2 Kennis maken in de eerste weken

Als onze opdrachtgever een kandidaat in dienst heeft genomen kunnen zij bij ons intern van start gaan met programma. Het is waarschijnlijk een goed idee om de nieuwe medewerker niet vanaf de eerste dag bij Label305 aan de slag te laten gaan. Het is beter dat de medewerker eerst één of twee weken bij de opdrachtgever kennis maakt met de organisatie. Laat eerst alles van het huidige bedrijfsproces zien, en laat hem/haar kennismaken met ins en outs van het softwareproduct.

Als deelnemers vervolgens aan de slag gaan bij Label305, is het verstandig om ze ook elke week één dag remote met ons samen te laten werken vanuit het kantoor van de opdrachtgever. Dit zorgt ervoor dat de deelnemers tijdens het programma continu een goede aansluiting heeft met het bedrijf, ook als ze bij ons intern mee-ontwikkelen. Bovendien biedt het ze de kans om vanaf het begin duidelijk uitzicht te hebben op de uiteindelijke werkwijze.

12.2.3 Opleiden en inwerken bij Label305

Om bekend te raken met onze ontwerp- en ontwikkelmethodieken gaat een deelnemer bij ons intern meedraaien voor een aantal maanden. Meestal ligt deze periode tussen de drie en negen maanden, afhankelijk van het niveau van de deelnemer en de huidige grootte van het ontwikkelteam van de opdrachtgever. De eerste deelnemer draait vaak langer bij ons intern mee. We vragen, vooral in de eerste paar maanden, dat een deelnemer een groot deel van de week bij Label305 op kantoor komt werken.

Een voorwaarde voor het ontwikkelingstransitieproces is dat gedurende het hele proces doorlopend een team van Label305 bezig is met de verdere ontwikkeling van het betreffende softwareproduct. Er zijn dus naast de deelnemers ook continu specialisten van Label305 met sprints voor het product bezig. Onze specialisten begeleiden deelnemers en betrekken ze volledig bij de ontwikkeling. Ze vertellen deelnemers over onze werkwijze en laten ze mee werken aan alle functies ingepland in de sprints. Dit kan een impact hebben op de velocity van de specialisten bij Label305.

Er wordt met de deelnemer gewerkt in de vorm van pair programming, zo kunnen wij meteen de deelnemer alles vertellen over de inhoud van de codebase terwijl we ook bezig zijn met het verbeteren van het softwareproduct. Een deelnemer krijgt ook alles mee van onze quality assurance- en ontwikkelprocedures, deze kunnen na de ontwikkelingstransitie daardoor gemakkelijk overgenomen en ingevoerd worden bij het interne ontwikkelteam.

Middelen

Wij zorgen ervoor dat een deelnemer een bureau, goede stoel en alle benodigde randapparatuur krijg. Denk hierbij aan professionele schermen, een goed toetsenbord en een goede muis. De deelnemer kan elke dag rekenen op inbegrepen lunch, fruit, snacks, koffie, thee en fris van Label305.

Wij kunnen ook een laptop en eventuele testapparaten regelen voor een deelnemer, deze brengen wij wel **in rekening** bij de opdrachtgever. Vanzelfsprekend zijn deze uitgebreider en prijziger dan regulier kantoorapparatuur. Zo is een goede en krachtige laptop belangrijk voor een productontwikkelaar. De apparatuur wordt eigendom van onze opdrachtgever.

Toegang

Zodra een deelnemer bij ons intern begint mee te draaien, wordt hij/zij gezien als volledig onderdeel van het team. De deelnemer krijgt daardoor ook automatisch toegang tot relevante omgevingen en systemen, inclusief de omgevingen waar de producteigenaar misschien geen toegang tot heeft. We vragen wel dat de deelnemer respecteert dat de procesverantwoordelijkheid bij ons ligt tot deze ook daadwerkelijk is overgedragen.

