Fakulteten för ekonomi, kommunikation och IT

Rapport

Distribuerad eXtreme Programming

Datum: 2006-05-01
Namn: Henrik Bäck
850611-6253
Kurs: CIT B01
Handledare: Tim Heyer

Innehållsförteckning

Värderingar

Kommunikation

Kommunikation skall enligt eXtreme Programming-modellen (hädanefter kallad XP-modellen) ske ofta och mycket. Ifall problem uppstår är det viktigt att kommunicera för att snabbt hitta en lösning på problemet. Kommunikation är också viktigt för att åstadkomma enkelhet i mjukvaran. XP-modellen säger inte något om hur kommunikationen skall ske dock att den måste ske hela tiden. Då utvecklingsteamen och kunder befinner sig på geografiskt skilda platser måste det finnas någon sorts system där information, frågor och svar snabbt kan publiceras så att alla gruppmedlemmar och kunder kan ta del av information omedelbart. Något man måste tänka på här är tidsskillnaden, mellan Sverige och Asien återfinns en tidsskillnad på mellan sju och åtta timmar. Detta bidrar starkt till hur kommunikation mellan grupperna kan ske. Om båda grupperna arbetar ordinarie arbetstider så kommer det bli så att det inte är några timmar som grupperna arbetar samtidigt. Det här bidrar till en fördröjning i kommunikationsprocessen och implicerar att frågor och svar grupperna emellan dröjer med svar.

Förutom detta är det fortfarande viktigt att alla som skriver programkod även dokumenterar denna. Detta är en viktig del av kommunikationen mellan de som utvecklar mjukvaran. Detta blir extra viktigt eftersom de två grupper som arbetar med mjukvaran befinner sig på geografiskt skilda platser. Dokumentationen bidrar starkt till kommunikationsprocessen.

Ett system för att lösa själva kommunikationsprocessen, som inte är programkodsbaserad, skulle enkelt kunna vara någon sorts e-post-listor. Bättre vore om någon sorts mjukvaru- eller webbaserad mötesplats användes. Dock gäller det att samtliga medlemmar kontinuerligt använder sig av ett sådant system för att vilja kommunicera med varandra i den mån det är möjligt.

Enkelhet

Enligt XP-modellen skall mjukvaran byggas för att enbart fylla dagens krav. Ingen hänsyn till vad som skall göras i morgon skall tas. Lösningarna som gjordes igår kan idag ses som mycket komplexa eller lika simpla som igår. Det viktigaste är att hitta en väg från den punkt där man nu befinner sig i och till målet. För att uppnå enkelhet i mjukvaran måste man än en gång kommunicera mycket med de andra i gruppen. Det är också viktigt att alla i gruppen har förståelse för det som utvecklas.

Teamet är delat på två geografiskt skilda platser, och som sagt i den föregående rubriken krävs det något enkelt och snabbt system att kommunicera genom. Dock finns fortfarande aspekten med tidsskillnaderna mellan grupperna kvar.

Det gäller också att alla i de båda grupperna veta vad de bygger för att uppnå enkelhet. Skulle ena gruppen få problem med något som den andra gruppen implementerat så kan det betyda att man måste vänta upp till en arbetsdag innan man kan få svar på de frågor som någon gruppmedlem vill ställa.

Återkoppling

Eftersom ingen plan för hur man skall ta sig till målet, håller speciellt länge gäller det att ha möjlighet att ändra denna med utvecklingens gång. För att över huvud taget kunna göra detta krävs feedback. Ett XP-team försöker generera så mycket feedback som de kan hantera, så snabbt som möjligt. Som tidigare nämnt finns det en viss tidsskillnad mellan de två grupperna, detta ger en viss fördröjning i feedbacken. Dock är inte denna tidsskillnad nödvändigtvis ett hinder för att följa XP-modellen. Det finns dessutom sexton (16) olika kunder som har åsikter och vill komma med förslag på förändringar. Dessa kunder finns dessutom enbart inom den ena tidzonen.

