Handelshögskolan
VID GÖTEBORGS UNIVERSITET
Institutionen för informatik
2004-01-14

LÄRANDE I EXTREME PROGRAMMING

Abstrakt

Ett antal enklare och mer flexibla utvecklingsmetoder har fått stor uppmärksamhet under senare år. En av dessa, Extreme Programming (XP), har tillämpats både inom akademiska och icke-akademiska miljöer. I denna uppsats undersöks var lärande och lärandeprocessen har förekommit i dessa tillämpningar i syfte att skapa förutsättningar för förbättringar av XP som metod för både lärande och systemutveckling. För denna litteraturstudie har en strukturerad metod använts med utgångspunkt i artiklar från två ledande vetenskapliga konferenser. Resultatet visar både skillnader och likheter i hur lärande uppstår i akademiska och icke-akademiska tillämpningar av XP. I akademiska tillämpningar uppstår lärande i aktiviteter som stöttar kommunikation mellan studenter, men också aktiviteter som tydligt minskar komplexitet och hjälper studenter att bättre förstå programmering och systemutveckling. Inom icke-akademiska tillämpningar är komplexitetsminskning och kommunikation också viktigt, men framför allt i kundnära aktiviteter. Sammanfattningsvis rekommenderas att XP utvecklas och att lärande ytterligare betonas i metoden för att förbättra lärande i fler och mer varierade tillämpningar.

Nyckelord: Extreme Programming, Lärande, Lärandeprocess

Författare: Martin Elvheim
Handledare: Helena Holmström
Examensarbete II, 10 poäng

Innehållsförteckning

Förord

I denna litteraturstudie har jag en koppling mellan lärande och Extreme Programming som kanske inte är helt självklar. I mitt arbete som lärare har jag experimenterat en hel del med pedagogiska former och med olika sätt att genomföra kurser inom systemutveckling och programmering. Ett upplägg som har varit intressant att utveckla är grupparbeten och deras betydelse för studenters lärande. Genom dessa experiment och studier så har även olika metoder inom systemutveckling varit förmål för intresse. Det är här Extreme Programming har kommit i fokus. Extreme Programming lämpar sig för programmering och systemutveckling i mindre team och med intensiva sociala kontakter mellan gruppmedlemmarna. Detta har tilltalat mig som en pedagogisk idé och jag har med detta perspektiv även genomfört en del experiment med Extreme Programming. Denna studie och uppsats är en fortsättning på detta arbete i syfte att för egen del undersöka vad Extreme Programming har betytt för individers lärande. Vad fortsättningen efter detta arbete blir har jag i skrivande stund ingen uppfattning om, annat än att det har givit mig ny inspiration och nya uppslag.

Uddevalla, januari 2004,

Martin Elvheim

1 Introduktion

Synen på systemutveckling har under senare år genomgått en rad förändringar. Dahlbom (1997) beskriver förändringarna utifrån hur fokus har förflyttats från att ha varit inriktat på maskiner med huvudsaklig uppgift att utföra beräkningar till större informationssystem med administrativa arbetsuppgifter och på senare tid till att tillhandahålla tjänster för spridning och delning av information. Den tidigare synen på datorer som maskiner för beräkningar, kallat ADB, var allmänt rådande även när det gällde utvecklingen av system och de bakomliggande utvecklingsprocesserna. Tidigare följde utvecklingen av informationssystem ofta vattenfallsmodellen eller livscykelmodellen, som innebär systemutveckling i ett sekventiellt förlopp utan iterationer (Andersen, 1994). Genom den förändrade synen på datorer blev det även viktigt att även förändra synen på systemutveckling. Under framförallt 1990-talet var detta en viktig fråga för utvecklare och forskare. Dessa förändringar av metoder och modeller har resulterat i nya idéer och tankar om kvalitetssäkring av utvecklingsprocessen. Några förändringar har varit att tillföra ett iterativt förlopp, en inkrementell utveckling samt möjligheter att föra tillbaka erfarenheter till tidigare skeden i utvecklingsprocessen (Mathiassen et al., 2001). Dahlbom (1997) menar också att förhållandet mellan maskiner, system och människa har förändrats och förändringarna berör således inte enbart huvuduppgiften för system och mjukvaror. Det har skett en förskjutning emot människor, användare och intressenter. Den nya synen på system som tjänsteleverantörer har ställt högre krav på systemens kapacitet, funktionalitet och omfattning. Dessa ökade krav har också inneburit ökade krav på hur utvecklingen skall bedrivas. Synen på utveckling och förbättring av utvecklingsmodeller har lett till att nya utvecklingsverktyg och verktygsrelaterade koncept har lanserats med skiftande framgång. Exempel på sådana koncept är Rational Unified Process (RUP) och Microsoft Solution Framework. Centralt i dessa modeller är i många fall kvalitetssäkring genom processer, praxis och dokumentationsstandarder. De nyare metoder för systemutveckling som har växt fram har också resulterat i ett behov av att förbättra utvecklingsprocesserna och inte minst att kunna definiera hur väl utvecklingsprocessen fungerar för att kunna mäta den i en utvecklingsorganisation. Ett exempel är Capability Maturity Model - CMM som har fått relativt stor betydelse för utveckling av själva processen (Sommerville, 2001).

En grupp av utvecklingsmetoder samlas numera under *Agile Methodologies*[1] och skulle förenklat kunna sägas innebära att metoder och processer raffinerats för att vara enklare, klara snabbare förändringar och underlätta för förändringar i utvecklingsmiljön (Newkirk, 2002). En av dom metoder som har fått stort genomslag är *Extreme Programming* (XP) (Beck, 1999). XP sätter fokus på programmerarna som grupp, i par och enskilt som drivande för att förbättra kvalitén på den mjukvara som utvecklas. Beck (1999) menar att en förenklad metod med lättimplementerade principer och praxis borgar för väl fungerande utvecklingsteam, därigenom bättre programvara och slutligen nöjdare kunder.

Även inom den akademiska världen förekommer XP allt oftare i grundutbildning och forskning (Kivi et al., 2000; Müller & Tichy, 2001; Keenan, 2002). Den ökande andelen

[1] Agile Methodologies är det allmänt accepterade samlingsnamnet för metoder och processer inom mjukvaruutveckling med huvudfokus på människor som är involverade i utvecklingsteamet. Agile betyder i detta sammanhang lättvikt, flexibilitet och anpassningsbarhet. Mer information om begreppet finns på http://www.martinfowler.com/articles/newMethodology.html

försök och tester inom industrin och den akademiska världen har också lett till att det blivit intressant att närmare studera vad som är intressant bakom fenomenet XP.

De metoder som återfinns inom Agile Methodologies beskrivs av Abrahamsson et al. (2002) som lättrörliga (eng. agile), modulariserade, iterativa, med korta tidscykler, fokuserade, adaptiva, inkrementella, riskminimerande, kollaborativa, kommunikativa och centrerade kring människor. De sista egenskaperna beträffande kollaboration, kommunikation och människor öppnar för intressanta sociala aspekter i samspelet mellan människor. Dessa egenskaper återfinns också i XP. Beck (2000) beskriver XP som centrerat och uppbyggt kring utvecklingsteamet och dess gemensamma målsättningar. I XP får individen en nedtonad roll och istället ägs hela programvaruutvecklingen demokratiskt av utvecklingsteamet. Försök med XP har visat att arbete tillsammans påverkar resultatet av ett projekt i en positiv riktning. (Dick och Zarnett 2002; Sharifabdi och Grot, 2002)

Lärande har visat sig vara intressant utifrån XPs sociala dimensioner. Elvheim (2003) visar att användningen av XP i exempelvis programmeringskurser kan stötta lärande genom interaktion och samverkan. Beck (2000) menar att metoden inte skall tala om hur utveckling skall utföras utan istället lära medarbetarna att lära sig hur man utvecklar god praxis. Han betonar att utbytet mellan medarbetare skall vara levande och några av principerna i XP som exempelvis parprogrammering är väldigt tydligt i utbytet mellan människor. XP som testas i många olika sammanhang har givetvis även betonat lärandeaspekter. Kerievsky (2001) beskriver vikten av lärande, inte minst kontinuerligt lärande, i XP-projekt. Kerievskys beskrivning visar emellertid problematiken bakom denna litteraturstudie. Trots att XP betonar sociala aspekter såsom lärande så saknas kopplingen till de sociala aspekterna på ett definierat, vetenskapligt och strukturerat sätt. Kerievsky (2001) nöjer sig i sin beskrivning av lärande i XP med att summariskt referera till den lärande organisationen. Visserligen befinner sig en stor del av de tester som görs inom området för *datavetenskap* genom att sociala och mänskliga aspekter på XP saknas. Således blir det intressant att närma sig XP med *informatik* som vetenskap och ha en syn på IT i ett socialt sammanhang.

Denna litteraturstudie syftar till att identifiera lärande i beskrivningar av XP. Vidare är tanken att med studien kunna koppla teorier om lärande och lärandeprocessen till dessa beskrivningar av lärande i XP för att kunna resonera och dra slutsatser kring hur lärande skall kunna tydliggöras och öka betydelsen av lärande inom användandet av XP. Detta i sin tur kan göra att XP förbättras som metod och att användningen av XP i utbildningssammanhang kan ge ett ökat lärande.

Frågan i denna studie är:

- I vilka aktiviteter uppstår lärande och lärandeprocesser i tillämpningar av Extreme Programming?

I denna fråga avgränsas lärande till att omfatta individuellt lärande och individuella lärandeprocesser.

2 Forskningssyn och metodval

2.1 Inledning

Järvinen (2001) beskriver *Information Systems* (IS) som ett område där informationsteknik studeras med olika infallsvinklar av datavetenskap och ytterligare en referensvetenskap för att stötta den egna forskningen. Ett likartat resonemang har Baskerville och Myers (2002) beträffande utvecklingen av IS som vetenskapsområde. De menar att IS relaterar sin vetenskap nära till t ex systemvetenskap, datavetenskap, psykologi och sociologi. I denna uppsats finns närvaron av ett referensområde genom lärande och lärandeprocesser. Detta område hämtar sin teori från psykologi och pedagogik för att beskriva och förklara fenomen. Här beskrivs *erfarenhetsbaserat lärande* som en process med begrepp, egenskaper och indikatorer för att identifiera lärandefenomenet i litteraturstudiens källor.

2.2 Forskningssyn

Valet av metod för att studera ett fenomen kan göras utifrån olika ställningstaganden. Ett sätt kan vara att skaffa sig en välgrundad forskningssyn. Den forskningssyn man väljer att ha kan då baseras på traditioner inom det egna ämnesområdet, synen på omvärlden och grunder för värdering av trovärdighet inom den egna forskningsgruppen.

Utgångspunkten för att undersöka lärandeprocesser är att individer på egen hand kan bedöma, värdera och skatta sina prestationer i relation till sin egen utgångspunkt. Easterby-Smith et al. (2002) beskriver detta som *social konstruktivism*. Inom social konstruktivism är det synen på verkligheten som en mänsklig konstruktion baserad på människors uppfattningar som är gemensam syn. För att få rätt förutsättningar skulle en situation, t ex ett avgränsat projekt vara lämpligt. Förutsättningarna för den inramade situationen är då en avgränsning i tid, omfattning och problematik. Då kan forskningsproblemet angripas genom att med hjälp av olika påståenden låta respondenterna skatta huruvida deras eget perspektiv, situation och bedömning överensstämmer med det påstådda. Eftersom människors uppfattningar av omvärlden är det som formar den, innebär studier av dessa uppfattningar att omvärlden betraktas (Easterby-Smith et al., 2002). Denna världssyn, att det går att dra slutsatser utifrån dessa bedömningar, brukar också hänföras till *hermeneutiken* (Wallén, 1996). Tolkning av innebörder är viktigt även i detta fall. Genom att låta tolkningen av både delar av de undersökta fenomenen och tolkningen av helheten få ett utrymme förstärks det som betraktas som ett hermeneutiskt forskningsperspektiv. Vidare arbetar hermeneutikforskare ofta med att i förväg ha en förståelse för det undersökta fenomenet och att kunna veta vad som skall betonas när det gäller att formulera problem och ansatser att undersöka. I denna studie skulle det ha en stark förankring genom tidigare upplevelser, arbete med och fokusering kring de undersökta aktiviteterna. Det finns således en tydlig koppling till den antagna förståelsen av de olika problemen. Vidare menar Wallén (1996) också att det är väsentligt att sätta tolkningarna i sitt sammanhang, vilket kan göras genom att ge problemen en tydlig ram och avgränsning.

