R: Ik ben nu de opname gestart. Allereerst bedankt dat jullie mee willen doen aan deze sessie. De sessie zal bestaan uit twee delen. In het eerste gedeelte zal ik jullie uitleggen wat precies de bedoeling is en wat de taak is die we gaan voltooien, en in het tweede deel gaan we de taak, of te wel een discussiegroep, daadwerkelijk uitvoeren. Dus even kort over het onderzoek: ik doe onderzoek naar software product kwaliteit, en specifiek naar de ISO 25010 standaard. Dit is een standaard die helpt om software product kwaliteit beter te benaderen aan de hand van acht karakteristieken. Die karakteristieken gaan we tijdens deze sessie bespreken. De taak die we zullen voltooien: we gaan voor elk van de acht karakteristieken brainstormen hoe we deze karakteristieken mogelijk concreet kunnen meten. Dus wat voor kwantitatieve waarden kunnen vinden voor de karakteristiek Belangrijk is dat we dus op zoek zijn naar zo objectief mogelijke waarden. Een voorbeeld daarvan is dat bij functional suitability je als metric kan noemen: “in hoeverre vind jij dat een product functional suitable is”, want dat introduceert subjectiviteit. Dus we zijn concreet op zoek naar datapunten die je kan meten om een benadering te vormen van de acht karakteristieken. Ik heb een set aan post-its klaarliggen en een pen. Jullie zullen voornamelijk een discussie voeren. Als onderzoeker mag ik niet leidend zijn en geen hints geven over wat mogelijke metrics kunnen zijn, dus ik wil jullie zometeen vragen dat wanneer jullie een metric hebben gevonden, deze op te schrijven op een blaadje en deze dan onder het rijtje te plakken waar deze bij hoort. Is het duidelijk wat voor taak we gaan doen?
P1, P2, P3: Ja
P1: Korte vraag, mag het in het nederlands?
R: Zeker, het mag zeker in het nederlands. Uiteindelijk heb ik een deel van de sessies in het engels gehouden en een deel in het nederlands, dus zal ik het vertalen naar het engels. Dat is ook een slag die ik kan maken, dus wat voor jullie het fijnst is. Oke, zullen we dan beginnen met het eerste karakteristiek. Ik zal ook de tijd meten, dus wanneer we nog twee minuten hebben voor een bepaald karakteristiek zal ik jullie ook duidelijk maken dat we de discussie een beetje moeten afronden. Het eerste karakteristiek is functional suitability, en functional suitability is “the degree to which a product or system provides functions that meet stated and implied needs”.
P2: En in het nederlands?
R: In het nederlands is het de mate waarin een product of systeem functies biedt die zowel de geschreven als impliciete needs van een gebruiker
P2: Doet het wat het moet doen?
R: Inderdaad. Dus de vraag die erbij hoort: Hoe zou je functional suitability kunnen meten? En als je moeite hebt met dat te bekijken zou je ook kunnen nadenken over wat zou jij kunnen doen om de functional suitability van je product te verhogen: wat voor acties moet je dan ondernemen? Daar kunnen natuurlijk ook metrics uit rollen. Dus dan laat ik nu de discussie aan jullie.
P1: Als het niet doet wat het moet doen dan stuur je een mailtje naar de klantenservice: het doet het niet. Dus die mails die kun je meten.
P3: Dus het aantal klachten bij wijze van spreken.
P2: En aan de andere kant heb je ook van die mensen die niet gaan communiceren, totdat het stuk gaat. Het doet niet helemaal wat je wilt, maar het is oke. Zulke mensen heb je ook. Dus daar vraag ik mij nog een beetje af.
R: Wanneer moet een klacht worden opgenomen is dan de vraag?
P2: Ja, nu heb ik het over mensen die over de drempel heen zijn dat ze contact op nemen, maar je hebt natuurlijk ook mensen die zeg maar wat minder geneigd zijn om al die moeite erin te stoppen, en gewoon accepteren dat het niet voldoet, maar dat het enigszins voldoet.

Maar functional suitability is: je wilt die honderd procent bereiken toch? Je wilt weten wat dat punt is
P3: Ja, of hoe verhoud het klachten zich tegenover het aantal actieve gebruikers
P2: Maar dan moet je dus klachten kunnen meten. Je hebt natuurlijk ook de persoon die een klacht heeft, maar dit niet meldt.
P2: Je hebt dat, maar ook kan ik vinden wat ik zoek. Je kan het meten aan de hand van een enquete. Of dat je de zoekfunctie gaat gebruiken in plaats van gewoon in de app gaat klikken. Met andere woorden: Je kan het niet snel vinden en moet het dan maar in de zoekfunctie gaan doen.
P1: Maar hoe zou je dat dan meten? Zoekgedrag van gebruikers?
P2: Zoekgedrag van gebruikers in de vorm van of die überhaupt die functie gebruikt, zoektermen op zoekfunctie. Die zijn te meten, en vervolgens als die daarna eruit komt, zoekt die naar iets wat een bekende knop is, of iets wat niet een bekende knop is. Zoekt die naar iets wat een bekende knop is binnen de voorpagina
P1: Maar dat zou klikgedrag zijn toch?
P2: Ja, klikgedrag binnen het zoeken. Zoeken naar het bekende, zoeken naar het onbekende
R: Hele goede punten die we nog aansnijden. Ook belangrijk bij het maken van deze metric is: ten opzichte waarvan meet ik? Bij het aantal klachten komt er misschien 1 numerieke waarde uit. Meet je dit aan de hand van een baseline. Of is dit puur een metric die je kan meten aan de hand van een trend. Je meet de metric nu voor het product, en een maand later, en je kijkt of het minder is geworden.
P2: Je bedoelt A/B testen
R: Bijvoorbeeld ja. Wanneer hebben deze metrics betekenis, want puur het meten van een numerieke waarde zegt nog niks
P1: De trend denk ik
R: Het kan natuurlijk per metric verschillen
P1: Voornamelijk die klachten bedoel ik dan
R: Er kunnen een aantal metrics zijn die aan de hand van een trend betekenis krijgen. Er kunnen metrics zijn die aan de hand van een industrie standaard betekenis krijgen. We kunnen metrics krijgen die aan de hand van organisatie standaarden betekenis krijgen. Dus ik denk dat het belangrijk is bij het maken van de metrics dat we specificeren hoe we het relateren aan iets. Wat is de baseline voor het meten?