Som tidigare nämnt skulle även detta kunna hanteras via någon sorts webbaserad mötesplats. Där skulle samtliga deltagare i utvecklingen kunna publicera feedback som alla kan ta del av och hantera. Eftersom det finns många kunder som har åsikter och förslag skulle ett sådant system, med en webbaserad mötesplats, även gynna dessa. Kunderna skulle enkelt kunna komma i kontakt med projektets medarbeta. Det kan också vara till fördel att lämna feedbacken via en webbaserad mötesplats då det inte känns lika svårt för personen som lämnar feedbacken att vara helt ärlig. Eftersom XP-modellen inte säger något om hur feedbacken skall ske så behövs inte XP-modellen förändras, dock är själva scenariot inte exakt det samma som den ordinarie XP-modellen beskriver.

Principer

Mänsklighet

Det gäller för samliga medlemmar i utvecklingsteamet att kunna koncentrera sig på arbetet. Vad som påverkar dem utanför teamet kan inte kontrolleras eller påverkas. Dock är det viktigt att de kan känna ett tillhörande i gruppen och att de känner att de bidrar med något värdefullt. Det är viktigt att finna en balans mellan individualitet och deltagandet som en medlem i teamet. Det handlar också om att ge och få förtroende av sina kollegor.

Hälften av de man arbetar med befinner sig på andra sidan jordklotet och det finns en stor risk att man aldrig träffat dem i verkliga livet. Det kan vara svårt att känna samhörighet med dessa och dessutom ännu svårare att kunna deras förtroende. Att skapa dessa relationer med de som arbetar i samma grupp är i sig mycket svårt, det gäller att alla gör sitt bästa för att hela teamet skall få känslan av att de är sedda och respekterade. Dock behöver inte relationen till arbetare i den andra gruppen leda till ett stort problem. Eftersom man åtminstone kan känna samhörighet med de i samma grupp som en själv så behöver inte just den här principen leda till att XP-modellen inte skulle kunna tillämpas.

Gemensam fördel

Det är viktigt att en aktivet inom teamet ger en fördel för alla i den. En aktivet skall inte gynna någon medan den missgynnar någon annan. Det här är en av de svåraste principerna i ett team. Dessutom blir det ännu svårare när teamet är delat i två grupper. Dock är det väldigt viktigt att man i så stor utsträckning som möjligt tillämpar detta.

Det är extra viktigt att man får alla deltagare förstående i att vissa moment inte ger någon fördel just nu men förhoppningsvis vid ett senare tillfälle. En viktig sak som bidrar till en gemensam fördel är att dokumentera ordenligt i koden. Det ger ingen direkt fördel just när man gör det dock ger det en stor fördel för den som skall fortsätta att bygga på det som någon annan tidigare påbörjat, speciellt om det är någon ur den andra gruppen som skall göra det. Detta gör att den här principen blir extra viktig genom att den kan underlättar även då kommunikationen mellan de två grupperna.

Reflektion

Reflektion är otroligt viktigt för att utvecklas. Teamet bör reflektera över hur och varför de arbetar. Det är, oavsett ifall flera mindre grupper eller en stor grupp används, viktigt att lära sig av sina egna misstag men även av andras misstag. XP-modellen ger tid till reflektion inom varje cykel och reflektionen sker efter att ett moment slutförts. På detta sätt ser man till att teamet inte fastnar i någon sorts cykel där de bara tänker utan att göra något. Då två grupper som arbetar tillsammans används kan det vara önskvärt att dessa två grupper kan reflektera tillsammans. XP-modellen talar om att hela teamet skall reflektera vilket gör att i detta fall blir det en förändring.

Två grupper på olika geografiska platser, med två olika tidszoner skall reflektera samtidigt med varandra. En sådan reflektion skulle enkelt och kostnadseffektivt kunna ske via någon sorts videotelefoni eller liknande. Dock finns ett problem, det är en tidsskillnaden på nästan åtta timmar mellan dessa grupper. Detta gör att de inbokade reflektionerna måste göras på valda tider så att det passar båda grupperna. Att finna någon tid som passar båda grupperna kan vara en svår uppgift.