12.2.4 Remote samenwerken met de deelnemer

Nadat een deelnemer een tijd bij ons heeft meegedraaid kan deze grotendeels op afstand deelnemen aan het ontwikkelingsteam. Vanaf dat moment is het ook niet meer noodzakelijk dat wij continu met sprints bezig zijn. Wel nemen wij, tot de overdracht van de procesverantwoordelijkheid, nog alle projectmanagement- en beheertaken op ons. Effectief is een deelnemer dus voor de overdracht van de procesverantwoordelijkheid eerst nog een remote onderdeel van ons team.

Gedurende de weken dat de deelnemer bezig is met ontwikkelen, maar geen sprints zijn ingepland, moeten wij nog steeds veel doen. Zo geven we middels code- en gebruiksreview nog steeds feedback en denken we mee met de deelnemer. Daarnaast zijn we druk bezig om samen met de producteigenaar een plan op te stellen voor de ontwikkeling in die week en in de daaropvolgende weken. Deze werkzaamheden worden in het geval dat geen sprint loopt gezien als normale losse werkzaamheden.

Naast de ontwikkeling blijven wij, voorafgaand aan de overdracht, verantwoordelijk voor service en onderhoud. We houden dus gewoon de serviceovereenkomst aan en dit werk blijven we afhandelen zoals voorheen. Dit betekent ook dat wij nog steeds de controle moeten hebben over alle lanceringen naar de productieomgeving, anders kunnen wij immers geen adequate service meer blijven verlenen. Later in het proces kan dit veranderen. Als de procesverantwoordelijkheid wordt overgedragen, wordt ook het interne team verantwoordelijk voor serviceverlening en lanceringen naar de productieomgeving.

12.3 Opbouwen van een volledig intern team

Het doel van het ontwikkelingstransitieproces is om uiteindelijk een intern team te hebben dat een groot deel van de ontwikkeling, al het onderhoud en alle technische ondersteuning uitvoert. De grootte die het team moet hebben om dit echt te

verwezenlijken loopt sterk uiteen en is afhankelijk van veel factoren, maar meestal moet gedacht worden aan minimaal drie personen.

Gedurende de gehele ontwikkelingstransitie doorlopen dus minimaal drie personen het programma. Eerst werken zij intern, en daarna remote met ons mee. Als op een gegeven moment het team van de opdrachtgever groot genoeg is, dragen we de procesverantwoordelijkheid over.

Het groeien van het team moet geleidelijk gebeuren. We willen iedereen genoeg aandacht kunnen geven. Het is alleen niet noodzakelijk dat de eerste deelnemer helemaal klaar is met het meedraaien bij ons voordat de volgende deelnemer kan beginnen.

Het is niet altijd vanzelfsprekend dat een deelnemer kan beginnen, we moeten per slot van rekening voldoende plek hebben. Dit stemmen we dus goed af met de opdrachtgever.

Zodra meerdere teamleden aan de slag zijn bij de opdrachtgever kunnen zij eerst met ons mee blijven draaien volgens onze werkwijze. Zij kunnen wel alvast gaan nadenken over hun eigen proces. We raden aan om onze werkwijze hierin als basis te nemen en ook al na te denken over verbeteringen die juist binnen de organisatie van de opdrachtgever handig zouden zijn. Als het interne team er klaar voor is, dragen we de procesverantwoordelijkheid over en is de ontwikkelingstransitie compleet.

12.3.1 Overdracht van de procesverantwoordelijkheid

Als we uiteindelijk samen met de opdrachtgever besluiten dat het team bij de opdrachtgever klaar is om alle procesverantwoordelijkheid op zich te nemen, dragen we deze over. Dit betekent effectief dat het team van de opdrachtgever de projectmanagementverantwoordelijkheden op zich neemt. Zij werken dan in detail nieuwe functies en taken uit – samen met de producteigenaar zonder tussenkomst van ons. Wij blijven betrokken en kunnen adviseren.

