Utforskande testning

SAST Stockholm, 2012-02-23

Rikard Edgren
Qamcom Karlstad
rikard.edgren@qamcom.se

qamcom

*“Utforskande testning är en **stil** för programvarutestning som betonar varje testares **frihet och ansvar** att kontinuerligt förbättra sitt arbete genom att betrakta testrelaterad inlärning, testdesign, testutförande och tolkning av resultat som **ömsesidigt stödjande** aktiviteter, som pågår parallellt genom hela projektet.”*

[fri översättning av Cem Kaner]

Varför är det bra?

från början vet vi inte allt som är viktigt

vi behöver lära oss mer

vi kommer designa tester, och lära oss under resans gång

Agenda

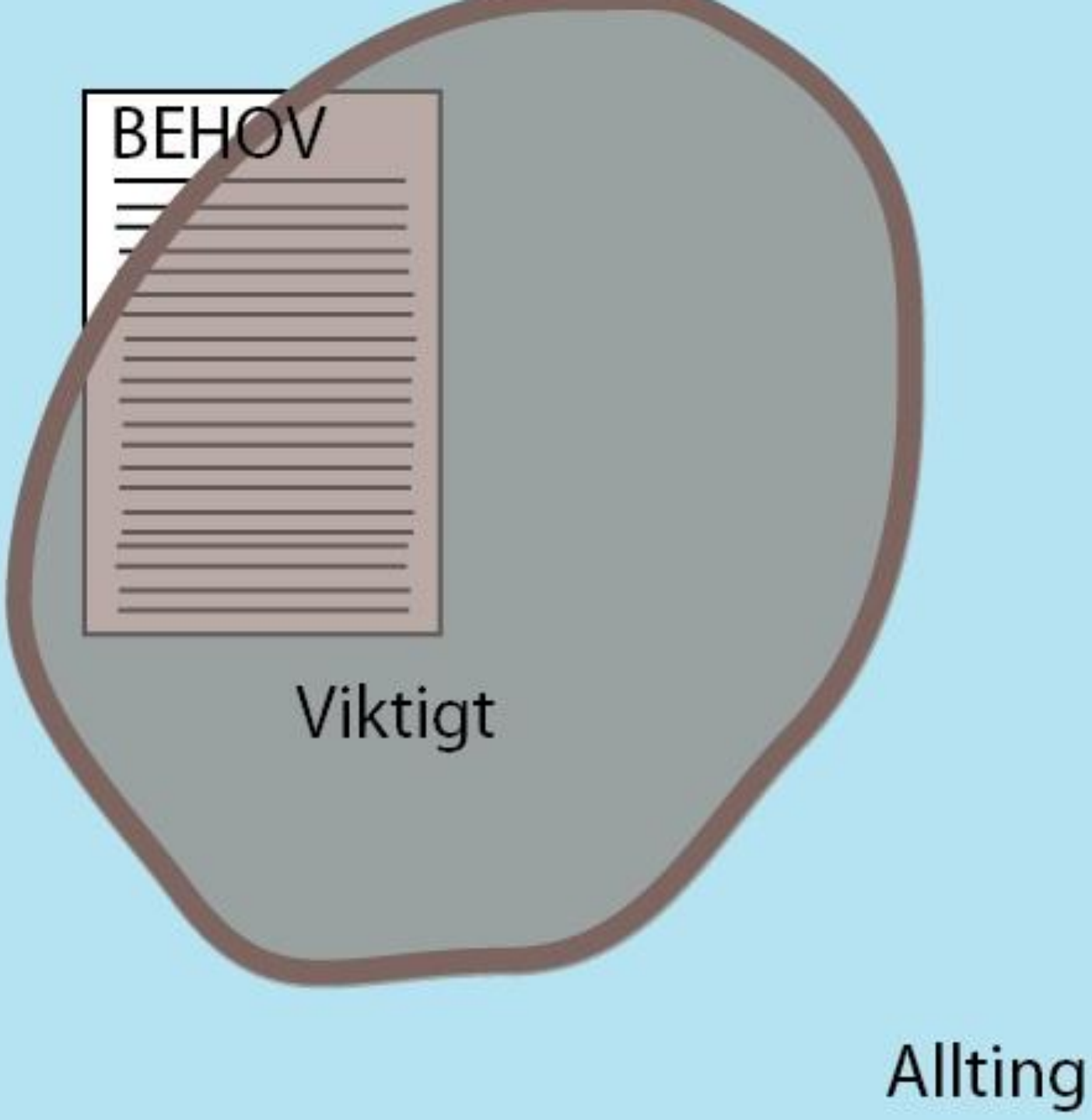
1. Åtta viktiga faktorer
2. Exempel
3. Innehållsrik testning
4. ”Standarder”
5. Sammanfattning