Förutom denna skillnad mot ordinarie modell är det också bra om team-medlemmarna reflekterar med andra än de som ingår i teamet. Här är det upp till var och en att lösa detta och det är inget som förändrar modellens utseende.

Flöde

XP-modellen säger att det är viktigt att ha ett kontinuerligt flöde av aktiviteter istället för diskreta faser. Gör små steg och gör dem ofta, flera gånger om dagen. På så sätt undviks problem med integreringen av mjukvaran. Det är också viktigt att mjukvaran fungerar flera gånger varje dag, inte enbart att den kompilerar och länkar. Flödet är viktigt för att kunna utveckla en mjukvara som fungerar och inte ställer till problem.

I det här scenariot, där två grupper som arbetar med samma mjukvara blir principen med flöde lika viktigt som i orginalmodellen, om inte än viktigare. Det måste kontinuerligt gå integrera sina ändringar i mjukvaran. Det är också viktigt att allt fungerar som det skall när den ena gruppen slutar arbeta, detta på grund av att den andra gruppen kommer att börja arbeta med den mjukvara där som den första gruppen slutade. Detta sker på grund av tidsskillnaden mellan grupperna. Om det skulle vara så att en grupp lämnade mjukvaran vid arbetsdagens slut då denna inte alls fungerade så skulle det göra att den andra gruppen skulle få åsidosätta sig ett

antal storyer för att först ordna upp detta. Detta skulle inte leda till något flöde ibland aktiviteterna hos denna grupp. Därför är det viktigt att båda grupperna har ett flöde av aktiveter som smidigt avlöser varandra och även ser till att mjukvaran fungerar flera gånger om dagen.

Ansvarsacceptans

Att ta ansvar för en uppgift är mycket viktigt. Det är mycket svårt att tilldela någon ansvar för något om denne inte accepterar detta ansvar. Därför är det viktigt att alla deltagare i teamet tar sitt arbeta på allvar och ansvarar för det de implementerar. De måste även, när de accepterat ansvaret, ansvara för design och testning.

Oavsett om XP-teamet består av en eller flera grupper kommer det alltid vara lika svårt att tilldela en person ett ansvar som denne inte accepterar. Eftersom alla utvecklingsgrupper, världen om, består av människor är detta den enkla anledning till att detta inte fungerar. Det är också lika viktigt som i orginalmodellen för XP att den som tar på sig ett ansvar även fullföljer detta med att deltaga i de moment som behöver göras. På grund av detta blir det ingen skillnad i den ordinarie XP-modellen och i jämförelse med dens om nu kommer att använda. En, två eller 100 grupper – det kommer alltid att vara svårt att tvinga en människa att ta ansvar för något.

Tillämpningar

Sitt tillsammans

XP-modellens tillämpning med ”Sitt tillsammans” talar om det viktiga i att mötas som människor i utvecklingen. Att använda sig av stora öppna ytor där hela teamet får plats att sitta. Det här är ett av detta projektets största utmaning. De två grupperna kan lätt, var och en för sig, sitta tillsammans och arbeta enligt XP-modellen. Det är dock högst önskvärt att även de två grupperna kan sitta och arbeta tillsammans. Detta kommer inte att vara möjligt på grund av det enorma geografiska avståndet mellan grupperna. Dock finns det andra lösningar på problemet och tillämpningen behöver inte reduceras. Det skulle enkelt, med dagens teknik, gå att ordna så att de båda grupperna kan ”sitta tillsammans” om inte annat virtuellt. Internet har ens stor potential och tillåter att båda grupperna kan, åtminstone för några timmar per dag, kan konferera över videotelefoni. I fallet sitter båda grupperna tillsammans var och en för sig och sedan sitter grupperna tillsammans med varandra. Detta skulle kunna vara ett steg på vägen mot att låta grupperna sitta tillsammans.