P2: De baseline kan ook zijn, dat je uberhaupt meet. De baseline is nul en dat is vanaf het moment dat de applicatie geïmplementeerd is. Dus de versie die nu staat is nu. Na de release.
P1: Na oplevering software toch?
P2: Dan heb je die metric inderdaad, en de baseline is dan nul, maar wat is honderd? En dat weet je dan niet. Want honderd kan alle kanten op gaan, honderd is indefnitie in dit geval
P3: Honderd is in dit geval het aantal gebruikers dat je hebt. Dus je cap is het aantal potentiele klagers mogelijk
P2: Mocht er een licentie op deze applicatie zetten, heb je je y max dan
R: Dus zouden jullie dan zeggen dat het aantal klachten gemeten moet worden aan de hand van het aantal gebruikers, of zijn dit losse metrics?
P1: Ik denk dat het aantal klachten pas wat zegt als je weet welke portie het is. Kijk als je 1 miljard gebruikers hebt en je hebt tien klachten, dan is het niet zo erg. En als je 20 gebruikers hebt en je hebt tien klachten is het in een keer heel anders
P2: Dus daar moet je een KPI op gaan zetten

P1: Het aantal actieve gebruikers versus klachten. En speelt beschikbaarheid hier nog een rol in? Een soort uptime, of zien we dat als een klacht?
P2: Ja die kun je afvangen in je klacht, maar het is een metric dus ik zal het er zeker bij zetten
P1: Dat moet je ook meten, want als het daar aan ligt is het functioneel ook niet echt
P2: En ook daar heb je je nul en je honderd en een threshold. Waar bepaal je dat het niet functional is? Is dat wanneer die 80% uptime heeft of is dat wanneer die 90% uptime heeft
P1: Nee precies, daar moet je iets over zeggen natuurlijk
P2: Daar kan je dan weer zon gestandaardiseerde maatstaf tegenaan zetten toch. Hoelang wil je doen over elk onderdeel?
R: We doen tien minuten over elk onderdeel en ik houd het bij. Dus wanneer we nog twee minuten hebben zal ik het zeggen
P1: Volgens mij zijn we er daarmee wel toch
P3: Er schiet mij niks meer te binnen
R: Wanneer niemand meer iets te binnen schiet kunnen we door gaan. Het is ook mogelijk om post its van de een naar de andere te wijzigen. We weten nu natuurlijk nog niet precies wat er allemaal gaat komen. Ik heb ze kort genoemd maar jullie weten nog niet wat ze betekenen. Dus het kan ook zo zijn dat we er een bedenken en dat we later erachter komen dat die toch beter bij een andere past. Het is altijd nog mogelijk om te schuiven. We hebben nu nog twee minuten.
P2: Wat ik mij bedenk is zeg maar, je hebt marktonderzoek, van dit is hoe je software eruit moet zien en wat een gebruiker vraagt en bijvoorbeeld een soort cookbook of blueprint gemaakt van hoe de applicatie eruit moet zien. En dan heb je de applicatie zelf. En het kan best wel zijn dat die cookbook niet helemaal tegenover die applicatie komt te staan. In andere woorden dat de vertaalslag van de marktvraag naar de applicatie een beetje scheef kan staan. Als in 80% klopt wel, maar 20% niet. Daar konden ze niet tegenaan bokken of technisch niet implementeerbaar wat daar de vraag was. Lijkt mij wel interessant om daar een soort maatstaf tegenaan te zetten. Je hebt feitelijke waarnemingen van de marktvraag in een soort cookbook gezet, en je daarna nog een check doet en een meetinstrument creëert vanuit je blueprint of cookbook van zitten alle functionaliteiten erin die vanuit de markt worden gevraagd
P1: Hoe zou je dat dan omschrijven? Marktanalyse versus
P2: Soort van blueprint versus actual build
P1: Ah ja zo, echt blueprint van je software
P2: Ja, want vaak heb je nog wel dat je eerste iteratie niet je laatste iteratie is. Maar je wilt wel zo snel mogelijk in productie want dat brengt waarde op. Dat is nog wel een goeie. Je baseline is je blueprint en je meetinstrument haal je gewoon letterlijk uit de tekst. Je maakt een soort csv.
P3: Hoeveel features zitten erin die we van te voren hadden gedefinieerd
P2: Ja een soort checklist. Want als je checklist volledig ingevuld is dan is het honderd, en als het niet volledig ingevuld is. En het zijn feitelijke observaties toch. Dus dan is het minder objectief.
P3: Ja goeie
R: Ik ben het er mee eens. Laten we deze opschrijven als closing remark voor functional suitability en daarna doorgaan naar performance efficiency, en dan lopen we precies goed. Andersom volgensmij, de plakrand zit hier. Geen idee. Goed, daarom doen we het ook zo. Ik heb ook een vorige sessie gehad waarin ik ze allemaal opplakte tot de post its niet goed genoeg plakten en ik continu aan het bukken was. Dus nu doen we het even zo.

Performance efficiency is de “performance relative to the amount of resources used under stated conditions”.
P2: Nog een keer?
R: Performance efficiency is de “performance relative to the amount of resources used under stated conditions”.
P2: En nu in het nederlands?
R: Dus hoe goed doet jouw product het gegeven de resources die die gebruikt. Resources zijn hier niet menselijke resources. Dus bijvoorbeeld de hardware omgeving waarin het runt
P2: Dit is echt alleen maar logging
R: Ja?
P2: In mijn optiek wel, misschien zijn er meer dingen. Hebben jullie een idee?
P3: Nog een keer lezen
P2: Hoe goed doet de applicatie het met de middelen die je hebt
R: Precies ja
P3: Aan wat relateer je het dan
P1: Je gaat dan meteen aan queries denken, niet?
P3: Ja, hij duurt twintig minuten, maar misschien is dat wel heel snel
R: Exact ja, exact, en dat is juist het interessante en complexe vraagstuk
P1: Het heeft dus ook te maken met wat die moet doen toch. Stel je zit aan queries te denken is een select sterretje natuurlijk honderd keer zo snel dan wanneer je een gekke 600 joins achter elkaar moet doen.
