

KVALITET KOSTAR

Gustaf Brandberg

”Kvalitet kostar” är fortfarande en utbredd myt inom mjukvaruutveckling och orsaken till att CMM5-certifierade företag fortfarande kan släppa produkter med tusentals kända buggar. Trots rigorösa och väldokumenterade processer tror man helt enkelt att det är för dyrt att släppa felfri kod. Jag hävdar motsatsen: Det är dålig kvalitet som kostar. Det är hög tid att avliva myten.

– Vadå myt? säger du. Det är klart att kvalitet kostar. Se på all tid vi lägger på spårbarhet mellan kod och funktionella krav och på att testa mjukvaran, dokumentera buggar i ett ärendehanteringssystem, hitta orsakerna till felen, rätta dem och sedan låta testavdelningen

verifiera både att vi faktiskt rättade buggen och dessutom inte införde något ny defekt medan vi höll på. Tror du att det är gratis? Vi har helt enkelt inte råd att öka kvaliteten i mjukvaran, det kostar redan alldeles för mycket.

– Det du just beskrev är inte kvalitet utan kvalitetsarbete, svarar jag.



Kvalitetsarbete är dyrt – särskilt om det är manuellt och om brister i kvalitet upptäcks sent och kräver mycket arbete för att rättas till. Frågan är: är det möjligt att öka kvaliteten i mjukvaran och samtidigt minska kostnaden för kvalitetsarbetet?

För att få svar på den frågan kan vi hämta inspiration från Taichii Ohno som är något av en pappa till Toyotas produktionssystem. Jag läste nyligen den engelska översättningen av en bok som han skrev redan 1978 som beskriver hans resa från 1950 och framåt. Hans approach till hög kvalitet till låg kostnad kan sammanfattas med två ord: Stoppa linjen!

Så här skriver han (alla citat är översatta från engelskan av mig): ”Den andra pelaren som Toyotas produktionssystem vilar på [den första är “Eliminera slöseri”] kallar jag för autonomisering, vilket inte ska förväxlas med enkel automatisering. Autonomisering kan också kallas “automatisering med en mänsklig touch”. [...] På Toyota är en maskin som är “automatiserad med en mänsklig touch” en maskin som är försedd med en automatiserad stoppmekanism” (1, s. 6).

Vid varje stopp samlades arbetslaget för att inte bara åtgärda problemet, utan även försöka hitta grundorsaken till att det uppstod och därefter förändra sin process så att problemet aldrig uppstod igen.

Stoppmekanismen på Toyota utlöstes med hjälp av röda rep. Ledningen krävde av arbetarna att de skulle dra i repen om ett kvalitetsproblem av något slag uppstod. Vid varje stopp samlades arbetslaget för att inte bara åtgärda problemet, utan även försöka hitta grundorsaken till att det uppstod och därefter förändra sin process så att problemet aldrig uppstod igen.

Toyotas approach gick precis tvärt emot det som ansågs vara det bästa arbetssättet inom bilindustrin på den tiden. Ford och General Motors resonerade precis tvärt om; maximal effektivitet uppnås genom att man låter de dyra maskinerna producera bildelar för fullt för att slå ut kostnaden för maskinen på så många delar som möjligt. Med den utgångspunkten är ett stopp i linjen något av det värsta man kan råka ut för och extremt kostsamt, något som ska undvikas till varje pris. Istället inspekterade man bilarna precis innan de lämnade fabriken och fixade alla problem där.

– Precis, ropar du! För att vi ska hinna leverera funktioner enligt vår tajta tidsplan måste vi prioritera bort bugggrättning, och vi hinner definitivt inte lägga tid på att förbättra våra processer för att undvika framtida buggar. Det är testavdelningens ansvar att hitta alla buggar som vi måste rätta innan vi kan leverera, inte utvecklingsavdelningens.

Vår mentor Taichii Ohno svarar: “Ett problem tidigt i processen leder alltid till en defekt produkt senare i processen. [...] Genom att bortse från sådana situationer och bara ta hänsyn till produktionsplanen reducerar man både produktiviteten och lönsamheten” (1, sid 4). Genom att stressa fram nya funktioner bygger ni upp teknisk skuld i mjukvaran, som ni förr eller senare måste åtgärda om ni vill fortsätta utveckla kodbasen. Om en snabb leverans nu innebär problem senare kan ni inte uppnå en långsiktigt hög takt på utvecklingen. Med Ohno-sans ord: “Hastighet är meningslös utan kontinuitet. Kom ihåg haren och sköldpaddan” (1, sid 63).

Nå, vad har autonomisering att göra med mjukvaruutveckling och med att minska kostnaden för kvalitetsarbete? Svaret är enkelt. Vi kan också automatisera våra verktyg. Tag till exempel den allt mer spridda vanan att kontinuerligt integrera all ny kod.

Kontinuerlig integration handlar om att regelbundet och automatiskt bygga hela det utvecklade systemet, centralt i en kontrollerad miljö. Så snart en ny version av systemet är byggd kör servern automatiska tester och meddelar direkt om någon kod orsakar problem – stoppar linjen – så att utvecklaren kan åtgärda problemet medan han eller hon fortfarande har koden färskt i minne och innan fler personer, processer och ärendehanteringssystem blir inblandade. Med andra ord kan man kalla det för automatisering med en mänsklig touch.

Även om bygg- och testprocesserna är automatiserade är det utvecklarna som bestämmer under vilka förutsättningar vidareutvecklingen ska stoppas genom att utveckla automatiska test. Det är även utvecklarna som måste agera när integrationsservern säger ifrån, både för att lösa det akuta problemet och för att se till att liknande problem inte inträffar i framtiden.

Nå. Kan man verkligen jämföra mjukvaruutveckling med bilproduktion? Är inte det helt olika saker? Den frågan lämnar jag till dig att avgöra, och det finns bara ett sätt för dig att ta reda på det genom att bilda dig en egen uppfattning om vilka era viktigaste utmaningar är, om något som jag skrivit om här skulle kunna hjälpa er och därefter testa själv. Taichii Ohno säger: ”Förståelse är mitt favoritord”.

Toyota hämtade för övrigt inspiration från amerikanska supermarkets som började dyka upp i Japan i början av 50-talet när de jobbade med att utveckla sitt produktionssystem och just-in-time-konceptet (1, s. 26), så varför skulle inte du kunna lära dig något av bilindustrin?

Kan man verkligen jämföra mjukvaruutveckling med bilproduktion? Är inte det helt olika saker?

För att sammanfatta. Kvalitet kostar inte. Kvalitetsarbete kostar, särskilt om det utförs sent i processen och med långa feedbackloopar och mycket formalia. Dålig kvalitet i releasen kostar ännu mer i form av support, missnöjda kunder och utebliven försäljning.

Döda myten. Våga stoppa linjen! ■ ■ ■

REFERENSER

- Ohno Taiichi (1988). Toyota Production System: Beyond Large Scale Production, engelsk översättning, ISBN . Productivity Press, New York.
- Poppendieck & Poppendieck (2003). Lean Software Development: An Agile Toolkit. ISBN . Addison Wesley.
- Privata samtal med Tom och Mary Poppendieck (oktober 2005).
- Fredriksson, Patrik (2004). Bygg kontinuerligt. Artikel i PNEHM! från Citerus AB.



Sedan han deltog i grundandet av Citerus 1996 har Gustaf Brandberg hållit facklan högt i kampen för bättre kvalitet i mjukvaruvärlden. Skicka honom några uppmuntrande ord på gustaf.punkt.brandberg@citerus.punkt.se.