Problemsynen är också väsentlig att belysa. Min tanke är att föra ett deskriptivt angrepp på det problem som formulerats. Wallén (1996) menar att denna ansats tydligt knyter an till

hermeneutikens tradition och arbetssätt. För att ytterligare understyrka detta krävs att teori tillförs som kan stödja genomförande, förklara och diskutera begrepp och tolkningar av resultaten.

I detta arbete är lärande och lärandeprocesser inom XP i fokus. Således påverkar synen på lärande och lärandeprocesser även forskningssynen. Stensmo (1994) presenterar två traditioner som bygger på olika epistemologier. Den ena med en syn på kunskap som något som skall efterlikna eller återge fenomen i verkligheten och den andra som bygger på att kunskap är resultatet av en omvandlingsprocess. Den senare av dessa betyder också att individen själv äger sin lärandeprocess och att den av utomstående enbart kan stimuleras på olika sätt. Vidare beskriver Stensmo (1994) att lärande i arbetslivet har en tydlig plats i epistemologiska sammanhang. I många fall skiljer man mellan kunskap som härrör från teori och kunskap som härrör från praktik och tillämpningar. Yrkeskunskap benämns exempelvis med termen ”tyst kunskap” eller förtrogenhetskunskap. Stensmo (1994) fortsätter med att presentera ett antal filosofers pedagogiska syn. Han skiljer utifrån ett epistemologiskt perspektiv på rationalism, empirism och konstruktivism. Den konstruktivistiska synen på epistemologin menar att kunskapsbildningen är en process som baseras på erfarenheter. Stensmo refererar Piaget som menar att kunskapskonstruktionen är en aktiv och kontinuerlig process som aldrig upphör. Stensmo beskriver även Lockes och Deweys syn på lärande i form av en process med gemensamma egenskaper som kontinuitet och en konstruktivistisk syn på att kunskaper uppstår ur lärande.

Den genomgående synen på forskning, lärandeprocessen och kunskap är i detta arbete konstruktivistisk. Detta förhållningssätt har valts med hänsyn till att göra metodval, analysmetod och övergripande perspektiv med en helhetssyn. Denna studie grundar sig i en litteraturstudie vars huvudkällor har sin empiri både i akademiska och icke-akademiska studier. Dessa källor har en konstruktivistisk syn på erfarenhet, individers egen förmåga till reflektioner och bedömningar och har fokus på ett socialt sammanhang.

2.3 Metod

Utgångspunkten i denna litteraturstudie är att beskriva ett fenomen utifrån vad som existerar i befintliga rapporter och litteratur. Det huvudsakliga förhållningssättet här är således deskriptivt snarare än normativt. Järvinen (2001) menar att detta förhållningssätt är ett bra val för forskning med konceptuell analys. Vidare menar han att den konceptuella analysen faller inom forskning på fenomen i verkligheten. Denna litteraturstudie använder konceptuell analys för att betrakta resultat och för att finna utgångspunkter för slutsatser och generalisering. Webster och Watson (2002) beskriver ett systematiskt sätt att arbeta med litteraturstudier. Deras förslag på uppbyggnad är följande:

1. Litteratursökning
    1.1. Sök huvudkällor för litteraturstudien
    1.2. Gå bakåt med källorna från 1.1
    1.3. Gå framåt med källorna från 1.1

2. Analys av litteratur

I följande avsnitt beskrivs hur de olika delarna har genomförts och vilken analys som har använts i litteraturstudien.

2.4 Litteratursökning

Första steget i denna litteraturstudie var att finna huvudkällor. Eftersom XP fortfarande är relativt nytt så begränsas antalet källor och inte minst antalet forum där källorna står att finna. Webster och Watson (2002) menar att de viktigaste källorna för en litteraturstudie finns i betydelsefulla tidskrifter. De hävdar vidare att *information systems* är ett multidisciplinärt område där flera vetenskaper ofta möts. I en litteraturstudie är det därför viktigt att även söka i källor som inte enbart är dedikerade till det egna vetenskapsområdet. I detta fall är XP synnerligen svårt att finna utanför *information systems* och *computer science*. Här har därför vidare alternativa sökningar inte genomförts. För att finna en bra utgångspunkt för denna studie har två parallella utgångspunkter använts, nämligen:

- Dedikerade konferenser
- Publicerad litteratur

Med dedikerade konferenser avses här vetenskapliga konferenser som i huvudsak berör XP. De konferenser som valdes som utgångspunkt för litteraturstudien är helt dedikerade till XP och Agile Methodologies, och har således en hög andel artiklar som berör området. De två konferenserna är:

- XP
- XP Universe

XP-konferensen är europeisk och startade 2000. Den har arrangerats fyra gånger och huvudarrangörerna har varit akademiska organisationer såsom Università di Cagliari, Free University of Bolzano – Bozen och Università di Genova. Programkommittén har allt sedan starten bestått av flera av upphovsmännen bakom XP. Första året hade konferensen 161 deltagare från 19 länder och sedan dess har antalet deltagare och omfattningen på konferensen ökat[1].

XP Universe-konferensen är amerikansk och startade 2001. Då hade konferensen 238 deltagare i huvudsak från USA. Konferensen arrangeras av ObjectMentor som är ett kommersiellt företag och således är inriktningen tydligare mot denna typ av intressenter. Programkommittén består liksom för XP-konferensen av framstående personer inom XP-området[2].

Ur alla elektroniskt publicerade artiklar på dessa två konferenser under åren 2001 och 2002 söktes efter begreppen *learning*[3] och *education*. Antalet artiklar i sökningen presenteras i tabell 1.

[1] Information om XP-konferensen är tillgänglig via senaste konferensens webbplats: http://www.xp2003.org/
[2] Information om XP Universe är tillgängligt via konferensens webbplats: http://www.xpuniverse.com/.
[3] Sökningen gjordes på alla begrepp som inleddes med learn* respektive educat* vilket även inkluderar böjningar av orden.

Konferens	Antal artiklar med ”learning” och/eller ”education”	Totalt antal
XP 2001	14	37
XP 2002	18	31
XP Universe 2001	20	30
XP Universe 2002	9	23

Tabell 1 – Artikelsökning från XP 2001-2002 respektive XP Universe 2001-2002

Webster och Watson (2002) föreslår att man söker bakåt med sina källor som utgångspunkt. Utifrån urvalet av artiklar från konferenserna söktes i referenslistorna efter ytterligare källor. I referenslistorna återfanns ytterligare elva artiklar och en doktorsavhandling som tillfördes litteraturstudien. (Se tabell 2)

Vidare menar Webster och Watson (2002) att man skall söka framåt och finna de källor som refererat till de man har i sin ursprungliga utsökning. För att göra detta föreslår de ISI Web of Science[4] där en källa kan sökas framåt för att finna var den är refererad. I denna studie genomfördes denna sökning för samtliga ingående källor och ytterligare tre artiklar återfanns för litteraturstudien[5]. (Se tabell 2)

För att ytterligare bygga vidare på litteraturstudien gjordes också sökningar efter bokkällor som återfanns i referenslistorna från artiklarna. De källor som bedömdes vara relevanta för Extreme Programming, lärande (learning) och/eller utbildning (education) införlivades även dessa i studien. (Se tabell 2)

Källtyp	Antal
Artiklar	11+3
Bokkällor	8
Doktorsavhandlingar	1

Tabell 2 – Sammanställning över ytterligare källor från referenssökningar[6]

2.5 Litteraturanalys

Järvinen (2001) menar att koncept, eller mer konkret termer, är centrala i all forskning. Således är analys av koncept också viktigt i analys av ett område eller fenomen. Han menar att vi med hjälp av koncept kan klassificera och gruppera det vi finner i vår forskning. Klassificering på detta sätt måste ske med några viktiga egenskaper som utgångspunkt. Dessa egenskaper är:

- Varaktighet
- Uttömmande
- Ömsesidigt uteslutande

[4] Web of Science är numera en del av ISI Web of Knowledge som återfinns på http://isiknowledge.com
[5] Dessa artiklar var emellertid inte åtkomliga vid tidpunkten för litteraturstudien p g a publicering i kommersiella databaser.
[6] Från dessa källor har dragits de som redan ingår i studien, men som varit refererade i källorna.

- Naturliga

Med varaktighet avser Järvinen att klassificeringen skall behållas oförändrad genom undersökningen och inte påverkas av våra resultat eller insikter. De skall täcka in alla tänkbara delar av vårt undersökta fenomen och inte lämna väsentliga delar utanför. De skall inte överlappa varandra utan göra det möjligt att verkligen tillhöra en specifik klass. Vidare menar han att de valda klasserna skall vara ett naturligt val beroende på ämne, fenomen eller område snarare än en konstruerad gruppering. Även Webster och Watson (2002) föreslår konceptuell analys. Inom respektive koncept menar de att man kan kategorisera efter mindre enheter som kan användas för analysen. Resultatet av en analys på detta sätt är en matris som placerar källorna i olika grupperingar. (Se Tabell 3)

Artikel	Koncept					
	A			B		
Analysenhet	i	ii	iii	j	jj	jjj
I						
II						
III						

Tabell 3 – Exempel på analystabell (Webster & Watson, 2002)

I denna studie klassificerades artiklarna utifrån akademisk eller icke-akademisk tillämpningsmiljö. Med akademisk tillämpningsmiljö avses här om den studerade artikeln har sin grund i en tillämpning av XP i utbildningssammanhang eller på något sätt har sitt ursprung i en akademisk miljö. De som bedömdes som icke-akademiska var alla som genomfördes i en miljö med verkliga kunder och oftast med ett bakomliggande kommersiellt syfte. För respektive artikel gjordes en datainsamling av hur, i vilket sammanhang och på vilket sätt presentation av lärande/lärandeprocess och lärandekontext gjordes i artikeln. Dessa delar benämndes och delades in i *presentation* respektive *kontext* i datainsamlingen.

Med dessa koncept som grund blev analysmatrisen som visas i tabell 4.

Artikel	Koncept			
	Tillämpningsmiljö		Lärande/Lärandeprocess	
Analysenhet	Icke-akademisk	Akademisk	Presentation	Kontext
I				
II				
III				

Tabell 4 – Analysmatris

Den akademiska tillämpningen var mindre intressant i bokkällorna då de inte baserade sig på en huvudtillämpning. Alltså gjordes inte samma klassificering. Istället insamlades data om lärande/lärandeprocess på samma sätt som för artiklarna.

Artiklar och bokkällor analyserades därefter med en kvalitativ ansats. Järvinen (2002) beskriver analys av dokument utifrån hur innehållet kan tolkas och värderas. Han beskriver detta utifrån exegetisk analys av innehållet i en text. De sex tekniker för exegetik som han presenterar är:

- Textanalys
- Lingvistisk analys
- Litterär analys
- Historisk analys
- Formanalys
- Reduktionsanalys

Den analys som användes här var textanalys och i viss mån formanalys, d v s den text som beskriver fenomenet och det sammanhang som kan utläsas ur helheten. Således bygger analysen på de formuleringar där lärandet presenteras och vilken betydelse detta har i den källa som undersöks. Textanalysen och formanalysen gjordes utifrån de klassificeringar som gjordes i analysmatrisen (Tabell 2).

2.6 Metoddiskussion

Backman (1998) diskuterar metoder utifrån ett kvalitativt och kvantitativt synsätt. Kvantitativa metoder är de som använder mätningar och sammanställningar som bygger på matematiska och statistiska principer. Kvalitativa metoder å andra sidan är sådana att de bygger på verbala beskrivningar. Han betonar att ingen av metoderna behöver innebära att ett visst forskningsperspektiv skall användas. Således skulle i detta fall både en kvalitativ och kvantitativ metod kunna vara tänkbar. Easterby-Smith et al. (2002) anför en storleksparameter i forskningsdesignen. Valet står emellan att välja en avgränsad förekomst av fenomenet eller i ett större antal sammanhang söka förekomsten. I detta fall kan kvantitativa metoder användas för att studera fenomenet i fler sammanhang än om en kvalitativ metod väljs. Valet mellan de båda metoderna kan falla tillbaka på hur omfattande i tid och arbete respektive metod är. Att exempelvis välja ett mindre omfattande urval av källor kan vara en snabb och resurseffektiv metod för datainsamling i jämförelse med en studie av samtliga påträffade källor.