P3: Misschien moet je het relateren aan een soortgelijke bestaande ofzo iets. Bijvoorbeeld met de verhuizing van Pentaho naar SQL. Dan weet je ongeveer hoe snel die het ongeveer kan doen
R: Wat natuurlijk ook een mogelijkheid is is dat je zegt: er is niet 1 baseline die waardevol om te gebruiken is. Hij is puur waardevol als je hem over tijd meet. Dus dezelfde query nu, en dezelfde query in een volgende iteratie van een product. Dat kan natuurlijk ook een antwoord zijn. Als je zegt, er is geen waardevolle baseline is het beter om te zeggen, er is geen baseline, hij is beter om als trend te meten in plaats van een fictieve baseline erbij te bedenken om een baseline te hebben
P3: Je hebt pas een baseline als je hem een keer gemaakt hebt
P2: Ik denk dat je deze ook wel mooi kan meten door deze te categoriseren als known knowns, known unknowns
R: Ja die vier
P2: Ik voel wel echt die vibe van, je bouwt nu een query en weet niet dat die sneller kan, totdat je weet dat die sneller kan. En daar zit natuurlijk een verschil in van wat je efficiency kan meten. Dus dat heb ik in mijn hoofd zitten. Je weet het pas wanneer je de optimalisatie hebt gevonden. Maar dat is vanuit een software idee met querys toch. Een software implementatie die gaat over syntax, syntax efficiency is hier een voorbeeld in.
P1: Ik zit meteen te denken aan games, FPS enzo snap je. Dat soort dingen. Dat kan je ook meten. Frames per seconde.
P3: Je hebt dan industriestandaarden. Moderne spellen hebben zoveel internetverbinding, zoveel frames per seconde
P1: Ja precies, maar dat is dan ook aan de achterkant. Elk stapje dat je zet is ook verwerkt. Maar dat hangt ook af van de resources die erachter hangen. Dus niet alleen de internetverbinding.
P3: Nee oke. Ik zit even na te denken.
P2: Logging is ook een ding. Performance is heel belangrijk toch

P1: Ik vergelijk het met alle situaties.
P2: Je zit nu ook te denken of je de query anders had moeten schrijven
P3: Nee dat is het eerste wat mij te binnen schiet.
P1: En dan voornamelijk is het bij refactoren toch
R: Maar oud versus nieuw, wat meet je dan in de oude en nieuwe situatie
P1: Doorlooptijd, dus tijd hoelang een proces duurt
P3: En wat normaal is, dat is deels ook ervaring toch?
P2: Hoe meet je je performance?
P2: Ik heb het ook ergens opgeschreven volgensmij. Functie versus doorlooptijd. Een query moet wat doen maar je weet dat een select ster niet tien minuten moet duren. Maar je weet ook dat zes joins niet een uur duren. Die had ik ook in mijn hoofd, de geschatte performance tijd versus de reële tijd, dat verschil is een ding
R: En hoe zou je die schatting maken?
P2: Ja dat is subjectief. Je doet het op basis van je eigen ervaring. Daar zit dan subjectiviteit in. Helaas is het niet anders. Je kan wel je subjectiviteit versterken met objectieve informatie. Dus bij wijze van spreken dat je het een beetje inkadert. Deze query moet zolang duren want we hebben zoveel joins en per join duurt het zolang. Dan maak je er een rekensommetje van. Maar dat is een schets
R: Ook nog steeds een antwoord natuurlijk, Een antwoord kan natuurlijk ook zijn, om dit te meten moet je subjectiviteit introduceren.
P3: Misschien weer query gerelateerd, maar je kan de query op verschillende platformen uitvoeren dan weet je beter wat performance wijs beter presteert. Dus dat is weer de doorlooptijd in principe
P1: Maar misschien ook memory use
P2: Je denkt nu vanaf een query standpunt maar het is gewoon syntax. Elke software heeft syntax en elk software is slim of niet snel in elkaar gezet waardoor het wel of niet snel gaat. Uiteindelijk, weet je wel.
R: We hebben nog twee minuten voor deze
P2: Dan hebben we nog, mocht het over resources zijn die we loggen tegenover de software die we erop hebben staan, meer hardware gericht nu. Hardware versus software heb je natuurlijk gewoon je ingeschatte relatie tegenover de reële resource caps die die heeft. Dat die gewoon aangeeft van je zit nu bijna op je max
R: Max als in welke resource?
P2: Stel je hebt een 32 gigabyte ram cpu en dat je gewoon had ingeschat van ik had 25 gigabyte nodig om überhaupt te runnen, maar hij capt hem er overheen. Ik zit op 31 ofzo. Dan weet je dat je eigenlijk al ver boven je performance capacity zit.
P1: Hoe zou je die omschrijven in twee woorden?
P2: Ja in een zin. Je hardware caps die je hebt neergezet tegenover de reële
P1: Gaps of caps?
P2: Caps
R: En dan hardware caps is nog vrij breed. Welke categorieën zou je daarbinnen hebben?
P2: RAM capacity
P1: RAM en CPU, storage
P2: Storage capacity ook ja, daar zit je ook een cap op. Je hebt niet 200 gigabyte nodig maar 50 en dan blijkt dat je 60 nodig hebt want je hebt iets niet meegerekend. Dan kan het zijn dat je misschien een inefficiente slag hebt gemaakt ofzo. Dus storage capacity, ram, wat heb je nog meer. Request response time

P1: Ik had het al opgeschreven ja, als je de reactietijd aan het loggen bent verwacht je ook binnen vijf seconden wat toch
P2: Ja, daar heb je wel een standaard maat voor nodig. Je kan vanaf subjectiviteit objectiviteit creëren door een alpha erop te gooien. Dus met andere woorden je hebt ongeveer 150 man nodig die vindt of laat zien van ik vind dit te langzaam of te snel
P1: Hoe noem je het?
P2: Een alpha toets
P1: Alpha toets gebruikers, x aantal gebruikers
P2: Of performance, en dan met name temporal performance denk ik dan. Of zeg ik iets heel wijs nu
R: Oke, laat dit de closing remark zijn voor performance effiency en doorgaan naar usability, akkoord?
P1: Zeker, graag
R: Usability is de “degree to which a product of system can be used by specified users to achieve goals with effectiveness, efficiency and satisfaction”. En dan komt nu waarschijnlijk de vraag: nederlands? Nederlands is dan in hoeverre jouw gebruikers je product kunnen gebruiken met effectiveness, efficiency en tevredenheid
P2: Dit gaat wel heel erg gelinkt met de eerste
R: Ja zeker. Het is inderdaad goed om op te merken als je noticed dat bepaalde misschien enigszins overlap hebben of hand in hand gaan, dat is ook een goed ding om op te merken
P1: Ik denk bijna dat je heel veel notes die liggen bij functional suitability ook kunt leggen bij usability. Want je zou ook weer die klachten dingen kunnen meten.