Vanaf het moment dat de procesverantwoordelijkheid is overgedragen, starten wij ook geen begeleidingsproces meer met nieuwe deelnemers. De groei, het selecteren en het opleiden van teamleden is dan volledig in handen van de opdrachtgever. Wij kunnen nog wel steeds als adviseurs ingeschakeld worden voor zowel ontwikkeling als voor meer procesmatige zaken.

Voordat de overdracht plaatsvindt, is het belangrijk dat wij de tijd hebben gekregen om alle code correct te annoteren als het gaat om intellectueel eigendom, zodat later hier ook geen onduidelijkheden over kunnen ontstaan.

Het team wordt verantwoordelijk voor alle hosting- en toolingomgevingen. Hiermee krijgen zij alle monitoringsystemen in beheer, en moeten ze zelf ingrijpen als iets fout gaat. Ook worden zij beheerders van de versiebeheeromgeving en beslissen zij

uiteindelijk welke codewijzigingen geaccepteerd worden, en welke code gelanceerd wordt naar productieomgevingen. Met de overdracht van de procesverantwoordelijkheid staakt ook het doorberekenen van alle hosting- en toolingkosten. Het is vanaf dat moment noodzakelijk dat dit intern georganiseerd wordt. We helpen bij het overzetten van alle tooling.

De lopende serviceovereenkomst wordt beëindigd bij de overdracht van de procesverantwoordelijkheid. Wij blijven wel bereikbaar en beschikbaar voor vragen en advies. Ook kunnen wij te hulp schieten als problemen ontstaan. Dit wordt dan gezien als los werk.

12.3.2 Advies en begeleiding na de overdracht

Na de overdracht van de procesverantwoordelijkheid kunnen we nog ingeschakeld worden om verder te helpen met het product en om het proces van het interne team te versoepelen. Zo kunnen we nog steeds het nieuwe team blijven assisteren met de productontwikkeling door ontwikkelsprints uit te voeren.

Soms wordt ook een nieuw project gestart, voor het ontwikkelen van een nieuw product. Hiervoor kan wederom onze reguliere werkwijze worden aangehouden.

12.4 Vergoeding of participatie

Het wordt ons vaak gevraagd maar Label305 participeert **niet bij aanvang** van de productontwikkeling. Onze opdrachtgever neemt altijd het productrisico op zich. We kunnen een ontwikkelingstransitietraject aanvangen zodra we beiden zien dat het product zich bewezen heeft bij de eerste klanten en verdere groeipotentie heeft. Wij zien dan ook de aanvang van een ontwikkelingstransitieproces als het eerste moment waarop wij participatie of een andere diepe samenwerking kunnen overwegen.

12.4.1 Vergoeding voor ontwikkelingstransitie

Normaalgesproken vragen wij een vergoeding voor het begeleiden van deelnemers gedurende het ontwikkelingstransitieproces. Dit is aan de orde als wij niet gaan participeren. De vergoeding komt bovenop de vergoeding voor de noodzakelijke continue doorlopende ontwikkeling tijdens het transitieproces.

€ ████,-* per deelnemer per week, exclusief btw

Deze vergoeding loopt door gedurende de weken waarin een deelnemer bij ons intern meedraait. De vergoeding is **niet afhankelijk** van het aantal dagen in een week dat een deelnemer bij ons zit. Zodra een deelnemer volledig remote gaat werken, wordt deze vergoeding niet meer in rekening gebracht.

Naast de vergoeding rekenen wij het **hoge uurtarief** voor alle losse werkzaamheden die noodzakelijk zijn voor het organiseren van een ontwikkelingstransitietraject. Dit geldt bijvoorbeeld voor het begeleiden en deelnemen aan de sollicitatiegesprekken, en ook het samenwerken met het productontwikkelingsteam van de opdrachtgever voor de overdracht van de procesverantwoordelijkheid.

