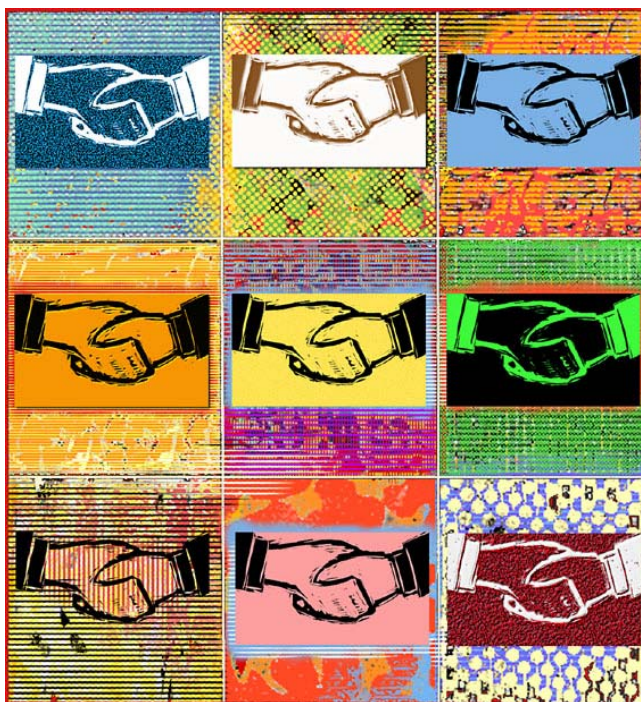


# OM DU ÄR NÖJD SÅ ÄR JAG NÖJD

Gustaf Brandberg

Några tankar om outsourcingkontrakt när man vill jobba lättrorligt



Fler och fler anammar ett lättrorligt arbetsätt med tätt samarbete mellan beställare och utvecklare, korta iterationer och inkrementella resultat. Istället för "vi och dom" jobbar man tillsammans som ett team. En annan trend är ju outsourcing av mjukvaru-utveckling till exempelvis låglöneländer. Men, vad händer när man outsourcar sin utveckling och "vi" jobbar på ett företag och "dom" på ett helt annat?

En uppenbar skillnad är att man nu måste skriva ett formellt och bindande avtal mellan två juridiska parter. Hur ska ett sådant avtal vara utformat för att bädda för framgång? Kan man överhuvudtaget jobba lättroligt?

Författarna till det agila manifestet säger ju redan i inledningen att ”we have come to value [...] customer collaboration over contract negotiation”. Samarbete är viktigare än kontraktsförhandlingar. Låt oss fundera lite på vad manifestatörerna egentligen kan ha menat med det...?

...

Har du funderat klart? Nähä. För all del, låt det ta den tid det tar. Det är ju inte så självklart som man först tror.

...

Nu då? Vad har du kommit fram till? Själv tror jag att tankegångarna kan ha gått ungefär så här:

För det första: kontrakt är viktiga. Det är ju en överenskommelse som beskriver hur parterna ska samarbeta och vilket ersättning som ska utgå. Om man siktar på en långsiktig relation mellan kund och leverantör vill det till att du som kund tycker att resultatet är värdefullare än priset och att det för leverantören är mindre kostsamt att skapa resultatet än vad ersättningen är. Hängde ni med? På ren managementfloskel-svenska handlar det om att skapa win-win.

Hur gör man då i praktiken då för att skriva ett kontrakt? De flesta standardavtal, till exempel IT-företagens standardavtal IT-projekt, bygger på att man först skriver en specifikation över slutresultatet. Det är alltså bara för leverantörens säljare att utgå från specifikationen, be sina utvecklare att tala om hur lång tid det tar, multiplicera det antal timmar de svarar med en schablontaxa, lägga på en lämplig marginal, skriva en offert med fast pris, häfta på ett standardavtal och skicka till kunden, pruta lite på priset, pruta lite till och då saknas bara en signatur här, hehe... sådärja, då var den säljbonusen bärgad!