P3: Succesvolle handelingen dan
P1: Hoe meet je dat dan?
P3: Logging toch?
R: Het verschil tussen deze twee is nog. Deze gaat om doet het wat het moet doen? En deze gaat nog een stukje verder: doet dit het voor alle gebruikers? Zijn er sommige gebruikers die een bepaald ding niet kunnen doen? En kunnen ze het efficient doen en met tevredenheid, dus het gaat nog verder dan. Doet het product wat het moet doen? Dus zijn alle features er. Is het bruikbaar?
P1: Heeft dit ook te maken met rollen en rechten?
R: In de zin van?
P1: Nou bepaalde gebruikers kunnen het een wel en het andere niet. Dan kan je ook je rollen en rechten security hebben bedacht, en dat functioneel testen
R: Ik denk dat die eerder gaat over een andere, security, die we nog gaan bespreken. Wat bijvoorbeeld hier een voorbeeld van kan zijn is
P3: Misschien een hele stomme maar bijvoorbeeld verleende licenses
R: Maar ook bijvoorbeeld, werkt het product ook voor elke leeftijdscategorie. Het heeft een bepaalde feature die
P2: Maar dat is usability toch?
R: Ja, we hebben het nu ook over usability toch
P2: Oh top, haha
R: Maar je product werkt bijvoorbeeld niet voor een kind van acht want die heeft niet de cognitieve capaciteiten om te werken met het product. Dat wordt dus bedoeld met specified users
P2: De beschrijving die erbij staat vind ik niet helemaal bij usability passen
R: Oke, omdat?

P2: Het is een remark omdat usability ook gaat over accessibility, het gaat best wel in hand. En het staat er niet bij dus dat is helemaal top. En dat gaat meer over moet ik een leesbril op om de applicatie te kunnen lezen
P1: Is dat zo?
P3: Misschien spreiding, demographie enzo? Dat er van alle leeftijdscategorieen evenveel gebruikers zijn, is het heel accessible toch.
R: Gegeven een target audience toch? Oke, interessant
P3: Stel je target iedereen en je bereikt maar een klein deel
P2: Je kan natuurlijk ook met een bepaalde toets meten als iemand de applicatie voor zijn hand krijgt en je geeft hem een taak, dat iedereen ongeveer dezelfde tijd erover doet om die taak te voltooien. En dan heb je dus een demographie die daarbij hoort. Dan kun je redelijk zien of iemand van 40 of 50 die digitaal niet zo geletterd is als een 18 jarige, of die door dezelfde snelheid door de applicatie komt. Of dat er een bepaalde benchmark is. Dat je kan zeggen van: weet je, je doet er twee minuten erover en die vijf. Maar als het tien is moeten we daar wat aan doen
R: Hoe zou je dat dan als concrete metric meten? Je hebt het nu over de delta tussen een slome gebruiker en een snelle gebruiker. Welke concrete numerieke waarde zou hieruit moeten rollen en hoe zou je die kunnen meten?
P2: Ik denk dat het je ideale gebruiker is. Een jonge gebruiker, want die is ongelooflijk geletterd. Die neem je als een soort baseline en daar gebruik je dan tegenover een delta van een oudere, of in ieder geval minder geletterde persoon. En literacy kan je dan ook meten van te voren dat je die demographische kenmerken hebt, een digital literacy vragenlijstje. En op basis daarvan kan je zeggen van. Iemand die een tien digitaal geletterd is zit hier zo ergens, en iemand die een vijf digitaal geletterd is zit daar zo misschien. En op basis van die twee knowns kun je zeggen van. Er moet een soort ratio zijn
P3: Je kan een soort voorspelling doen waar je in moeet zitten. En als je er teveel van afwijkt
P1: Dan zit je weer op klikgedrag toch?
P2: Taak voltooid is dan een ding.
P3: Hoe weet je dat dan?
P2: Wanneer de taak voltooid is. Bijvoorbeeld ik zeg tegen jou schrijf een email naar die in die. Je moet bijvoorbeeld in gmail klikken en dan vervolgens allemaal dingen doen om dat mailtje te schrijven. En de mailtekst moet je bij wijze van spreken gewoon knippen en plakken, gewoon om tijd te besparen. Misschien wil je wel een hallo mailtje sturen. Een bedachte taak compleet uitgevoerd. En die taken kunnen dan gebaseerd zijn op de features die in een applicatie zitten. Bijvoorbeeld, schrijf een query, klaar doe hem. Krijg hem dan terug. Taak voltooid. Bouw een nieuwe table, misschien ga je dan teveel in de syntax zitten. Maar ja SQL is ook software. Taken definiëren en die dan voltooien. Ten opzichte van de demografische kenmerken of iedereen er op dezelfde manier doorheen komt. Er kan natuurlijk een situatie zijn waarin iedereen er heel langzaam doorheen gaat omdat je app niet werkt.
R: Wordt dat dan niet afgevangen door de metrics die aan performance effiency hangen? Die zouden moeten zeggen, je app werkt traag
P2: Nee dat kan eraan liggen dat niemand je app begrijpt
P3: Een soort front end performance. Je kan een super goede backend hebben maar je weet niet waar je heen moet
P2: Stel je moet een API request doen en je weet de helft van de headers niet eens. Iemand heeft je uitgelegd hoe het werkt. Dan zit je soms er wel twee dagen aan te werken om een curl request te doen.

R: Dus jij zou zeggen dat naast het meten van de delta tussen een snelle en langzame gebruiker het ook zinvol is om iets te zeggen over de average gebruiker. Is de average gebruiker traag ten opzichte van wat jij ervan verwacht
P2: Goeie, dan moet je wel weten wat er traag aan is. Wanneer weet je dat iemand er niet snel doorheen gaat. Je zou bijna kunnen zeggen, degene die de applicatie heeft gebouwd moet die taak voltooien. De manier waarop hij er doorheen klikt is bijna hoe je wilt dat de gebruiker er doorheen klikt
R: Twee minuten nog voor deze
P2: Dat is je honderd, en je wilt iedereen naar de honderd krijgen
P1: Ik denk mooie punten, denk je ook niet?
