Exploratief testen gedefinieerd
Hardnekkige mythes ontkracht!
Huib Schoots

Dit artikel is gepubliceerd in TestNetNieuws 2013-2 in Oktober 2013
(https://www.testnet.org/bibliotheek.html)

Het is niet de sterkste van een soort die overleeft, ook niet de intelligentste. Wel degene die zich het beste aan veranderingen kan aanpassen. (Charles Darwin)

Er is veel gezegd en geschreven over exploratief testen (engels: exploratory testing). Helaas blijkt dat exploratief testen nog vaak onbegrepen is. Nog veel te vaak lees ik onjuistheden over exploratief testen. Dit artikel zet kort en bondig uiteen wat exploratief testen is, hoe het kan worden gepland, gestructureerd en traceerbaarheid biedt. Daarnaast worden enkele onjuistheden die gepubliceerd zijn gecorrigeerd.

Definitie
Exploratief testen (ET) is een benadering van het testen van software die bestaat uit simultaan leren, testontwerp en testuitvoering. Cem Kaner, die de term in 1983 voor het eerst gebruikte, definieert het als "een manier van testen van software die de persoonlijke vrijheid en de verantwoordelijkheid van de individuele tester benadrukt door voortdurend de kwaliteit van zijn/haar werk te optimaliseren door test-gerelateerd leren, testontwerp, testuitvoering en interpretatie van de testresultaten als onderling ondersteunende activiteiten te beschouwen die gedurende het project parallel lopen."

Het wezenlijke kenmerk van ET is dat de tester actief bezig is met de software. Het testen is een cognitief proces: actief, doelgericht, nieuwsgierig wordt de software onderzocht. Terwijl de software wordt getest, doet de tester kennis op die weer input is voor nieuwe testen.

Test techniek?
Exploratief testen wordt vaak gezien als een black box-testtechniek, maar de context-driven-beweging beschouwt het als een manier van testen, een testaanpak of methodiek die kan worden toegepast op elk test proces, in ieder stadium van het ontwikkelingsproces. Het is niet afhankelijk van gekozen de testtechniek, noch het object dat wordt getest. Iedere testtechniek kan exploratief of vooraf gescript worden uitgevoerd.

Ad-hoc
Op wikipedia staat het volgende: "*Ook is het mogelijk en heel populair om minder gestructureerd (ad hoc) te testen zoals "exploratory testing" of "error guessing". Voordeel daarvan is dat het weinig voorbereiding kost, nadeel is dat er mogelijk niet volledig getest wordt. Ook kunnen gevonden fouten niet reproduceerbaar zijn, waardoor ze moeilijker te vinden en te herstellen zijn.*" [1]

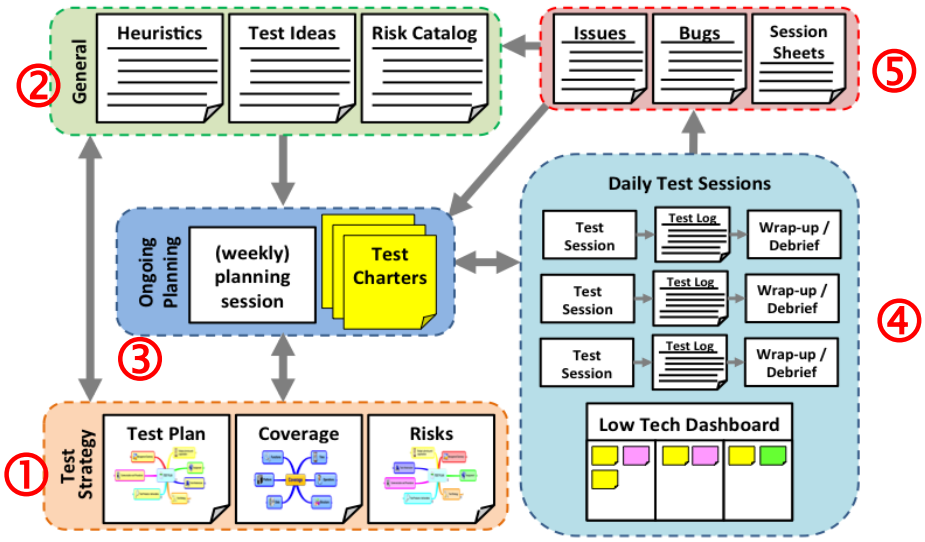
ET is inderdaad ad hoc, want ad hoc betekent letterlijk "voor dit speciale geval" of "eenmalig" of "voor één bepaald doel" [2]. Maar dat wil niet zeggen dat het niet gestructureerd is. ET is uitermate gestructureerd, daarop kom ik later in de paragraaf structuur op terug. Daarnaast beweert wikipedia dat er mogelijk niet volledig getest wordt. Door ET toe te passen zoals in dit artikel beschreven, zorgt de tester ervoor dat er precies genoeg getest wordt en de belangrijkste testen als eerste worden uitgevoerd. De laatste opmerking in de tekst van wikipedia zegt dat gevonden fouten niet reproduceerbaar zijn. Door goede bevindingen te schrijven hoeft dit helemaal geen probleem te zijn. Onderzoek toont zelfs aan dat met ET de bevindingen beter zijn.

