EFFEKTIV IT

LEDTIDER & KVALITET

RAPPORT NR 2 – MARS 1994

MÄTNING FÖR EFFEKTIV SYSTEMUTVECKLING

Tapani Kinnula

SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING

SISU

SISU bedriver ett program för forskning och utveckling inom informationsteknologins tillämpningsområden – Effektiv IT. Grunden till programmet är en förstudie inom detta område som SISU genomfört på uppdrag av Näringsdepartementet och NUTEK. Forskningen koncentreras till områden som har stor ekonomisk relevans för svenskt näringsliv och förvaltning.

Målet med programmet är att svenskt näringsliv och förvaltning ska kunna använda resultaten för att:

- Effektivare styra och utveckla verksamheter
- Minska kostnaderna för informationsförsörjningen
- Bättre utnyttja befintliga informationssystem
- Använda bättre värderings- och kalkyleringsprinciper
- Minska ledtiderna vid införande av nya system
- Förbättra intern och extern kommunikation

Arbetet under första året drivs inom fem forskningsområden: *Systemutvecklingens ledtider och kvalitet*, *Systemarvet*, *Affärskommunikation*, *IT:s ekonomi och management* samt *Verktyg för verksamhetsutveckling*.

Kista april 1994

Bäste läsare!

Du får nu ett exemplar av rapport nr 2 *Mätning för effektiv systemutveckling*, från projektet Systemutvecklingens ledtider och kvalitet inom SISUs forskningsprogram Effektiv IT.

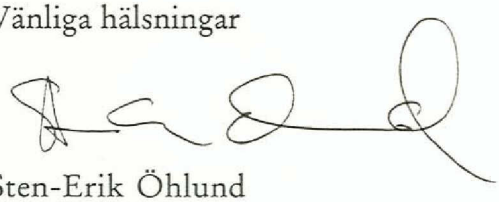
Syftet med rapporten är att ge en inblick i vad mätning av systemutveckling kan vara, vad som kan mätas, samt redovisa och jämföra resultat från en finsk mätdatabas som är kopplad till mät- och projektuppskattningsverktyget Laturi.

Med denna rapport vill vi också slå ett slag för en ökad mätning av system, utvecklingsprocesser och projekt, vilket vi menar har stor betydelse för att åstadkomma en effektivare systemutveckling med kortare ledtider, höjd kvalitet och sänkta kostnader.

Vi har under arbetet med rapporten fått en del värdefulla synpunkter som vi kommer att analysera, bearbeta och vidareutveckla under nästa fas av projektet. Vi tycker ändå det är viktigt att nu komma ut med en första version för att se vilket intresse som finns och förhoppningsvis få feedback till nästa rapport.

Vi vill gärna ha dina synpunkter.

Vänliga hälsningar



Sten-Erik Öhlund
Projektledare

SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING • SWEDISH INSTITUTE FOR SYSTEMS DEVELOPMENT

ELECTRUM 212, 164 40 KISTA • BESÖK: ISAFJORDSGATAN 26 • TELEFON 08-752 16 00 • TELEFAX 08-752 68 00

SISU

Innehåll

1. Inledning

1.1. Mätning – en försummad kunskapskälla inom systemutveckling

Mätning är sedan länge grundläggande inom de flesta ingenjörsvetenskaper och inom konstruktion. Vi mäter elkonsumtion, föremåls fysiska dimensioner, temperatur, strömstyrka, astronomisk strålning, ljustyrka etc. Mätning är en nödvändig källa till kunskap inom ett stort antal områden och nya upptäckter och innovationer är ofta en följd av den kunskap och insikt som mätresultaten ger.

Tyvärr är mätning en inte särskilt etablerad verksamhet inom systemutveckling. Även då dataprogram ingår i tekniska produkter vars fysiska egenskaper prestanda och tillverkningsprocesser noggrant mäts ur alla tänkbara aspekter, sker ingen eller liten mätning av mjukvarukomponenterna och deras utvecklingsprocess.

En förklaring till detta förhållande är att dataprogram och datasystem i grunden är ganska abstrakta företeelser. En annan förklaring är att systemutveckling är en relativt ung verksamhet, som ute på företagen ofta uppfattats som stödverksamhet snarare än kärnverksamhet. Företagen har därför prioriterat frågor som marknadpositioner, tillverkningsprocesser och produktutveckling på bekostnad av den stödverksamhet, som bara ansetts bidra med underlättande arbetsinstrument (datasystem) till de viktiga verksamhetsdelarna.

1.2. Varför har mätning i systemutveckling blivit viktigt?

Synen på dataprogram och datasystem har dock radikalt förändrats under det senaste decenniet. Värdet av informationen i ett företags datasystem anses idag allmänt långt överstiga värdet av dess produktionsapparat. Ett företag med god ekonomi kan snabbt bygga upp en ny produktionsapparat, medan det vore dödsdömt om all information i dess datasystem försvann. Att återskapa underlag för planering och produktion och all nödvändig information om kunder, produkter, service och kontakter, och att styra denna information till rätt segment, skulle bli så kostsamt och tidskrävande att företaget skulle hinna bli utslaget av sina konkurrenter.

Datasystemen är idag en integrerad del i de flesta produktionssystem. Alltmer av produktionen planeras och styrs med datasystem, samtidigt som företagens ekonomiska rutiner, kundinformation och marknadsuppföljning i allt högre grad förlitar sig på fungerande datasystem. Samtidigt har kostnaderna för utveckling och underhåll av alla dessa datasystem blivit en allt större del av företagens totala kostnader. Detta har gjort att datasystemen börjat uppfattas som en livsviktig del av företaget. En del vars kvalitet och pålitlighet är en förutsättning för kvaliteten och pålitligheten i företagets processer och tjänster och även i dess produkter. Datasystem, och därmed även systemutveckling, har blivit någonting som påverkar ens konkurrenskraft, och som man kan påverka sin egen konkurrenskraft med.

1.3. Signaler om mätningens ökande betydelse

Intresset för bättre organisering av systemutveckling, och för effektivisering och optimering av dess arbetsprocesser ökar kontinuerligt. Det avspeglar sig i det intresse som Concurrent Engineering [2] för systemutveckling börjat väcka, och i det faktum att allt fler företag börjat organisera sin systemutveckling för ISO9000-certifiering. Metoder

för att diagnosticera och öka mognaden i företagens systemutvecklingsprocess har också börjat tillämpas. Ett exempel på en sådan metod är Software Engineering Institutes (SEI) Capability Maturity Model. Flera av EG:s ESPRIT-projekt handlar också om mätning i systemutveckling i någon form. Även ISO har initierat ett projekt (SPICE) för framtagning av ett ramverk för analys och utveckling av systemutvecklingsprocessen med utgångspunkt i just SEI:s Capability Maturity Model.

Gemensamt för alla dessa ansatser och modeller är att mätning utgör en grundläggande del i dem, och att en väl fungerande mätningsverksamhet anses vara en nödvändig förutsättning för att medvetet och framgångsrikt effektivisera systemutvecklings-processen.

1.4. Översikt av denna skrift

Denna rapport behandlar mätning i systemutveckling i allmänhet.

Kapitel 2 tar upp mätningens grunder och ger en del definitioner för begrepp. Exempel ges på vad som kan mätas och kapitlet avslutas med ett avsnitt om skillnaderna mellan taktisk och strategisk mätning och betydelsen av strategisk mätning för ett företags konkurrensförmåga.

Kapitel 3 tar upp några mätmetoder och mått som används i dag, främst storleksmått och mått för resursförbrukning och produktivitet.

Kapitel 4 redovisar en del av de praktiska effekter ett fungerande mätprogram kan ha på systemutvecklingsprocessen, och hur mätresultat kan användas för bättre projektkalkyl och projekthantering.

Kapitel 5 utgör en stor och viktig del av rapporten. Här redovisas resultat från två studier om systemutveckling. Den första undersökningen, utförd av Capers Jones I USA, redovisar statistik över produktivitet och kvalitet i amerikansk systemutveckling. Den andra undersökningen har genomförts av författaren och bygger på en projektdatabas med 117 projekt från Finland. Den finska statistiken redovisar också produktivitetskillnader i olika branscher och tekniska utvecklingsmiljöer.

Kapitel 6 avslutar rapporten med några slutsatser och ger också tips om vidare läsning för den intresserade, främst om hur man inför mätning i ett företag.

Kapitel 7 är en förteckning över omnämnd litteratur.

2. Mätningens grunder

Mätning förekommer såväl inom vetenskap och teknologi som i vardagligt liv. Vi mäter för att utvärdera och för att öka vår förståelse, men också för att bättre kunna förutsäga och styra. Inte så att man kan se in i framtiden, men med kunskap om tidigare händelser och deras premisser kan man bilda sig en mening om en trolig eller tänkbar utveckling under liknande betingelser. Därmed kan man öka sin beredskap och sina möjligheter att agera på bästa möjliga sätt.

I detta kapitel belyser vi en del av mätningens teoretiska grunder för att sedan behandla mer praktiska frågor om vad som kan eller bör mätas och hur detta kan ske.

2.1. Definitioner och tolkningar

Vi har tidigare givit exempel på olika mätaktiviteter och på olika situationer som motiverar mätning. Men hur definieras mätning som aktivitet?

> *Definition 1: Mätning* är den aktivitet som tilldelar numeriska eller symboliska värden till egenskaper hos företeelser i verkligheten på ett sådant sätt att de kan beskrivas och analyseras enligt givna och väldefinierade regler.

Mätning handlar alltså om att inhämta information om egenskaper hos företeelser. Dessa företeelser kan vara ett fysiskt objekt eller något mer abstrakt, som till exempel en händelse eller en programmodul eller en testfas i ett systemutvecklingsprojekt. Vilka egenskaper man väljer att mäta beror på vad man vill veta om företeelsen.