Åtta viktiga faktorer

- Testdesign (utfrågningar, modeller)
- Noggrann observation (och titta på många ställen)
- Kritiskt tänkande (ifrågasätt även ditt eget tänkande)
- Diversifierade idéer (olika angreppssätt, tumregler)
- Rikliga resurser (verktygslåda, vänner)
- Styrning av ditt eget arbete (relatera till uppdrag)
- Snabb inlärning (övning ger färdighet)
- Statusrapportering (vad har testats och hur)

Exempel på användningsområden

- När det är extremt noga att det blir rätt
  - Då vill man testa på alla möjliga sätt, inklusive utforskande.
- När det ska gå snabbt
  - Det tar för lång tid att i förväg skriva ned vad som ska testas.
- Och överallt annars med…
  - Förståelsen av vad som är viktigt förändras alltid.
- Täckningsgrad för utforskande testare
  - Att fånga det som är viktigt.


BEHOV
Viktigt
Allting

Innehållsrik testning

- Att köra samma tester om och om igen kan fånga saker som gått sönder (regressionstestning)
- För att hitta ny information, så ska du köra nya tester, eller variationer på de gamla.

- Mer komplexitet ger rikare tester med större chans att hitta ny, viktig information.
  - Komplex data
  - Komplex omgivning
  - Komplexa sekvenser
  - Komplexa användare

…men gärna enkelt

- Håll det enkelt, du kommer långt utan krusiduller.
- ALAP (As Late As Possible) – bestäm detaljer så sent som möjligt.
- Skriv essensen av tester i enradingsformat (granskningsbart)
- Det är bättre att testa ganska bra på många olika sätt, än perfekt på ett eller två. [Lessons Learned in Software Testing]

Serendipitet

- Att leta efter något, men hitta något annat, som är värdefullt.

- Oerhört vanligt för testare som har ögonen vidöppna.
- Vi testar något, men ser något annat, viktigt, på vägen.

- Specar är en bra start, men det finns mycket mer…


Artefakter
Produkthistoria
Kvalitetsegenskaper
Felhantering
Sökning
Skuld
Användare
Risker
Leverabler
Rädslor
Omgivning
Verktyg
Du
Förmågor
Teknologier
Lagen
Modeller
Standarder
Scenarion
Informationsmål
Konversationer
Image
Data
Kontextanalys
White-box
Konkurrenter
Viktigt
Samlingar
Publika samlingar
Programmet-i-sig
Syfte
Branschkunskap
Fältinformation
Referenser
Kreativa idéer
Bakgrund
Allting

”Standard”-verktyg #1

- Ett väldigt användbart sätt att skapa dina egna strukturer är att ugå från James Bach’s produktelement SFDPOT i Heuristic Test Strategy Model

- **Struktur** – det som utgör själva produkten
- **Funktioner** – det som produkten gör
- **Data** – det som produkten använder
- **Plattform** – det som produkten är beroende av
- **Aktiviteter** – hur produkten kommer att användas
- **Tid** – relationer mellan produkten och tid

”Standard”-verktyg #2

- Att utgå från kvalitetsegenskaper som tvingar dig tänka själv.
  - Ex: *”en erfaren användare kan utföra vanliga uppgifter väldigt snabbt”*
- Dessa kan du ha i bakhuvudet som pågående testidéer, körandes gratis, och beredda att se intressanta saker.