I denna litteraturstudie valdes att göra en mindre omfattande sökning och utifrån den göra en kvalitativ studie. Detta val påverkar studiens två huvudsakliga kritiska moment: sökning av källor och analys av materialet. Beträffande sökning av källor inverkar det främst på tillförlitligheten i studien. Osäkerheten ligger mycket i om de källor som finns med i studien kan sägas vara giltiga för det som skall uppnås. De två konferenser som har valts som utgångspunkt för studien har ganska kort historik, de har förhållandevis liten ämnesomfattning och de har en tydlig inriktning på att vara positivt inställda till sin inriktning. Detta är förvisso faktorer som inverkar på innehållet i de källor som kommer från dessa konferenser. Å andra sidan är dessa konferenser ledande i den bemärkelse att de har representation av de flesta upphovsmännen bakom metoden som är i fokus i undersökningen, de har en spridning mellan både akademiska och icke-akademiska aktörer och de har olika arrangörsstrukturer. Samtliga dessa faktorer talar för att det skall finnas möjlighet till vida perspektiv på de fenomen som är av intresse. Ytterligare något som ökar möjligheten till många perspektiv är att koncentrationen av bidrag högt. Således är det på dessa konferenser

goda möjligheter att finna en stor andel av de bidrag som faktiskt presenterar det eftersökta fenomenet.

Analysen av materialet påverkar förvisso tillförlitligheten, men i större utsträckning giltigheten i denna studie. Webster och Watson (2002) och Järvinen (2001) föreslår en mer omfattande analys med hjälp av koncept och analysenheter inom varje koncept. Detta kräver emellertid en stor kännedom om området och det förväntade resultatet i undersökningen. Denna kunskap om resultatet finns inte i förväg i denna litteraturstudie. Det alternativa sätt som har tillämpats här är att göra en kvalitativ analys av innehållet i källorna och i efterhand göra klassificeringar av det material som framkommer. I detta avseende ökar därmed giltigheten genom att med kvalitativt angreppssätt studera fenomenet utifrån dess egna förutsättningar. Eftersom det har varit svårt att skapa koncept som uppfyller de egenskaper som Järvinen (2002) ställer på koncept i konceptuell analys så har det varit lättare att ha ett mer förutsättningslöst arbetssätt. Detta har också givit möjlighet att fånga upp de olika nyanser som finns i källmaterialets beskrivningar.

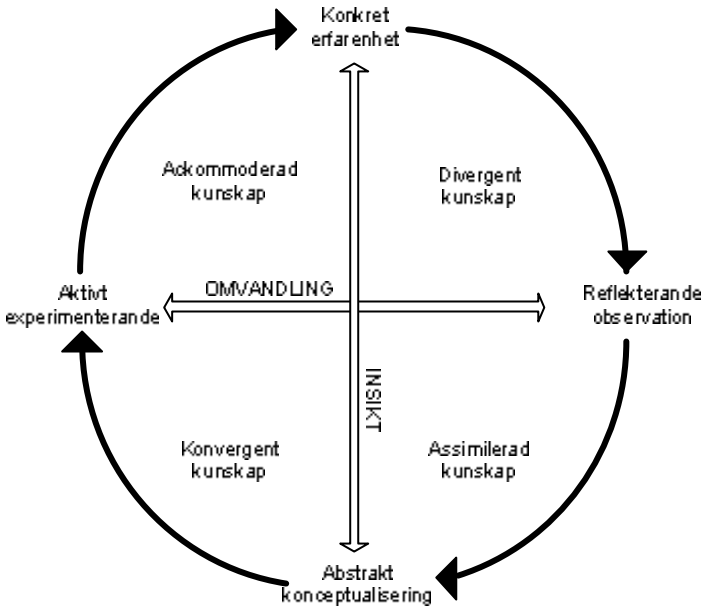
3 Erfarenhetsbaserat lärande

Hergenhahn och Olson (1993) definierar lärande i sin enklaste form som den process som leder fram till kunskap. De menar att lärande är en ständig process som inte är knuten till specifika platser, tidpunkter eller aktiviteter. Denna process som definierats ett stort antal gånger har i många avseenden detta oberoende av kontext som gemensam nämnare. Kolb (1984) ansluter sig till denna definition av lärande och fogar samtidigt till dimensionen om erfarenheter:

> *”Learning is the process whereby knowledge is created through the transformation of experience.” (Kolb, 1984, s. 38)*

Kolb (1984) beskriver lärandeprocessen som *erfarenhetsbaserat lärande*. Han menar att lärande måste beskrivas utifrån en process som har vissa väsentliga särdrag. För det första menar han att lärandeprocessen betonar anammande och därför som inte kan sammanfattas med resultat och prestationer. Vidare menar han att lärandeprocessen består av en kontinuerlig transformation av erfarenheter. Denna transformation kan inte isoleras eller ses som oberoende av andra aktiviteter. Transformationen sker av både objektiva och subjektiva aspekter på en individs erfarenheter. Således är lärandeprocessen en kunskapsbildande process som ständigt pågår och som samlar en individs alla erfarenheter och bildar grund för ny kunskap.

Enligt Kolb går lärandeprocessen från *konkret erfarenhet* över *reflektiv observation* och *abstrakt konceptualisering* till *aktivt experimenterande* (Se figur 1). Dessa fyra faser är att betrakta som en slags aktiviteter eller dimensioner i lärandet där kunskapen formas genom omvandling (transformation) och insikt (förståelse). Mellan de olika faserna återfinns fyra typer av kunskaper: *divergent kunskap, assimilerad kunskap, konvergent kunskap* och *ackommoderad kunskap*.



Figur 1 – Strukturen i lärandeprocessen (Kolb, 1984)

Under fasen *konkret erfarenhet* samlas individens erfarenheter och bildar en ostrukturerad kunskapsmängd (divergent). Dessa erfarenheter ligger till grund för att individen skall börja reflektera över vad han eller hon har genomgått eller upplevt. I den *reflekterande observationen* bearbetas erfarenhetsintrycken och den divergenta kunskapen omvandlas och upptas till en mer strukturerad kunskapsmängd (assimilerad). Genom denna process övergår individen skapa abstrakta uppfattningar om verkligheten eller det aktuella fenomenet. Denna kunskap ger förståelse av komplexitet och ligger till grund för kritiskt tänkande. Då individen kan göra *abstrakta konceptualiseringar* av sin verklighet så ökar också möjligheten att de nya kunskaperna appliceras på nya situationer och individen börjar undersöka tillämpbarheten i andra sammanhang (konvergens). Genom att *aktivt experimentera* med de nya insikterna kommer individen således att kunna göra nya erfarenheter om situationer, fenomen och upplevelser. Därigenom anpassas kunskapen för att kunna användas i nya sammanhang (ackommoderad).

Processerna i modellen är insikt och omvandling. Individen omvandlar kunskap genom att reflektera över erfarenheter och experimentera med sina insikter. Förståelsen går från att ha direkta erfarenheter av ett fenomen till att ha en djupare kunskap om en komplex situation.

Kolb (1984) menar att lärandeprocessen kan stimuleras genom att stötta individen i de olika faserna och därigenom påskynda eller underlätta aktiviteterna i desamma. I en arbetsmiljö menar Kolb att lärandet gynnas av att tillväxt och förändringar är starka, ledning och övervakning är positiv samt möjligheter till avancemang, självtänkande och eget ansvar finns. Inom utbildningar så menar han vidare att det måste finnas tre huvudsaktliga aktiviteter för att stimulera lärandeprocessen: Inhämtning av kunskaper och färdigheter, specialisering av färdigheter och integration av kunskap till djupare förståelse. Avsaknad av dessa egenskaper i arbetsmiljö respektive aktiviteter i studiemiljöer kommer enligt Kolb oavkortat att försvåra lärandeprocessen.

I denna litteraturstudie är definitionen av lärande och lärandeprocessen de som kommer att ligga till grund för analys. Visserligen är även kunskapsbegreppen väsentliga för lärandeprocessen och det resultat som många gånger eftersträvas i mätningar eller värderingar av lärandet. Emellertid ger de erfarenhetsbaserade lärandeprocessen inget svar på hur man kan mäta kunskaper. Däremot kan processen i sig identifieras genom sina egenskaper vilket är vad som görs i denna studie.

4 Tillämpningar av XP

4.1 XPs värden, aktiviteter och principer

XP har sitt ursprung i ett antal mer eller mindre framgångsrika tillämpningsförsök under senare delen av 1990-talet som har teoretiserats och beskrivits av Beck (1999, 2000). Referenserna till Beck (2000) som huvudkälla för hur XP skall bedrivas är väldigt många[7]. Becks beskrivning av XP är emellertid den första i en rad böcker om hur XP skall genomföras (Beck & Fowler, 2001; Baird, 2002, Jeffries et al., 2001), om hur det har genomförts (Wake, 2002; Lippert et al., 2002) och som mer kritiskt granskar XP (Stephens & Rosenberg, 2003; McBreen, 2003).

Beck (2000) menar att metoden är en nedbantad utvecklingsmetod speciellt anpassad för utvecklingsteam som är små eller medelstora. Han menar att metoden särskilt väl är anpassad för programvara och system som utvecklas i miljöer där kraven är oklara och/eller starkt föränderliga under utvecklingsprocessen. XP handlar som Beck betonar inte om att arbeta extremt med programmering utan snarare att utnyttja ett mindre antal förnuftiga principer på ett extremt sätt och därigenom säkerställa att alla som är inblandade i processen arbetar med rätt saker, producerar nytta och minskar fel eller eliminerar felrisker i alla lägen. Avseende de aktiviteter som genomförs i XP så skiljer sig inte XP speciellt från andra utvecklingsmetoder. I XP återfinns testning, kodning, integrering, iterationer, analys och design, arkitektur mm. Således skiljer sig inte XP nämnvärt i det avseendet. När det gäller verktyg för utövandet så är XP inte heller anmärkningsvärt. Exempelvis har XP testats tillsammans med ett antal olika programspråk såsom Small Talk, Java, C++, med olika notationer (exempelvis UML) och med olika grad av dokumentation. Därigenom skiljer sig inte möjligheterna att mer eller mindre fritt välja utvecklingsmiljö heller.

Det som skiljer XP framför andra metoder är snabbheten till anpassningar efter förändrade och förnyade krav, samt att på ett bättre sätt ta tillvara kompetenser i utvecklingsteamet. XP verkar i extremt korta iterationer som alla innehåller minimala utvecklingar stegvis av den programvara som skall utvecklas. Fokus i XP ligger också på att beslut skall fattas av dem som berörs av det. Således ligger i princip alla beslut om utvecklingen hos utvecklarna i teamet. Vidare är XP extremt grupporienterat och betonar vikten av att utvecklarna verkligen jobbar i grupp med beslut, med vad som produceras och inte minst hur arbetsmiljön skall vara för individerna. Således finns i XP även arbetsprinciper som främjar att individer kontinuerligt skall lära av varandra eftersom arbetet är att som grupp producera bättre system och bättre mjukvara. Ytterligare aspekter på XP är att det i många avseenden förkastar onödig dokumentation. Fokus i metoden ligger, kanske inte oväntat, på just programmering. T ex

[7] I denna litteraturstudie refererar mer än 98% av alla ingående artiklar till Beck (2000).

menas i XP att programkod i sig utgör en stor del av dokumentationen i ett utvecklingsprojekt.

I XP finns ett antal värdegrunder, huvudaktiviteter och principer som alla främjar de bakomliggande idéerna i metoden. Värdegrunderna *kommunikation, enkelhet, feedback* och *kurage* syftar alla till att utvecklingsteamet skall ha en väl fungerande och stabil bas. Beck (2000) menar att dessa värden är sociala värden som inte skall förbises utan vara väsentliga delar av teamets arbete.

Kommunikation är väsentligt i de flesta av aktiviteterna i XP. En stor del av aktiviteterna utförs i grupp eller som minst två och två. Att bortse från muntlig kommunikation i detta avseende vore fel. Förutom muntlig kommunikation måste teamet kommunicera skriftligt via programkod och den begränsade mängd dokumentation som XP i övrigt föreskriver.