Enligt definitionen ovan mäter vi alltså inte i egentlig bemärkelse ett system-utvecklingsprojekt. Istället mäter vi olika egenskaper hos projektet, vilket ger oss kunskap om projektet som sådant, men också kunskap om likheter och gemensamma mönster och regler som gäller för projekt i allmänhet. Då man mäter ett projekt måste man fråga sig vad man mäter. Är det projektets längd, dess resursnyttjande, dess totala kostnad eller vad?

Mätvärden som tilldelas ett projekts egenskaper beskriver projektet för oss. De ger oss värdefull information och hjälper oss att skapa en uppfattning om projektet och att analysera det. Utan att ha deltagit i ett projekt, kan vi med några viktiga mätvärden göra oss en bild av projektet och skilja det från andra. Följande beskrivning kan till exempel uppfattas som om det ger oss en mängd översiktlig kunskap om ett systemutvecklings-projekt:

> Projektet utvecklade ett stort skadeförsäkringssystem. Projektets längd var 3,5 år. Det förbrukade 210 personår, därav 150 det sista året och sysselsatte som mest 40 personer samtidigt. Totalt kostade projektet 70 miljoner kr.

Beskrivningen förefaller informativ, och hjälper oss att jämföra projektet med liknande beskrivningar av andra projekt. Men förstår vi verkligen vad författaren menar? Förmodligen tolkar inte alla beskrivningen likadant. Orsaken till detta är att en del av de mått som används inte är entydigt definierade. Termer som personår och projektkostnad kan definieras på ett antal olika sätt och tolkningen beror på den definition som läsaren utgår från. Vad är då ett mått?

> *Definition 2:* Ett *mått* är den enhet som används för att beskriva värdet från en mätning.

Ett sätt att betrakta mått är att se det som hjälp till att tolka ett värde. Om vi till exempel säger att ett datasystems storlek är 295 000, vet vi inte vad detta värde står för, eftersom det finns flera mått på ett systems storlek, t ex kodrader, funktionspoäng eller

exekverbara programsatser. En angivelse av mätvärdet tillsammans med måttet, 295 000 COBOL-rader, ger oss en klarare uppfattning om hur stort systemet är, men fortfarande kan olika tolkningar göras. Definitionen av måttet borde förtydligas, till exempel genom tillägget att med en rad COBOL-kod menas en icke-blank, okommenterad rad av COBOL.

Vissa mått är enkla och orsakar inga tolkningsproblem, medan andra kan kräva noggranna definitioner, vilket exemplet med COBOL-rader ovan visar.

2.2. Vad mäta och varför

Det finns enligt en grov uppdelning fyra faktorer som är avgörande för ett företags framgång på en marknad med hård konkurrens:

- Produktionskostnad
- Ledtid (time to market)
- Kvalitet
- Kundtillfredsställelse

Det bör rimligtvis finnas ett motiv för varje beslut om mätning. Med andra ord, mätresultatet, och den kunskap det ger, måste vara så värdefullt och användbart i något sammanhang att det motiverar det besvär och den kostnad som mätningen innebär. En allmän riktlinje för mätning i systemutveckling är att mätresultaten skall vara användbara för att åstadkomma något av:

- högre produktkvalitet
- ökad produktivitet
- ökad projekthanteringsförmåga (förmåga att uppskatta, planera och genomföra projekt inom ramen för tids-, resurs- och kostnadsbudget med ökad eller bibehållen produktkvalitet)
- minskade utvecklings- och förvaltningskostnader

Det som mäts kallas mätobjekt. Ett mätobjekt är en egenskap hos en företeelse, snarare än företeelsen som sådan. Vilka egenskaper hos vilka företeelser som skall mätas beror på vilken typ av kunskap och effekt man är ute efter. I systemutveckling är man ofta intresserad av utvecklingsprojekt och deras egenskaper, till exempel projektkostnader, produktivitet, leveranstider och kvalitet. Dessa projektegenskaper är således intressanta att etablera som mätobjekt. En del av dessa kan dock inte mätas direkt, utan kan bara studeras indirekt genom andra, mera grundläggande, egenskaper. Följande resonemang exemplifierar härledning av intressanta mätobjekt och den kunskapsutveckling som kan följa av dessa.

> Kostnaden för ett projekt beror i stor utsträckning på arbetsinsatsen, dvs persontiden. Produktiviteten är förhållandet mellan produktionsvolymen (datasystem) och arbetsinsatsen, dvs produktionsvolym/persontid. Vad gäller kvalitet begränsar vi oss till antal upptäckta fel i granskning eller testning med en given testmetod. Vi har därmed åtminstone följande intressanta mätobjekt: persontid, projektstorlek (produktionsvolym), ledtider och felfrekvens i granskning/testning. Dessa kan också mätas på olika detaljeringsnivåer, t ex på projektnivå eller per projektaktivitet/etapp. Med tillägg av siffror från kostnadsredovisning eller nyckeltal som kostnad per persontimme kan vi få fram några viktiga tal om ett projekt: total kostnad, kostnad för persontiden, total persontid, persontid per etapp, kostnad per etapp, produktivitet, felfrekvens per etapp, förhållande mellan antal fel upptäckta i sista testning och fel upptäckta i specifikationsdokument. Om vi nu har dessa siffror, för låt oss säga 30 projekt, kan vi börja studera trender, systematiska mönster, orsakssamband mellan fel i tidiga faser och fel i sena faser, produktivitetskillnader mellan projekt etc. Den kunskap, som sådana studier genererar, hjälper oss att förbättra vår egen systemutvecklingsprocess.

Det finns generella skäl att etablera vissa mätobjekt, till exempel för att de ger underlag för styrning av projekt och processer[1]. Men det finns även mer individuella behov och mål som avgör valet av mätobjekt. Produktansvariga måste kunna mäta kostnad och resursbehov vid underhåll av en viss produkt, för att kunna se om produktens existens är ekonomiskt försvarbar. Testansvariga behöver kunna mäta felfrekvenser och feltyper i olika systemkomponenter, från analys till implementation, för att senare, med feldata från t ex systemdesign, kunna förutsäga var i ett system olika fel bör sökas och vilka testmetoder som bör användas för att hitta de viktigaste felen.

2.3. Mätobjekt

Om man betraktar systemutveckling som en process bestående av delprocesser, utkristalliserar sig en tänkbar klassificering av företeelser som man kan tänkas vilja mäta ur olika aspekter:

- *processer* som är aktiviteter vilka utförs under systemutveckling
- *produkter* som skapas av processerna
- *resurser* som nyttjas av processerna

Dessa tre klasser av företeelser innefattar dock inte på ett naturligt sätt en annan viktig aspekt med stor betydelse för både processer och produkter. Denna aspekt ger upphov till en ytterligare klass:

- *problemdomän* som betecknar processens/projektets tillämpningsområde såsom bransch och systemtyp.

Det vi på något sätt vill mäta är egenskaper hos företeelser i någon av dessa klasser. Vissa av egenskaperna är direkt relaterade till företeelsers *inneboende* natur (t ex en process löptid), medan andra måste mätas och tolkas med hänsyn till deras *omgivning* (t ex processens totalkostnad räknat från persontid och utrustningskostnader). Vidare kan vissa egenskaper mätas *direkt* medan andra är *indirekta*, dvs kombinationer av andra egenskaper. Ett projekts resursnyttjande i persontimmar är t ex en direkt mätbar egenskap, medan projektets produktivitet är indirekt, eftersom den härleds från två andra egenskaper, nämligen antal producerade kodrader och resursnyttjandet (kodrader/persontimme).

I en projektbaserad organisation utförs mätning ofta på genomförda projekt (dvs tillämpningar av en viss utvecklingsprocess), medan mätresultaten kan relateras även till de underliggande processerna, deras produkter och resurser.

Processer består av relaterade aktiviteter som kräver resurser och som skall producera resultat. Egenskaper som kan mätas relativt enkelt och objektivt kallas hårda egenskaper. Mätbara egenskaper hos processer är bl a :

---

[1]Observera distinktionen mellan projekt och process. Ett projekt är en i tiden avgränsad arbetsinsats för att få fram ett visst resultat. En process är en mängd aktiviteter som tillsammans åstadkommer en viss effekt. Ett projekt kan innefatta arbetsmoment som återfinns som aktiviteter i en process, dvs processen eller dess aktiviteter kan omsättas som arbetsmoment i ett projekt. T ex omsätts testning (process) som testning av en viss komponent (arbetsmoment) i ett visst projekt. Systmeutvecklingsprocessen omsätts på olika sätt i genomförda projekt. Genom att mäta på projekt som bygger på en viss utvecklingsprocess, kan generell kunskap om processen och dess kvalitéer utvinnas.

- *resursnyttjande*, t ex antal persontimmar eller personmånader, eventuellt uppdelat per personalkategori
- *bemanning*, dvs antal personer involverade i processens genomförande
- *produktionsvolym*, t ex systemvolym räknat i antal kodrader eller funktionspoäng, dokumentvolymer, testdatavolymer etc
- *ledtid*, dvs kalendertiden för processens genomförande
- *kvalitetsindikatorer*, t ex antal upptäckta programfel före och under testning samt fel upptäckta efter driftsättning, konstruktionsfel, antal ändringar i kravspecifikationen och i designdokument. (Dessa indikatorer kan senare utnyttjas för diverse kvalitetsnyckeltal såsom antal fel per personmånad, felelimineringseffektivitet, vilket är förhållandet mellan fel upptäckta under utveckling och efter driftsättning samt genomsnittlig kostnad per felrättning, dvs antal rättade fel/persontid för rättningar)
- *kostnad*, gärna uppdelad på kostnadsslag (persontid, lokaler, inköpt/hyrd mjukvara och hårdvara, etc)
- *produktivitet*, t ex antal funktionspoäng eller kodrader per personmånad, antal sidor användardokumentation per personmånad eller antal modultester per personmånad
- *avvikelser från tidsplanen*

Det är här på sin plats att påpeka att processegenskaper som produktionsvolym och kvalitet är nära knutna till motsvarande egenskaper hos de produkter som processen producerar, dvs produkterna volym respektive kvalitet. Mera subjektiva och svårbedömda, men likväl betydelsefulla, processegenskaper är de som kräver en mänsklig bedömning eller värdering. Dessa egenskaper överlappar eller påverkar dessutom ofta varandra. Exempel på dessa så kallade mjuka egenskaper är bl a:

- *stabilitet*, dvs processens känslighet för störningar t ex pga resursbrist
- *standardisering* i processens genomförande
- *styrbarhet*, dvs hur lätt processen är att planera och styra enligt planerna, och planera om eller anpassa till oväntade situationer

Processegenskaper kan mätas per process/aktivitet ner på lägsta efterfrågade nivå. De kan också sammanställas per projekt och projektetapp. Mätdata från mätning av processegenskaper kan uppdelas på egenskapskategorier och kopplas till viktiga process-/ aktivitetstyper. Insamlade mätdata enligt genomtänkt systematik är en förutsättning för analys och effektivisering av utvecklingsprocessen.