- CRUSSPIC STMPL finns i Heuristic Test Strategy Model.
- thetesteye.com har gjort en fördjupad kategorisering, där du kan välja mellan mer än 100 egenskaper, som **kanske** är viktiga i ditt sammanhang.
  Kvalitetsegenskaper för programvara

Kvalitetsegenskaper

- **Förmågor** - Kan produkten utföra värdefulla funktioner?
- **Pålitlig** - Kan du lita på produkten i många och svåra situationer?
- **Användbarhet** - Är produkten lätt att använda?
- **Karisma** - Har produkten “det”?
- **Säkerhet** - Skyddar produkten mot oönskat användande?
- **Prestanda** - Är produkten tillräckligt snabb?
- **IT-vänlig** - Är produkten lätt att installera och underhålla?
- **Kompatibilitet** - Hur väl interagerar produkten med sin omgivning?
- **Support** - Kan kundernas användning och problem understödjas?
- **Testbarhet** - Är det lätt att verifiera och testa produkten?
- **Underhåll** - Kan produkten underhållas och utökas till låg kostnad?
- **Flyttbarhet** – Är det möjligt att flytta produkten till andra miljöer och språk?

Från thetesteye.com

Exempel: Pålitlighet

- Kan du lita på produkten i många och svåra situationer?

- **Stabilitet:** produkten ska inte krascha , orsaka undantag eller skriptfel.
- **Robust:** produkten hanterar (o)förutsedda fel på ett behagligt sätt.
- **Stresstålighet:** hur beter sig systemet när olika gränser överskrids?
- **Återhämtning:** det är möjligt att starta om och fortsätta efter ett allvarligt fel.
- **Dataintegritet:** all sorts data behålls intakt genom hela produkten.
- **Säkerhet:** produkten medverkar inte till att skada personer eller egendom.
- **Katastrofhantering:** vad händer om någonting riktigt, riktigt allvarligt inträffar?
- **Trovärdighet:** är produktens beteende konsekvent förutsägbart och trovärdigt?

Från thetesteye.com

”Standard”-verktyg #3

- Cem Kaners RIMGEA för buggrapportering (som är en sorts utforskande testning)

- Reproducera
- Isolera
- Maximera
- Generalisera
- Förkroppsliga
- Använd ett sakligt språk

- *Testningen är inte bättre än kommunikationen av resultaten.*

”Standard”-verktyg #4

- SBTM (Session-Based Test Management) – hett!
- Att använda sessioner där testningen dokumenteras gör att man kan visa vad som testas.
- ”Debrief” gör att man kommer på fler idéer.
- Man blir en bättre testare av att synliggöra sitt tänkande.

Utforskande testning

	Utforskande
Lärande	från många ställen
Design	under tiden
Exekvering	med variationer
Tolkning	orakel och omdöme

- Dessa aktiviteter interagerar, och hjälper varandra.

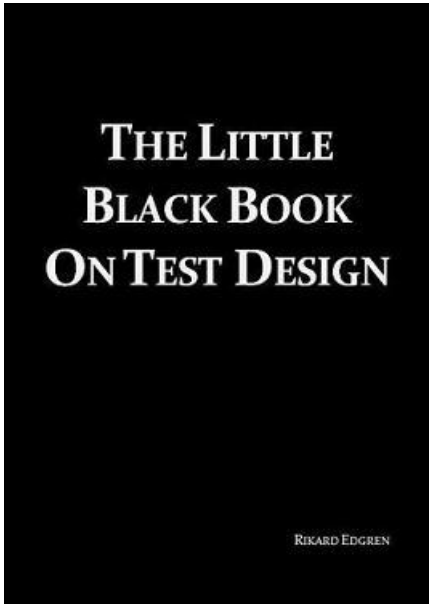
Slutord

- Det handlar om hur du tänker.
- Nackdelar?
  - Kan vara svårt att veta när man ska sluta
  - Svårt veta att man täckt allt viktigt
  - Kräver att man litar på testarna
  - Ifrågasätts av gammalmodiga granskare
- Du behöver hitta DIN EGEN utforskande testning
- Gör ditt bästa, samarbeta, lär dig förstå
  vad som är viktigt

Frågor

- ???

- Litteraturtips:
  - Heuristic Test Strategy Model (Bach)
  - BBST Test Design (Kaner, Fiedler)
  - The Little Black Book on Test Design (Edgren)



www.thetesteye.com

rikard.edgren@qamcom.se