P2: Zal ik hem even overnemen?
P1: Waarom?
P2: Of zit je er lekker in? Chill
R: Oke, voor usability, dit zijn alle punten, doorgaan naar de volgende?
P1, P2, P3: Yes
R: Oke, dan nemen we reliability, dat wordt vanuit de standaard, het zijn allemaal definities uit de standaard overigens ik heb ze niet zelf bedacht, de “degree to which a product, system or component performs specified functions under specified conditions, for a specified period of time”. Wat het dus eigenlijk betekent is dat wanneer je het systeem een bepaalde load geeft, dus een bepaalde condition voor een bepaalde tijd. Ik hoeverre voltooid die dan nogsteeds de functies die die zou moeten voltooien
P3: Een soort stress test eigenlijk
R: Ja
P1: Heeft het dan ook te maken met schaalbaarheid?
R: Ja, zeker
P3: Aan de interpreteerder, hoe je hem invult
R: Ja, dat is dus ook waar we in deze studie naar kijken. Dit zijn allemaal definities die vanuit de standaard worden gebruikt. Je zei net al, ik vind deze niet helemaal duidelijk. Dat is ook een interessante onderzoeksuitkomst, dat de standaard in bepaalde opzicht niet zo bruikbaar is dat als we hopen dat die is. Er zijn dus geen goede of foute antwoorden, als jij denkt dat het erbij hoort dan hoort het erbij
P1: Is mijn responstijd even groot als ik duizend gebruikers heb of tien miljoen? Daarmee meet je schaalbaarheid
P3: Reliability. Dan denk ik ook wel aan uptime
P2: Ik ook
P1: Dan kunnen we hem toch gewoon opschrijven
P3: Of response time van tickets denk ik
R: Aan de hand waarvan zou je je uptime dan benaderen? Zou je zeggen honderd procent is desired. Want in de praktijk is honderd procent niet mogelijk. Of is dat aan de hand van een SLA?
P1: Er zijn ook SLA’s waarin staat 99 komma 999999 procent. Dan dekt elke organisatie zich aan dat als het een keer niet werkt. Dit is dus die ene keer dat het niet werkt.
P2: We hebben het ooit gedaan, nee dat kan ik niet zeggen
P3: Dat hebben we ooit gemeten, bij bedrijf X
P1: Intern bij X. Dan heb je natuurlijk nog servers die altijd up zijn. En dan kan je gewoon uitrekenen de tijd en dan op een bepaalde manier waar ze een keer down zijn
P2: En wat was de measure? Wat was de uitkomst?
P1: Nou we hadden in de SLA staan dat het 99 komma 98 procent moest zijn

P2: En bij jullie was dat?
P1: Ja
P3: Wat een ster he. Wat is er verder nog?
P2: Reliability
P1: Ja P2, brand los
P2: Reliability, dat mocht de syntax dynamisch zijn, dat je alle cases ook afvangt. Maar dan weet je nog niet wat alle cases zijn.
P3: Responstijd van tickets, of zoiets. Dat als het niet goed ging
P1: Je zit in de ticket flow he?
P3: Ik ben een soort getraumatiseerd. Als ik het t woord hoor word ik gek
P1: De responstijd op incidenten dan?
P3: Ja, dat is wel een goede denk ik. Alleen wat is normaal dat is dan de volgende vraag
P1: Ja maar dat spreek je af
R: Of wederom het kan zijn dat er niet perse een standaard voor normaal moet zijn en dat dit een waarde is die zo laag mogelijk moet zijn. In je volgende iteratie wil je dat die lager is dan in de volgende
P2: Reliability haal je ook uit version control
P3: Een soort check of je het hebt?
P2: Version control implemented, yes or no?
P1: Ik zit ook meteen te denken aan of er zeg maar een disaster is, een soort recovery time. Toch? Hoe snel ben je weer up to speed, of komt er nog een andere categorie die dit bedacht heeft?
R: Nee
P1: Maar wat zei je eerst?
P2: Nee, je spreekt uit ervaring, toch?
P1: Onder andere ja. Wij hebben SLA's gehad vroeger
P3: De cloud is daar heel groot op dat ze dat hebben
P2: Back-up recovery time bedoel je dan
P3: Ik ben de term even vergeten. Availability zones
P1: Of regions
P2: En ik heb zoiets van, de ratio tussen de dynamische software en de controleerbaarheid. Je kan inderdaad zeggen dat syntax dynamisch wordt. Maar vang je ook alle routes af? Bijvoorbeeld je hebt je availability zones. Maar als je bijvoorbeeld spot zones gebruikt weet je niet waar het vandaan komt. Dus hoe reliable is dat? Transparantie tegenover dynamiek. Het is heel moeilijk te meten. Wanneer is iets transparant?
P3: Dat is dan misschien huiswerk voor R
P2: Hoe de ratio van black box, maar ja hoe schrijf je dat op. Hoe maak je dat meetbaar. Je weet gewoon bepaalde dingen op een gegeven moment niet. En dat moet je gewoon meten. Maar hoe meet je iets wat je niet weet. Door te meten wat je wel weet. Oke, laat maar. Laten we naar de volgende gaan toch?
P1, P3: Ja
R: Alright, dan gaan we naar maintainability, en maintainability is de "degree of effectiveness and efficiency to which a product or system can be modified by the intended maintainers". Dus in hoeverre kan je aanpassingen brengen aan een bestaand product.
P2: Oke, chill
P1: Belangrijk aan deze definitie is wel dat het gaat om die intended maintainers, dus niet in hoeverre kan Pietje jouw codebase onderhouden, als Pietje niet binnen jouw organisatie is

P2: Gebruik maken van programming paradigms toch? Dus niet alles in functional programming zetten, maar alles in class based programming zetten? Hoeveel functions heb je tegenover hoeveel classes je hebt?
P1: Hoe schrijf je dat op? Hoe schrijf je dat woord wat je zegt?
P3: P - a - r - a - d - i - g - m -s. Paradigms. Dat mag R uitzoeken hoe je dat schrijft.
P1: Maar heeft dit ook niet deels te maken met hoe je het afdicht aan de achterkant? Ja, maar dit is voor de intended, dus ja
P3: Waar ik in zit is lage maintainability, maar waar licht dat dan aan?
R: Ja, waar licht dat dan aan?
P3: Ja, documentatie, domeinkennis. Maar hoe meet je dat, dat is dan ook de vraag
P2: Stel je zou een vraag stellen, dat had je al door denk ik?