Produkter är resultat av processer. Produkter kan t ex vara ett helt informationssystem, systemkomponeneter, specifikationsdokument, konstruktionsdokument, testdata eller användardokumentation. Produkter har även de egenskaper som är mätbara. Några eller alla av följande egenskaper kan användas för att objektivt mäta produkter.

- *storlek* eller *volym*
- *komplexitet*
- *modularitet* eller *struktur*

- *funktionalitet*
- *återanvändningsgrad*
- *stabilitet*
- *kvalitet*

Ibland flyter måtten för t ex storlek och funktionalitet ihop. Ett mått på ett systems funktionalitet kan vara antalet slutanvändarfunktioner, något som samtidigt också indikerar systemets storlek. På samma sätt hävdas att systemstorlek med fördel kan anges i funktionspoäng, vilket många ser som ett mått på den funktionalitet som systemet tillhandahåller slutanvändarna.

Andra egenskaper som inte är lika enkla att mäta objektivt, utan som regel kräver en viss subjektiv bedömning är:

- *underhållsvänlighet*
- *enkelhet*
- *användartillfredställelse*
- *tillförlitlighet*

Även kvalitet kan i viss mån anses tillhöra dessa subjektiva, mjuka egenskaper beroende på vad man lägger för innebörd i kvalitetsbegreppet. På samma sätt kan också tillförlitlighet betraktas som en objektiv egenskap beroende på definition. Om ett systems tillförlitlighet, å ena sidan, definieras som genomsnittligt antal driftavbrott per månad, skulle det kunna mätas objektivt. Å andra sidan kan man definiera tillförlitlighet som användarnas uppfattning av hur robust och tillförlitligt systemet är och därmed få ett subjektivt mått. Slutligen kan båda definitionerna samtidigt användas för att mäta tillförlitlighet ur dessa två synvinklar. Det har ibland visat sig att system som ofta får driftsavbrott eller innehåller fel, ändå av användarna kan uppfattas som mera tillförlitliga än system som mera sällan drabbas av sådant. Detta helt enkelt därför att faktorer som användningsfrekvensen, systemets betydelse för arbetsuppgifterna, i vilka situationer driftavbrotten uppstår och tiden för återstart, påverkar användarnas uppfattning om tillförlitligheten.

Produktegenskaper är naturligtvis inte alltid oberoende av varandra. Till exempel påverkar en produkts komplexitet och modularitet dess underhållsvänlighet. En enkel och välstrukturerad systemmodul är med stor säkerhet mycket enklare att underhålla än en komplicerad spagettikonstruktion. Likaså påverkar komplexitet och modularitet åtminstone indirekt vissa kvalitetsaspekter, såsom risken för att oupptäckta fel finns i systemet. Detta beror ofta på oöverskådlig struktur och komplexitet i möjliga exekveringsvägar vilket ger dålig testbarhet. För många av dessa egenskaper gäller också att det inte finns ett enda mått som väl uttrycker egenskapen, utan snarare en samling mått som tillsammans ger en bild av egenskapen. Komplexitet kan till exempel anges med antal mått som i kombination beskriver komplexiteten ur olika synvinklar och i olika delar hos produkten.

Resurser kan delas upp i personal och verktyg. Vissa egenskaper är gemensamma för både personal och verktyg. En sådan objektivt mätbar storhet är:

- *kostnad*, t ex månadshyra, avskrivning eller inköpspris för verktyg eller timkostnad för personal

En annan viktig egenskap som är gemensam för både personal och verktyg är dock svårare att bedöma objektivt:

- *tillgång*, dvs möjligheten att ha tillgång till resursen när och om den behövs för en viss process eller ett visst projekt

Förutom dessa två egenskaper har både personal och verktyg egna viktiga egenskaper som det kan vara värt att mäta. För personal är dessa:

- *storlek* på projektgrupper
- *kompetens* inom viktiga områden (metoder, projektarbete, verktyg, problemdomän), som kan anges som subjektiva värderingar eller med mer objektiva mått såsom erfarenhet i antal år eller utbildningsnivå
- *produktivitet*, på gruppnivå eller per individ (vilket dock är känsligt och kan orsaka negativa reaktioner)
- *kommunikationsnivå* som anger hur många andra individer/projektgrupper en individ/projektgrupp måste samordna sitt arbete med

Verktyg som utnyttjas i en process kan ha stor betydelse för processens (och personalens) produktivitet och dess kostnad. Förutom de två ovannämnda egenskaperna, kostnad och tillgång, finns det några andra verktygsegenskaper vars mätning och analys kan bringa klarhet i den roll verktygen spelar för produktiviteten eller ge insikt i verktygens kvalitéer relaterade till företagets utvecklingsmiljö:

- *tillförlitlighet*, dvs hur robust och pålitligt verktyget är
- *integrationsnivå* med andra verktyg
- *metodstöd*, dvs hur väl verktyget stödjer de använda metoderna
- *täckningsgrad* för den process de används i, dvs hur väl ett verktyg eller en verktygsuppsättning täcker eller stödjer processen eller projektet
- *nyttjandegrad*, dvs hur väl verktyget nyttjas av personalen (ofta finns ett verktyg tillgängligt men det nyttjas inte effektivt pga bristande kunskap/träning eller pga kompatibilitetsproblem med den övriga verktygsmiljön)

Problemdomänens karaktär har en stor betydelse, både för produkters utformning och för svårigheten att lösa de problem som uppstår under utveckling. Däremot kan det vara svårt, ibland kanske omöjligt, att på ett objektivt sätt mäta själva problemdomänen. Till exempel för ett utvecklingsprojekt, dvs tillämpningsområdet för det system som projektet skall producera. Det huvudsakliga problemet här är att avgränsa problemdomänen till endast relevanta områden. Däremot är det möjligt göra en subjektiv bedömning av olika egenskaper hos problemdomäner. Sådana subjektiva bedömningar kan vara nog så värdefulla och t ex ge insikt i produktivitetsskillnader mellan projekt och komplexitetsskillnader mellan produkter inom olika problemdomäner. Exempel på subjektiva mått för problemdomäner är:

- *storlek* eller omfattning vilket betecknar mängden delområden och företeelser som måste beaktas
- *komplexitet* som uttrycker mängden och karaktären hos samband mellan företeelser som måste beaktas
- *stabilitet*, dvs tillämpningsområdets benägenhet till förändring.

Både storlek och komplexitet ställer krav på kunskap och expertis hos projektpersonalen. Särskilt som det gäller att problemdomänens karaktär inte bara skall förstås och kunna analyseras. Vunnen kunskap skall även kunna överföras till de datasystem som utvecklas inom problemdomänen i termer av funktionalitet, beteende och konstruktion. Stabiliteten å sin sida påverkar stabililiten hos kraven på funktionalitet och beteende hos datasystemen. Därmed påverkar den även produktiviteten i utveckling och förvaltning samt systemens livslängd och/eller underhållsbehov. Att förstå och ta hänsyn till problemdomänen är alltså viktigt för att förstå de krav som ställs på projektpersonalen (projektbemanning) och för att kunna förstå och förutsäga kostnader för både utveckling och förvaltning av datasystem (budget, prognoser, planering).

2.4. Strategisk och taktisk mätning

Mätning kan utföras på olika nivåer och för olika syften i ett företag. Capers Jones [5] skiljer mellan strategisk och taktisk mätning. Strategisk mätning innefattar faktorer som berör hela företaget och dess framgång. Taktisk mätning koncentrerar sig på enskilda projekt och faktorer som påverkar projektutfall och projektförhållanden.

Tabell 1 ger exempel på strategiska och taktiska mätobjekt.

Många av de strategiska mätobjekten är summor och genomsnitt av deras taktiska motsvarigheter. Gemensamt för de strategiska mätobjekten är att de uttrycker tillståndet och förhållandet i företaget som helhet. Deras värden kan jämföras över tiden för att få reda på tendenser och variationer inom strategiskt viktiga områden, såsom kostnadsutveckling, kostnadsfördelning och produktivitetsutveckling. De kan också användas för att sätta upp mål och följa måluppfyllelse. Målen kan vara absoluta (t ex "1995 skall vi producera minst 100 funktionspoäng per anställd"), relateras till olika tidpunkter (t ex "Produktivitetsökningen skall vara minst 5% per år de närmaste 5 åren") eller relaterade till andra företag (t ex "Vi skall vara bland de 5 mest produktiva företagen i vår bransch 1995").

Strategiska mätobjekt	Taktiska mätobjekt
Total personalstyrka	Projektbemanning
Fördelning mellan yrkeskategorier	Persontid per projekt/aktivitet
Projektportföljens storlek	Kostnad per projekt/aktivitet
Lönsamhetsstudier	Projektrisker
Marknadsandelar	Projektlönsamhet
Årlig mjukvarukostnad	Användartillfredsställelse
Årlig hårdvarukostnad	Personalkompetens
Årlig personalkostnad	Produktion per projekt
Total produktionsvolym (ex. funktionspoäng)	Projektproduktivitet
Årlig produktionsvolym	Projektkostnad
Företagsproduktivitet	Felfrekvenser
	Verktygseffektivitet

Tabell 1. Strategiska och taktiska mätobjekt

Ibland kan skillnaderna mellan de taktiska och strategiska mätvärden vara radikala. Låt oss betrakta produktivitet som exempel. Taktisk mätning av produktivitet ger oss produktiviteten i projekt som har avslutats och som har levererat sitt resultat. Vi kan därifrån räkna fram genomsnittsproduktiviteten för under året slutförda projekt. Detta produktivitetsvärde kan med fördel användas för att uppskatta produktivitet, resursbehov och kostnader i kommande projekt.

