TESTEN VAN DE API ACHTER EEN MOBILE APP

Door Marc van 't Veer • marc.vantveer@polteq.com

'There's an app for that' is een officiële slogan en commercial van Apple die direct begrepen wordt door de meeste mensen. Ongeacht welke functionaliteit gewenst is: er kan een app voor gevonden worden. De marketingafdeling van T-Mobile denkt precies zo en wil klanten direct toegang geven tot hun essentiële data zoals facturen en belstatus. Maar hoe geef je deze ervaring aan iedereen en niet alleen aan de iPhone klanten? Wat doe je dan voor je klanten met een Blackberry, Windows of Android telefoon? 'En we willen ook nog een Facebook integratie!' voegde een marketing medewerker eraan toe.

Met deze diversiteit aan vragen heb je een API nodig! Maar wat is de toegevoegde waarde van een systeemtester in een cloud enabling project? Waarom niet direct een app met een API verbinden en zo testen?

Screen scrapers

Op dit moment bestaan er zogeheten screen-scraper-apps die de T-Mobile website scannen, de daar gevonden data interpreteren en presenteren. Als website verandert, dan valt een app om. De interpretatie is niet altijd correct en het wachtwoord van T-Mobile wordt opnieuw gebruikt in de app. Als antwoordt hierop wil T-Mobile een beveiligde toegang geven en controle krijgen over de data die wordt uitgewisseld door de introductie van een API.

Wat is een API?

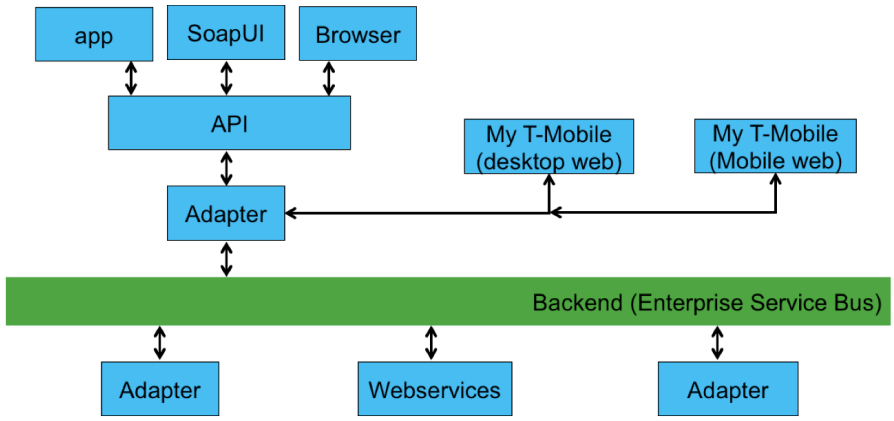
Voordat je een API kunt (systeem)testen, moet eerst begrepen worden wat het is. Een definitie van een API kan gevonden worden op Wikipedia: een *Application Programming Interface*. Dit is een verzameling definities op basis waarvan een computerprogramma kan communiceren met een ander programma of onderdeel. Hier is een API een publieke webservice op internet. Deze API definieert een generieke toegang tot de functionaliteit daarachter. In het kort is een API een raamwerk voor de communicatie tussen systemen. Voor T-Mobile is de API een zelfstandige interface en is voortgebouwd op de interne Service Oriented Architecture (SOA). Hiermee wordt de keten een stap groter. Vanuit systeemtest gezien, zit in de korte API-definitie alles:

- het raamwerk en alle elementen hierin;
- de communicatie;
- de data die verzonden wordt en gepresenteerd;
- een 'tussensysteem' dat aangeeft dat er een integratie is.

Systeemtest van een API en SOA

Zo bekeken bestaat de systeemtest uit drie onderdelen: testen van het raamwerk (de structuur), testen van de data die gecommuniceerd wordt (functionaliteit) en testen van de integratie met andere systemen. Een eerste belangrijke stap is te begrijpen hoe een API bediend kan (moet) worden. De API is een publieke service die bijvoorbeeld via een URL in een browser aan te spreken is. De URL bevat het adres van de service en het uit te voeren commando. Een voorbeeld van een publieke API van de openbaar vervoerder Nederlandse Spoorwegen (NS): http://nsapi.xmpp.openov.nl/stations. Dit is het adres voor alle NS stations in Nederland, als je hierachter stations typt, dan krijg je de stations namen en met stations/ (met forward slash) de afkortingen om verder te navigeren. ➔

Het is te vergelijken met de Windows command line. Er is geen front-end aanwezig alleen cryptische commando's en op basis van de output navigeer je verder.



