37 källor till testidéer

Vi rekommenderar att du kontinuerligt samlar testidéer från många olika informationskällor.
Titta igenom nedanstående och tänk på risker och värdet av testningen; hitta genvägar för att täcka det som är viktigt.

Produkt	**1. Förmågor.** De mest uppenbara testidéerna handlar om vad produkten förväntas göra. En bra start är krav, exempel och andra specifikationer, eller en funktionslista skapad utifrån programmet. Försök också identifiera outtalade krav, saker som användarna förväntar sig, men som inte dokumenterats. Var beredd på oönskade förmågor. **2. Fel-lägen.** Programvara kan fallera på många sätt, så ställ "tänk om"-frågor för att ta fram tester som undersöker hanteringen av interna/externa, förväntade/oväntade, (o)avsiktliga, realistiska/provocerade fel. Ifrågasätt systemets feltolerans; alla objekt och komponenter kan gå sönder. **3. Modeller.** En tillståndsgraf hjälper dig identifiera testidéer kring tillstånd, övergångar och vägar. En karta över systemets anatomi visar vad som kan testas, och vad som interagerar. Skapa din egen anpassade modell med strukturer som SFDPOT från Heuristic Test Strategy Model. En visuell modell är lättare att kommunicera. Modelleringsaktiviteten brukar ge större förståelse och nya idéer. Många och rika modeller ger bättre testidéer. **4. Data.** När du identifierar vanlig och ovanlig data (det finns alltid brus) så får du en bra start för ett gäng testidéer. Följ enkel och knepig data genom systemet, var inom och utanför gränserna, utmana datatyperna, använd CRUD (Create, Read, Update, Delete), utnyttja beroenden, och titta på datan på många ställen. **5. Omgivning.** Ingen produkt är en ö, så kompatibilitet (hårdvara, OS, program, inställningar, språk) är ett av många viktiga testningsproblem, och undersök också aktiviteter man gör i anslutning till produkten. Om du förstår det stora systemet, så kan du få trovärdiga testidéer, som är svåra att se om man tittar på en funktion i taget. **6. White-box.** En kombination av testarens kritiska tankesätt och utvecklarnas perspektiv utmanar antaganden, och kan hitta fel som är snabba att åtgärda. Titta extra noga på kodvägar och alternativ som är svåra att se ur ett systemperspektiv. Och kodtäckning är inte värdelöst, det kan användas för att hitta saker som inte testats. **7. Produkthistoria.** Gamla problem uppträder gärna i ny skepnad. Sök i ditt bugg- eller supportsystem, skapa en felkatalog, minns kritiska problem och deras rotorsaker. Använd gamla versioner av programvaran som inspiration och orakel. **8. Rykten.** Det finns ofta rykten om produktens kvalitet och problem, varav en del skadar produkten och företaget. Använd ryktena som testidéer; det är ditt uppdrag att ta reda på om de stämmer eller inte. **9. Programmet-i-sig.** När du använder produkten så kommer du få massor av idéer kring vad som är felbenäget, relaterat, intressant. Om du kan äta din egen hundmat (eufemism: sippa din egen champagne), så är det lättare att avgöra vad som är viktigt. **10. Teknologier.