Projekt som lagts ned eller inte hunnit avslutas före årets slut, har inte kunnat mätas med avseende på produktivitet och syns inte i detta genomsnittsvärde. Men om vi nu vill mäta den strategiska produktiviteten, dvs företagets produktivitet, får vi kanske ett värde som

inte ens är i närheten av den genomsnittliga projektproduktiviteten. Orsaken är att företaget bara har kunnat leverera resultat från de slutförda projekten, medan nedlagda eller oavslutade projekt inte bidragit till produktionsvolymen men väl till konsumerad persontid. Den strategiska (och strategiskt mycket viktiga) produktiviteten kan få långt sämre värden än den taktiska produktiviteten.

Både strategisk och taktisk mätning är viktig, men ger olika insikter och kunskaper. De har också skilda användningsområden och målgrupper. Strategisk mätning är av stort värde för företagets ledning, som ansvarar för företagets positionering på marknaden, kostnadsutveckling och förmåga att skapa intäkter. Taktisk mätning däremot är till mest praktisk nytta för chefer på dataavdelningar, projektledare, men även för projektanställda. Det är en hjälp att uppskatta, planera och genomföra enskilda projekt, och att skapa en utvecklingsprocess som kan anpassas till och optimeras för både företagets och de individuella projektens förutsättningar.

Väl att märka är dock att strategisk mätning inte kan utföras utan föregående taktisk mätning. En företagsledning som vill ha ordentligt underlag för sin strategiska planering och sina strategiskt viktiga beslut, måste också vara beredd att investera i och stödja taktisk mätning på projekt- och aktivitetsnivå.

3. Mått och mätmetoder

3.1. Storleksmått

Produkter från systemutveckling är datasystem med tillhörande anvisningar, dokumentation och ofta även utbildningstjänster. Det finns inget enskilt mått som ger en sammanfattning av t ex ett projekts produktionsvolym (storlek) så att alla ingående produkterna täcks. I stället bör storleken anges separat per produktkategori. Detta innebär dock inte att det inte finns mått som i allmänhet ger en bra indikation på ett projekts storlek. Den produkt eller produktkategori som är huvudingrediensen och som kräver den största arbetsinsatsen, kan vara en bra approximation av ett projekts storlek för många syften t.ex. resurs- och tidsuppskattning. Systemstorlek kan också vara en bra indikator för storleken hos ett projekt, särskilt som volymen för andra produkter/produktkategorier ofta är linjärt beroende av just systemstorleken. Exempel på produkter vars volym ofta kan relateras till systemstorlek är specifikationsdokument, konstruktionsdokument, testdatauppsättningar, användardokumentation, utbildningsmaterial etc. När det gäller att mäta produktionen från enskilda aktiviteter är det ofta enklast att använda sig av dokumentvolymer. Att mäta dokumentvolymer är också ett bra sätt att få en uppfattning om mängden "pappersarbete" i ett projekt och relatera det till systemstorlek, projekttyp och andra typer av arbete såsom programmering och felrättning.

3.1.1. Projektstorlek och systemstorlek

Två enkla och populära mått för systemstorlek, som också kan uttrycka projektstorlek vid nyutveckling, är antal kodrader (LOC, lines of code) i ett system och antal ingående program. Fördelen med kodrader som mått är att det är lätt beräkna, är konkret och objektivt. Tyvärr varierar antalet kodrader kraftigt beroende på språkval, hur en kodrad definieras, programmeringskonventioner och hur programmeraren strukturerar och löser uppgiften. Dessutom är allt större delar av datasystem realiserade utan någon egentlig kod alls (grafiska specifikationer, skärmlayouter med direkt referens till fält i databaser etc). Antal program, dvs självständiga exekverbara programmoduler, är också lätt att beräkna, men tyvärr lika oprecist som antalet kodrader och av liknande skäl.

Under det senaste decenniet har så kallade funktionsbaserade mått börjat vinna terräng. Gemensamt för dessa är att de utgår från antal funktioner som ett system består av. Till dessa hör Tom DeMarco's Bang och det alltmer populära funktionspoäng (function points) varav det finns ett antal varianter som t ex Basic function points, IFPUG function points, Mark 2 function points, feature points och Laturi function points (som alltså används för projekt i Laturi-databasen).

Enligt funktionspoängmetoden (Function Point Analysis, ofta förkortat till FPA) består ett system av en mängd användarfunktioner. Dessa funktioner, eller systemkomponenter, delas in i fem separata typer: indata, utdata, frågor, externa gränssnitt och filer (dvs dataobjekt eller grupperingar av data). Dessa vägs efter sitt informationsinnehåll, dvs hur mycket data de innehåller eller hanterar. En typisk funktion får 4-10 poäng beroende på typ. Ett medelstort ADB-system består av ca 150 funktioner med totalt ca 700 funktionspoäng. Funktionspoäng är i stort sett oberoende av teknik och metoder som används. De kan dessutom med fördel användas för uppskattning av systemstorlek i tidiga utvecklingsfaser, t ex för projektkalkyler, eftersom mycket av den information som krävs för funktionspoänganalys, finns tillgänglig i kravspecifikationer eller till och med i förstudiedokument. Nackdelen med funktionspoäng är att det inte är lika objektivt som antal kodrader utan lider av viss känslighet för vilka tolkningar och räkneregler som tillämpas i analysen. En standardisering av funktionspoäng tillsammans med tolknings- och räkneregler är dock på väg, främst genom försorg av International Function Point

User Group (IFPUG), men även genom att ISO-standardisering av funktionspoäng-metoden inletts.

Systemstorlek enligt Bang (även kallad "specification weight metrics") anges dels som ett funktionsmått kallat "function bang" och dels som datamått kallat "data bang". Function-bang beräknas utifrån antalet funktionella primitiver (bubblor) i dataflödesdiagram vilka vägs enligt typ och antal dataelement de använder. Data-bang beräknas från entiteter i datamodellen, där varje entitet vägs efter antalet ingående attribut och relationer. Det kan påpekas här att en del moderna function point -varianter för automatisk beräkning av funktionspoäng ur systemspecifikationer påminner om DeMarco's Bang-metod. Bang-metoden är dock inte särskilt spridd och tillämpas främst i USA.

Utöver ovanstående finns det flera andra sätt att mäta volym hos system, projekt eller hos ADB-avdelningars produktion, men de allra flesta har sin utgångspunkt antingen i programkod eller systemfunktioner/-komponenter.

3.2. Mått för resursanvändning

Resurser som används innebär kostnader. Dessutom påverkar relationer mellan behov av och tillgång på resurser ett projekts tidsplanering och förmåga att hålla de planerade ledtiderna.

Standardmåttet för angivelse av personresurser är persontimme (ptim) vilket motsvarar den arbetsinsats som en person kan utföra under en timme. Persontimmar kan lätt transformeras till persondagar och personmånader, dock med den varningen att en personmånad kan vara olika många effektiva persontimmar i olika företag, beroende på arbetstider och företagets interna praxis.

Det är också relativt enkelt att fastställa den genomsnittliga timkostnaden för en personresurs och därmed översätta den till kronor och ören.

En mera fullständig resursangivelse skulle även täcka åtgången av andra resurstyper såsom hård- och mjukvaruresurser. Mätning av dessa är dock i praktiken svår eftersom de ofta är gemensamma resurser för projekt och avdelningar, och för att det inte finns något enkelt sätt att mäta deras nyttjande i ett specifikt sammanhang. Dessutom upplevs det som ointressant om nyttjandet av utrustning inte innebär direkta kostnader för ett projekt. I de fall resurser direkt belastar projektets konto kan det vara intressant att mäta resursnyttjande i antal volymer eller nyttjandetimmar. Exempel på mått för hård- och mjukvaruresurser är:

- *CPU-tid* för datorkraft
- *diskvolym* (MB eller GB)
- *antal arbetsstationer*
- *antal installationer/licenser* för olika programvaror

Om resursnyttjande verkligen mäts och kostnaden per enhet för de olika resurstyperna är känd, är det enkelt att räkna fram den totala rörliga kostnad för ett projekt eller en process . Den fasta projektkostnaden så som lokaler, gemensam utrustning och gemensam administration måste i regel hanteras som schablonbelopp vilket relateras till något lämpligt projektmått (exempelvis per projekt, per projektmånad eller per personmånad). I vissa fall kan även projektets egen konsumtion av lokaler mätas utan schablontal. Måttet för nyttjande av lokaler skulle då kunna vara antal nyttjade kvadratmeter.

3.3. Produktivitetsmått

För att kunna mäta och ange produktivitet måste man först fastställa resursåtgång och produktionsvolym. Vi har redan redovisat ett antal mått för resursåtgång och produktvolymer. Om vi främst koncentrerar oss på systemstorlek som ett uttryck för produktionsvolym, har vi några alternativ för produktivitetsangivelser:

- kodrader/persontimme (rad/ptim)
- program/persontimme (prog/ptim)
- funktionspoäng/persontimme (fp/ptim)
- bang/persontimme (bang/ptim)

Mätning av ett projekts produktivitet med något av ovanstående mått innebär sålunda att ett systems storlek bedöms och att den totala mängden konsumerad persontid, inklusive administration beräknas. Detta kräver att tillräckligt underlag finns tillgängligt. Adekvat projektdokumentation och tidsredovisning är med andra ord nödvändigt för produktivitetsmätning. Bra och nyanserat underlag ger dessutom möjligheter att ta fram produktivitetstal för olika typer av projekt, delprocesser per produktkategori och per resurskategori (olika personalkategorier och icke-personella resurser).