[1] http://nl.wikipedia.org/wiki/Testen (software)
[2] http://www.encyclo.nl/begrip/ad hoc

Onderzoek bewijst…
In 2007 is er onderzoek gedaan naar de effectiviteit van het vinden van bugs[3]. Opvallend genoeg bleek dat traditioneel en exploratief testen even effectief waren als je kijkt naar de test uitvoer tijd en het aantal en de ernst van de gevonden fouten. Maar het traditionele testen nam vijf (!) maal zoveel tijd in beslag vanwege de voorbereiding. Bovendien werden bij het traditionele testen twee keer zoveel foutieve bugs gemeld.

Informatie verzamelen en leren
Testen wordt vaak gezien als een exacte wetenschap. Maar ik denk dat testen veel meer een sociale wetenschap is. Testen is een zoektocht is naar informatie, het is het verzamelen en vastleggen van informatie over dingen die belangrijk zijn. Jerry Weinberg definieert testen als: "testen is het verzamelen van informatie om een beslissing te informeren". Rikard Edgren (key-note op het komende najaarsevenement) schreef recent een briljante "open brief" waarin hij testen definieert[4]. Testen is veel meer dan bugs vinden en kijken of aan requirements is voldaan. Testen is vooral leren en dat is een aspect dat in ET centraal staat!

Structuur
Ik leg ET graag uit aan de hand van de volgende illustratie:



1. Test strategie
Ook in een ET traject zal er een test plan worden opgesteld. Ik gebruik hiervoor graag mind maps en maak daarbij gebruik van het Heuristic Test Strategy Model (HTSM) van James Bach[5]. Mind maps zorgen dat mijn plannen kort en bondig blijven, het HTSM geeft me houvast (structuur) om de juiste zaken te beschrijven. Met de klant bespreek ik de gewenste dekking en leg die vast met behulp van "San Francisco Depot" [6] in een aparte mind map. Dit zal tevens het uitgangspunt zijn van mijn testen. Indien nodig maak ik nog een mind map met de product risico's.

---

[3] "Defect Detection Efficiency: Test Case Based vs. Exploratory Testing" by Itkonen, Mäntylä and Lassenius (proceedings of the International Symposium on Empirical Software Engineering and Measurement, pp. 61-70, 2007)
[4] http://thetesteye.com/blog/2013/03/open-letter-to-eurostar-organizers-testing-introduction/
[5] http://www.satisfice.com/tools/htsm.pdf
[6] See Heuristic Test Strategy Model. SFDIPOT: structure, function, data, interface, platform, operations and time.

2. General

Als input gebruik ik documenten en lijstjes die ik voor meerdere projecten gebruik, je zou ze kunnen zien als een soort libraries. Een overzicht van alle product risico's kan een mooie checklist zijn om te zorgen dat we geen risico's over het hoofd zien. Een lijst met test ideeën uit voorgaande projecten heeft hetzelfde doel, maar dan voor de uit te voeren testen.

