De ketentest is vaak een hete aardappel die mensen in een organisatie vaak het liefst zo snel mogelijk doorspelen. De organisatie waar ik inmiddels een tijdje werk, worstelt ook al lang met de ketentest. Waarom is het zo lastig om een werkende ketentest te organiseren? Hoe neem je de blokkades weg en hoe maak je de ketentest onderdeel van continuous delivery? Ik begin met een overzicht van de voetangels en klemmen in het werkveld van de ketentest.

Niemand voelt zich verantwoordelijk voor de keten

Testers in de verschillende teams willen meestal wel bijdragen aan ketentesten: zij zien de risico's wel. Maar ketentesten kost aandacht, tijd, inspanning, geld en die krijg je niet zomaar beschikbaar. Lastig punt daarbij is dat vaak niemand zich verantwoordelijk voelt voor de keten. De silo's in een organisatie zorgen voor focus op de eigen processen. 'Voor ons verandert er niet zo veel', aldus een functioneel beheerder van een bronsysteem in de keten. Maar de verandering in de data uit dat systeem kan grote gevolgen hebben in de keten.

Systemen worden ontwikkeld en getest op eilandjes

Ontwikkelaars en testers hebben hun handen vol aan het doorgronden van hun eigen systeem. Koppelingen met andere systemen vormen daarbij een extra complexiteit. Men doet beperkt kennis op over 'het andere systeem'. Laat staan dat men weet hoe informatie zijn weg precies vindt langs een keten van systemen dat hoort bij het van A tot Z doorlopen van een bedrijfsproces. Je hoort 'Ik test tot de grens van mijn systeem en niet verder'.
Het *is* vaak ook complex in de zin van veel informatie, het vergt veel van iemand om al die kennis op te doen.

Testdata komt uit productie

Organisaties gebruiken vaak testdata uit productie. Dat gebeurt bijvoorbeeld omdat men dan productie-like kan testen. Klinkt op zich logisch, maar het is om verschillende redenen niet gewenst om een functionele ketentest op te zetten rond productiedata. We dekken in de ketentest risico's af, waarbij de werking van een systeem afhangt van de gegevens die doorkomen uit een ander systeem in de keten. Daarvoor maken we specifieke tests waarbij we specifieke testdata nodig hebben, waarvoor we niet afhankelijk willen zijn van wat er 'toevallig' in productie staat.

Daarnaast willen we tests kunnen herhalen, wat lastig is als na een verversing de productiedata veranderd zijn en ik opnieuw moet gaan zoeken naar een geschikte casus.

Een heel ander punt met betrekking tot gebruik van productiedata is dat in veel organisaties persoonlijke gegevens in de productiedata zitten. Het gebruik van persoonlijke gegevens voor testdoeleinden is wettelijk niet zonder meer toegestaan. Het wettelijke regime op gebruik van persoonlijke gegevens (dat trouwens al heel lang bestaat) is met de AVG-wetgeving verder aangescherpt en de handhaving wordt strakker.