Däremot riktar vi en varning mot att mäta produktivitet på individnivå. Dels för att det kräver mycket arbete och är kostsamt, dels för att det kan skapa en negativ stämning mot hela mätprogrammet och därmed äventyra det. Faktum är att i så gott som alla lyckade satsningar på mätning har man undvikit mätning på individnivå, medan de flesta satsningar där individer har mätts och värderats har misslyckats. Ett ytterligare argument mot mätning av individer är att den mänskliga naturen har en tendens att överskatta, underskatta, bortse från saker och ting eller ta med litet extra, allt beroende på vad det innebär för individen. Med andra ord, mätning av individer när dessa själva levererar en del av mätresultaten, eller kan påverka dem, är ett säkert sätt att skaffa sig missvisande mätresultat. Missvisande mätresultat leder till felaktiga slutsatser som leder till felaktiga åtgärder, i värsta fall med motsatta effekter än de avsedda. En lyckad etablering av ett mätprogram kräver att man tar hänsyn till "mätningens sociologi", dvs både kulturella och sociala förhållanden samt emotionella stämningar i grupper och individers sätt att förhålla sig till olika slag av värderingsinsatser .

4. Nyttjande av mätdata

Kvantitativa mått såsom systemstorlek, resursnyttjande, produktivitet, ledtider, komplexitetsnivåer och kvalitetstal kan användas som indikatorer för tillståndet i systemutvecklingsprocessen och kan ge möjligheter till förbättring av processen. Förändringsinsatser, som dessa insikter leder till, bör innefatta kontinuerlig erfarenhetsåtervinning. De bör gärna ingå som en del i ett större benchmarkingprogram[2] vilket syftar till att höja effektiviteten och kvaliteten i såväl produktionen som i hela utvecklingsverksamheten [7]. Konsekventa mätningar, analyser och jämförelser internt och med andra företag ger insikt och hjälper till att staka ut vägen till effektivare verksamhet. Även om omedelbara åtgärder ofta identifieras, särskilt i början, handlar det främst om en långsiktig satsning. Mätresultat och den kunskap de ger kan användas för effektivisering av utvecklingsprocessen inom ett antal viktiga områden. Exempel på sådana områden är kvalitetsutveckling, produktivitetsutveckling, minskning av ledtider och utveckling av metoder för projekthantering. Mätning för bättre projekthantering tas upp i ett separat avsnitt i detta kapitel.

Kvalitetsutveckling. Ett mätningsprogram kan med fördel kombineras med ett kvalitetsutvecklingsprogram som t ex Total Quality Management (TQM). Genom att identifiera mätbara kvalitetspåverkande faktorer och kvalitetsindikatorer (t ex olika komplexitetstal) kan ett mätprogram även ta sig an kvalitetsfrågor.

Produktivitetsutveckling. Det är ett välkänt faktum att kvalitetsförbättringar, som en följd av etablerad mätning i produktionen och processen, även leder till ökad produktivitet. Dessa effekter kan komma mycket snabbt efter införd mätning på grund av den ökade medvetenheten om dessa förhållanden i företaget. Till detta kommer de uppenbara vinsterna som bra mätstatistik över olika produktivitetsförhållanden ger i form av underlag för direkta produktivtetshöjande åtgärder.

Minskade ledtider. Ökad produktivitet ger förutsättningar för snabbare utveckling och därmed kortare ledtider, dock bara till en viss gräns. För att verkligen kapa ledtiderna måste ytterligare åtgärder till. Underlag till sådana får man delvis genom att mäta och analysera utvecklingsprocessen ur ledtidsperspektiv, snarare än ur rent produktivitetsperspektiv. Detta innebär att man identifierar och mäter de faktorer som påverkar ledtiderna. Det brukar dock vara svårt att minska ledtiderna dramatiskt enbart genom yttre faktorer och förbättringar i den tekniska miljön. För att nå stora vinster i ledtider krävs det ofta omfattande ändringar i sättet att driva utvecklingsprocessen och att strukturera och styra projektarbetet (t ex enligt principerna i Concurrent Engineering).

Processutveckling. Alla ovannämnda områden bidrar var för sig och tillsammans till förbättring av utveckingsprocessen. Om mätning av projekthantering och problemlösning tillämpas konsekvent och med samma principer för projekt med olika metoder och tillvägagångssätt, blir det möjligt att identifiera de mest effektiva tillvägagångssätten ("best practices"). Trender och utvecklingsvägar inom intressanta områden kan studeras i ett tidsperspektiv och förbättringsåtgärder kan sättas in och deras effekter följas upp. Mätning är en nödvändig förutsättning för ett framgångsrikt och fortlöpande förbättringsarbete.

---

[2]Benchmarking är en systematisk process för att jämföra effektiviteten, kvaliteten och ledtiderna i arbetsprocesser i den egna organisationen med andra organisationer med motsvarande förutsättningar, gärna sådana som tillhör de bästa inom de aktuella områdena. Syftet härmed är att identifiera de bästa arbetssätten och att själv tillämpa dem för att uppnå ökad effektivitet och kvalitet.

Ett exempel på detta är mätning av fel som upptäcks vid granskningar, inspektioner och tester samt fel som upptäcks och rapporteras av kunden. Dessa mätdata tillsammans med data om de tillämpade utvecklingsmetoderna (inklusive sätten att granska och testa) kan användas för att spåra felkällor och beräkna "felpotential" per felkälla. Nästa steg är att använda denna kunskap till att förutsäga och hitta fel, förebygga fel och identifiera effektiva sätt att eliminera felkällor. Resultatet är ökad förmåga att förebygga fel och ökad felelimineringseffektivitet.

4.1. Projekthantering

Kunskap om produktivitetsförhållanden tillsammans med metoder för uppskattning av systemstorlek ger automatiskt bättre möjligheter till förkalkyl av projektkostnader samt resurs- och tidsbehov. Mätningar som äger rum under genomförandet av ett projekt ger också bidrag till projektstyrning och uppföljning.

Mätning av projekt på aktivitetsnivå tillsammans med registrering av ett antal projektegenskaper ger kunskap om vilka aktiviteter som brukar förekomma i olika typer av projekt (problemdomän, systemstorlek, metodmiljö, kvalitetskrav etc). Förutsättningen är att det finns ett antal "aktivitetskonton" för mätresultat, dvs att det finns en standardiserad mängd aktivitetstyper för vilka mätdata samlas. Att enbart mäta på övergripande projektnivå räcker inte för ekonomiska studier av generella mönster i resurs- och tidsförbrukning och kostnadsutveckling. Den lägsta ambitionsnivån bör vara att åtminstone mäta på en detaljeringsnivå som ger besked om resultat, resursförbrukning, ledtider, kvalitetsdata och kostnader för:

1. Analys/kravspecifikation
2. Systemkonstruktion
3. Tillverkning (programmering)
4. Testning
5. Projektledning.

Förutom kvantitativa data för dessa "aktivitetskonton" bör ett antal projektförutsättningar registreras. Detta är en förutsättning för att man senare skall kunna analysera samband mellan projektets förutsättningar och dess utfall, med avseende på t ex resursförbrukning, ledtider, produktivitet, kvalitetsdata och kostnader. Dessa samband är nyckeln till ökad förmåga att förutsäga, planera och styra nya projekt, eftersom de utifrån kända förutsättningar ger möjligheter att med rimlig säkerhet uppskatta andra faktorer, projektets troliga utfall samt påvisa osäkerheter och risker.

Projektförutsättningar kan grovt grupperas i tre klasser:

- Produktfaktorer: Systemets storlek, komplexitet, kvalitetskrav, tillämpnings-område, livscykel (nyutveckling/förvaltning), effektivitets- och prestandakrav, antal samtidiga användare, antal installationer och utbildningsinsatser

- Processfaktorer: Projektstruktur, metod(er) för projekthantering, övergripande metod/utvecklingsmodell, metoder/tekniker och verktyg för analys, kravspecifikation, konstruktion, testning och införande, tillämpade standarder och konventioner

- Resursfaktorer: Tillgång på personal, utrustning och mjukvara, användarmedverkan under olika projektstadier, personalens erfarenhet inom tillämpningsområdet, analys/kravspecifikation, systemkonstruktion, testning och projektarbete, personalens erfarenhet/kunskap om de tillämpade metoderna och verktygen

Det kräver en viss tid och möda att mäta och samla in mätdata. För att det skall flyta smidigt krävs det i regel att det finns en fast instans som ser till att alla nödvändiga data levereras av projekten. Erfarenheter från flera företag visar dock att en välorganiserad datainsamling med etablerade rapporteringsrutiner till stor del kan automatiseras i de rutinmässiga delarna, medan mätresultatens direkta nyttoeffekter för enskilda projekt efter kort tid anses motivera det extra besväret.

Att utnyttja mätdata för effektivare projekthantering och för att skapa underlag för processförbättringar är däremot mycket enklare. Med hjälp av datorstödda verktyg för registrering av mätdata och projektegenskaper i mätdatabaser, är det idag relativt enkelt att genomföra analyser av samband mellan ett stort antal projektrelaterade faktorer och projektets utfall ur olika perspektiv. Det finns också möjligheter att direkt nyttja dessa samband för uppskattning och planering av nya projekt. Man kan även följa projektets utveckling gentemot de uppskattade parametrarna, och justera dessa efter hand.

5. Statistik om systemutveckling

Detta kapitel redovisar resultaten från två studier om systemutveckling. Den första studien har utförts av Capers Jones från Software Productivity Research (SPR) och analyserar produktivitet och kvalitet i amerikansk systemutveckling [5]. Den andra studien har genomförts av författaren och har som underlag en finsk mätdatabas, kopplad till mät- och projektuppskattningsverktyget Laturi. Laturi finns idag även i Sverige, men någon användbar svenskt mätdatabas för statistiska analyser finns ännu inte.