Enkelhet innebär att teamet alltid skall sträva efter att minska komplexitet i det som produceras och det som skall vara gemensamt. Eftersom utvecklingsteamet gemensamt producerar och äger sina ”produkter” så menar Beck att detta är en av de viktigare värdena för att alla i teamet skall kunna förstå och arbeta med alla delar.

Med feedback avser Beck att det är ovärderligt att få exakt, tydlig och konstruktiv feedback på vad man har åstadkommit och hur långt man kommit mot målet. Detsamma gäller feedback mellan individer och vad man som individ har producerat. I XP betonas dels omedelbar feedback mellan individer och kollektiv feedback för gruppen som helhet.

Det sista värdet, kurage, kan vara det svåraste att tillämpa i praktiken. Med det menar Beck att man behöver våga öppna sig som individ för att fungera i ett team som arbetar enligt XP. Han menar rent av att om man är en utpräglad individualist så kan man inte arbeta enligt XP. Kurage avser i detta avseende att man behöver visa sina svaga sidor och vara mottaglig för feedback, men också att våga ta över andras lösningar, förbättra och återkoppla det man arbetar med till sina kollegor.

Vidare har XP fyra huvudaktiviteter: *koda, testa, lyssna* och *designa.* XP är en metod som utgår från programmerarna. Beck menar att programmerarna kan förbättra sin kommunikation genom att programmera (koda) tillsammans. Han menar att muntliga beskrivningar lätt missuppfattas medan programkod förklarar tankelogik på ett sätt som inte framkommer genom samtal. Testning är centralt i XP och syftar till att eliminera fel så tidigt som möjligt. Tester av programkod skrivs före själva koden och det menar Beck hjälper programmerare att förstå vilket syfte en viss programkod har. Beck menar med aktiviteten att lyssna att programmerare behöver öka sitt lyssnande på kunder, uppdragsgivare och andra intressenter för att därigenom bättre fånga upp krav, syften och mål med utvecklingen. Slutligen menar Beck att design skall bedrivas kontinuerligt på daglig basis av programmerare i ett XP-team. I designaktiviteten menar Beck att komplexitet är upphov till dålig design. Således föreslår han att komplexitet genom goda principer skall elimineras och därmed öka kvaliteten i den producerade programvaran.

Förutom de värdegrunder och huvudaktiviteter som är grundläggande i XP så arbetar man efter ett antal principer. Dessa principer är tillämpningar, genomförande och praxis då ett team använder XP. Beck förespråkar att man tillämpar dessa och gör det aktivt. Han menar att de tolv principerna ingalunda är nyheter utan snarare en sammanställning av välkända aktiviteter och en betoning av hur dessa bör genomföras. Samtliga av dessa principer bygger

på huvudaktiviteterna som förekommer i olika balans i de olika principerna. De tolv principerna är:

- Planeringsspelet (The Planning Game)
- Små versionsreleaser (Small releases)
- Projektmetafor (Metaphor)
- Enkel design (Simple design)
- Testning (Testing)
- Omstrukturering (Refactoring)
- Parprogrammering (Pair programming)
- Kollektivt ägande (Collective ownership)
- Kontinuerlig integration (Continous integration)
- 40-timmarsvecka (40-hour week)
- Kund i teamet (On-site customer)
- Kodstandard (Coding standards)

Explicit finns i XP egentligen bara en princip som talar om lärande. Beck benämner den *lära att lära* och placerar inte den bland de centrala principerna. Han menar att denna princip betyder att utvecklare som praktiserar XP själva skall lära sig att lära hur tillämpningar bäst görs eller som han skriver:

> *Rather than make a bunch of doctrine statements like 'Thou must do testing like XYZ', we will focus on teaching strategies for learning how much testing you should do. Also how much design, and refactoring, and everything else you should do.* (Beck: 2000, s.39)

Flera författare menar att lärande är närvarande genom värdegrunderna, huvudaktiviteterna och principerna trots att det inte explicit nämns. Baird (2002) menar att en av fördelarna med *parprogrammering* är att underlätta delning av kunskap genom att utvecklare arbetar tillsammans med att *koda*. Han menar vidare att synergieffekterna från två programmerare som arbetar tillsammans bidrar till ökade erfarenheter och därigenom möjligheter till nya kunskaper. Lippert et al. (2002) exemplifierar kunskapsspridningen mellan programmerarna genom *kollektivt ägande*. De menar att en förutsättning för att det faktiskt skall gå att dela programkod mellan programmerare är att de befinner sig på samma kunskapsnivå. För att åstadkomma detta föreslår de utbildning och övning. De menar att också *kontinuerlig integration* bidrar till denna kunskapsspridning genom att alla programmerare har tillgång till all kod och därigenom kan dra nytta av den. Wake (2002) menar att *kodstandard* hänger samman med alla de ovanstående genom att vara en förutsättning för *kommunikation* mellan programmerare. McBreen (2003) pekar på ytterligare principer som väsentliga för att minska komplexitet och därigenom öka förståelsen hos programmerare. Han nämner *projektmetafor*, *enkel design* och *omstrukturering* som viktiga principer för att gynna förståelse.

Sammanfattningsvis menar alltså Beck (2000), Baird(2002), Wake (2002), Lippert et al.(2002) och McBreen (2003) att ett antal av XPs principer, huvudaktiviteter och värdegrunder gynnar kunskapsspridning, ökad förståelse och större möjligheter till erfarenheter.

4.2 Akademiska tillämpningar av XP

I denna litteraturstudie finns arton artiklar som tydligt kan kopplas till XP i en akademisk kontext. Den akademiska kontexten innebär i nästan alla artiklar att XP genomfördes som en del av en eller flera universitetskurser. Gemensamt för denna kontext var att det i de flesta fall saknades ”riktiga” kunder att producera programvara till och att det i samtliga fall fanns ett underförstått mål med lärande, som mer eller mindre alltid är närvarande i universitetskurser. Undantaget från detta presenteras av Holcombe et al. (2001) som istället utgår från ett studentdrivet företag (sanktionerat av universitetet). Det gemensamma med övriga inom denna grupp av artiklar är istället att de studenter som utövat XP har motsvarande erfarenheter från yrkesmässig programmering som andra studenter, d v s relativt ringa.

Holcombe et al. (2001) har i övrigt en typisk erfarenhet i hur studenter adopterar XP som metod och hur de tar till sig nyckelprinciperna, nämligen med stor entusiasm. De menar att den speciella miljö med verkliga och intresserade kunder, motiverar studenter och entusiasmerar dem mer än vad som annars vore normalt. Holcombe et al. pekar inte ut några delar av XP som speciellt framgångsrika eller misslyckade, de visar inte heller på särskilda fördelar med XP gentemot tidigare metoder i sitt studentprojekt utan framhåller mer betydelsen av att studenterna verkligen får erfarenheter av att arbeta med programmering och mjukvaruutveckling med riktiga kunder som intressenter. Lappo (2002) menar liksom Holcombe et al., att låta studenterna lära genom praktiska erfarenheter är det viktigaste när det gäller programmering i allmänhet och XP i synnerhet.

> *However, there is more to learning XP than sitting in a lecture and learning some facts.* (Lappo: 2002, s.4)

Han konstaterar emellertid att de erfarenheter som krävs för att verkligen kunna göra jämförelser, reflektioner och bilda sig en förståelse för XP och programmeringsprojekt, tar lång tid att skaffa sig. Dessa erfarenheter är något som studenter inte har tillräckligt med tid eller resurser för att hinna skaffa sig. Däremot betonar han vikten av att studenter får påbörja denna erfarenhet och åtminstone ges en chans att ta till sig kunskaper från andras erfarenheter. Även Sanders (2001) reflekterar över de nyttiga erfarenheter som XP bidrar till i universitetskurser. Hans pilotstudie grundar sig i försök med XP både hos oerfarna studenter i introduktionskurser och mer erfarna studenter i senare kurser. Försöken visar visserligen att XP bidrar till nyttiga erfarenheter för samtliga studenter, men att mer erfarna studenter har bättre förutsättningar för att förstå de erfarenheter de gör genom att tillämpa XP.

> *…but using it [XP] for a project should be delayed until the students become more proficient in the traditional phases of software development.* (Sanders: 2001, s.5)

Således menar han att kunskaper i traditionell systemutveckling är nödvändiga för att kunna reflektera över fördelar och nackdelar med XP och därigenom få nytta av de erfarenheter som ett XP-projekt ger. Sanders (2001), Lappo (2002) och Holcombe et al. (2001) ger en gemensam bild av hur XP tillämpas akademiskt. De är inte ensamma om att betona hur viktigt det är för lärandet att studenter får erfarenheter av programmering och systemutveckling. Astrachan et al. (2001) går ytterligare ett steg längre och understryker inte bara vikten av erfarenheter för programmeringsstudenter. De skapar en genomgående erfarenhetsbaserad

miljö för studenterna från första till sista programmeringskurs och ser studenterna mer som klienter/kunder i detta arbete. Deras erfarenheter baseras mycket på aktiva inlärningsmetoder som syftar till att studenterna i alla lärtillfällen skall vara aktiva. Speciellt betonar Astrachan et al. (2001) *parprogrammering* som viktig i detta avseende. De har testat parprogrammering enligt Beck (2000), men har också utvecklat den till att omfatta föreläsningar inom programmering. Deras tillämpning är då med läraren som ena part i paret och samtliga studenter i auditoriet som andra part. Trots att de understryker att XPs principer inte kan isoleras menar de att parprogrammering är den absolut viktigaste delen av XP, inte minst i universitetsmiljöer. Genom att tillämpa XP och parprogrammering menar de att studenterna förbättrar sitt sätt att tänka kring programmering och därmed får en bättre inblick i hur programvara är uppbyggd.

Parprogrammering är den princip som de flesta akademiska XP-försök rapporterar som mest tillämpad, viktigast för lärande och lätt att praktisera. Back et al. (2002) menar att lärandet är kontinuerligt mellan programmerare som tillämpar parprogrammering. Detta är speciellt betydelsefullt för studenter eftersom de ofta har olika kunskaper, skiftande erfarenheter och olika kunskapsnivåer. Med parprogrammering menar de att man kan förvänta sig att mer erfarna studenter bidrar till att öka programmeringskunskaper till sina mindre erfarna kollegor. Även Wilson (2001) instämmer i de positiva effekterna av parprogrammering. Han menar visserligen att studenter ofta arbetar tillsammans i par utan att det organiseras genom en metod, men att parprogrammering bidrar till en bättre inlärning genom att vara del av utvecklingsmetoden. Keenan (2002) är även han positiv till parprogrammering, men hans försök visar att skiftande erfarenheter inte enbart bidrar till kunskapsutbyte mellan studenter. Istället menar han att parbildningen skall ske med försiktighet och att likartade kunskapsnivåer ger bäst resultat. Hans erfarenheter pekar på att det annars tenderar att vara den starkare programmeraren som dominerar i parprogrammeringen. Lappo (2002) beskriver parprogrammering som misslyckat i sina försök. Han härleder problemen till personlighetsskillnader och konkurrenssituationen mellan studenter. Den miljö som Lappo har testat XP inom innebär till stor del att studenterna konkurrerar om betyg och prestationer. Detta visade sig vara det som i stor utsträckning bidrog till att de kollektiva och sociala aspekterna av XP blev svåra att genomföra för honom. Baheti et al. (2002) har testat parprogrammering i en distansmiljö i ett experiment med studenter som programmerare. Deras jämförelse mellan parprogrammerare på samma plats respektive distribuerat visar att det även är möjligt att nå lika hög produktivitet på distans som samlokaliserat. Visserligen har Baheti et al. inte undersökt lärandet inom parprogrammering distanserat, men de menar att utbytet av erfarenheter kan ske lika bra med enkla distanshjälpmedel som exempelvis konferensprogramvaror. Laurie Williams är refererad av dem som har testat parprogrammering i utbildning. Hon och några medförfattare har i ett antal artiklar (Williams et al., 2000; Williams & Kessler, 2000; Williams & Kessler 2001; Cockburn & Williams, 2001) och en avhandling (Williams, 2000) studerat parprogrammering och lärande. De menar att parprogrammering förbättrar resultatet jämfört med individuella prestationer, ökar färdigheter och kunskaper hos programmerarna och ökar möjligheterna till kommunikation inom projektgruppen. De menar att just aktiviteten parprogrammering starkt bidrar till lärande vars resultat är just högre produktivitet, programvarukvalitet och kommunikation. Med bättre kunskaper och insikter menar de att framtida fel förhindras och undviks vilket leder till dessa förbättringar. De flesta av studierna bakom dessa slutsatser gjordes i akademiska miljöer och isolerade det kollaborativa inslaget i XP (främst parprogrammering) utan att låta det ingå i något XP-projekt.