Dock finns fortfarande problemet med den stora tidsskillnaden mellan grupperna. Att implementera den här tillämpningen skulle kräva att båda grupperna fick justera sina arbetstider något. Detta skulle leda till något som skulle kunna tolkas som att den ena gruppen avlöser den andra. Till en viss del kanske detta kan vara önskvärt. Trots allt detta kommer åtminstone de båda grupperna att kunna sitta tillsammans var och en för sig, vilket förhoppningsvis tillför grupperna i sig något. Dessutom bidrar den webbaserade mötesplatsen till att teamet, om även virtuellt, att teamet sitter tillsammans.

Hela teamet

All expertis som behövs för att utföra kundernas beställningar skall finnas med i teamet enligt XP-modellen. Dock skall dessa personer tas med i teamet som en medlem, inte som någon som kan något. Eftersom teamet är delat i två grupper skulle detta betyda att det behövs två uppsättningar av vissa kunskaper för att möta båda gruppernas behov av kompetenta kollegor. Dock skulle detta kunna lösas genom att dela upp de storyer som erhållits från kunderna och koncentrera vissa kunskaper till en viss grupp. En annan aspekt på att ha full kompetens i båda grupperna är att det är dyrt att ha specialkompetens på två ställen.

En faktor återstår och den kan leda till en del problem. Teamet kan få problem med att känna samhörighet, alltså att det gör det här tillsammans. Det är antagligen inget problem för de två grupperna att känna samhörighet inom grupperna var för sig. Dock kan det här leda till att det blir ett scenario; ”vi och dem”. Ett sådant scenario är aldrig bra då det kan leda till att medlemmarna i de respektive grupperna börjar tycka att de är bättre än de i den andra gruppen. Ett sådant scenario kan aldrig leda till att lärdom dras av varandras misstag, att deltagarna kan stötta varandra eller de kan växa som ett team. Det finns många vardagliga exempel när sådana här situationer uppstår och här skull det finnas stor riska att det gör det även här.

Ett sätt att göra så att hela teamet känner sig delaktig är att låta den webbaserade mötesplatsen, där kommunikation sker, i första hand publicera meddelanden globalt. På så sätt kommer alla som deltar i projektet ha möjlighet att läsa och besvara vad någon publicerat. På så sätt får man igång en diskussion och alla kan känna sig delaktiga.

Informativ arbetsyta

Det är enligt XP-modellen mycket viktigt att ha en informativ arbetsyta. Det skall vara möjligt att med en snabb överblicka få reda på hur det går för projektet och var det står just nu. Detta brukar vanligtvis implementeras genom att använda sig av någon sorts stor tavla där man kan placera upp de storyer som finns i olika grupper. Detta sätt, med en tavla som arbetsyta, fungerar inte så bra då två grupper med stort geografiskt avstånd används. Det skulle inte vara praktiskt möjligt att synkronisera de två tavlorna med information och det skulle snabbt bli en mindre informativ arbetsyta.

Det skulle vara möjligt att lösa det här genom att använda sig av någon sorts mjukvara där det finns möjlighet att placera ut någon form av ”lappar” på en arbetsyta. Genom att använda en sådan mjukvara skulle det var mycket enklare att synkronisera de två arbetsytorna. Sedan visas denna bild upp på en storbildskärm, vägg-TV eller med hjälp av en projektor och en informativ arbetsyta, av liknande slag som den med tavlan, uppstår. Den här sortens arbetsyta har fördelen att den kan administreras från båda grupper och förändringarna får omedelbar genomslagskraft.

Genom användning en sådan här mjukvara gynnas även kunderna, då även dessa finns på flera geografiskt skilda platser. Det skulle vara möjligt att de, från sina lokala placeringar, kunde se denna arbetsyta via exempelvis World Wide Web. På detta sätt

kan de mycket enklare deltaga i vad som händer i en viss cykel men även se att projektet går framåt och de får valuta för pengarna.