P1: Stel, toen dacht ik al: vraag
P2: Maar ja, stel je hebt een ideale wereld voor je en je komt op project X, en je moet maintainen wat er staat. En ze vragen of je de code wilt optimaliseren, en een versie wilt updaten. Je komt er binnen en er is geen documentatie
P3: Hoofdpijn
P2: Maar stel, er is een ideale wereld waarin alles helemaal compleet en perfect voor je is, wat zou dat dan zijn? Waardoor jij meteen aan de slag kan gaan zonder dat er iets gefixed moet worden?
P3: Dat zijn de drie dingen die niet goed meetbaar zijn. Is er nog iemand aanwezig die betrokken is geweest bij het bouwen?
P1: Aantal kennishouders
P2: Ja dat is een vet meetbare
R: En we hoeven natuurlijk ook niet, bij sommige van deze abstracte concepten is het onmogelijk om binnen de tijd die we hebben een objectieve benadering te hebben. We proberen onze benadering zo objectief mogelijk maken. Je kan bijvoorbeeld zeggen, domein kennis kan je meten aan de hand van A, B en C. Dat is by far niet compleet om het totale plaatje te capturen maar dan hebben we in ieder geval een paar pijlers om dit te kunnen doen. Dat is ook een antwoord.
P3: Ik denk een actieve community achter de techniek
P2: Ik denk dat je het aantal kennishouders prima kunt meten door een ticket systeem te meten, en dan hoevaak staat zijn naam bij tickets?
R: Ja bijvoorbeeld
P3: En de derde is documentatie. Je gaat niet het aantal pagina's tellen
P2: Nee, maar wat je wel kan doen is per class, per software, per functionaliteit, dit is de verwachte input, dit is de verwachte output, dit zijn de processen die erin zitten. Dus per functie documentatie. Per software functie een documentatie
P3: Daar heb je verschillende dingen voor? PEP8 is dat dan ook? Die leveren vast wel standaarden aan waar het minimaal aan moet voldoen. Die kan je stelen. Maar dat is dan wel voor Python
P2: Ja want, we hebben die drie dingen nu. Zijn er nog meer dingen dat je kan denken van: dat sowieso
P1: Ik zit te denken, in principe zou je geen domein kennishouder moeten hebben als je een hele simpele handleiding hebt. Waar je gewoon stap voor stap, heel dummy jip en janneke uit legt wat je moet doen
P3: Maar ik denk dat documentatie ook uitleg vereist
P1: Ja, dat is ook zeker waar. Maar als je een jip en janneke handleiding hebt, als die goed is hoef je niet heel veel vragen te stellen

P2: Maar in casu Github, je gaat naar Github, staat een readme bij. Die lees je door, en vervolgens implementeer je de code zonder dat je ooit die gast hebt gesproken. Dus kennelijk zijn er voorbeelden waarbij je dus wel
P3: Maar dat gaat over gebruiken, niet over onderhoud
P2: Ja goede ja
P3: Maar zo'n readme is niet voldoende om het hele proces op te breken
P2: Ja, want je gaat niet erin zitten, goeie. Oke, onderhoud. Ja, het gaat voornamelijk om de snelheid waarin je de applicatie leert te begrijpen zonder dat je er kennis van hebt. Dat is de vibe waar wij nu in zitten. Stel je voor je bent echt vanaf het begin af aan betrokken bij het proces en je weet alles over die applicatie, hoe meet je dan dat die gast het onderhoudbaar vindt?
P3: Ja, maar stel dat hij dan de benchmark stelt. Dat is een utopia
P2: De manier hoe ik het vaak met *naam persoon* oefende of onze applicatie bij project X te onderhouden was om de readme of de confluence pagina aan iemand mee te geven, en dan snap je dit. Op het moment dat die vragen ging stellen was het moment dat we de readme moesten aanpassen.
P1: Zou het aantal vragen dat je terugkrijgt van een collega die het overdraagt een meetbaar iets kunnen zijn?
P2: Ja natuurlijk, en dan de dimensie van de vragen. Waar gaat het over? Wanneer doen we koffie, dat is geen
P3: Het is ook heel erg afhankelijk van degene die de dingen ontvangt. Ik ben geneigd weinig vragen te stellen, maar ik ken er genoeg die geneigd zijn een heleboel vragen te stellen. Je moet een soort baseline te hebben
P2: Ervaringsniveau van degene die het moet maintainen, die staat er natuurlijk niet bij
P3: Dat is uit te drukken in aantal jaren in principe
P2: Dat doen we bij onze club wel, maar ergens anders hebben ze weer een andere maatstaf daarvoor. Een IQ toets. Ik vind het mooi. Compatibility?
R: Yes, we lopen nog binnen de tijd. Alright, compatibility is de "degree to which a product or system can exchange information with other products or systems". Dus in hoeverre kan jouw product informatie uitwisselen met andere systemen? Wanneer kan jouw product goed informatie uitwisselen met een ander systeem?
P3: Dus het is de bedoeling dat die het moet kunnen
R: Exact ja. Het is inderdaad bij al deze dingen de aanname dat die dit bepaalde ding moet kunnen. Het kan natuurlijk ook zijn dat je zegt voor bepaalde producten: dit is een interne applicatie die in no way informatie uit dient te wisselen, dus boeien. Dus dit is altijd onder de aanname dat het product het wel moet kunnen. Want als een product dit niet moet kunnen ga je deze simpelweg niet meten. Maar we moeten natuurlijk nog steeds meetbare punten hebben voor producten die dit wel moeten kunnen
P1: Het aantal native ingebouwde in de software zelf
P3: Ja
P1: Koppelingen met systemen
P3: Ja, zoiets als snowflake heeft dat
P1: Ja, maar ook zoiets als Pentaho, daar zitten heel veel integraties in. Weet ik veel, SAP, SHP, weet je misschien wat. Al die systemen zitten daar al in.
P2: Ja, ik vind deze best wel goed gedefinieerd in de vorm dat je eigenlijk door te zeggen van, de mate waarop die met elk mogelijk systeem voor de use case kan werken zonder enige issues is, dat is het enige wat ik in mijn hoofd heb zitten. En dat bij elke versie verandering dat er nog steeds is. Met andere woorden: ik heb software dit en de use case is

dat er voor die use case ook andere software worden gebruikt om met die use case te kunnen werken, dus je wilt eigenlijk dat als de gebruiker vraagt om integratie tussen die software en deze sofware, dat die mogelijkheid er is
P3: De industrie bepaalt de standaard
P2: Gewoon de amount of connectivity clicks die die heeft.