Testning och *Omstrukturering* var ytterligare två XP-principer som studenter tog till sig i en större utsträckning. Steinberg (2001) menar att dessa principer främjar lärande bättre än andra eftersom de är starkt fokuserade till just programmering, där han menar att de viktigaste tillämpningarna av XP görs. Han menar att studenterna har lärt sig mer om programmeringskoncept, abstraktion och begrepp genom att aktivt tillämpa testning före programmering och kontinuerlig omstrukturering av sin egen programkod. Dessutom menar han att det har inneburit vissa fördelar i undervisningen av Java som programspråk.

> *…an additional pedagogical benefit was the delay in having to specify a main() method.* (Steinberg: 2002, s.4)

Steinberg antyder slutligen att det har funnits sociala fördelar med att studenterna har praktiserat XP. Han menar att studenterna hade mycket lättare för att diskutera sina programmeringsproblem med varandra och att de i allt större utsträckning vågade vända sig till varandra inom gruppen med problem.

4.3 Icke-akademiska tillämpningar av XP

Av artiklarna i studien hade 54 artiklar en tydlig icke-akademisk kontext, d v s de grundade sig på studier som inte genomförts i universitetsmiljö. De flesta av dessa artiklar berör erfarenheter från införande av XP som metod i ett utvecklingsteam, på ett företag eller inom ett specifikt projekt. Erfarenheterna som rapporteras berör XPs principer (Griffin, 2001; Kirkpatrick, 2001, Karlström, 2002), tester av ny teknik i samband med XP (Henninger et al., 2002; Finsterwalder, 2001) eller alternativa sätt för genomförande av XP (Pine, 2001; Kircher et al. 2001; Eckman, 2002). De flesta av dessa rapporter från icke-akademiska miljöer saknar koppling till individuellt lärande. Några behandlar lärande på organisationsnivå och intar ett lärandeperspektiv i form av kunskapshantering med XP i fokus (Putman, 2002, Maurer, 2002, Bossi & Cirillo, 2001).

Kerievsky (2001) är den som presenterar lärande mest ingående och föreslår att lärande skall få en mer framskjuten plats, rent av som huvudprincip i XP. Han föreslår att lärandet skall vara kontinuerligt och utövas som egna aktivteter i form av kunskapsbanker, retrospektiva iterationer och studiegrupper. Syftet med det kontinuerliga lärandet som Kerievsky förslår är tekniskt för programmerarna i studiegrupper, kunskapsbanker för att samla idéer till utveckling och som han skriver om retrospektiva iterationer:

> *…participants take the lessons learned during the retrospectives and turn them into concrete ideas for improving their development process.* (Kerievsky: 2001. s. 4)

Således menar Kerievsky (2001) att hela utvecklingsteamet, kunder såväl som utvecklare, skall ingå i dessa aktiviteter för att utveckla själva processen. Insikter och kunskaper skall alltså göras tillgängliga och tillgodogöras för individer såväl som gruppen som helhet. Collins och Miller (2001) beskriver även dom retrospektion och lägger till introspektion som ytterligare en metodförbättring för XP. Introspektion menar de är när projektdeltagare skall utveckla ett lärande genom erfarenheter i den egna projektprocessen och genom att lära sig av tidigare misstag och framgångar. De menar precis som Kerievsky (2001) att visserligen är lärande ”inbyggt” i XPs principer, men att inte säkerställa lärande genom att avstå från särskilda aktiviteter ökar riskerna för att förändringar aldrig genomförs.

Bailey et al. (2002) presenterar lärande utifrån var svårare lärmoment finns i XP. De menar att en av de största utmaningarna i XP finns i principen *testning*. Testning i sig är svårt och att lära sig skriva effektiva och bra tester är komplicerat och att dessutom lära sig att göra det före man kodar det som skall testas gör det hela ännu mer omständligt. De menar att utmaningen i lärande finns i de principer som är svårast att tillämpa och testning är en princip som fundamentalt ändrar arbetssättet för de flesta utvecklare och programmerare. Ytterligare några principer som Bailey et al. (2002) identifierar som utmanande avseende lärande är *kund i teamet* och *planeringsspelet*. Principen kund i teamet innebär att klienter/kunder aktivt skall delta i utvecklingsarbetet. Det kräver att de förstår och är insatta i hur utveckling skall bedrivas med XP, förutom att utvecklarna skall hantera XP och kund/klient. Planeringsspelet innebär att användarfall skall beskriva olika delar av programvaran som utvecklas. Att skriva bra och begripliga användarfall som samtidigt är användbara i programmering och testning är mycket svårt menar de och något som kräver träning och god kommunikation med klient/kund. Van Cauwenberghe (2001) belyser även han vikten av kundens närvaro i ett utvecklingsprojekt:

> *We learn all the time from the customer, from the system being developed.* (Van Cauwenberghe: 2001, s. 2)

Han menar att XPs inkrementella utveckling säkerställer kontinuerligt lärande från kunden och från de erfarenheter utvecklarna gör i sitt arbete. Genom det kontinuerliga lärandet förbättras förståelsen för problemområdet, designen och programvaran som utvecklas. Han menar då att utvecklarna kan göra abstraktioner av tidigare erfarenheter och hitta enklare och bättre lösningar på nya problem.

Lovaasen (2001) och Grenning (2001) identifierar det huvudsakliga lärandet genom erfarenhetsutbytet i *parprogrammering*. De menar att det är oerhört värdefullt att sprida kunskaper och erfarenheter mellan programmerare. Grenning (2001) poängterar närvaron av åtminstone en rutinerad programmerare för att sprida kunskaper medan Lovaasen (2001) menar att kunskapsspridningen är stor bara genom aktiviteten parprogrammering och inte minst genom kommunikation mellan medarbetarna.

> *The rate of learning is so much higher because of pair programming and the excellent level of communication among the entire team.* (Lovaasen: 2001, s. 9)

Grenning (2001) menar liksom Lovaasen att även mer erfarna programmerare lär sig genom parprogrammering, men att de bidrar på ett annat sätt till kunskapsspridning.

> *…senior people are needed. They help spread the wealth of knowledge.* (Grenning: 2001, s. 6)

Johansen et al. (2001) beskriver de erfarenheter som gjorts vid introduktionen av XP i ett mindre utvecklingsteam och beskriver vikten av att lära av erfarenheter.

> *…the value isn't learned until you are deeply involved in actually doing XP.* (Johansen et al.: 2001, s. 1)

Johansen et al. (2001) beskriver hur de introducerade XP i sitt utvecklingsteam genom interaktion mellan medarbetare, demonstrationer och aktivt deltagande. Genom att praktisera de olika principerna i XP så konstaterade de att de gradvis fick bättre inblick i bakomliggande faktorer till de olika principerna. Genom detta menar de att de löste många problem från tidigare projekt och att de fick en samsyn på testning, planering, prioritering, ansvarsområden, kodförbättring och vikten av att arbeta kollektivt. Johansen et al. menar alltså att de lärt sig XP just genom att tillämpa principerna och genom att reflektera tillsammans.

5 Lärandeprocessen och XP

5.1 Inledande analys

Källorna i denna litteraturstudie är möjliga att gruppera utifrån olika perspektiv. Det huvudsakliga perspektivet har här varit vilken typ av tillämpning som har gjorts av XP. Denna gruppering har använts för att klassificera artiklarna som ingår i studien.

Utöver artiklarna så ingår åtta bokkällor som inte faller inom denna klassificering p g a att de inte har den typ av empiriska utgångspunkt som artiklarna i litteraturstudien har. Bokkällorna i studien är dock möjliga att gruppera på andra sätt. Den viktigaste bokkällan för XP är Beck (2000). Då i princip samtliga källor i litteraturstudien refererar Beck (2000) och använder de beskrivningar han har av XP som metod, måste den anses vara den viktigaste bokkällan. Övriga åtta bokkällor kan indelas i tre olika typer. De som beskriver XP som metod i syfte att förmedla metoden som sådan (I), de som beskriver hur XP som metod har tillämpats i utvecklingsprojekt (II) och slutligen de som kritiskt granskar XP som metod i relation till andra metoder (III) (Se tabell 5).

Gruppering	**Referenser**
I	Beck (2000), Beck och Fowler (2001), Baird (2002) och Jeffries et al. (2001)
II	Wake (2002), Lippert et al. (2002)
III	Stephens och Rosenberg (2003), McBreen (2003)

Tabell 5 – Grupperingar av bokkällor

Dessa tre grupper av bokkällor var inledningsvis intressanta att titta närmare på. Emellertid visade det sig att de delvis saknade relevans och delvis saknade den inriktning som ovanstående gruppering indikerar. De bokkällor som fanns inom den första gruppen (I) är mer eller mindre direkt uppbyggda efter Beck (2000) och ger ytterligare förtydligande av det han har beskrivit, samt ger instruktioner om hur de olika värdena, aktiviteterna och principerna skall tillämpas. De ger således inte egentligen någon ytterligare analys av metoden eller nyanserar den på något mer omfattande sätt. Inom grupp två (II) finns bokkällor som förvisso har tillämpat XP och sammanfattar detta. De försöker dessutom dra slutsatser av hur de har lyckats genomföra XP i sina respektive organisationer och projekt. Dessa slutsatser är däremot inte sådana att de förs tillbaka i form av kritik mot metoden i första hand utan snarare självkritiska granskningar av hur de har genomfört XP och hur man skall göra för att bättre följa de rekommendationer Beck (2000) har. Den sista gruppen (III) av bokkällor utger sig för att vara mer kritiska mot XP som metod. Visserligen har de en mer kritisk ton gentemot XP och de avfärdar XP då förutsättningarna för att genomföra XP inte uppfylls. De har dessutom

synpunkter på metoden som sådan och några av principerna men de föreslår sällan några omfattande metodförändringar eller förbättringar. I detta avseende är de ingalunda grundat kritiska mot XP. Således kan sammanfattningsvis sägas om bokkällorna i denna litteraturstudie att samtliga bygger på Beck (2000) och accepterar hans beskrivning av XP som metod. Därigenom blir Beck (2000) den metodbeskrivning som alltjämt är den rådande av XP som metod. Bokkällornas beskrivningar ger alltså inte mycket ytterligare om hur XP som metod skall genomföras, har genomförts eller inte borde genomföras.

Artiklarna som klassificerades efter akademiska och icke-akademiska tillämpningar fördelades enligt följande:

Klassificering	**Antal artiklar**
Akademiska tillämpningar	13+5
Icke-akademiska tillämpningar	48+7

Tabell 6 – Klassificering av artiklar (de artiklar som adderas tillkom genom referenssökningarna)

Av de artiklar som ingick i den ursprungliga sökningen var det möjligt att klassificera samtliga med denna utgångspunkt. Två av de artiklar som tillkom i sökningarna av referenser visade sig inte möjliga att göra denna klassificering på. Den ena saknar helt beskrivningar av vilken tillämpningsmiljö som hade använts och den andra har båda miljöerna med i sin utgångspunkt. Klassificeringen efter akademisk och icke-akademisk tillämpningsmiljö visade sig vara lämplig i denna litteraturstudie. Artiklarna med de akademiska tillämpningarna har genomgående en högre kvalitet genom att vara tydligare disponerade, genom bättre akademiska utgångspunkter med metoder, teorier och tydligare referenser till annan forskning. Dessutom har de genomgående ett syfte med lärande som fångades in genom denna klassificering. De icke-akademiska tillämpningarna av XP visade sig dock ha genomfört XP som metod på ett bättre sätt och ger därför en bättre bild av hur XP fungerar i ett utvecklingsteam. De har också genomgående ett fokus på en kund som nästan helt saknas i de akademiska tillämpningarna, vilket är till XPs fördel eftersom metoden är fokuserad på att producera programvara åt en kund. Många av artiklarna visade sig ha liten eller ingen relevans för denna studie och de som har relevans beskrivs i de följande analysavsnitten. Intressant att notera är också att inte heller artiklarna avviker från Beck (2000) som huvudreferens. Med ett par undantag utgår samtliga artiklar från hans metodbeskrivning och bygger upp tillämpningarna av metoden kring framför allt de principer som han beskriver. Ingen av artiklarna bygger på någon annan metodbeskrivning av XP (de som inte refererar Beck (2000) saknar referens till XP som metod).