Parprogrammering

Att använda sig av parprogrammering när man skriver mjukvara underlättar mycket. Det är enkelt att ta hjälp av någon annan när man kört fast eller när man vill förenkla någon idé. Det hjälper också till att hålla fokus på det som skall göras. Det är viktigt att båda personerna har en korrekt arbetsmiljö vid den dator där de två personerna sitter. Det skall vara bra med plats så att båda personerna får plats att sitta korrekt vid datorn.

Båda grupperna har ett jämt antal medlemmar vilket gör at det är enkelt att införa parprogrammering så länge ingen är frånvarande från arbetet. Det går att skapa par och det går enkelt att rotera paren. Hur ofta paren bör rotera är något som gruppen får känna sig fram för att hitta den tid som passar dem bäst. Det finns inga skillnader på hur tillämpningen med parprogrammering förändras på grund av att två grupper används. Detta eftersom att oavsett om de vore en eller två grupper skulle parprogrammering kunna användas enligt det som XP-modellen föreskriver.

Storyer[1]

De kunder ett projekt har skriver det de vill ha implementerat i mjukvaran på så kallade storykort. Därefter estimeras tiden det tar att implementera korten. Det gäller att så tidigt som möjligt bestämma det som skall finnas på korten inför nästa cykel. Kunderna är i det här fallet sexton (16) stycken och finns på olika ställen i landet. Eftersom kunderna skall skriva och väja ut de storyer som de tycker är viktiga och dessutom gemensamt ta beslut om vad som är viktigt och mindre viktigt måste även dessa kommunicera. Det finns fler mer eller mindre lönsamma sätt att lösa det här på.

Hur urvalet av storyer kommer att gå till kommer att belysas under rubriken ”Veckovisa cykler” nedan, dock återstår problemet med de att två distribuerade grupperna måste ha tillgång till sina storyer och även till storyer som är värda att ta med ifall tid blir över. Detta skulle kunna lösas med hjälp av en tidigare lösning i denna rapport. Som tidigare föreslagits skulle den informativa arbetsyta bestå av en vägg-TV eller en projektion av de aktiveter som föreligger. I samma mjukvara som hanterar denna projektion skulle det vara möjligt att implementera en yta där man kan se de aktuella storyer som ligger i denna cykel. På så sätt får man all information samlad på ett ställe samt presenterad på ett för ändamålet bra sätt.

Problemet med att lägga alla storyer direkt i en mjukvara kan vara att de inte riktigt tas på allvar eller att de glöms bort (främst av kunder). Att de inte tas på allvar är ett problem som ligger hos utvecklarna och det kan inte accepteras. Skulle det vara nödvändigt, på grund av att storyer är virtuella, skulle man kunna implementera en funktion i arbetsytan för utskrift av storykort. På detta sätt överför man virtuella

[1] Story [stå´ry] –er; ~berättelse
SAOL Trettonde Upplagan

storykort till reella objekt som förhoppningsvis, för de som behöver, bidrar till att acceptansen för korten ökar.

Veckovisa cykler

Veckovisa cykler är mycket bra för de som arbetar i projektet. På så sätt finner man att varje vecka är en veckas storyer implementerade och utvecklarna kan gå hem och återhämta sig över helgen, vilket är mycket viktigt för dem.

Dock har projektet inte enbart mer än en utvecklingsgrupp utan de har även flera kunder knutna till samma projekt. Detta innebär ett visst problem vid hantering av storyer kommer att förekomma. Det är önskvärt att samtliga kunder närvarar vid urvalet av de storyer som skall implementeras den kommande veckan, dock kan detta vara ett problem med tanke på hur kunderna är fördelade över landet. Att försöka se till att varje kund deltar vid varje veckovis cykel skulle varken vara kostnadseffektivt eller särskilt eftertraktat. Man skulle få kunder som infinner sig på plats men inte ”har någon lust” att arbeta, det skulle också kosta en del pengar att transportera personer från hela landet till ett ställe.