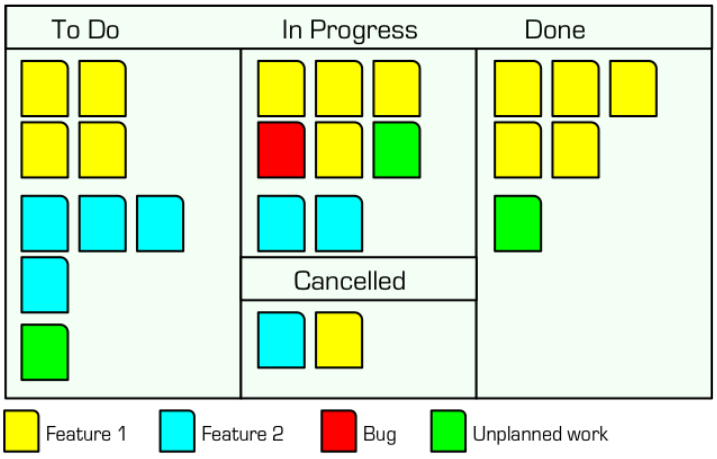
Test Ideeën

Uit mijn ervaring blijkt dat het bedenken van test ideeën één van de moeilijkste dingen is in het testen. Om dit probleem te tackelen, gebruik ik vele bronnen die me kunnen inspireren:

- Heuristic Test Strategy Model
  http://www.satisfice.com/tools/htsm.pdf
- ET dynamics (te vinden in de appendices van Rapid Software testing)
  http://www.satisfice.com/rst-appendices.pdf
- The Little Black Book on Test Design
  http://www.thetesteye.com/papers/TheLittleBlackBookOnTestDesign.pdf
- Software Quality Characteristics (UK)
  http://thetesteye.com/posters/TheTestEye_SoftwareQualityCharacteristics.pdf
- Software Kwaliteit Kenmerken (NL)
  http://dewt.files.wordpress.com/2013/03/thetesteye_softwarekwaliteitkenmerken1.pdf
- Test Heuristics Cheat Sheet
  http://testobsessed.com/wp-content/uploads/2011/04/testheuristicscheatsheetv1.pdf
- Several Checklists
  http://sqa.fyicenter.com/FAQ/Software-Testing-Check-List/
- Touring Heuristic
  http://michaeldkelly.com/blog/2005/9/20/touring-heuristic.html
- You Are Not Done Yet
  http://www.thebraidytester.com/downloads/YouAreNotDoneYet.pdf
- 8-layer testing model
  http://www.shino.de/2012/03/23/testbash-an-8-layer-model-for-exploratory-testing/

Heuristieken

Een heuristiek is een feilbare methode voor het oplossen van een probleem of het maken van een beslissing. Heuristieken zijn informele, intuïtieve oplossingsstrategieën, die mensen ontwikkelen om bepaalde problemen aan te pakken. In tegenstelling tot algoritmen, die altijd en overal werken, zijn heuristieken specifieke strategieën die we leren gebruiken in specifieke situaties en die niet altijd een oplossing garanderen[7]. Het komt dus aan op de vaardigheden van de tester om deze goed en effectief te gebruiken.



3. Planning

Net zoals in agile, geef ik de voorkeur aan een dagelijkse planningssessie met behulp van een whiteboard. Zodoende zorgen we dagelijks dat we de meest belangrijke testen uitvoeren. Bovendien is zo'n sessie een uitstekende manier om elkaar op de hoogte te houden.

Op het whiteboard plak ik stickies die test charters voorstellen. Test charters zijn missies voor een testsessie beschreven in één tot drie regels.

[7] Bron: http://nl.wikipedia.org/wiki/Heuristiek

Voorbeelden van charters zijn:

- "Lees hoofdstuk 4 van de productspecificatie. Bereid een mindmap, en bespreek het met Peter (programmeur) en David (architect).
- Onderzoek het menu import van applicatie X. Identificeer belangrijke functies, maak een dekkingsoverzicht en een risico lijst.
- Test de interface van Applicatie A met Applicatie B.
- Controleer of de schermen van Applicatie C voldoen aan de windows standaard.

4. Dagelijkse test sessies.

Testen worden uitgevoerd in onafgebroken sessies van maximaal 90 minuten. Een tester kan er dus maximaal 3 per dag doen. Per sessie wordt een charter getest. Gedurende de sessie maakt de tester uitgebreide aantekeningen. Indien mogelijk wordt een tester dagelijks gedebriefd door een collega waarin ze elkaar ondervragen over de uitgevoerde testen. Deze review/coaching sessies zorgen ervoor dat er niet over het hoofd gezien wordt. Tijdens het testen, maar ook tijdens de debrief kunnen nieuwe charters ontstaan.