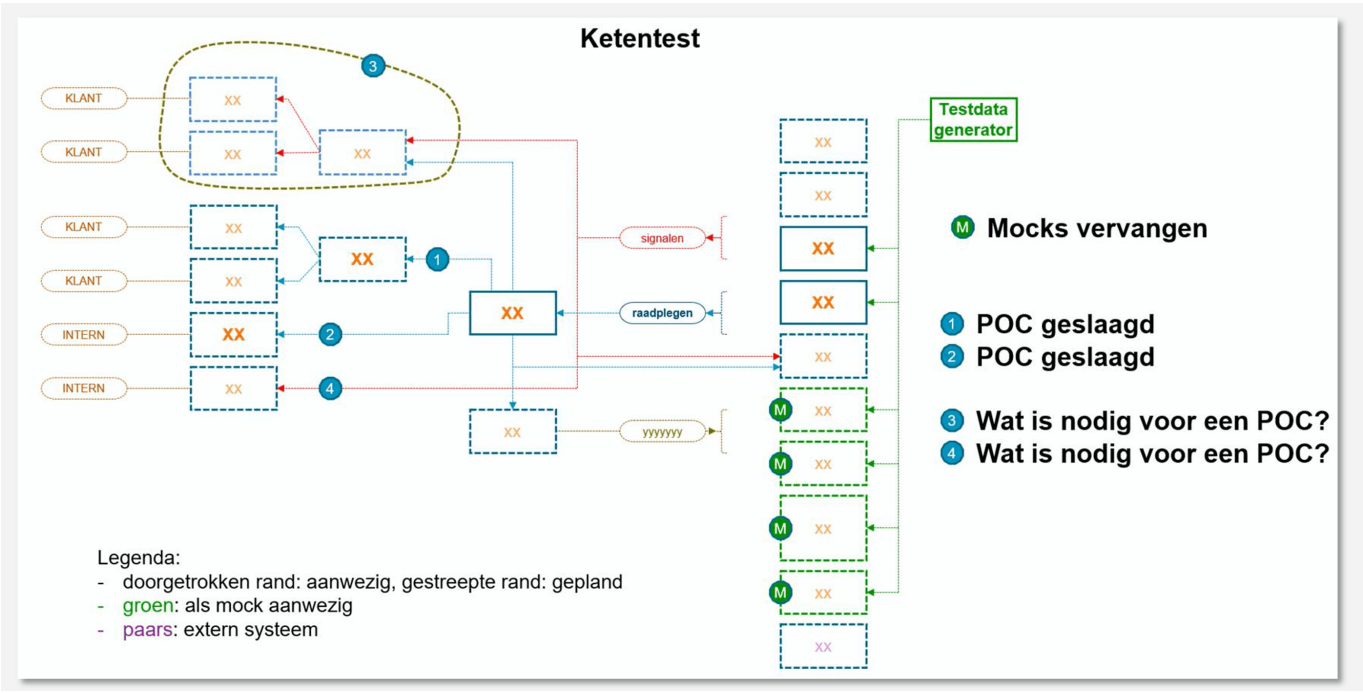
Testdata sluiten niet op elkaar aan

Organisaties hebben een praktisch punt als ze een dump van productie in de systemen in de ketentestomgeving willen plaatsen: de data in de verschillende systemen passen op elkaar. Maar goed, die route heb ik in het vorige punt afgesneden. De meeste ontwikkelteams hebben wel een vulling voor hun systeem met testdata ten behoeve van het functioneel testen. Stel nou dat de testdata in de verschillende systemen op elkaar zouden aansluiten en zouden passen op de ketentestgevallen, dan is het een kwestie van de systemen koppelen en gaan. Maar ja, de systemen worden ontwikkeld op eilandjes met hun eigen focus en testdoelen. Daar hebben we het al over gehad. Helaas.

Verversen van data is moeilijk of niet mogelijk

Dus moeten we kijken hoe we in alle systemen testdata kunnen toevoegen die passen bij de ketentests. Dat lijkt een simpele kwestie, maar het blijkt dat dit voor sommige systemen niet per se makkelijk is. Bijvoorbeeld omdat men ooit is begonnen met gegevens uit productie, die op zich wel heeft geanonimiseerd en hun tests daarop hebben geënt. Voor het toevoegen van specifieke, gefingeerde data heeft men dan geen werkende procedure.

Met consistente data in de systemen in de ketentestomgeving kunnen we testen. Dan doet zich het volgende voor: in een ketentest ‘verbruiken’ we meestal testdata. Transacties muteren data in systemen, waardoor dezelfde test herhalen niet gaat. Dat is nodig voor het reproduceren van bevindingen, voor het testen van opgeloste bevindingen en voor regressietesten. Bepaalde systemen blijken dan geen knopje te hebben om terug te gaan naar de beginstand of op een andere eenvoudige manier de mutaties te vergeten.



Daarnaast willen we ook nog dat …

Oké, het is gelukt om met alle partijen een draaiboek te maken voor het uitvoeren van een ketentest. Met het klaarmaken van testdata, uitvoeren van andere voorbereidende handelingen, draaien van de tests, analyseren van de testresultaten, oplossen van bevindingen is dagen, zo niet weken, doorlooptijd gemoeid. Een volgende testronde moet opnieuw worden ingepland en vergt opnieuw een fikse doorlooptijd. Dat terwijl steeds meer systemen in een geroutineerd CI/CD proces vrolijk tien keer per dag wijzigingen uitrollen naar productie. Zonder een ketenregressietest is er theoretisch tien keer per dag een risico dat één van de ketens waar het systeem deel van uitmaakt stuk gaat. Hoe trekken we de ketentest in de pipelines van de betrokken systemen?

Daarvoor moeten we:

- de ketentest makkelijk kunnen herhalen;
- geautomatiseerd testdata kunnen resetten en herladen;
- de tests geautomatiseerd kunnen draaien;
- de geautomatiseerde tests opnemen in de deployment pipelines van de systemen.

Met alle knelpunten en uitdagingen die we al hadden: hoe gaan we dit bereiken?

Nou, hier gaan we dan.

Iedereen is verantwoordelijk voor de keten

Als niemand eindverantwoordelijk is voor de keten, dan is dus *iedereen* verantwoordelijk voor de keten. Iemands betrokkenheid eindigt niet bij de systeemgrens: een systeem is pas succesvol als het systeem zijn rol goed vervult in de context van de hele keten. Business value is pas geleverd als de hele keten werkt. Uiteraard zijn er afhankelijkheden en daar komt het aan op teamwork en samenwerking. Dit is een kwestie van mindset. Product owners moeten teams ruimte geven om hun bijdrage aan de ‘gezamenlijkheid’ in te vullen. Systeemtesters moeten energie steken in het meer leren kennen van ‘de andere systemen’ en dat is best pittig. Een ketentestmanager faciliteert de ketentest en geeft ondersteuning aan de betrokken mensen om hun rol in de ketentest in te vullen. Door zo weinig mogelijk centraal te beleggen, houdt de organisatie de verantwoordelijkheid voor de ketentest bij de gezamenlijke systemen.