Det låter enkelt. Problemet är bara att projektet ofta är dödsdömt redan nu, redan innan bläcket har torkat. Risken är stor att du just har köpt en meter mjölk, någonting du inte vill ha. De flesta standardavtal, exempelvis IT-företagens IT-projekt, förutsätter nämligen att du som kund har specificerat exakt vad du vill ha när du vet minst om vad du behöver. Under utvecklingens gång, när du ser hur arbetet framskrider och hur omvärlden förändras, kommer du att lära dig massor, men då får du inte ändra dig med mindre än att varje ändring ska dras genom ett change-control-board, ofta med omförhandlingar av priset som följd.

Dessutom brukar man ju säga att djävulen gömmer sig i detaljerna; det är inte förrän utvecklarna sätter tänderna i problemet på allvar som de har en realistisk chans att uppskatta omfattningen av arbetet och inser att alla estimat exploderar. Shit pommes frites, snacka om lose-lose! Leverantören får jobba exkrementet ur sig för att leverera något som du som kund inte vill ha!

Ni ser väl paradoxen: alla viktiga, bindande beslut fattas innan vare sig du eller leverantören har hunnit lära er något av att försöka angripa problemet. Maximalt med konsekvenser baserade på minimalt med information. Båda parter tar en enorm risk i pengar, verksamhetsstörningar och förlorat anseende.

Fastprisprojekt med fryst specifikation är alltså inget framgångsrecept, men vad finns det för alternativ? Löpande räkning? Det skapar en flexibilitet för dig som kund att prioritera om och skära i funktionaliteten under projektets gång, men den flexibiliteten har ett högt pris: plötsligt flyttas all risk över till dig som kund; ju ineffektivare leverantören jobbar, desto mer pengar får hon. Andra kontraktsformer som försöker kontrollera och sprida risken är att dela upp arbetet i flera fastprisetapper med möjlighet till omförhandling mellan varje, målpriskonstruktioner, co-sourcing (där både kunden och leverantören arbetar aktivt i projektet) och vinstdelningsmodeller. Mary och Tom Poppendieck skriver mer utförligt om alla dessa kontraktsformer i sin bok *Lean Software Development: An Agile Toolkit* (som för övrigt finns till försäljning i vår nyöppnade webbshop, billigare än Bokus).

Gemensamt för dessa modeller är det öppna omfånget (Optional Scope Contracts). David Schmalz har i PNEHM! #2 2005 liknat projekt vid en konversation där parterna så tidigt som möjligt måste utforska vad kunden förväntar sig och vad leverantören kan åstadkomma för att reda ut vad som egentligen är realistiskt att hoppas på. Med ett annat ord kan det kallas förväntanshantering. Ett bra kontrakt måste stödja denna lärandeprocess, eller i alla fall inte motarbeta den. Faktum är att jag just upptäckte att Den Norske Dataforening i flera år har haft ett standardavtal för iterativ utveckling, PS2000, som skall vara utformat för att fånga upp lärandet under projektets gång. Det är definitivt värt en närmare undersökning.

Å andra sidan har jag varit inblandad i flera projekt som har gått utmärkt trots en kontraktsform som motverkar samarbete. För en tid sedan hjälpte jag en kund att upphandla ett system och anpassningar av Datarutin. Trots ett fastprisupplägg på anpassningsprojektet levererades det i tid med alla de funktioner vi visste och upptäckte att vi behövde. Det berodde till stor del på att samarbetet klaffade, inte minst med Datarutins utvecklingschef Jan Adolfson och tekniska projektledare Patrik Thölin. Det hände flera gånger att Datarutin utvecklade mer än vad som stod i specifikationen

när vi upptäckte att det saknades någon kritisk funktion. På samma sätt strök vi funktioner som de hade lovat att ta fram eftersom de visade sig vara väldigt svåra att lösa tekniskt och att de i slutändan inte var så viktiga för oss. När Patrik tyckte att våra önskemål på modifieringar blev för krävande i förhållande till värdet för oss sa han ifrån och då backade vi oftast (men inte alltid). Kort sagt, under hela projektets gång pågick en ständig omförhandling av vad som skulle göras.