R: En een connectivity click is dan?
P2: Nouja, de ratio tussen de packages of de wat dan ook dat je mogelijk zou kunnen hebben binnen in die wereld, tegenover de hoeveelheid connecties die die kan maken in realiteit, zonder dat er errors ontstaan
P1: Je hebt ook het gebruik van packages
P2: Ja, dat is ook software
P1: Package gebruikers, ofzo
P2: Met voorbeeld, je hebt PHP en je hebt Python, en ze kunnen allebei een website laten zien, maar je wilt niet dat, het is een beetje gek als PHP en Python met elkaar kunnen communiceren. Dus dat is wel een beetje een use case verhaal waarbij twee dingen niet met elkaar willen communiceren. Maar misschien wil je het wel, weet je wel. Noem maar een voorbeeld waarin wel. Stel je hebt een website met een machine learning element, misschien wil je dan wel python gebruiken. Is er dan compatibility mogelijk, dat soort dingen. Of bijvoorbeeld een andere taal waarin men packages kan delen. Je hebt bijvoorbeeld R en Python en die kunnen wel met elkaar werken. Dat soort dingen. Python kan SQL gebruiken om queries te doen naar Athena. Dat is compatibility in mijn zien. En als ik dan de Python versie ga veranderen wil ik wel nog steeds dat het werkt
P1: Dus versie van de software versus de hoeveelheid integraties die mogelijk is per use case
P3: Dus in de context
P2: In de context van je use case ja. Of bedoel je iets anders?
P3: Nee, zeker weten. Dat is eigenlijk goed te omvatten
P2: Ja, ik vind van wel
P1: Dan is het toch perfect
P3: Gewoon in eentje hebben we het
R: Alright, dan zijn jullie ready om door te gaan naar de volgende?
P3: Ja
P1: Zeker
P2: Ja
R: Oke, top. Dan hebben we er nog twee. De een na laatste is portability, dat is de “degree of effectiveness or efficiency in which a system or product can be transferred from one usage environment to another”. Dus in hoeverre kan je jouw product bewegen van de ene omgeving naar de andere. Hoe effectief is dat, en hoe efficiënt gaat dat proces
P1: Wat is dan een omgeving?
R: Bijvoorbeeld, een voorbeeld zou bijvoorbeeld kunnen zijn
P1: Cloud migratie
R: Exact ja, van een cloud omgeving naar een andere cloud omgeving. Dat kan een migratie zijn. De omgeving waarin je product is gedeployed en dat dan naar een andere.
P1: Wij komen dat in die zin veel tegen, maar wat wij bouwen is eigenlijk altijd portable
P3: De cloud definiteert dat eigenlijk voor je, toch?
P1: Bij het deployen van software snap je. Omgeving onafhankelijk

P2: Ja, als het allemaal automatisch gaat heb je dus een tien, en mocht het niet allemaal automatisch gaan moet je alle taken die je gaat doen noteren in een ticket systeem, en daarin beschrijf je alle handelingen die je moet doen.
R: Exact ja, de definitie gaat niet specifiek alleen over effectiveness. Jullie zeggen dus, het kan bij ons, maar het gaat ook over de efficiency. Hoe snel kan je dat doen? Of wat jouw definitie van efficiency is
P3: Ja, en hoe meet je dat
P2: Ik heb dan gewoon een ticket systeem in mijn hoofd. Je hebt taken en handelingen. Taken is je ticket, en handelingen zijn de hoeveelheid dingen die je daadwerkelijk moet doen. Die kun je best wel subjectief definiëren, maar objectief zijn ze ook al interessant. De ene handeling duurt langer dan de andere. Maar het gaat om de amount of
P3: Misschien een heel concreet voorbeeld, van AWS naar Azure heeft vijftien handelingen. Van AWS naar GCP heeft er maar drie nodig
P2: En mochten die vijftien allemaal kleine handelingen zijn. Prima, maar het zijn nog steeds vijftien handelingen die je cognitief moet verwerken om te doen. Er zit een bepaalde soort van zwaarte in
P1: Dat vind ik heel mooi
P2: Beetje hetzelfde idee als je bijvoorbeeld, ik werk nu heel goed met Docker en het idee daarvan is dat je al je software zet in een operating system in een virtual machine, zet je overal neer en het connect gewoon
P1: Ja dat zeggen we net, het maakt niet uit welke cloud je gebruikt
P2: Maar ga je lokaal builden en deploy je hem dan pas, dan krijg je issues. Dus daar zit dan weer een portability issue in die je zeg maar kan afvangen door te zeggen. Als je die oplossing met een container build met dezelfde operating system host als waar die gerund wordt. Maar dat is wel een handeling, of een taak. En binnen die taak zitten allemaal handelingen om het te doen. We willen er snel doorheen lijkt het wel
R: Ik denk dat we hiermee efficiency heel goed afhangen. We gaan kijken hoe snel en wat voor taak we moeten voldoen om hem draagbaar te maken. Maar misschien kunnen we ook nog kijken naar effectiveness. Wanneer heb je een product effective geport van de ene omgeving naar de andere. Ik denk dat we dat dan momenteel nog niet afhangen
P3: Behoud van functionaliteit toch?
P2: Op dezelfde manier als je de metrics van hiervoor gebruikt. Ik denk dat je daarin een set van deze metrics zou kunnen gebruiken om te meten of het effectief is. Effectiveness is volgensmij sowieso een hele moeilijke. Oh nee, of wel toch? Effectiveness is een hele moeilijke om heel kwantificeerbaar te maken
R: Ja?
P2: Is het effectief
P1: Werkt het allemaal nog hetzelfde?
P2: Ja maar je hebt die functional
P3: Je kan het op verschillende assen gooien
P2: Ja, maar als het functioneel is doet het wat het moet doen, en als het effectief is doet het meer dan wat het moet doen?
P3: Ja, of op een bepaalde graadmeter van hoe goed het iets doet. Als iets tien minuten doet is het wel functioneel, maar niet effectief, of efficient is dat meer
P2: Efficient is dat meer
P1: Daar ga je al, de eeuwige discussie wat het verschil is. Efficiency gaat altijd over tijd
P3: En effectiviteit gaat dan over?