Om man istället för att välja ut storyer varje vecka gjorde detta varje månad skulle det var möjligt att låta kunderna en gång per månad bestämma vad som skall implementeras. På detta sätt får man ner kostanden för resor för de alla kunder som skall resa från landets alla hörn. Det skulle också vara möjligt att göra samma sak var annan månad om det visade sig att varje månad är för mycket. Det viktigaste är att det inför varje månad/var annan månad finns bestämt vilka storyer som skall implementeras från vecka till vecka. Skulle ändringar behöva göras skulle detta senare kunna göras via den webbaserade mötesplatsen.

I de fall någon programmerar har slut på storyer att implementera kan fler hämtas från en kö/stack. Mjukvaran som tillhandahåller alla storyer tilldelar då en programmerare ett nytt uppdrag och denne får på så sätt inte välja vad denne vill göra. Det här är bra på många sätt, dels sätter det en programmerare på prov och denne får utvecklas som programmera dessutom eliminerar det konflikter som beror på vilken sorts storyer en viss programmerare valt.

Slöhet

Slöhet inom teamet är aldrig bra. Det kan bidra till att projektet inte levererar det man lovat och att det då uppstår en mycket dålig stämning mellan kund och företag. Det är viktigt att se till att de som arbetar gör det de ska och kommer i tid till arbetet. Skulle fallet inte vara så är det svårt att estimera hur långt det tar att implementera en story. En annan orsak till att någon form av slöhet uppstår hos utvecklarna kan vara stress, stress kan i vissa fall leda till att människor inte orkar arbeta i sin normala takt. Dock måste det finnas, när det kärvar ihop sig, möjlighet att arbeta snabbare och mer effektivt för att klara av att nå målet.

Oavsett om man använder sig av en eller flera grupper så är det viktigt att alla har samma arbetsmoral och inställning till det de gör. Dessutom är det viktigt att ledningen uttrycker för utvecklarna att det är tillåtet att säga ifrån när det blir för

mycket att göra. Detta eftersom, om de skulle ha mer att göra än de klarar av, i sin tur leder till att mindre av de storyer som är utlovade kan implementeras. Det är också viktigt att de två gruppernas medlemmar är punktliga vad det gäller tider, att komma i tid till arbetet och att inte sluta innan arbetsdagen är slut. Om detta inte efterföljs kommer mycket tid tas från de storyer som lovats att implementeras. Man bör känna av var varje grupp ligger angående hur mycket storyer de kan implementera varje cykel. Det som fungerar i ena gruppen behöver inte fungera i andra gruppen och kan leda till att gruppens utvecklare blir stressade.

Kontinuerlig integrering

Det är viktigt, för hela gruppen och för den som programmerar att man ägnar sig åt kontinuerlig integrering. Genom att integrera varje story och låta integreringen ske asynkront kan utvecklarna rätta till de fel som eventuellt uppstår direkt istället för att göra det senare. Genom att användandet av automatiserade tester kommer även den denna del att utföras varje gång. Under tiden programmeringsparet väntar på att allt detta ska utföras får man tid till den viktiga reflektionen. Skulle inte asynkron integrering användas skulle man vara tvungen att göra upp fasta tider då kompilering och testning skulle ske. Det skulle innebära att den ena gruppen inte arbetade när felen uppstår, detta på grund av tidsskillnaden mellan grupperna. För att rätta dessa fel får den andra gruppen antingen vänta till dagen där på eller rätta felen själva. Detta skulle vara mycket mindre effektivt och invecklat än att kompilera och testa vid varje gång ett programmeringspar laddat upp sin nya version av programkoden.

Om detta skall vara möjligt måste båda utvecklingsgrupperna arbeta mot samma integreringsserver. Detta borde vara möjligt att göra genom att låta all datatrafik mot servern gå vi någon WAN-länk. På så sätt får båda grupperna tillgång till den senaste integrerade mjukvara hela tiden. Det finns några fler problem med detta, så som administration av dessa servrar, som inte hör till denna rapport.