** Er kunnen geen rechten worden ontleend aan dit tarief en het is dus enkel ter indicatie. Bovendien is de vergoeding voor ontwikkelingstransitie onderhevig aan eventuele prijsverhogingen net als onze andere tarieven.*

12.4.2 Participatie

Een alternatief voor de vergoeding is het starten ontwikkelingstransitietraject en tegelijk een participatieovereenkomst aangaan. Wij staan hier soms voor open, maar dit hangt af van veel verschillende factoren. Als de opdrachtgever en wij beiden geïnteresseerd zijn in participatie, moeten we hierover meerdere keren uitgebreid om tafel gaan zitten.

Onze insteek bij participatie is altijd dat wij onszelf niet zien als durfkapitalist en we enkel beperkte risico's willen nemen. Wij doen graag waar we van houden, en willen dat graag blijven doen. Een investering moet inhoudelijk en zakelijk interessant zijn, en een aanzienlijke kans hebben om zichzelf binnen een redelijke periode terug te verdienen. We participeren dus nooit bij aanvang van een productontwikkelingsproces. Ook zien we nooit reguliere werkzaamheden volledig als investering, dit doen we maximaal tot 25% van onze normale tarieven.

Bij participatie kunnen verschillende vergoedingen die gepaard gaan met het ontwikkelingstransitietraject gezien worden als investering en **100%** aan ons vergoed worden in de vorm van aandelen. Hierbij moet gedacht worden aan alle vergoedingen besproken in de vorige paragraaf zoals de wekelijkse vergoeding en de werkzaamheden noodzakelijk voor de organisatie van het traject. Voor de ontwikkelingskosten, zoals sprints, vragen we altijd om alsnog 75% via de reguliere facturen te voldoen. Zo kan ook **25%** van de toekomstige ontwikkelkosten gezien worden als investering, en worden vergoed in de vorm van aandelen.

Wij ontvangen bij participatie altijd aandelen in het bedrijf dat al het intellectueel eigendom van het product in zijn bezit heeft. Meestal is dit ook het bedrijf dat de directe baten van de verkoop van het product ontvangt en het uiteindelijke interne ontwikkelteam op de loonlijst krijgt. In een andere situatie kan dit gebeuren bij een 100% dochter van het bedrijf, bijvoorbeeld in het geval van een holdingmaatschappij. In dat geval vragen wij om aandelen van de holding.

Vaak spreken we voorafgaand aan een ontwikkelingstransitietraject af wat de precieze financiële omvang van het traject zal zijn. We stellen op basis van die omvang een investeringsbedrag vast. Daarnaast is er ook een bedrag dat voldaan wordt via

reguliere facturatie. Als het investeringsbedrag is vastgesteld moet door een onafhankelijke derde partij de waarde van het bedrijf worden bepaald, en in het verlengde daarvan de waarde van een aandeel.

Een sterk versimpeld voorbeeld: Een opdrachtgever wil een ontwikkelingstransitietraject starten. De waarde van het bedrijf is op dit moment € ██████████, onderverdeeld in ██████████ aandelen. We besluiten een traject te starten van één jaar waarin sowieso 50 éénweekse sprints worden afgenomen. In elk van deze sprints werkt een team van twee van onze specialisten aan het product. Tegelijkertijd stomen we gedurende het jaar drie deelnemers klaar voor een intern productontwikkelingsteam. We komen overeen dat in het aankomende jaar gezamenlijk € ██████████ aan diensten wordt afgenomen, hieronder vallen de sprints, een SLA en de diensten die gepaard gaan met het ontwikkelingstransitietraject. Van het totaalbedrag wordt € ██████████ voldaan via reguliere facturen. Daarnaast worden ██████████ nieuwe aandelen aan ons uitgeschreven t.w.v. € ██████████. Na afloop van het jaar is er door 5 mensen gewerkt aan de verbetering en verdere ontwikkeling van het product, is een goed draaiend intern productontwikkelingsteam opgezet en hebben wij een aandeel van 10% in het bedrijf achter het softwareproduct.

Zodra de ontwikkelingstransitie met participatieovereenkomst is afgelopen blijven wij nauw betrokken bij de organisatie. Bovendien blijven we hierna soms een gereduceerd tarief hanteren, ook na de vooraf overeengekomen transitieperiode.