P2: Effectiviteit gaat vaak over impact toch?

P3: Is er geen overlap tussen effectief en functioneel. Dat is eigenlijk een beetje een synoniem toch. In een hele grote zin
R: In grote zin wel
P2: Een hamer is functioneel voor het slaan van een spijker, maar het is ook best wel effectief want je kan er ook ruiten of hoofden mee slaan
P3: Je kan het ook prima met een lepel doen, een spijker erin slaan
R: Dan is het wel effectief, maar dat is efficiency dan weer. Je hebt meer slagen nodig dus het is minder efficiënt. Het is nog steeds even effectief, want bereik je je doel? Je kan nog steeds die spijker met een lepel erin slaan. Dus dat is effectiviteit. Maar goed we dwalen een beetje af van software product kwaliteit
P3: Maar het is dan nog functioneel? Want hij doet nog steeds precies wat je wilt dat die doet
P2: Het is niet effectief, maar vooral niet efficient toch?
R: Goed, een kwestie voor buiten deze focus group
P3: Ja, anders zitten we hier morgen nog
R: Coveren we portability goed?
P1: Ja, zeker
R: Laten we dan doorgaan naar security, de laatste. Security is de “degree to which a product or system protects information and data so that persons or other products have the degree of data access appropriate to their level of authorization”
P1: Daar gaan we
R: Dus hoe meet je of jouw applicatie veilig is
P1: Loggen, wie heeft access
R: Dus in hoeverre zijn assets zo beschermd dat alleen mensen die er toegang tot hebben toegang hebben tot die assets
P1: Deze is weer heel vaag. Ja gewoon dat toch?
P3: Er is meer een checklist toch?
P1: We doen gewoon SSO
R: Dus wat zou je doen qua logging, wie heeft toegang tot wat en wanneer? Hoe zou je dit tot een numerieke waarde kunnen omsmelten
P3: Aantal access groups?
P1: Het gaat er toch om, hoe is het numeriek te maken?
P3: Access groups ofzo, als je er honderd hebt. Maar dat is afhankelijk van hoe groot je applicatie is
P1: Ja, hoe is het numeriek te maken?
P2: Heel simpel toch, je wens. Honderd is nul, of tien is nul. Dus als je iemand hebt die niet in je lijst staat om het zo simpel te houden, en die klikt wel op je website op een knopje waar je niet bij mag. Dat is dan 1, en dat is dan al een minpunt in feite. Want hij mag niet op dat knopje klikken, maar het gebeurt wel
P1: Als je het moet zeggen is het gewoon het aantal illegale gebruikers. Maar hoe weet je dat dan?
P2: Ja, complement regel. Dus alles behalve degene waar je access aan mag geven. Wat doen die op jouw applicatie
P1: Maar het moet toch numeriek zijn, het schiet niet op
P3: Of stel je hebt honderd gebruikers, hoeveel zijn er toegekend aan users groups
P1: Je kan een licentie opnemen
P3: In hoeverre is je gebruikersgroep gecovered. Dat is iets numerieks
P1: Hoe is je gebruikersgroep gecovered?

P3: Zit iedereen in een emmer van toegang? Of zijn er mensen die zich overal doorheen bewegen
P2: Bedoel je hier de principal of least responsible of zoiets. Least privileged. POLP.
P3: Ja maar stel alles is gebasseerd op groepen, groep analyst mag alleen daarbij, groep developer mag overal bij. Maar hoeveel gebruikers vallen niet binnen een groep.
P2: Wie is niet gecovered, en waarom niet. Zeg maar gecertificeerde niet user group users, versus mensen die er wel
P3: Hoeveel overlap zit er in de groepen, is dat gewenst
P2: Implementatie van security lagen zoals firewall is vanzelfsprekend denk ik. Security layers
P1: Niet perse, altijd, maar het zegt wel iets. Als je nul hebt is het niet best
P2: Heeft een applicatie internet nodig of niet, dat is vaak wel een dingentje. Een applicatie die internet nodig heeft dan kan je je afvragen of die niet een soort backport open is waar mensen in kunnen om in te breken
P1: Wat is dan de numerieke waarde?
P2: Ja, een ja of nee vraag. Wel toch? True of false
R: Ja zeker, dat moet zeker kunnen
P1: Dus internet access, vraagteken
P2: Internet access, true false. Want met internet access krijg je weer allemaal van die dingen als zit het binnen een VPC
P1: Is er SSL, HTTPS?
P2: SSH
P1: Aantal security protocols
P3: Ja maar je moet wel voorzichting zijn, dat alles met internet niet perse lager secure vindt als dingen zonder internet. Het is pas niet secure als de dingen die erachter zit niet
P2: Het kan best zijn dat je juist internet access wilt hebben zodat je weet als iemand lokaal in loopt te breken op je applicatie, maar goed
P3: Ja, daar kan je heel lang over
P1: Heel stom, maar misschien ook heeft iedereen een account
P3: Ja, hoe krijg je uberhaupt toegang zeg maar. Maar dat is weer moeilijk te kwantificeren
P2: Je kan ook zeggen, hoeveel versies loopt een applicatie achter op wat de laatste update is. Met andere woorden, je hebt een software die versie een is, en er wordt nog 0.8 gebruikt, en hoeveel versies mag die achterliggen voordat de applicatie werkelijk gewoon wordt gezegd vanuit de controller: je mag hem gewoon niet meer gebruiken
P3: Op die manier bedoel je, dat je hem zelf niet hoeft up te daten
P2: Stel je voor je hebt een Iphone en die komt uit 2011 die werkt niet meer met de software van nu, maar daar hebben ze eigenlijk ook gewoon gezegd, we gaan niet eens proberen of het werkt. Het werkt gewoon niet meer. En wat is je policy daarin en hoe dat afgedekt is
P3: Ja, dat is een goeie
P1: We hebben er echt veel
P2: Goed punt om te stoppen zou ik zeggen
P3: Ik merk dat ik ook een beetje leeg ben
R: Dan is het de perfecte tijd om af te ronden. Super bedankt voor al jullie input. Ik zal in ieder geval de resultaten naar jullie toesturen in de uiteindelijke thesis. En dan volgt dus ook nog een kleine questionnaire met jaren van ervaring in je huidige rol. Dat was m dan. Super bedankt. Dan ga
]ik nu de deelname uitzetten
P3: Dankjewel