We gebruiken geen data uit productie voor de ketentest

Het definiëren van ketentestgevallen doe je samen. Kies daarvoor bijvoorbeeld de volgende benadering voor de dekking:

1. We doen geen ketentests op risico’s die ook op systeemniveau gedekt kunnen worden;

2. Ketentests raken alle betrokken systemen;
3. We voegen alleen tests toe als de risico's de inspanningen verantwoord maken (PRA basis).

Vervolgens stellen we een dataset samen waarmee die tests kunnen werken en vullen de systemen precies daarmee.

Systemen moeten data kunnen resetten, aanvullen

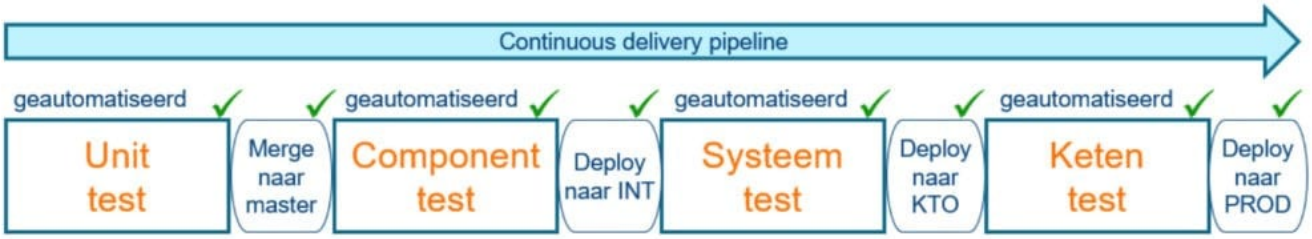
De testdataset bij de ketentestgevallen wordt ingeladen in de systemen. Een systeem mag pas 'meedoen' in de ketentest als ze dit 'met een druk op de knop' kunnen. Systemen die dat niet kunnen, vervangen we (hopelijk voorlopig…) door mocks. Daarmee leggen we de verantwoordelijkheid voor de ketentestbaarheid nadrukkelijk bij de teams. Toch is het zaak mocks zo snel mogelijk te vervangen door de echte systemen, omdat het onderhoud van mocks bewerkelijk is en het uitbreiden van de ketentest arbeidsintensief maakt.

De ketentestomgeving is een machinale testomgeving

Als een systeem geen geautomatiseerde tests kent, dan moet dat eerst worden geregeld, voordat aan de ketentest kan worden meegedaan. De ketentestomgeving is namelijk het terrein van machines: pipeline jobs besturen het resetten en inladen van testdata, het draaien van de testen en het loggen van de resultaten. De omgeving is daarmee in principe niet beschikbaar voor testers om met de hand tests te doen. Ik zeg hierbij in principe, want als er een bevinding is, dan is de omgeving nodig voor probleemanalyse – door mensen. Ik voorzie hierbij dat zodra ketentests in de deployment pipelines van de systemen worden uitgevoerd, één ketentestomgeving onvoldoende is. Er is ook een omgeving nodig om bevindingen te analyseren en ook om nieuwe geautomatiseerde ketentest te ontwikkelen.

De dekkingsopbouw van tests van ieder systeem is op orde

Bij het opstellen van testgevallen op de ketenomgeving stellen we de controlevraag: kan dit risico ook op systeemniveau gedekt worden? Het komt in de praktijk namelijk voor dat testers zeggen een ketenomgeving nodig te hebben om hun eigen systeem te testen. De ketentestomgeving is hier niet voor bedoeld. In plaats daarvan moet het ontwikkelteam (betere) mocks bouwen voor hun systeemtest en daarmee de afhankelijkheid van de keten oplossen.



Hoe staat het er nu mee?

De organisatie waarvoor ik werk, heeft een ketentestomgeving draaiend waarop iedere nacht de testdata opnieuw wordt uitgerold en waarop in een aantal ketens geautomatiseerde tests draaien. Veel systemen zijn nog gemocked, maar de komende maanden verwachten we een aantal systemen toe te kunnen voegen die in de plaats komen van 'hun' mocks. Een belangrijk bedrijfsproces kan helaas nog niet worden getest omdat een cruciaal systeem pas over een jaar wordt herbouwd, waarmee het kan voldoen aan de voorwaarden voor de ketentestomgeving. We wachten daar op voordat we een ketentest gaan opnemen in de deployment pipeline van systemen. Tot die tijd hebben we wel een dagelijks vangnet van regressie in een aantal ketens, heel wat meer dan wat we daarvoor hadden!

Lange adem

Voor het opzetten van een geautomatiseerde ketentest is een lange adem nodig en de nodige support aan teams om hun systeem te laten voldoen aan de instapeisen voor de ketentestomgeving. Maar het gaat lukken, we willen het en het kan!

Kees Blokland