12.4.3 Samenwerkingsovereenkomst

Soms is het door de bedrijfsstructuur van onze opdrachtgever lastig om een participatieovereenkomst aan te gaan. Het kan zijn dat we alsnog beiden geïnteresseerd zijn in een afspraak waarbij wij beiden langdurig profiteren van een succesvolle ontwikkelingstransitie. In een dergelijke situatie kunnen we een onder vergelijkbare voorwaarden werken als bij een participatieovereenkomst, maar maken we bijvoorbeeld een afspraak over omzetsdeling in plaats van het uitschrijven van aandelen. De omzetsdeling kan beperkt blijven tot de producten waar wij en het nieuwe team aan samenwerken. Zo kan een vaste vergoeding per verkocht product of maandelijkse vergoeding per abonnee worden overeengekomen.

Ook een samenwerkingsovereenkomst gaan we **niet** aan bij aanvang van de ontwikkeling, maar pas bij de start van een ontwikkelingstransitietraject. We moeten hiervoor, net als bij participatie, regelmatig uitgebreid om de tafel zitten. Uitgangspunt is hier opnieuw dat de afspraak inhoudelijk en zakelijk interessant moet zijn voor ons. Daarbij willen wij beperkte risico's nemen en moet er een aanzienlijke kans zijn dat de investering zichzelf binnen een redelijke periode terugverdient.

Een sterk versimpeld voorbeeld: Een opdrachtgever wil een ontwikkelingstransitietraject starten. Er is al een eerste versie van een B2B SaaS-

product gelanceerd en er zijn op dit moment 200 klanten die ieder € [REDACTED] per maand betalen voor het gebruik van het product. We besluiten een traject te starten van één jaar waarin sowieso 50 éénweekse sprints worden afgenomen, in elk van deze sprints werkt een team van twee van onze specialisten aan het product. Tegelijkertijd stomen we gedurende het jaar drie nieuwe deelnemers klaar voor een intern productontwikkelingsteam. We komen overeen dat in het aankomende jaar diensten t.w.v. € [REDACTED] worden afgenomen, hieronder vallen de sprints, een SLA en de diensten die gepaard gaan met het ontwikkelingstransitietraject. Van het totaalbedrag wordt € [REDACTED] voldaan via reguliere facturen en wordt afgesproken dat wij voor een periode van 5 jaar 4% van de terugkerende omzet uit het product ontvangen als tegenprestatie voor de investering van de overige € [REDACTED]. Na afloop van het jaar is er door 5 mensen gewerkt aan de verbetering en verdere ontwikkeling van het product en is een goed draaiend intern productontwikkelingsteam opgezet.

13 Sunsetting

Soms is het nodig om een softwareproduct dat in productie draait uit te faseren. Levert het ontwikkelde softwareproduct niet het gewenste resultaat? Is het lastig om het softwareproduct te verkopen, of de benodigde omzet te behalen? Wordt er ingezien dat dit ook met doorontwikkeling niet alsnog kan lukken, bijvoorbeeld door een sterk veranderde markt? Helaas zijn niet alle softwareproducten die worden ontwikkeld succesvol. Er moet soms besloten worden om te stoppen met de verdere ontwikkeling en ondersteuning van een product.

13.1 Sunsetprocedure

Wordt er besloten om te starten met sunsetting? Dan ondernemen we verschillende stappen om ervoor te zorgen dat het project netjes gearchiveerd kan worden. Dit gaat om zowel de verzamelde data als de ontwikkelde software. Bij sunsetting stopt automatisch de eventuele serviceovereenkomst.

We halen de applicatie-infrastructuur offline. Hierbij schrijven we een kopie van de data in de databases en blobstorage-diensten correct weg naar een data-archief. Apps die in de applicatiewinkels staan, maar niet meer werken met een applicatie-infrastructuur die offline is, halen we uit de applicatiewinkels.