Sammanfattningsvis kan sägas att både artiklar och bokkällor i denna litteraturstudie är uppbyggda kring den beskrivning av XP som metod som Beck (2000) förmedlar. Det finns förvisso alltså andra metodbeskrivningar, men dessa lyser med sin frånvaro. I följande kapitel utgår jag från Becks beskrivningar av XP som metod (Beck 1999, 2000) för att analysera lärandeprocessen i XP.

5.2 Lärandeprocessen och metodbeskrivningar av XP

I metodbeskrivningar av XP, med Beck (2000) som främsta källa, finns få explicita uttryck för lärande. Beck (2000) beskriver lärande i XP som en mindre framstående princip – *lära att lära*. Denna princip innebär att utövare av XP inte skall lära sig exakt hur XPs huvudprinciper

och aktiviteter skall tillämpas utan snarare bygga en strategi för att själva förstå tillämpningen. Genom denna princip om lärande menar Beck (2000) implicit att utvecklare som skall använda XP måste skaffa sig *erfarenheter* av XP som metod för att kunna tillämpa den. Beck (2000) knyter alltså samman lärande och erfarenheter genom att värdera erfarenheterna som viktiga för utvecklarna och genom att de betonas i principen om lärande. Vidare är XP en metod som tillämpas för att utveckla programvara (utvecklingsprocessen). Beck (2000) menar alltså att samlandet av erfarenheter skall ske under utvecklingsprocessen och kontinuerligt, främst genom tillämpningen av XPs huvudaktiviteter och genom de tolv principerna. Sambandet mellan erfarenheter, lärande och utvecklingsprocessens kontinuitet hänger väl samman med den definition av lärande som Kolb (1984) gör:

> *"Learning is the process whereby knowledge is created through the transformation of experience." (Kolb, 1984, s. 38)*

Beck (2000) förordar en metod där *kommunikation, enkelhet, feedback* och *kurage* som alla är värdegrunder som betonar samverkan, förståelse mellan utvecklarna, ständiga förbättringar och möjligheter till personlig utveckling. Därigenom är det tydligt så att Beck (2000) utan att explicit beskriva det, starkt förespråkar att ett utvecklingsteam inte bara skall ha en kontinuerlig utvecklingsprocess utan även en kontinuerlig lärandeprocess inom teamet. Kommunikation, enkelhet, feedback och kurage är de värdegrunder som finns närvarande i XP och de är samtidigt som de beskrivs av Beck (2000) viktiga instrument för förändring och ständig förbättring. Således underlättar värdegrunderna det som Kolb (1984) sammanfattar med förändring (transformation). Implicit finns alltså erfarenhetsförändring i XPs värdegrunder.

I sammanhanget är det intressant att belysa de tolv principer Beck (2000) beskriver i XP. Han pekar inte ut någon som speciellt gynnsam i ett lärandeperspektiv. Emellertid belyses de genom de fyra huvudaktiviteterna *koda, testa, lyssna* och *designa*. Dessa fyra aktiviteter förekommer i olika sammansättningar i de tolv principerna och de innehåller liksom värdegrunderna förutsättningar för erfarenhetsutbyte mellan individer. Kodning som bedrivs i huvudsak parvis i XP ger möjligheter till att underlätta problemlösning och kommunikation mellan utvecklare. Detta gemensamma arbete stöttar alltså möjligheterna till erfarenhetsinsamling och förändring av erfarenheter. Testning menar Beck (2000) hjälper programmerare att förstå sin egen och andras programkod och ökar alltså möjligheterna till förändring av erfarenheter. Genom att lyssna ökar en utvecklare sina möjligheter att samla erfarenheter och öka sin förståelse. Slutligen måste utvecklaren genom design bryta ner komplexitet för att kunna bygga upp programvara. Därigenom ökar alltså möjligheterna till att ytterligare öka förståelsen av ett utvecklingsprojekt och dess problematik. De fyra huvudaktiviteterna har alltså ett uttalat syfte att öka antalet *erfarenhetsutbyten*, förbättra *förståelse* och underlätta *kommunikation*. Samtliga av dessa aktiviteter har alltså betydelse för en lärandeprocess inom XP.

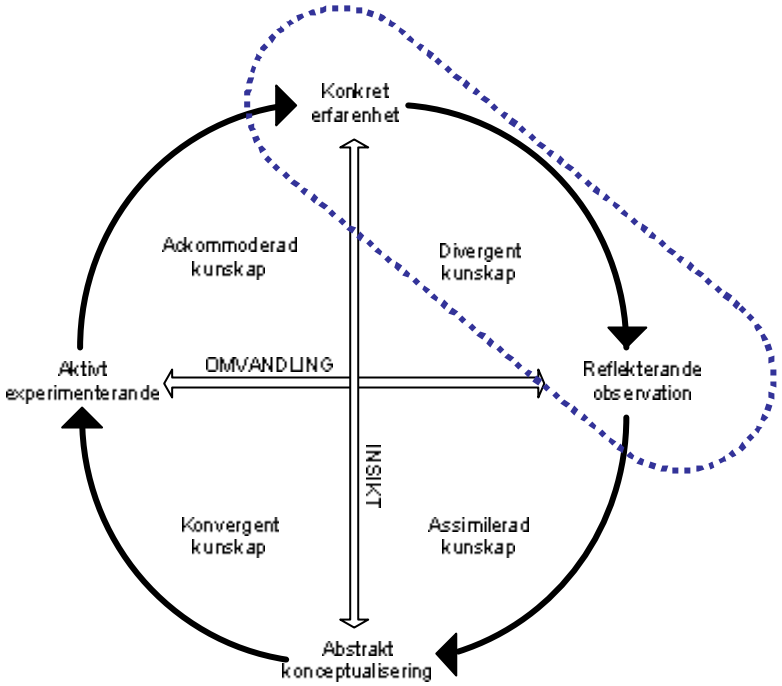
Beck (2000) beskriver inte i detalj hur han har tänkt att sammansättningen av aktiviteter skall se ut för de olika principerna. Än mindre kopplar han lärande till de tolv huvudprinciperna i XP. Däremot belyses lärande i några av principerna hos de andra bokkällorna i denna litteraturstudie. Kunskapsutbyte betonas genom *parprogrammering* av Baird (2002), genom *kollektivt ägande* och *kontinuerlig integration* av Lippert et al. (2002) och genom *kodstandard* av Wake (2002). Således menar de att dessa principer särskilt skulle betona kommunikation och erfarenhetsutbyte mellan utvecklarna. De menar också att detta utbyte ökar kunskaperna hos utvecklarna. Dessa principer bör alltså innehålla aktiviteter som främjar både insamling

av erfarenheter och förändring av dessa till kunskaper. Med andra ord bör det i dessa principer finnas en lärandeprocess närvarande. McBreen (2003) menar att det utöver dessa finns ytterligare principer som särskilt ökar förståelse hos främst utvecklare och programmerare i ett team som utövar XP nämligen *projektmetafor, enkel design* och *omstrukturering.* Således kan sägas att de principer där lärandeprocessen bör vara särskilt tydlig är:

- Parprogrammering
- Kollektivt ägande
- Kontinuerlig integration
- Kodstandard
- Projektmetafor
- Enkel design
- Omstrukturering

I förhållande till de fem övriga principerna som Beck (2000) beskriver så karaktäriseras dessa principer av att vara kollektiva, betona vikten av en fungerande kommunikation och att de innehåller komplexitetsminskande aktiviteter. Dessa principer är alltså sådana att de främjar sådant som är förutsättningar för en lärandeprocess.

Vidare är det intressant att betrakta hur dessa principer förhåller sig till den erfarenhetsbaserade lärandeprocess som Kolb (1984) presenterar (Se figur 2). Baserat på bokkällornas beskrivningar av XP är det rimligt att placera XP som metod i faserna *Konkret erfarenhet* och *Reflekterande observation.* Detta baseras på att XP i första hand är beskrivet utifrån aktiviteter som främjar insamling av erfarenheter och viss reflektion i form av exempelvis feedback och kommunikation. Det som inte finns i bokkällorna är mer konkreta aktiviteter som skapar abstraktioner ur erfarenheterna. Inte heller finns det någon explicit och uttalad metodförbättring där XP som metod eller de enskilda programmerings- och utvecklingserfarenheterna kan utvecklas utifrån de erfarenheter som utvecklare och programmerare gör. Således når man metodmässigt sällan ett aktivt experimenterande som kan leda till en kontinuerlig erfarenhetsbaserad kunskapsprocess. Emellertid är dock detta resonemang enbart utifrån XP som den är presenterad rent metodmässigt. I den konkreta tillämpningen sker givetvis denna process kontinuerligt vilket också framgår av artiklarna i litteraturstudien.



Figur 2 – Strukturen i lärandeprocessen (Kolb, 1984) och XP som metod

Metodbeskrivningarna i litteraturstudiens bokkällor menar att några av XPs principer har större förutsättningar än andra att främja lärandeprocesser. Detta baserat på att dessa principer mer än andra betonar kommunikation, kollektivt arbete och möjligheter till utbyte och reflektion. Vidare placerar sig metodbeskrivningarna i området för *divergent kunskap* där erfarenheter och kunskaper samlas respektive börjar reflektera kring. XP som metod har inte i dessa källor några uttalade eller organiserade aktiviteter som leder metoden till abstraktioner eller aktivt experimenterande med nya insikter. Däremot finns förstås förutsättningar för dessa abstraktioner och experiment i tillämpningar av XP, vilket belyses i följande kapitel.

5.3 Lärandeprocessen och akademiska tillämpningar

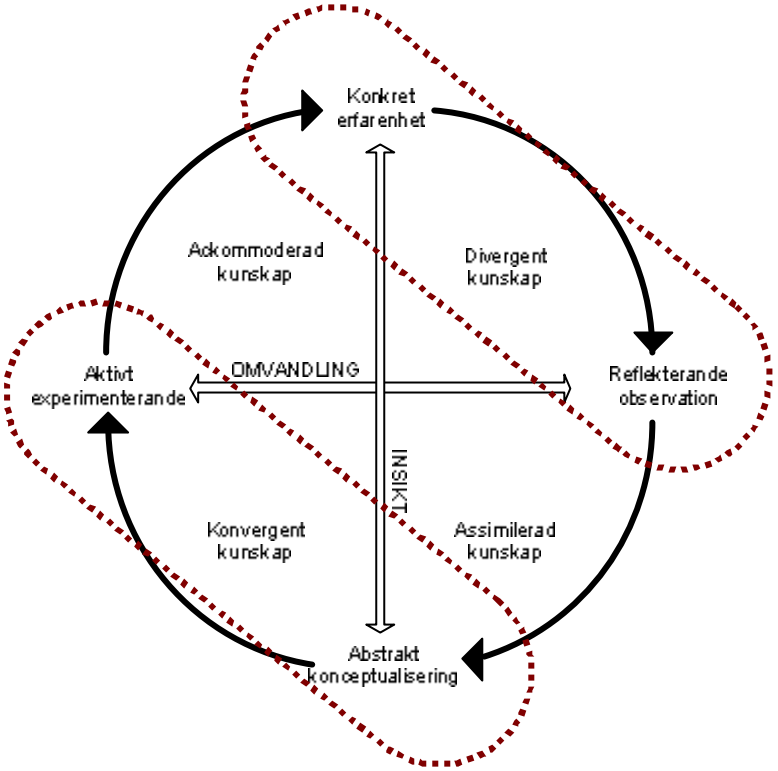
I akademiska tillämpningar av XP har studenter i princip alltid ett underförstått syfte att lära sig något. De flesta av de akademiska tillämpningar som ingår i denna studie härrör från kurser i systemutveckling och/eller programmering. I dessa kurser har XP använts som metod för att genomföra utvecklingsprojekt och uppgifter som genomförs i grupp. En viktig aspekt i dessa kurser är att ge studenter erfarenheter av programutveckling i verkliga eller simulerade miljöer. Genomgående i alla akademiska tillämpningar betonas studenternas möjligheter att samla erfarenheter genom tillämpning av XP. Exempelvis visar Holcombe et al. (2001), Lappo (2002), Sanders (2001) och Astrachan et al. (2001) tydligt på att studenterna entusiastiskt tar sig an uppgifter och projekt med verkliga eller fiktiva kunder och tillämpar många av XPs principer. Samtliga berör vikten av erfarenheter för att förstå programmering och systemutveckling och lägger betydelsen i just samlande av erfarenheter.