Orsaken till att just Capers Jones' undersökning och Laturi-databasen valts är att de båda använder funktionspoäng (function points) som basmått. Som redan nämnts är detta mått oberoende av programmeringsspråk och tekniska utvecklingsmiljöer och ger därför mycket bättre möjligheter till storskaliga statistiska undersökningar som spänner över flera branscher, utvecklingsmiljöer och systemtyper än vad kodbaserade mått någonsin kan ge. Dessutom är funktionspoängmetoden på frammarsch i hela världen och kommer med all sannolikhet att standardiseras av ISO. Intresset för funktionspoäng är mycket stort även i Sverige och flera svenska företag har redan satsat på denna metod.

Capers Jones' undersökning är också värd att redovisa på grund av sin omfattning och bredd. Den ger en god inblick i förhållanden i amerikansk systemutveckling, betydelsefull konkurrent, men också leverantör till svensk systemutveckling. Trots att Laturi-databasen för närvarande bara innehåller finska projekt, är den likväl mycket intressant ur svenskt perspektiv. Finland och Sverige är båda nordiska länder med mycket likartade förhållanden, till exempel vad gäller näringslivets struktur, lagstiftning, sociala förhållanden samt konkurrensförhållanden. Metodanvändning, arbetsmetoder och de tekniska miljöerna i systemutvecklingen är också mycket lika i de båda länderna.

5.1. Felkällor i materialet

Läsaren bör vara medveten om att projektmaterialet i de båda undersökningarna inte kan göra anspråk på extrem korrekthet. Felkällorna är flera. Den för analysen mycket viktiga storleksbestämningen av projekt kan innehålla fel. Särskilt stor är osäkerheten i den amerikanska undersökningen, eftersom en stor del av projektmaterialet inte har beräknats med funktionspoängsmetoden från början (en del projektmaterial dateras så långt tillbaka som 1950, medan funktionspoängsmetoden utvecklades i slutet av 70-talet). Projektstorlek har i många fall uppskattats utifrån antal kodrader med hjälp av statistiskt framräknade samband mellan kodrader och funktionspoäng.

En annan potentiell felkälla är osäkerheten vad gäller resursanvändning, dvs arbetstimmar som lagts ned i projekten. Här finns det flera hinder för en korrekt bestämning av arbetstid. Bristande bokföring och projektredovisning är en uppenbar försvårande faktor, medan varierande praxis vad gäller tidsrapportering och kontering av arbetsuppgifter är en annan. Det är också i många fall svårt att avgöra när ett projekt har påbörjats, eftersom den egentliga projektstarten i regel har föregåtts av inledande studier och analyser. Denna osäkerhet påverkar förutom bestämning av nedlagd arbetstid även bestämning av projektets ledtid.

Ytterligare en potentiell felkälla är statistiken i sig och dess tolkning. Trots upprepade kontroller kan fel ha smugit sig in. Allt från rena räknefel till dåligt valda statistiska metoder och algoritmer. Och även om statistiken i sig är korrekt framräknad från det aktuella underlaget, måste man hålla i minnet att underlaget i sig inte nödvändigtvis är representativt nog för att man skall kunna dra generella slutsatser om systemutveckling i allmänhet. Orsaken till detta är att projektpopulationen inte bygger på statistiskt urval, utan består av de projekt som man lyckats samla in. Statistiken här redovisar bara vad man kan få fram ur underlaget. Detta gäller särskilt Laturi-databasen, eftersom

projektunderlaget är relativt litet, 117 projekt. Det kan därmed omöjligen ge ordentlig täckning vad beträffar förhållanden i olika branscher och utvecklingsmiljöer.

Osäkerheten och alla brasklapparna till trots, är det ändå intressant att analysera och jämföra resultaten från studierna. Även om siffrorna i sig kanske inte är helt jämförbara, torde de dock ge läsaren en fingervisning om trender och allmänna förhållanden i dessa länder. Förhoppningsvis kan dessa trender och förhållanden extrapoleras till svenska förhållanden, i väntan på motsvarande svensk statistik. Trots bristerna är statistiken som presenteras här, och i andra publikationer, bättre än ingen alls. Den ger upphov till ökade insikter och fördjupad kunskap även inom svensk mjukvaruindustri, kunskap som det skulle vara svårt att inhämta på annat sätt. Förhoppningsvis stimulerar denna undersökning till andra, mer exakta och relevanta studier, som kan redovisa och korrigera brister inom svensk mjukvaruindustri. Med tiden kommer vi att få en mer pålitlig och exakt statistik. Med ökad erfarenhet och kommande standardisering, kommer även vår kunskap om mätning och statistisk analys av mätdata från systemutveckling att öka. Men fram till dess får vi göra så gott vi kan med de data och de kunskaper vi har.

5.2. The United States Software Measurement Study

I sin bok Applied Software Measurement redovisar Capers Jones resultaten från sin mycket omfattande undersökning om produktivitet och kvalitet i amerikansk systemutveckling – the United States Software Measurement Study. Undersökningen omfattar över 4000 projekt daterade från 1950 till 1990. Vi kommer här att redovisa en del av resultaten med kommentarer och slutsatser.

5.2.1. Översikt av projektmaterialet

Period	Antal projekt	Andel %
1951-1960	150	3,6
1961-1970	500	11,9
1972-1980	1000	23,8
1980-1990	2550	60,7
Totalt	4200	100,0

Tabell 2. Tidsperioder för projekt i undersökningen

Projekt typ	Nyutv.	Ändring	Felkorr.	Totalt	Andel %
System	700	1500	200	2400	57
Inf.system	350	600	125	1075	26
Inbäddad	50	150	25	225	5
Militär	75	100	0	175	4
Telekomm.	30	50	75	155	3
Processtyrn.	40	100	0	140	2
Vetenskapl.ig	50	50	0	100	2
Artif. Intelligens	20	10	0	30	1
Totalt	1315	2560	425	4300	100

Tabell 3. Fördelning av projekt mellan projekttyper.

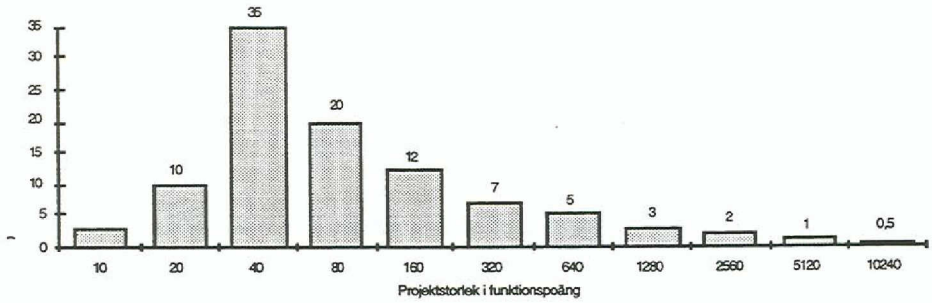
(Tabellen korrigerad av oss efter felaktighet i källan.)

Materialet i den amerikanska undersökningen spänner över nästan 40 år av systemutveckling. Tabell 2 redovisar fördelningen av projekt över de olika tidsperioderna. Som framgår av tabellen är ca 40 procent av projekten över tio år gamla. Detta bör man ta hänsyn till när man drar slutsatser från en del av den presenterade statistiken. Vidare gäller att angivna medelvärden är aritmetiska medelvärden av motsvarande värden för enskilda projekt om inte annat uttryckligen anges.

Förutom nyutveckling omfattar insamlade projektdata även förvaltningsprojekt. Förvaltning har i denna undersökning delats in i systemändringar (dvs tillägg och modifieringar) och felkorrigeringar. En sammanställning av volymer för de olika projekttyperna, nyutveckling, ändringar och felkorrigeringar visas i tabell 3. System-projekt och informationssystemprojekt har en bra täckning inom områden för nyutveckling, ändringar och felkorrigeringar. Statistiken visar att över 80 procent av alla projekt i undersökningen är just systemprojekt eller informationssystemprojekt och att de flesta projekt handlar om ändringar i befintliga system. Av 4200 projekt handlar 2460 (59%) om ändringar och 425 (10%) om felkorrigering medan endast 1315 (31%) är nyutveckling.

Tabell 3 kan dock inte utan vidare ses som en generell fördelning, utan avser endast det undersökta projektmaterialet. Capers Jones själv uppskattar att 26% av alla projekt i USA handlar om systemjukvara, 52% om informationssystem (vid tolkning) och 22% är militära projekt. Vidare har han kommit fram till att en realistisk uppskattning är att 41% av all systemutveckling i USA är nyutveckling, 45% systemändringar och 14% felkorrigering. Förvaltning, inklusive både systemändringar och felkorrigering, utgör enligt honom alltså ca 60% av all systemutveckling, samtidigt som andelen förvaltning stadigt ökar.

Enligt en försiktig uppskattning är ca 4 miljoner av totalt 7 miljoner programmerare idag sysselsatta med systemförvaltning i hela världen. År 2000 bedöms 6 miljoner av totalt 10 miljoner programmerare huvudsakligen arbeta med systemförvaltning medan 4 miljoner ägnar sig åt nyutveckling. Dessa siffror är givetvis bara grova uppskattningar, men budskapet är dock att systemförvaltningen blir en allt viktigare faktor i systemutveckling, om ingen radikal ändring inom informationsteknologin gör att merparten av alla gamla system snabbt måste ersättas med nya.