5. Issues, bevindingen en session sheets

Charters worden zoveel mogelijk na iedere sessie uitgewerkt tot session sheets waarin de volgenden zaken worden vastgelegd: charter, tester, datum, geraakte functionaliteit, gebruikte bestanden en data, bevindingen, issues en de eerder gemaakte aantekeningen. Net als in ieder ander test traject worden issues en bevindingen vastgelegd. Issues zijn problemen die de voortgang van het testen in gevaar brengen, bevindingen zijn problemen die de waarde van het product in gevaar brengen.

ET is lastig

Testen is niet makkelijk, althans niet als je het goed wilt doen. Zoals eerder gezegd is goede test gevallen bedenken vaak lastig. Exploratief testen is dus helemaal niet makkelijk, maar daarom juist een leuke uitdaging. De grootste uitdagingen in ET zijn: goede, bruikbare notities maken, voldoende test ideeën verzinnen op het juiste moment, rapportage van de dekking en het managen van het gehele proces.

Leren

ET is niet makkelijk, maar het is te leren. Een cursus is een goede begin en helpt je prima op weg. Maar dan ben je er nog niet. Om het te echt onder de knie te krijgen is veel oefening nodig. Door samen met anderen te testen (pair testing) leer je van elkaar en krijg je feedback op jouw werk. Naast allerlei test relateerde zaken, is het ook nuttig om meer te leren over observeren, experimenten ontwerp, vooringenomenheid, sociale wetenschappen.

De kracht van ET

ET is een krachtige manier van werken. Door de dagelijkse planning en de constant bijsturing (ook tijdens het testen zelf) wordt constant datgene getest dat het belangrijkste is. De tester is veel meer betrokken, juist doordat kritisch denken constant noodzakelijk is. Daarnaast maakt ET volop gebruik van onbewuste kennis[8]. Iedere test die uitgevoerd wordt, is input voor de volgende. Vaak is vooraf niet te voorspellen hoe de software zal reageren. De uitkomst van een experiment, bepaalt de volgende stap van de tester. De creativiteit van de tester wordt hierdoor volop benut.

Toekomst

Exploratief testen wordt gelukkig steeds serieuzer genomen. Steeds meer mensen passen het succesvol toe en het neemt een steeds belangrijkere plaats in, in ons mooie vakgebied. Het is wachten totdat er literatuur gaat verschijnen die het belang van ET onderstreept.

---

[8] Onbewuste kennis is bijvoorbeeld ervaring, attitude of een metal model in je hoofd. Het is vaak moeilijk over te dragen, maar is ontzettend belangrijk. Kan je uitleggen hoe je je evenwicht bewaard terwijl je aan het fietsen bent? Lees het boek "Tacit and explicit knowledge" van Harry Collins over dit onderwerp.

De blogs van Cem Kaner[9], James Bach[10] en Michael Bolton[11] zijn fantastische bronnen van informatie. Ook de blogs van James Lyndsay[12], Jonathan Kohl[13] en Elisabeth Hendrickson[14] bevatten een schat van informatie. Die laatste heeft recent een goed boek gepubliceerd over ET genaamd Explore it! Dit is een absolute aanrader!

Meer informatie:

- Bach, James. "What Is Exploratory Testing?" http://www.satisfice.com/articles/what_is_et.shtml
- Bach, James & Jonathan. "Session-Based Test Management" http://www.satisfice.com/sbtm/
- Rapid Software Testing http://www.satisfice.com/info_rst.shtml
- Bolton, Michael. "ET Resources" http://www.developsense.com/resources.html
- Meer interessante links: www.huibschoots.nl/links

If you cannot trust your testers, you do not make them write more detailed test cases. But you train them! (Rikard Edgren – EuroStar 2012 & Gitte Ottosen – ATD 2012)

[9] http://Kaner.com
[10] http://satisfice.com/blog
[11] http://developsense.com/blog
[12] http://workroomprds.blogspot.com
[13] http://kohl.ca/blog
[14] http://testobsessed.com