Wat betekenen de 10 SIG-richtlijnen en wat zijn de voordelen voor jou als opdrachtgever?

Bij OVSoftware staan we garant voor de goede kwaliteit van de software die we opleveren. Dit doen we onder andere door bij de ontwikkeling van software de tien richtlijnen voor codekwaliteit van de Software Improvement Group (SIG) toe te passen die zijn gebaseerd op de ISO25001 standaard. Maar wat betekenen de richtlijnen en welke voordelen ze hebben voor jou als opdrachtgever?

1

Write short units of code

Schrijf kleine functies (15 regels)

Wat betekent het?
Divers onderzoek heeft uitgewezen dat de menselijke hersenen maar een beperkt aantal regels kunnen bevatten. Bij maximaal 15 coderegels per functie, kan een developer deze goed overzien en begrijpen.

Voordeel
Korte functies maken software inzichtelijker waardoor iedereen het snapt en ermee kan werken. Onderhoud en aanpassingen zijn eenvoudig en kunnen door wisselende personen worden gedaan. Op termijn kost onderhoud van kleine functies minder tijd en geld.

Write simple units of code

Schrijf simpele code units

Wat betekent dit?
Elke methode (een aantal regels code dat ervoor zorgt dat een actie wordt uitgevoerd) bevat maximaal 15 regels code. Een andere manier om objectief te bepalen hoe ingewikkeld de code al dan niet is, is door te kijken hoeveel ontwikkelpaden een methode bevat. Het aantal ontwikkelpaden (beslisbomen voor uiteenlopende scenario's) beperken tot vier per methode, draagt ook bij aan de overzichtelijkheid ervan.

2

Voordeel
Een kwalitatief goede codebase betekent dat je gedurende de hele levensloop van de ontwikkelde software profiteert van de voordelen. Minder bugs, minder downtime, snel kunnen doorvoeren en testen van correcties en aanvullingen. En hoe minder tijd developers aan dit soort zaken besteden, hoe minder het de opdrachtgever kost.

3

Write code once

Schrijf code één keer

Wat betekent het?
Deze richtlijn gaat over het eenmalig schrijven van code die je vervolgens op meerdere plaatsen kunt hergebruiken. Het stuk code dat ergens opnieuw gebruikt moet worden, wordt uit een eerder geschreven functie gehaald en in een nieuwe functie gezet. Hierdoor kun je functionaliteit hergebruiken zonder de code te kopiëren.

Voordeel
Tijdens het ontwikkelen zelf kan het iets meer werk zijn, maar de initiële investering verdient zich snel terug omdat developers veel minder tijd kwijt zijn met het zoeken en aanpassen van code.

Keep unit interfaces small

Houd de interfaces van units klein

Wat betekent dit?
Een unit is een functie in de softwarecode die een bepaald aantal regels code uitvoert, bijvoorbeeld het versturen van een e-mail. Een e-mail is echter niet compleet zonder een aantal aanvullende stukjes informatie, zoals de aanhef en een onderwerpregel: dat zijn de parameters van een functie. Deze vijfde richtlijn stelt het aantal parameters per functie op maximaal vier.

4

Voordeel
Deze richtlijn leidt tot het eenvoudiger kunnen begrijpen, (her-)gebruiken en onderhouden van software, met op termijn lagere kosten als gevolg. Ook draagt het bij aan de tevredenheid van developers omdat het prettiger werken is met goed onderhoudbare code.

5

Separate concerns in modules

Scheid concerns binnen modules

Wat betekent dit?
Door er bij het ontwikkelen van softwarecode voor te zorgen dat je binnen een module niet meer dan één concern* hebt, kun je naderhand wijzigingen in de code doorvoeren of nieuwe functionaliteit toevoegen zonder dat je daarbij het risico loopt dat er ergens anders ongewild (en soms ook ongemerkt) dingen fout gaan.