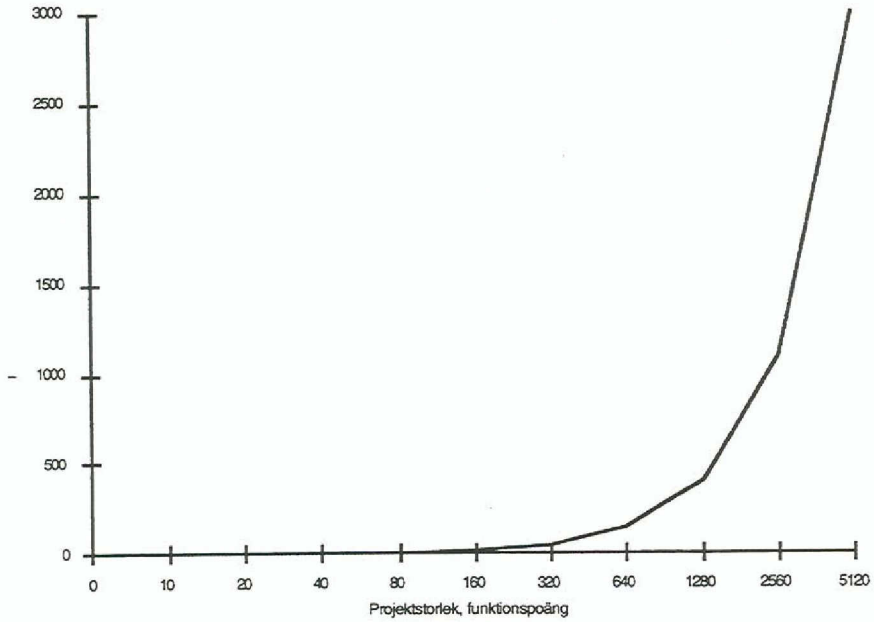


Figur 1. Fördelning av projekt mellan olika projektstorlekar.

Hur projekten fördelar sig mellan olika storleksklasser visas i figur 1. Observera att skalan på storleksaxeln inte ökar linjärt utan med en fördubbling av systemstorlek. Vad som förefaller överraskande är att projekten är jämförelsevis små, de flesta mindre än 100 funktionspoäng. Om man betänker att en typisk användarfunktion enligt funktionspoängmetoden ger 4-5 funktionspoäng, så innebär det att de merparten av de undersökta projekten innehåller färre än 20 funktioner, eller att funktionerna är mycket enkla, dvs ger få funktionspoäng. Uttryckt på annat sätt: Om en funktionspoäng motsvarar drygt 100 rader COBOL, ger 100 funktionspoäng ett projekt om ca 10 000 rader COBOL-kod. Detta innebär dock inte att de flesta datasystem i USA är så små. Förklaringen ligger i att projektmaterialet i undersökningen till stor del består av förvaltningsprojekt (se tabell 3), dvs ingrepp i redan existerande system, som definitionsmässigt bör vara mindre än nyutvecklingsprojekt där all funktionalitet måste skapas från början.

5.2.2. Resursnyttjande, ledtider, personal och produktivitet

Innan vi med hjälp av diagram börjar studera samband och tendenser i projektmaterialet, presenterar vi, i tabellform, en översikt . Tabell 4 sammanställer projektmaterialet med projektstorlek som utgångspunkt och redovisar sedan motsvarande procentuell andel av alla projekt, persontid i personmånader, ledtid i kalendermånader, personal i heltidsekvivalenter, genomsnittlig arbetstilldelning i funktionspoäng per person och produktivitet i funktionspoäng per personmånad. Tabellen omfattar projekt inom områdena systemmjukvara, militära system och informationssystem. Man bör iaktta en viss försiktighet innan man drar slutsatser för specifika situationer från tabellens data, eftersom materialet är så omfattande. En analys där projekten har delats upp i olika systemtyper och branscher skulle förmodligen resultera i helt andra genomsnittsvärden (vi kommer senare se ett exempel på detta vid analys av projekt från Laturi-databasen).



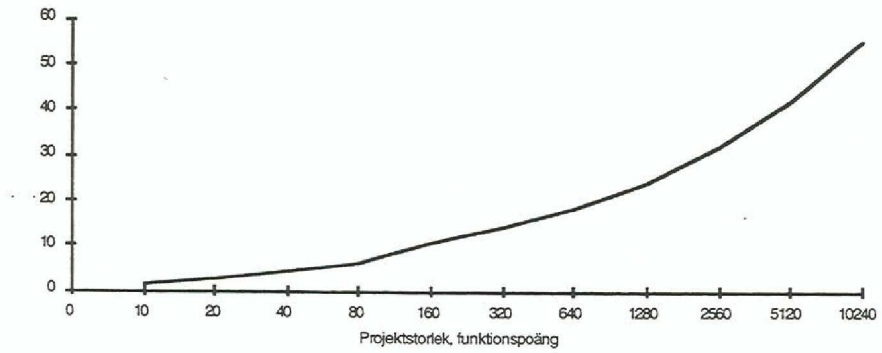
Figur 2. Sambandet mellan projektstorlek i funktionspoäng och resursanvändning i personmånader.

Observera att axeln för projektstorlek i ovanstående diagram inte är linjär utan att storleksvärden fördubblas för varje steg.

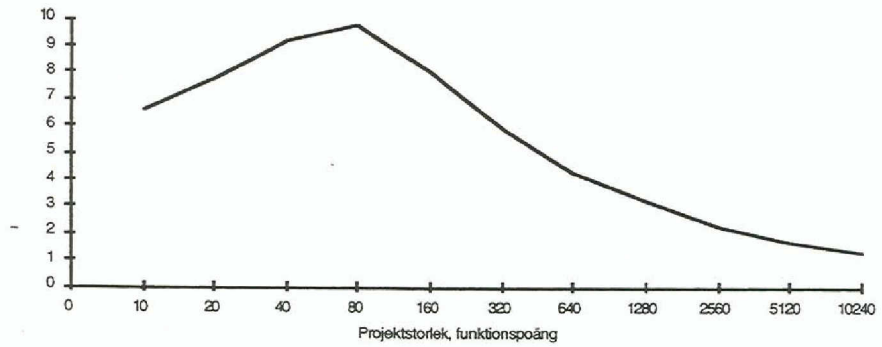
Storlek	Andel %	Pers.mån	Ledtid	Personal heltidsekv	Arbetstilldelning	Produktivitet
10	3	1,5	1,5	1	10	6,6
20	10	2,6	2,6	1	20	7,8
40	35	4,4	4,5	1	40	9,2
80	20	8,1	6	1,4	64	9,8
160	12	20	11	2	80	8,0
320	7	55	14	4	80	5,9
640	5	149	18	8	80	4,3
1280	3	405	24	27	75	3,2
2560	2	1099	32	34	78	2,3
5120	1	2994	42	71	72	1,7
10240	0,5	8192	55	149	69	1,3

Tabell 4. Resursanvändning, ledtider och personalstorlek, arbetstilldelning och produktivitet för olika projektstorlekar i Capers Jones stora undersökning från 1990.

Figur 2 visar ett diagram som illustrerar sambandet mellan projektstorlek och resursanvändning i form av personmånader. Inte oväntat ökar resursbehovet i takt med projektstorleken. Företag som inte bokför och konterar alla projektrelaterade aktiviteter på själva projekten, utan betraktar många av dem som allmän stödverksamhet eller konterar dem under andra rubriker, kan tycka att resursanvändningen förefaller väl stor. Det är då lämpligt att tänka på hur mycket persontid som "försvinner" från projektkonton pga konteringsprinciperna.



Figur 3. Sambandet mellan projektstorlek och ledtid.



Figur 4. Sambandet mellan projektstorlek och produktivitet för alla typer av projekt.

Figur 3 visar sambandet mellan projektstorlek och ledtid. Även här gäller att ledtiden ökar i takt med projektstorleken. Ledtiden ökar dock inte alls lika brant som resursbehovet. Faktum är att ökningen av ledtiden börjar avta när systemen/projekten börjar blir riktigt stora, även om detta inte riktigt framgår ur diagrammet på grund av att storleksaxeln inte är linjär. En naturlig orsak till detta avtagande är att det finns gränser för acceptabla ledtider, samtidigt som flera aktiviteter kan utföras parallellt vid stora projekt där ett stort antal personer är inblandade.

En mycket intressant bild framträder när man relaterar projektproduktiviteten till projektets storlek. Som vi redan tidigare sett ökar resursanvändning med ökad projektstorlek, men systemstorlekens betydelse för ett projekts effektivitet får ett mer dramatisk innebörd när man studerar produktivitetsutvecklingen vid ökad projektstorlek. Figur 4 visar hur produktiviteten varierar med systemstorlek. Det visar sig att det finns en optimal projektstorlek runt 80 funktionspoäng, när genomsnittet räknas för både nyutveckling och förvaltning. Mindre projekt drabbas av att diverse arbetskrävande, men nödvändiga aktiviteter, har en proportionellt större andel av resursförbrukningen än aktiviteter som direkt bidrar till att framställa den slutliga produkten. Exempel på nödvändiga aktiviteter vilka kräver både tid och arbete utan att öka produktens volym, är diverse projektadministration, ständiga omkompileringar, länkningar och testkörningar. Detta

gäller särskilt för felrättning, ändring och tillägg av enstaka funktioner vilka måste in i det ramverk som det befintliga systemet utgör och som inte får påverka gamla funktioner eller tillföra nya fel och bieffekter.

Stora system å andra sidan lider av ökad komplexitet, vilket omfattar projekthantering och kommunikationsproblem. Ytterligare en förklaring till varför produktiviteten drastiskt sjunker för stora projekt finns att hämta i projektorganisationen. Stora projekt som inte delas upp i delprojekt blir lätt oöverskådliga. Kommunikationsvägarna blir ohanterliga vilket gör projekten mycket svåra att styra och följa upp. Uppdelning av projekt bör dock göras med omtanke och hänsyn till systemets struktur. Ofta anpassas ett projekts struktur till organisationens utformning: Projekt delas upp i delprojekt. Abetsuppgifter ges till olika avdelningar och projektgrupper efter deras formella roll, status och etablerade revirgränser inom företaget. Detta kan få ödesdigra konsekvenser i de fall då ansvaret för systemkomponenter med inbördes beroendeförhållande tilldelas projektgrupper som har kommunikations- och samarbetssvårigheter. Konsekvensen blir att delprojekt arbetar utifrån sina egna förutsättningar och åsikter utan att vara överens om de övergripande riktlinjerna och utan förmåga att samordna sitt arbete. Problem uppstår och stämningen i projektet försämras snabbt. Alla skyller på alla och ingen vill ta ansvar för svårigheter och misslyckanden.

