

GLÖM INTE DIN QA!

Mikael Lundgren

De flesta lättrorliga metoder för produktutveckling, exempelvis Scrum, ställer annorlunda krav på resultatet av utvecklingsarbetet än traditionella metoder. En effekt av detta som ofta förbises i början är att brister i byggprocess och testning synliggörs som tydliga hinder för att uppnå målen istället för att bakas dolt in i den övergripande planen som "ställtider" och liknande. När man börjar försöka pröva arbeta lättrorligt är detta ett skäl till att det kan upplevas som att en mängd problem börjar uppträda i utvecklingscykeln efter en eller ett par initiala iterationer av framgång.

I Scrum är projektledaren, eller Scrum Mastern, ansvarig för att hålla reda på de problem som uppstår, och arbeta aktivt med företaget för att åtgärda dem. I den här artikeln ska vi titta på vad som kan hända med testningen i en omgivning där resten av avdelningen börjar komma igång med att arbeta lättrorligt.

Projektet kommer att börja gå fortare, i takt med att projektmedlemmarna börjar arbeta tillsammans med produktägaren med att bearbeta och implementera krav, hela tiden med sikte på att ha körbar kod med definierat affärsvärde klart till iterationens slut. Om man redan har en kultur med hög grad av automatisering och kanske testdriven utveckling har



man goda förutsättningar att nå målen. Men om man glömmer testavdelningen i bakvattnet har man snart problem, eftersom lättroblig metodik efter ett tag börjar kräva samarbete mellan programmerare och QA på ett nytt sätt. Istället för överlämningar, färdiga testspecar och fullständiga kravdokument före testningen påbörjas krävs samarbete, peer programming (programmerare och testare sitter tillsammans), rollerna börjar suddas ut och alla tar ansvar för kvaliteten. Följderna av att ha en QA-avdelning som sitter och väntar på fullständiga kravspecar innan testningen kan börja, blir snart tydliga:

- Istället för parallell utveckling och kvalitetssäkring börjar testningen ofta sent i iterationen
- Testningen fortsätter in i nästa iteration med oförminskad kraft
- Buggrättning på förra iterationens funktioner kommer in som okända element i nästa iteration (kräver ställtid, kanske omställning av utvecklingsmiljö osv)
- När testningen av föregående iteration avslutats har programmerarna redan hunnit långt med nästa iterations krav, vilket gör att QA kommer för sent in igen
- Cykeln är igång, och är mycket svår att bryta "i farten".

Efter ett tag kan kontentan faktiskt bli att utvecklarna upplever kontroll och mindre stress, trots högre omvandlingstakt från abstrakta krav till körbar implementation, medan testavdelningen bränner ut sig. Och en annan effekt är att tidsestimaten i iterationerna (och på hela projektet) inte förbättras, eftersom en lejonsvans av osäker kvalitet ständigt släpar efter all körbar kod, och återför teamet till redan implementerade features gång på gång.

Medicinen är enkel att beskriva, men tar tid att genomföra. Det lättrobliga paradigmskiftet måste även innefatta vår testavdelning, som trodde sig kunna luta sig tillbaka och skörda frukterna av det nya arbetssättet!

Gränsen mellan utveckling och test måste göras suddig. Eftersom vi premierar nära samarbete mellan utveckling och beställare måste vi nu även ha samma inställning mellan utveckling och test!

Varför separerar vi utveckling och test traditionellt? Varför har programmerare högre löner än testare, och varför sitter de i olika rum? Svaret är inte självskrivet, men bottnar förmodligen i att programmerare ofta är högutbildade, medan testare rekryterats bland

“Varför separerar vi utveckling och test traditionellt? Varför har programmerare högre löner än testare, och varför sitter de i olika rum?”

“mannen på gatan”. Upplägget med en pipeline, där producerad kod överlämnats till lägre kvalificerad personal för godkännande har länge förhärskat, men nu börjar andra vindar blåsa. Idag finns aspekter av kvalitetssäkring i alla led. Som testare kan man förväntas skriva såväl fixturer som automatiska tester och simulatorer, vilket kräver programmeringskunskap. I USA finns sedan flera år kvalificerade högskoleutbildningar för testare, och möjligheterna att arbeta med både och växer för var dag.

När jag pratar om utveckling brukar jag säga att jag använder ordet “utvecklare” i sitt vidaste begrepp – det innefattar både programmerare och testare! Och varför inte grafiker? Tekniska skribenter? Kort sagt, alla som är med och *utvecklar* produkten. I vissa sammanhang är det provokativt, i andra självklart.

- Låt dina testare sitta tillsammans med programmerarna. För att de ska kunna ta aktiv del i arbetet från dag 1 måste de ju ha färsk information – och programmerarna måste hålla QA ajour med arbetet i utbyte mot många små tester dagligen.
- Många små dagliga byggen (i en kontinuerligt integrerad miljö) istället för stora överlämningar vid spridda tillfällen.
- Detaljering och utredning av krav går hand i hand med utformning av testspec. Är de nödvändigtvis två olika dokument?
- Automatisera tester! Utvecklare måste ta kvalitetsansvar – lägg inte ansvaret för regressionstestning på QA!

Detta är några tumregler. Men det viktigaste är – när du vänder skutan och börjar öka farten, se för allt i världen till att du inte lämnar din QA kvar på kajen! ■ ■ ■



Mikael Lundgren är konsult på Citerus och har arbetat som utvecklare, projektledare och utvecklingschef. Han anser att företag som har en nära relation mellan sin marknadsavdelning och sin utvecklingsavdelning har en enorm strategisk fördel. Om du är av en annan åsikt, eller inte förstår hur det ska gå till, ta upp det med honom på mikael punkt lundgren at citerus punkt se.