Dessa rapporter om samlande av erfarenheter i akademiska miljöer ger indikationer på Kolbs (1984) erfarenhetsbaserade lärandeprocess. Studenterna ges möjlighet att samla erfarenheter som kanske inte annars vore möjligt. Vidare borde rimligen också XP bereda möjligheter att förvandla (transformera) dessa kunskaper genom reflektion. Eftersom XP är kollektivt och

betonar kommunikation borde rimligen även detta främjas vid akademiska tillämpningar av XP. Den princip som de flesta betonar vara väsentlig för denna kommunikation och reflektion är *parprogrammering.* Wilson (2001), Keenan (2002) och Baheti et al. (2002) rapporterar alla att parprogrammering har varit den mest använda principen och att den har varit betydelsefull för studenternas möjligheter till gemensam problemlösning och reflektioner kring uppgifter. Laurie Williams har genomfört ett antal studier om parprogrammering och kollektiva arbetsuppgifter för studenter. Hennes resultat visar att parprogrammering är lätt för de flesta studenter att ta till sig och göra till en aktiv del av studierna. Parprogrammering innehåller aktiviteter som stimulerar insamling av *erfarenheter* då studenterna arbetar med problemlösning tillsammans. Det uppmuntrar till *kommunikation* genom att studenterna måste kommunicera genom programmeringen såväl muntligt som genom programkod och det förbättrar möjligheterna till *reflektion* eftersom studenterna förväntas ge varandra omedelbar feedback på varandras arbete.

Förutom parprogrammering är det inte många av XPs principer som visat sig vara särskilt intressanta ur lärandesynpunkt. *Testning* och *omstrukturering* är två principer där lärande påtalas i de ingående artiklarna. Steinberg (2001) menar att dessa aktiviteter som är starkt orienterade kring programmering, är lätta för studenter att ta till sig och tillämpa då de tämligen isolerat kan genomföras. Utifrån ett perspektiv med utgångspunkter i erfarenheter och reflektioner så befrämjar principerna testning och omstrukturering lärande främst genom att underlätta förståelse, problemlösning och minskning av komplexitet. Båda principerna är mindre kommunikativa än parprogrammering.

Även om XPs principer inte är tillämpade i någon större omfattning så menar de flesta akademiska tillämpningar att användningen av XP har inneburit ett ökat och i vissa fall mer fokuserat lärande. Astrachan et al. (2002) menar att de studenter som har tillämpat XP har förbättrat sitt tänkande kring programmering och fått en bättre inblick i hur programvara byggs upp. Mot bakgrund av det och i relation till Kolbs (1984) erfarenhetsbaserade modell så är det rimligt att anta att studenter har ett mer lärandefokuserat perspektiv, utgår från mindre omfattande och mindre komplexa erfarenheter, gör reflektioner utifrån dessa och skapar sig abstrakta konceptualiseringar som senare kan omsättas i ett mer aktivt experimenterande. Mycket tyder på att så skulle vara fallet. Det genomgående och tydligaste tecknet på detta är hur lätt studenterna i de akademiska tillämpningarna har att förändra och ändra sina arbetssätt baserat på de erfarenheter de gör. Många studenter är givetvis pragmatiska och släpper lätt ett arbetssätt för att, åtminstone som ambition, bättre, snabbare och lättare klara en uppgift. Emellertid har den entusiasm som XP har genererat också möjliggjort ett visst mått av experimenterande med metoden XP (Se figur 3).



Figur 3 – Strukturen i lärandeprocessen (Kolb, 1984) och XP i akademiska tillämpningar

De abstraktioner och det aktiva experimenterande som förekommit i akademiska tillämpningar av XP har emellertid inte inneburit något återförande av kunskaper till XP som metod utan snarare en för studenterna större förståelse av programmering och utvecklingsmetoder i allmänhet och för XP som metod i synnerhet.

5.4 Lärandeprocessen och icke-akademiska tillämpningar

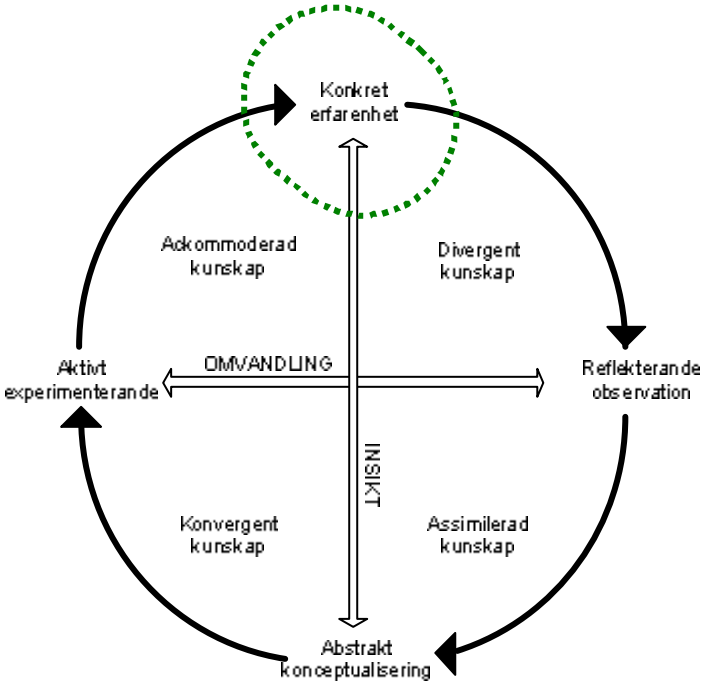
Det lärande som finns underförstått i akademiska tillämpningar av XP saknas i icke-akademiska tillämpningar. I de artiklar som ingår i denna studie är det få som explicit beskriver lärande i någon större omfattning. Några artiklar beskriver lärande på organisationsnivå (10 st) vilket ligger utanför denna litteraturstudie som har ett individuellt lärandeperspektiv. De allra flesta icke-akademiska artiklarna i denna studie beskriver ett lärande genom erfarenheter av XP, men berör lärande bara implicit och flyktigt och specificerar det inte närmare.

Några få specificerar lärandet närmare genom XPs principer. Inte oväntat är *parprogrammering* en sådan princip. Lovaasen (2001) och Grenning (2001) ringar in parprogrammering som den viktigaste principen för kunskapsutbyte, inte minst mellan rutinerade programmerare och utvecklare och mindre rutinerade sådana. Van Cauwenberghe (2001) och Bailey et al. (2002) tar upp principen *kund i teamet* som viktig för förståelse av problemområdet bakom projektet och de uppgifter som skall lösas genom programvaruutvecklingen. Bailey et al. (2002) berör vidare *planeringsspelet* där kunden skall vara delaktig i skapandet av s.k. användarfall. I dessa principer är kommunikation viktig för att förmedla kunskaper mellan kunder och utvecklare. Vidare berör Bailey et al. (2002)

testning som en viktig princip för lärande. De menar visserligen inte att lärande naturligt uppstår genom testning utan omvänt att testning är så omfattande och krävande att ett lärande är oundvikligt för att testning skall fungera som det är tänkt. Denna princip berör både insamlande av erfarenheter, reflektioner och nya tillämpningar för att generera detta lärande. Det Bailey et al. (2002) berör här är således en möjlighet till att testning leder till att utvecklare måste generera nya kunskaper för att kunna tillämpa testning.

Några artiklar rapporterar om alternativa sätt att genomföra XP på, exempelvis distribuerat med utvecklare som inte sitter på samma plats eller med särskilda typer av utvecklingsprojekt. Visserligen kan dessa artiklar indikera att XP förändras genom aktivt experimenterande, men i detta fall visade det sig inte vara det. Dessa alternativa sätt var i samtliga fall snarare försök att visa på att den metod som Beck (2000) presenterar också är möjlig att förlänga till miljöer där XP normalt skulle vara svårt att genomföra. Ett undantag från detta är Kerievsky (2001) som faktiskt försöker att tillföra lärande som princip i XP. Kerievsky (2001) föreslår vilka aktiviteter som skall finnas med i denna nya princip och de bygger på att samla *erfarenheter*, *kommunicera* dessa till utvecklare och i organiserade former *reflektera* över desamma. Kerievsky (2001) lutar sig inte mot någon definition för lärande, men han hänvisar till dessa egenskaper i lärande som direkt kan hänföras till Kolbs (1984) erfarenhetsbaserade lärandeprocess.

De flesta icke-akademiska tillämpningar berör som nämnts insamling av erfarenheter. I förhållnade till den erfarenhetsbaserade lärandeprocessen befinner sig de icke-akademiska tillämpningarna av XP då i första fasen (se figur 4). De flesta artiklar menar också att denna insamling av erfarenheter skall leda till lärande. I förhållande till lärandeprocessen så saknas emellertid aktiviteter som leder till reflektion och än färre artiklar påvisar abstraktion och nytt aktivt experimenterande. Visserligen finns det undantag från detta i Bailey et al. (2002) och Kerievsky (2001), men de allra flesta beskriver inte hur lärandet av erfarenheter skall ske, utan skriver bara att det sker.



Figur 4 – Strukturen i lärandeprocessen (Kolb, 1984) och XP i icke-akademiska tillämpningar

5.5 Lärande och XP

Syftet med denna litteraturstudie har varit att identifiera lärandefenomenet i beskrivningar av XP genom följande frågeställning:

- I vilka aktiviteter uppstår lärande och lärandeprocesser i tillämpningar av XP?

Denna litteraturstudie har visat att metodbeskrivningar (här: bokkällor) beskriver väldigt få explicita aktiviteter som främjar lärande. Huvudkällan bakom XP, Beck (2000) har en mindre viktig princip som explicit betonar lärande. När det gäller tillämpningar som rapporterats genom vetenskapliga artiklar så visade det sig vara främst akademiska tillämpningar, där studenter var delaktiga i användningen av XP, som betonade lärande. I icke-akademiska källor visade sig lärandet vara betydligt svårare att hitta och de flesta artiklar rapporterar kursivt om var lärandeprocesser har varit uppenbara.

Vidare har XP som metod granskats närmare i förhållande till den erfarenhetsbaserade process som Kolb (1984) presenterar. I metodbeskrivningar, akademiska tillämpningar och icke-akademiska tillämpningar har lärande i XPs principer sökts. Mellan de tre olika källtyperna finns både skillnader och likheter (se tabell 7).

Metodbeskrivningar	**Akademiska tillämpningar**	**Icke-akademiska tillämpningar**
Parprogrammering Kollektivt ägande Kontinuerlig integration Kodstandard Projektmetafor Enkel design Omstrukturering	Parprogrammering Testning Omstrukturering	Parprogrammering Testning Kund i teamet Planeringsspelet

Tabell 7 – Principer med lärande i XP för metodbeskrivningar respektive tillämpningsområden

Parprogrammering är den princip som samtliga källor beskriver som gynnsam för lärande. Testning beskrivs i både akademiska och icke-akademiska tillämpningar som gynnsam för lärande. Parprogrammering är gynnsam för lärande dels med avseende på erfarenhetsutbyte och dels med avseende på kommunikation och reflektion. Testning gynnar förståelse av komplexitet och abstraktion snarare än erfarenhetsutbyte.

Övriga skillnader mellan de olika källtyperna är svårare att analysera. De principer som har rapporterats gynnsamma i metodbeskrivningarna och inte har samma resultat i tillämpningarna kan vara det av olika orsaker. En sådan kan vara att de inte rapporterats lika omfattande i de tillämpningar som ingår i denna litteraturstudie. Det kan också vara så att de inte lika frekvent eller på rätt sätt har tillämpats i de rapporterade studierna. Ytterligare en förklaring kan vara att metodbeskrivningarna inte ger en korrekt bild av hur principerna faktiskt tillämpas inom XP-projekt.