Det är just detta ständiga förhandlingsspel kring krav och innehåll som är institutionaliserat i lättrorlig utveckling. Inom Extreme Programming kallas det för The Planning Game och i Scrum genomför man planeringsdagar i början av varje iteration där beställare och utvecklare förhandlar om vad nästa leverans ska innehålla. Ett avtal med öppet omfång stödjer detta arbetssätt.

Min erfarenhet från projektet med Datarutin visar ändå på det som jag tror att de agila manifestatorerna försöker säga angående customer collaboration; relationen och parternas goda vilja är viktigare än kontraktsformen. Om båda parter har inställningen att "om du är nöjd så är jag nöjd" och ingen försöker gynna sig själv på den andres bekostnad så har man alla förutsättningar att lyckas, även om kontraktsformen motarbetar samarbete.

Frågan: Hur skriver jag ett kontrakt för ett lättrorligt projekt? kanske är felformulerad. Frågan kanske snarare borde lyda: Hur kan jag lita på att min leverantör av utvecklingstjänster levererar mycket värde för pengarna?

### **Hur kan jag lita på att min leverantör av utvecklingstjänster levererar mycket värde för pengarna?**

Börja med att ställa följande frågor till potentiella leverantörer:

- Kan ni jobba iterativt? Kan ni snarast visa upp en grov utvecklingsplan och sedan producera en detaljplan i början av varje iteration?
- Vi finns tillgängliga för att detaljera kraven på daglig basis. Är era utvecklare vana att fråga snarare än att förutsätta och att ta ansvar för att dokumentera resultatet av kravdiskussionerna?
- Kan ni demonstrera körbar kod efter varje iteration?
- Kan ni efter varje iteration bevisa att den tekniska skulden i koden är låg? Kan ni visa att den blir godkänd av en heltäckande svit av automatiserade testfall? Får vi genom stickprov eller på annat sätt kontrollera att koden inte luktar refaktoring?

Glöm slutligen inte att kolla referenser; hur har leverantören skött sig mot tidigare kunder? En konkurrent till Datarutin med ett bra system försvann under upphandlingen eftersom de hade utnyttjat ett dåligt skrivet kontrakt i ett tidigare uppdrag vilket blev väldigt dyrt för den kunden. En sådan leverantör vill man inte jobba med.

Komplettera med ett kontrakt som ger båda parter incitament att anstränga sig till det yttersta för att samarbeta och lyckas.

Inspiration hittar du genom länkarna nedan. ■ ■ ■

#### LÄS MER OM KONTRAKT FÖR LÄTTRÖRLIG UTVECKLING

- Det agila manifestet: <http://www.agilemanifesto.org/>
- Beck, Cleal: Optional Scope Contracts:  
<http://www.xprogramming.com/ftp/Optional+scope+contracts.pdf>
- Vägledning till det norska standardavtalet PS2000:  
[http://dataforeningen.no/filestore/veiledning\\_ps2000.pdf](http://dataforeningen.no/filestore/veiledning_ps2000.pdf)
- Poppendieck, Tom & Mary: Lean Software Development: An Agile Toolkit, ISBN 0-321-15078-3
- Poppendieck, Tom & Mary: Agile Contracts Workshop:  
<http://www.poppendieck.com/agilecontracts.htm>



**Gustaf Brandberg är VD på Citerus AB och ansvarig för alla företagets förhandlingar och avtal. Kontakta honom på [gustaf.brandberg@citerus.se](mailto:gustaf.brandberg@citerus.se) om du vill veta hur Citerus är som leverantör av mjukvara, eller om du vill utbyta tankar om kontraktsfrågor i största allmänhet.**