Hou er rekening mee dat het uitvoeren van vergeet, wijzig en inzichtverzoeken veel meer tijd in beslag neemt zodra persoonsgegevens in een data-archief beland zijn. Het uitvoeren van een sunsetprocedure geeft geen vrijwaring van GDPR-verantwoordelijkheden.

We gaan nog een laatste keer over alle code heen om het intellectuele eigendom correct te annoteren. Bovendien kijken we voor de laatste keer of het technische readme-document up-to-date is. Dit document vertelt alles over de randvoorwaarden van het draaien van het softwareproduct, en hoe het eventueel weer opgestart zou kunnen worden. Alle code halen we uit actief versiebeheer en schrijven we weg naar het data-archief.

Alle overige projectbestanden, zoals ontwerpdocumenten, een export van de projectmanagementsoftware, kopieën van alle facturen, offertes en akkoorden en een uitdraai van de urenverantwoording kunnen ook worden toegevoegd aan het data-archief op aanvraag. Het data-archief wordt direct aan de opdrachtgever beschikbaar gesteld. We bewaren zelf het data-archief nog maximaal 3 maanden na sunsetting. Dit is relatief kort i.v.m. de GDPR.

We houden domeinnamen 5 jaar na de sunsetprocedure aan, om domeinnaamkaping te voorkomen. Hierna zeggen we deze domeinnamen op. Indien gewenst kunnen wij een tekst van onze opdrachtgever weergeven op het domein, waarin uitleg wordt gegeven aan eindgebruikers over het staken van het product.

Kosten

We brengen voor de sunsetprocedure losse werkzaamheden in rekening tegen het **hoge uurtarief**. De tijd die de procedure kost kan sterk uiteenlopen, maar er moet rekening mee gehouden worden dat een van onze specialisten hier minstens één of twee dagen mee bezig is.

Op de laatste factuur staan de uren die we hebben besteed aan de sunsetprocedure. Daarnaast rekenen we direct voor de aankomende 5 jaar opslagkosten voor het archief, CDN-kosten voor de informatieve pagina en kosten voor domeinnaamregistratie door.

13.2 Sunsetting bij betalingsverzuim of faillissement

In het geval van betalingsverzuim **45 dagen** na het verlopen van een servicefactuur (75 dagen na facturatie) starten we automatisch met de sunsetprocedure. Voor deze sunsetprocedure wordt ook een factuur nagestuurd.

Speciale situatie bij faillissement

Wanneer een opdrachtgever uitstel van betaling of faillissement heeft aangevraagd maar de nog bestaande omzet van het bedrijf afhankelijk is van het softwareproduct kan een uitzondering gemaakt worden. Maak hier eerst melding van. Dit kan gedaan worden door zowel het (voormalig) bestuur als de curator. We stellen in het geval van uitstel van betaling of een faillissement de sunsetprocedure uit. Wel brengen we alvast de kosten voor het sunsetten in rekening. Als uiteindelijk geen sunsetprocedure uitgevoerd hoeft te worden omdat geen faillissement plaatsvindt of een doorstart wordt gemaakt, crediteren we deze kosten weer. Bij faillissement vragen wij feitelijk dat we gezien worden als zogenaamde *dwangcrediteur*. Indien hier geen gehoor aan wordt gegeven zetten we alsnog de sunsetprocedure in gang.

13.3 Opnieuw starten na sunsetting

Ook als de sunsetprocedure helemaal is uitgevoerd, is het mogelijk om weer opnieuw te starten. Dit proces noemen we ook wel *sunrisen*. Hiervoor moeten we opnieuw applicatie-infrastructuur en ontwikkelomgevingen inrichten. Afhankelijk van de duur tussen het sunsetten en sunrisen kan dit een aanzienlijke hoeveelheid tijd in beslag nemen. Zo is het soms handiger om de applicatie-infrastructuur meteen anders in te richten, om direct een verbeterslag te maken.

We kunnen het opnieuw starten oppakken in de eerste sprint, van een reeks sprints, of als los werk. Er kan vanuit gegaan worden dat één van onze specialisten meerdere dagen bezig is met *sunrisen*.