** Om du känner till detaljer kring teknologierna som produkten använder sig av, så förstår du vilka områden som är problematiska, du förstår möjligheter och säkerhetsaspekter, du vet vilka parametrar som ändras, och när. Du kan göra rätt variationer, och ha fruktbara, tekniska diskussioner med utvecklarna. **11. Konkurrenter.** Om du tittar på andra lösningar på liknande problem, så kan du se vad som behöver testas, men också få en känsla för vilka egenskaper som slutanvändarna är intresserade av. Det kan finnas interna lösningar (ex. Excel-applikation) att inspireras av, och ofta finns det analoga lösningar för samma syften. Kanske kan du få en briljant idé genom att titta på konkurrenternas support, FAQ med mera?
Bransch	**12. Syfte.** Produktens syften ger dig mål för dina testidéer. Fråga "varför?" några gånger till för att ta reda på de egentliga målen. Dessa kan ge testningen en bred, välvillig start som hittar väldigt viktiga problem, fort. **13. Affärsmål.** Vilka är de viktigaste målen för företaget (och delmål för avdelningar)? Finns det några krav som bryter mot de målen? Känner du till det stora sammanhanget, vet du vad produktens vision och värde är? **14. Produktens image.** Produktens önskade beteende och egenskaper kan vara uttalade eller outtalade, sittandes i väggarna eller i producenternas och konsumenternas tankar. Du kan skriva motiverande buggrapporter om du kan peka på hot mot produktens image, t.ex. genom att visa på avvikelser gentemot marknadsföringsmaterial. **15. Branschkunskap.** Om du känner till sammanhanget produkten befinner sig i, så kan du förstå hur det ger värde till kunderna. Om du inte kan skaffa den kunskapen, samarbeta med någon som har god kunskap om behoven, processerna och miljön. **16. Lagliga aspekter.** Behöver du bekymra dig om kontrakt, böter eller andra legala aspekter? Vad skulle kunna kosta företaget väldigt mycket pengar? Finns det en advokat som kan ge tips om vad som till varje pris ska undvikas?
Team	**17. Kreativa idéer.** Alla produkter är unika och kräver helt nya testidéer. Prova lateralt tänkande (t. ex. Edward De Bonos Sex Tänkarhattar, provocerande operatorer, tvärtom-metoden, slumpmässig stimulans, Google Goggles) för att få kreativa tankar. Metaforer och analogier är ett bra sätt att tänka i nya banor. **18. Interna samlingar.** Använd eller skapa listor över saker som ofta är viktiga i ditt sammanhang. En del kallar detta kvalitetsmönster, andra har produkt-specifika snabbtester. **19. Du.** Du är en användare. Du kan vara en intressent. Du är viktig. Utnyttja dina styrkor från erfarenhet, färdigheter, kunskap och problemkännedom. Använd dina känslor och din subjektivitet för att förstå vad som är viktigt. Om "Kvalitet är värde för en person ", så är det rätt bra om den personen är "jag". Men glöm inte bort dina svagheter och blinda fläckar.