** Een concern van een module bepaalt welke belangen deze module heeft en welke acties deze verzorgt.*

Voordeel
Het achteraf aanpassen van software met sterk verweven concerns is risicovol en kostbaar. Aanpassingen zijn duurder omdat je meer tijd kwijt bent aan het zekerstellen dat deze geen negatief effect hebben op andere delen van de applicatie. Het leggen van een goede basis kost geld, maar is altijd beter en goedkoper dan achteraf dingen aan moeten passen.

Couple architecture components loosely

Verweef softwarecomponenten op een losse manier

Wat betekent dit?
In deze richtlijn gaat het vooral om hoe componenten onderling samenhangen en met elkaar communiceren en dat de lijnen waarlangs de communicatie verloopt niet kriskras alle kanten opgaan. Het primaire uitgangspunt van deze richtlijn is om componenten zoveel mogelijk geïsoleerd te laten functioneren.

6

Voordeel
Code wordt beter gestructureerd en de impact van een aanpassing wordt verkleind omdat je slechts een onderdeel van het systeem wijzigt. Aanpassingen kunnen hierdoor in minder tijd worden gerealiseerd, de foutgevoeligheid gereduceerd en de stabiliteit van de software verbeterd.

7

Keep architecture components balanced

Houd het aantal componenten gelijk

Wat betekent het?
Een code component is een aantal pagina's met code bij elkaar die verschillende dingen doen. Het in balans houden van code componenten doe je door ze onderling niet te veel van elkaar te laten verschillen qua omvang en hen goed te groeperen.

Voordeel
Als je code componenten in balans zijn, kunnen developers vlot traceren waar problemen zitten, preciezer ingrijpen en sneller corrigeren. Het zorgt ervoor dat onderhoud minder tijd en geld kost en dat fouten sneller worden opgepikt waardoor de applicatie zo kort mogelijk uit de lucht is.

Keep your code base small

Houd de code basis klein

Wat betekent het?
Een codebase is een verzameling broncode die wordt gebruikt om een bepaald softwaresysteem, applicatie of softwarecomponent te bouwen. Hoe kleiner de hoeveelheid code, hoe makkelijker het is om iets terug te vinden en om de software te onderhouden.

8

Voordeel
Als een softwareproject klein blijft, gaat de kwaliteit ervan omhoog, voorkom je onnodige vertraging van het project of het falen ervan (= applicatie wordt nooit in productie genomen). Een kleine codebase betekent hoge betrouwbaarheid (door beperken van complexiteit), betere onderhoudbaarheid en snellere resultaten. Kwalitatief goede software betekent een kostenbesparing over de gehele levensduur van een applicatie.

9

Automate tests

Automatiseer testen

Wat betekent dit?
Bij geautomatiseerd testen van de code units kijk je of zij doen wat ze moeten doen. Niet alleen of de werking van de functie goed uitpakt, maar juist naar dingen die fout kunnen gaan. Geautomatiseerde testen worden vaak als regressietest ingezet. Bij elke aanpassing of uitbreiding van de code wordt daarbij gecontroleerd of er niet onbedoeld iets is stukgegaan wat je dan direct kunt corrigeren.

Voordeel
Het automatiseren van testen leidt op termijn altijd tot grotere efficiency, betere onderhoudbaarheid en eenvoudiger aan te passen softwarecode.

Write clean code

Zorg voor goede leesbaarheid van je code

Wat bekent dit?
Deze richtlijn is erop gericht ongebruikte code zoveel mogelijk te verwijderen en telkens als je software bewerkt, tegelijkertijd kleine verbeteringen op andere plekken door te voeren.

10

Voordeel
Leesbare code is makkelijk overdraagbare code. Als opdrachtgever ben je dan niet meer enkel aangewezen op die één of twee specifieke personen die ooit aan de wieg hebben gestaan van een bepaald softwaresysteem. Het is dus vooral een bijdrage aan de continuïteit van processen en daarmee indirect ook aan het voortbestaan van je bedrijf.

OVSoftware