Omstrukturering som rapporteras vara gynnsam i akademiska tillämpningar gynnar liksom testning förståelse av komplexitet vilket kan vara en förklaring till detta. När det gäller kund i teamet och planeringsspelet (även där med kunden närvarande) gynnar tillämpningar i icke-akademiska miljöer eftersom kundens närvaro ökar möjligheterna att göra rätt och skapa programvara. Därigenom kan det vara så att lärandet blir tydligare då utvecklarna har kommunikation och feedback med kunden.

När det gäller XP i den erfarenhetsbaserade lärandeprocessen så placerar sig XP som metod främst inom faserna *konkret erfarenhet* och *reflektiv observation*. Få av källorna i litteraturstudien tar erfarenheterna längre än till att just att konstatera dom och möjligen reflektera över dom. I icke-akademiska tillämpningar är det ännu tydligare att de flesta artiklar handlar om erfarenheter från tillämpningar oftast utan att återföra dessa erfarenheter till vare sig individen eller XP som metod. I akademiska tillämpningar sker kunskapsprocessen för studenterna som tillämpar XP, men i ett annat huvudsyfte än det XP har nämligen att inhämta kunskaper relaterat till den kurs där XP tillämpas.

6 Slutsatser

XP är en utvecklingsmetod som betonar sociala aspekter som kollektivt arbete, ansvar, erfarenhetsspridning och kommunikation. I metodbeskrivningar av XP framstår lärande som en del av XP som rimligen borde vara del av metoden. Lärande är en förutsättning för att XP skall fungera och utveckla det team som tillämpar metoden. Huvudsyftet med XP är att producera programvara av hög kvalitet, snabbare och bättre än andra metoder. Genom att integrera lärande som ett sätt att kontinuerligt förbättra tillämpningen av XP kan metoden utvecklas och bli en mer flexibel och bättre anpassningsbar metod. Idag saknar XP aktiviteter som återför erfarenheter och förbättrar XP som metod. Dessa aktiviteter skulle med fördel kunna vara kopplade till lärandeprocessen för att tydliggöra att erfarenheter behöver bearbetas för att kunna tillföra nya kunskaper genom aktivt experimenterande med de djupare slutsatser som dragits av erfarenheter från tillämpningar av XP.

XP tillämpas främst inom icke-akademiska miljöer. Metoden är främst skapad för utveckling av programvara åt kunder och dåligt anpassad till akademiska tillämpningar. De akademiska tillämpningar som ändå sker med XP genomförs ofta med simuleringar av verkliga miljöer eller med projekt med motsvarande förutsättningar som icke-akademiska miljöer har. Emellertid har akademiska tillämpningar i huvudsak andra mål än icke-akademiska och det främsta av dessa är ofta lärande. Båda tillämpningsmiljöerna kan ha nytta av att mötas för att utveckla det lärande som faktiskt förekommer om än i olika form inom tillämpningarna. Akademiska tillämpare är bra på att tillämpa XP-aktiviteter och principer som främjar huvudsyftet med den miljö de befinner sig i, nämligen lärande och studier. Icke-akademiska tillämpare har ofta kommersiella mål och syften med XP och är bra på att tillämpa aktiviteter och principer som gynnar det egna projektet. Genom att förena dessa och förändra XP genom att utveckla principer som tydligare betonar olika miljöer kan XP bli en metod som passar bra för både akademiska lärandemiljöer och icke-akademiska kommersiella miljöer. Utifrån denna litteraturstudie föreslår jag att XP behåller sina principer och utvecklar tillämpningsrekommendationer för akademiska miljöer så XP kan främja lärande på fler plan i dessa miljöer. På så vis skulle XP även kunna utvecklas till att vara en metod för att inte bara utveckla programvara utan även för att lära sig att utveckla programvara. På motsvarande sätt skulle XP i icke-akademiska miljöer vara en metod för att utveckla team som skall utveckla programvaror, d v s en metod som är självförbättrande för utvecklingsteamet.

I denna litteraturstudie har syftet varit att finna lärande och lärandeprocesser i beskrivningar av XP. Den har visat att XP har förutsättningar för erfarenhetsbaserat lärande men för närvarande bara i en tidig del av den erfarenhetsbaserade lärandeprocessen. Därför behöver XP utvecklas som metod för att bättre gynna lärande och tydligare kunna föra tillbaka dessa kunskaper till metoden och till utvecklarna. Inom detta område finns således fortsatta intressanta områden att utforska.

7 Referenser

Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J. (2002). *Agile Software Development Methods.* Espoo: VTT.

Andersen, E. S. (1994). *Systemutveckling – principer, metoder och tekniker.* Lund: Studentlitteratur.

Astrachan, O., Duvall, R.C., Wallingford, E. (2001), Bringing Extreme Programming to the Classroom. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Back, R.J., Milovanov, L., Porres, I., Preoteasa, V. (2002), XP as a Framework for Practical Software Engineering Experiments. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Backman, J. (1998), *Rapporter och uppsatser,* Lund: Studentlitteratur.

Baheti, P., Gehringer, E., Stotts, D. (2002), Exploring the Efficacy of Distributed Pair Programming. In Wells, D., Williams, L. (Eds.), *Extreme Programming and Agile Methods – XP/Agile Universe 2002,* Berlin: Springer.

Bailey, P., Ashworth, N., Wallace, N. (2002), Challenges for Stakeholders in Adopting XP. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Baird, S. (2002). *Extreme Programming in 24 Hours.* Indiana: SAMS.

Baskerville, R., Myers, M. (2002). Information Systems as a Research Discipline. *MIS Quarterly,* Vol. 26, No.1, (sid. 1-14).

Beck, K. (1999). Embracing Change with Extreme Programming. *Computer*, Volume 32 Issue: 10, October 1999. (sid. 70-77).

Beck, K. (2000). *Extreme Programming Explained.* Upper Saddle River, NJ: Addison-Wesley.

Beck, K., Fowler, M (2001). *Planning Extreme Programming.* Upper Saddle River, NJ: Addison-Wesley.

Bossi, P., Cirillo, F. (2001), Repo Margining System: Applying in the Financial Industry. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Cockburn, A., Williams, L. In Succi, G. Marchesi, M., Extreme Programmin Examined;: Upper Saddle River: Addison-Wesley.

Collins, C.T., Miller, R.W. (2001), Adaptation: XP Style, In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Dahlbom, B. (1997). The New Informatics. *Scandinavian Journal of Information Systems*, vol 8, no 2, 1997.

Dick, A.J., Zarnett, B. (2002). Paired Programming & Personality Traits. *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering*. Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Easterby-Smith, M., Lowe, A., Thorpe, R. (2002), *Management Research*, London: SAGE.

Eckman, E.C. (2002), Transition Roadmap. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Elvheim, M. (2003), Extreme Learning with Extreme Programming, In *Conference Proceedings 26th Information Systems Research Seminar in Scandinavia*, Haikko Manor: Helsinki School of Economics.

Finsterwalder, M. (2001), Automating Acceptance Tests for GUI Applications in an Extreme Programming Environment. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Grenning, J., Using XP in a Big Process Company. In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Griffin, L.A. (2001), A Customer Experience: Implementing XP. In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Henninger, S., Ivaturi, A., Krishna, N., Thirunavukkaras, A. (2002), Supporting Adaptable Methodologies to Meet Evolving Project Needs. In Wells, D., Williams, L. (Eds.), *Extreme Programming and Agile Methods – XP/Agile Universe 2002*, Berlin: Springer.

Hergenhahn, B.R., Olson, M.H. (1993). *An Introduction to Theories of Learning*. Englewood Cliffs, NJ: Prentice Hall.

Holcombe, M., Gheorghe, M., Macias, F. (2001), Teaching XP for Real: Some Initial Observations And Plans. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Jeffries, R., Anderson, A., Hendrickson, C. (2001). *Extreme Programming Installed.* Upper Saddle River, NJ: Addison-Wesley.

Johansen, K., Stauffer, R., Turner, D., Learning by Doing: Why XP Doesn't Sell. In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Järvinen, P. (2001). *On Research Methods*. Tampere: Järvinen.

Karlström, D. (2002), Introducing Extreme Programming – An Experience Report. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Keenan, F. (2002). Teaching And Learning XP. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Kerievsky, J. (2001). Continous Learning. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Kircher, M., P. Jain, Corsaro, A., Levine, D. (2001), Distributed Extreme Programming. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Kirkpatrick, D. (2001), Finding the Right Process Mix in the Real World. In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Kivi, J, Haydon, D., Hayes, J., Schneider R., Succi, G. (2000). *Extreme Programming: A University Team Design Experience*. 2000 Canadian Conference on Electrical and Computer Engineering, Vol 2000:2. (sid. 816-820)

Kolb, D. (1984). *Experiential Learning*. Englewood Cliffs, NJ: Prentice Hall.

Lappo, P. (2002), No Pain, No XP. Observations on Teaching and Mentoring Extreme Programming to University Students. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Lovaasen, G., Brokering With Extreme Programming. In *Conference Proceedings 2001 XP Universe*. Raleigh: Object Mentor.

Lippert, M., Roock, S., Wolf, H. (2002). *Extreme Programmin in Action.* West Sussex: John Wiley & Sons.

Mathiassen, L., Munk-Madsen, A., Nielsen, P.A., Stage, J. (2001). *Objektorienterad analys och design.* Lund: Studentlitteratur.

Maurer, F. (2002), Supporting Distributed Extreme Programming. In Wells, D., Williams, L. (Eds.), *Extreme Programming and Agile Methods – XP/Agile Universe 2002,* Berlin: Springer.

McBreen, P. (2003). *Questioning Extreme Programming.* Upper Saddle River, NJ: Addison-Wesley.

Müller, M.M., Tichy, W.F. (2001). Case Study: Extreme Programming in A University Environment, *Proceedings of the 23rd International Conference on Software Engineering*. ICSE 2001. (sid. 537-544).

Newkirk, J. (2002). Introduction to Agile Processes and Extreme Programming. *Proceedings of the 24th International Conference on Software Engineering, 2002*. ICSE 2002. (sid. 695-696).

Putman, D. (2002), Where Has All the Management Gone? In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Pine, S. (2001), An Application of XP in a Multiple Team/Multi-process Environment. In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Sanders, D. (2001), Student Perceptions of the Suitability of Extreme and Pair Programming, In *Conference Proceedings 2001 XP Universe.* Raleigh: Object Mentor.

Sharifabdi, K., Grot, C. (2002). Team Development and Pair Programming – tasks and challenges of the XP coach. In *Conference Proceedings XP 2002, 3rd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia & Free University of Bolzano-Bolzen.

Sommerville, I. (2000). *Software Engineering 6th Ed.* Essex: Pearson Education Ltd.

Steinberg, D.H. (2001), The Effect of Unit Tests On Entry Points, Coupling And Cohesion in An Introductory Java Programming Course. In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Stensmo, C. (1994). *Pedagogisk filosofi.* Lund: Studentlitteratur.

Stephens, M., Rosenberg, D (2003). *Extreme Programming Refactored: The Case Against XP*. New York, NY: Apress.

Van Cauwenberghe, P., Refactoring or Upfront Design? In *Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering.* Alghero: Università di Cagliari, Italia.

Wake, W.C. (2002). *Extreme Programming Explored.* Upper Saddle River, NJ: Addison-Wesley.

Wallén, G. (1996), *Vetenskapsteori och forskningsmetodik*, Lund: Studentlitteratur.

Webster, J., Watson, R., (2002). Analyzing the Past to Prepare for the Future: Writing A Literature Review. *MIS Quaterly,* 26 (2), xiii-xxiii.

Williams, L. (2000). *The Collaborative Software Process*. (PhD Thesis, University of Utah)

Williams, L, Kessler, R.R. (2000), All I Really Need to Know about Pair Programming I Learned In Kindergarden, Communications of the ACM, May 2000.

Williams, L., Kessler, R.R., Cunningham, W., Jeffries, R. (2000). Strengthening the Case for Pair-Programming. *IEEE Computer*, July/August 2000, (s.19-25).

Williams, L.A., Kessler, R.R. (2001). Experimenting with Industry's "Pair-Programming" Model in the Computer Science Classroom. *Computer Science Education*, March 2001.

Wilson, D. (2001), Teaching XP: A Case Study. In *Conference Proceedings 2001 XP Universe*. Raleigh: Object Mentor.