Projekt	**20. Projektets bakgrund.** Orsakerna bakom projektet driver många beslut, och historien bakom tidigare (liknande) projekt är bra att känna till för att kunna göra effektiv testning. **21. Informationsmål.** Det är grundläggande att förstå de uttalade och outtalade målen med testningen. Om du inte har några, skapa egna kvalitets- och informationsmål som kan guida din testning. **22. Projektrisker.** Projektets svårigheter kan delvis hanteras med testning. Du vill veta vilken funktionalitet som utvecklarna har problem med, och du kan ändra din plan beroende på vilka risker som behöver adresseras först. **23. Testartefakter.** Använd dina testidéer, loggar och resultat för fortsatt testning, men titta också på resultat från andra projekt, beta-rapporter, användbarhetsutvärderingar, testresultat från tredjepartskomponenter. Vilka frågor vill du kunna besvara i dina statusrapporter? **24. Skuld.** Genvägar som tas skapar ofta en ökande skuld. Det kan vara projektskuld, lednings-skuld, teknisk skuld, programvaru-skuld, testningsskuld eller vad du vill kalla det. Om teamet har koll på listan över skulder, så kan du skapa testidéer utifrån dem. **25. Konversationer.** Samtal kan ge dig viktigare information än det som finns i specifikationerna. Det finns många som kan hjälpa dig med din testdesign, en del kan bättre bedöma viktighetsgrad, vad kan du få ut av att nämna saker i förbigående? Om utvecklarna litar på att du kan hitta intressanta saker, så kommer de ge dig insider-information om tvivelaktiga delar av programmet. Fråga utvecklarna *"vad tycker du vi borde testa?"* och "*vilken del av koden hade du velat gjort bättre?*" **26. Kontextanalys.** Vad mer i sammanhanget borde påverka vad du testar, och hur? Känner du till marknadsvillkoren och projektets drivkrafter? Är det något som ändrats som leder till nya sätt att testa? Vad testas av andra? Vilka styrkor och svagheter har projektet och dess medlemmar? **27. Många leverabler.** Det finns många saker att testa; det körbara programmet, installatören, API, tillägg, kod och kommentarer, filegenskaper, Hjälp, annan dokumentation, Release Notes, readme:s, reklam, träningsmaterial, demos etc. Alla dessa kan också ha information som kan inspirera. **28. Verktyg.** Om något kan göras väldigt fort är det ofta en bra idé att prova. Verktyg är inte bara ett medel, de kan också vara en startpunkt för utforskning.
Intressenter	**29. Kvalitetsegenskaper.** Det finns alltid kvalitetsegenskaper som avgör om projektet lyckas; ibland är de lätta att nå, ibland kritiska och svåra. Vår definition inkluderar förmågor, pålitlighet, användbarhet, karisma, säkerhet, prestanda, IT-vänlig, kompatibilitet, support, testbarhet, underhåll, flyttbarhet, och en mängd underkategorier. Många av dessa kan användas som pågående testidéer att ha i bakhuvudet; gratis, men redo att identifiera överträdelser. **30. Produkträdslor.** Saker som intressenter är riktigt oroliga för är mycket starkare än risker; de behöver inte prioriteras, de behöver testas. Typiska rädslor som är svåra att verifiera, men värdefulla för testning är: dålig image, fel beslut, skada, ingen gillar produkten. Olika människor har olika rädslor; ta reda på vilka som är viktigast. **31. Scenarion.** Användarna vill åstadkomma eller uppleva något med programvaran, så skapa tester som på ett varierat sätt simulerar sekvenser av produktens beteenden (testa inte bara funktionerna isolerat.) Ju mer trovärdiga användningsmönster du känner till, desto mer realistiska tester kan du utföra. Prova också excentriska såpoperatester för att bredda testtäckningen. **32. Information från fältet.** Förutom kunskap om kundernas problem, miljö, behov och känslor kan du ta dig tid att förstå hur kunderna både lyckas och misslyckas. Intervjua kunder, säljare, marknadsförare, konsulter, support, eller ännu hellre: jobba där ett tag. **33. Användare.** Tänk på olika sorters användare (personer du känner, personas), olika behov, olika känslor och olika situationer. Ta reda på vad de gillar och ogillar; vad de gör i anslutning till din programvara. Du kan spela upp en scen i ditt testlabb och ha rollspel med olika användartyper, vad ger det för testidéer? Bäst är såklart ofiltrerad information direkt från slutanvändarna, i deras sammanhang. Kom ihåg att två liknande användare kan tänka väldigt olika om samma saker.
Externt	**34. Publika samlingar.** Dra nytta av generella eller specifika listor på buggar, kodningsmisstag, eller testidéer. När du bygger upp din egen minneslista, ta en titt på de här: • Appendix A of Testing Computer Software (Kaner, Falk, and Nguyen) • Boris Beizer Taxonomy (Otto Vinter) • Shopping Cart Taxonomy (Giri Vijayaraghavan) • Testing Heuristics Cheat Sheet (Elisabeth Hendrickson) • You Are Not Done Yet (Michael Hunter) Lär dig några tricks och tekniker från böcker, bloggar, konferenser, sök efter tumregler för testdesign, eller uppfinn de som är bäst för dig. **35. Standarder.** Leta upp relevanta bransch-standards, lagar och reglementen. Läs och förstå UI-standards, säkerhetsregler, policy. Kan du hitta information som pekar på luckor i regelverken? **36. Referenser.** Referensmaterial av olika slag är en bra källa till orakel och testinspiration, t.ex. en atlas för en geografisk produkt. All sorts kunskap kan vara användbar, och Wikipedia kan räcka för att få en snabb förståelse av exempelvis en statistisk modell. **37. Sökning.** Att googla är ett bra sätt att hitta saker du letar efter, och saker du inte visste att du behövde (serendipitet.)

Rikard Edgren, Martin Jansson and Henrik Emilsson - thetesteye.com v1.0

inspired by Sabourin's 10 Sources for Testing Ideas, Bach/Bolton's HICCUPPS(F) and Kaner's writings.