Schematisch overzicht uitbreiding SOA architectuur met API

Risico's

Een typisch risico voor het testen van een API is dat toekomstige gebruikers onbekend zijn en waarschijnlijk nog niet bestaan. Een ander risico is dat integraties tussen bijvoorbeeld app en API nog niet zijn uitgevoerd. De toekomst is nog niet gespecificeerd, hoe kun je hier dan testen voor opstellen en uitvoeren? Andere voorbeelden waarbij risico's optreden zijn (test)data en scope. Met welke data kan een toekomstige gebruiker tegen de API 'praten'? Waar begint de verantwoordelijkheid van een tester en waar eindigt deze als je de API moet testen?

My T-Mobile app met testdata van de API.

Acht fasen

Bij het realiseren, bouwen, van software doorloopt T-Mobile normaal vier fasen, van development tot en met automated regression test. Bijzonder bij de introductie van de API zijn de extra integratiefasen, zoals productie-integratietests. De strategie voor het testen van de API bij T-Mobile bestaat uit: 'Het gecontroleerd integreren in de complete infrastructuur (SOA-architectuur)'. ➔

In totaal worden er acht fasen doorlopen:

1. Development, Systeemtest en Systeem Integratie Test (SIT) door T-Mobile;
2. Integration (internal: API and complete backend);
3. app Development door app leverancier;
4. Acceptance;
5. Integration (external: API with prototype app;
6. Automated regression test;
7. Productie-integratietest;
8. Aftercare.

De zevende fase, productie-integratietest, bestaat uit meerdere stappen, waaronder geselecteerde T-Mobile medewerkers die op productie als eerste de app kunnen downloaden en testen of de complete keten werkt. De aanpak van systeemtest wordt gestructureerd door de REST architectuur van de API. (Voor informatie over REST zie: http://nl.wikipedia.org/wiki/REST en https://capi.t-mobile.nl/documentation/rest.) De systeemtest heeft vooral een technische insteek en niet zozeer een functionele. Onderdelen van de systeemtest zijn bijvoorbeeld het testen van de resources, internet media types, de caching van output van een API-aanroep en de verschillende presentatie vormen (HTML, XML en JSON).
(Informatie over internet media types, zie: http://en.wikipedia.org/wiki/Internet_media_type.)

Beperkingen testomgeving

De systeemtest is een gecontroleerde fase in een gecontroleerde omgeving. De meeste defects vind je op het grensvlak tussen de API en de app. Zo bleek de app niet om te kunnen gaan met de standaard HTTP status codes, zoals 401 - Unauthorized. Een reden voor deze melding kan zijn dat er een 'Invalid username or token' is, maar ook 'Account has been temporarily locked out'. Welke melding moet de app nu laten zien? Een oplossing is een implementatie van een unieke API Error Code voor elke situatie. Dit is slechts één voorbeeld van een integratiedefect (tussen app en API). Een belangrijke les is dat een aantal tests niet uitgevoerd kan worden op een testomgeving. Dat komt doordat op deze omgeving niet alle klant/product configuraties getest kunnen worden, waardoor mogelijke defects pas gevonden worden op de productie omgeving. Er is dus behoefte aan een testfase als 'Productie Tests'. Als een bepaald abonnement niet getoond wordt zoals verwacht, dan bestaat er voor de website een backup (trick) om dit te corrigeren in de presentatie, omdat deze door T-Mobile zelf beheerd wordt. De belangrijke les hieruit is dat voor de API geen controle is over de presentatie van de data op de app.

Conclusie

Na alle testfases is de API succesvol live gegaan. De API is sinds december 2011 door 250.000 gebruikers aangeroepen via een app op een iPhone of Android telefoon. Verder is de API voorbereidt voor een Windows en Blackberry telefoon, maar ook voor Facebook. Voor een systeemtester blijkt het bij het testen van een API een groot voordeel te zijn als je ervaring met SOA-omgevingen hebt. Als systeemtester heb je bovendien ook nieuwe verantwoordelijkheden bij alle fasen van het hele project, bijvoorbeeld voor het voorbereiden van testdata voor andere testfases/testafdelingen en het uitvoeren van productietests.

Een API's is een enabling technologie die de deur opent naar de nieuwe wereld van cloud-computing! ←