Ett projekts komplexitet bör hållas nere så långt det går. Detta går att göra genom att etablera välavgränsade delprojekt som i sig har låg komplexitet. Samtidigt bör projektstrukturen förbli enkel och delprojekten vara beroende av varandra. I sin book Software Reliability Handbook från 1990 påvisar P. Rook [9] de principiella likheterna mellan systemstruktur och projektstruktur och de gemensamma sunda riktlinjerna som detta medför:

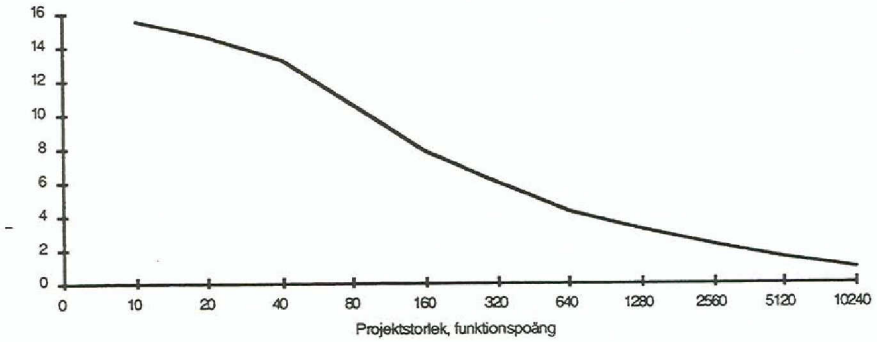
Systemstruktur	Projektstruktur
Varje systemkomponent bör vara så liten och enkel att den lätt kan förstås	Varje projektgrupp bör vara så liten att den lätt kan styras och kontrolleras
Varje systemkomponent bör vara så självständig som möjligt och endast löst kopplad till andra komponenter	Varje projektgrupp bör i görligaste mån tilldelas en arbetsuppgift som är oberoende av andra projektgruppers arbete och som minimerar onödig kommunikation mellan projektgrupper
Delarna i varje systemkomponent bör vara väl sammanhängande och på ett sammanhängande sätt bidra till komponentens roll i systemet	Varje projektgrupp bör tilldelas arbetsuppgifter som är sammanhängande och bidrar till projektgruppens gemensamma uppgift
Systemkomponenter som ömsesidigt påverkar varandra bör falla under gemensam kontrollerande komponent	Projektgrupper bör grupperas under gemensam ledning så att beslut som fattas av en ledningsgrupp har minsta möjliga effekt på det arbet som andra ledningsgrupper svarar för.
Kommunikationslänkar som inte följer systemets logiska och fysiska struktur (tvärs över hierarkier och komponentgränser) bör undvikas eller åtminstone fullständigt dokumenteras .	En organisation bör inte vara hänvisad till att lita till godtycklig och slumpvis kommunikation mellan projektgrupper och individer för att få reda på konsekvenserna av designbeslut.

Dessa likheter tyder på att strukturen hos ett system som följer dessa principer också är en bra utgångspunkt för att strukturera projektet. Om en projektgrupp svarar för en väl sammanhängande och oberoende komponent, innebär det också att projektgruppens arbetsuppgifter är sammanhängande och oberoende av andra projektgruppers arbete på motsvarande sätt. Dessa principers betydelse ökar i takt med att projektstorleken ökar, och visar på nödvändigheten av att strukturera, planera och styra projektet och delprojekt.

5.2.3. Jämförelse mellan nyutveckling och förvaltning

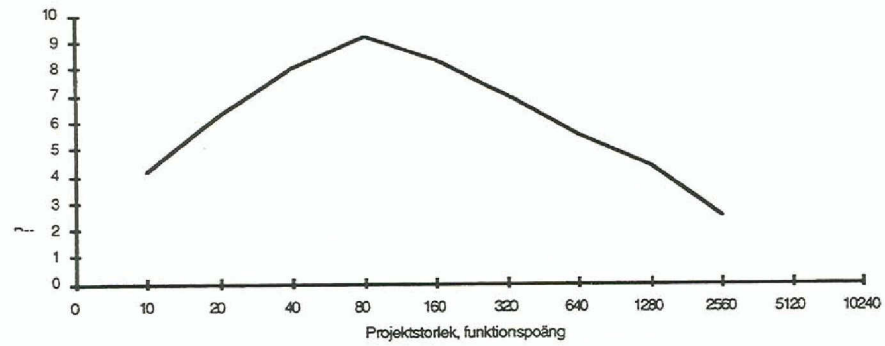
Eftersom den amerikanska undersökningen omfattar både nyutveckling och förvaltning, är det intressant att studera likheter och skillnader mellan dessa ur produktivitets-

perspektiv. Figur 5 visar produktiviteten för projekt som handlade om nyutveckling. Kurvan har ingen "puckel" som skulle tyda på någon optimal projektstorlek för nyutveckling. Detta förefaller något förvånande. Antingen har små projekt i USA ingen administrativ apparat som drar ner deras relativa produktivitet eller också har sådan uppgifter inte registrerats (varmed statistiken på denna punkt skulle vara missvisande). Dessutom känns inte siffrorna för nyutvecklingsprojekt på 10-80 funktionspoäng särskilt trovärdiga. Vilka system har så få funktionspoäng (redan ett normalt dataobjekt (t ex en enda tabell) med 20 termer ger 10 poäng)? Är det rimligt att de räknas som ett eget projekt vilket följs upp och bokförs med tillräcklig noggrannhet för att ge någorlunda rättvisande produktivitetsangivelser?

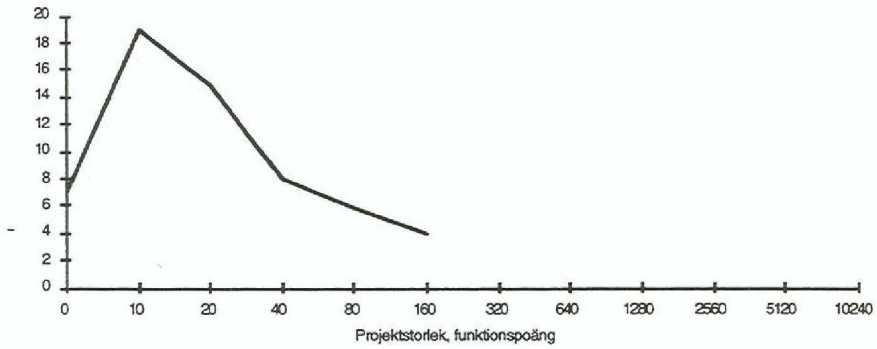


Figur 5. Produktivitetskurva för nyutveckling

Produktivitetskurvan för systemförbättringar i figur 6 visar liknande beteende som kurvan för alla projekt sammantagna i figur 4. Ändringsprojekt är mest produktiva då projektstorleken är runt 80 funktionspoäng. Här ser man också tydligare att produktiviteten för små ändringsprojekt sjunker, faktiskt mera dramatiskt än för stora projekt om man tar hänsyn till den icke-linjära skalan på storleksaxeln. Orsakerna till detta är förmodligen just att små projekt påverkas av närvaron av aktiviteter som kräver tid och resurser, men inte bidrar till ökad produktionsvolym, vilket vi konstaterade ovan. Stora ändringsprojekt påverkas av den ökade komplexiteten och den svårhanterliga projektapparaten, men dessutom av den komplexitet som det befintliga system medför. Dock förefaller stora ändringsprojekt vara mera produktiva än motsvarande stora nyutvecklingsprojekt. Förklaringen kan vara att när nya funktioner och komponenter läggs till finns det bättre möjligheter att återanvända algoritmer, procedurer och mekanismer som redan implementerats i det befintliga systemet (detta kräver givetvis kunskap om systemet och ordentlig dokumentation).



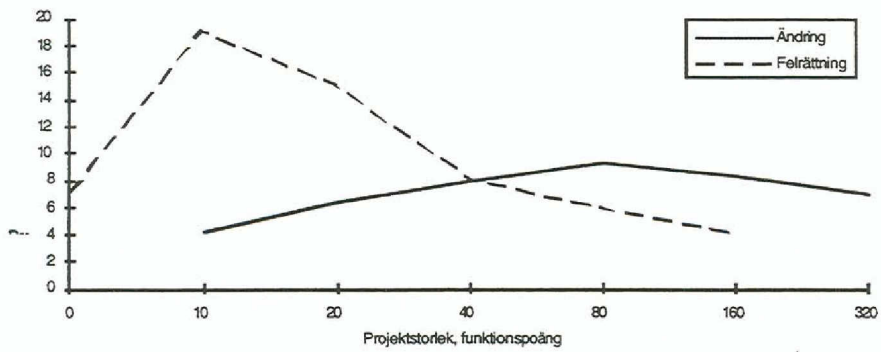
Figur 6. Produktivitetskurva för systemförbättring



Figur 7. Produktivitetskurva för underhåll (felkorrigering)

Produktiviteten för felrättning ser mer dramatisk ut, vilket framgår av diagrammet i figur 7. Den optimala produktiviteten uppnås redan för mycket små insatser, cirka 10 funktionspoäng. Detta verkar rimligt. Ju mindre ingrepp som behöver göras för att rätta ett fel, desto enklare är det. Att de allra minsta rättningarna är mindre produktiva har sin troliga förklaring i att det trots allt tar en viss tid att identifiera och hitta det inrapporterade felet hur litet det än är. Att produktiviteten för mer omfattande felrättning sjunker så hastigt känns också naturligt. Ju fler ställen som blir föremål för ingrepp och ändringar, och ju fler samband och beroenden man måste hålla reda på, desto mer mödosamt blir arbetet. Risken för att plantera in nya fel ökar också kraftig, vilket leder till en ökad andel testningar, där ofta hela systemet måste testas och inte bara de delar som modifierats.

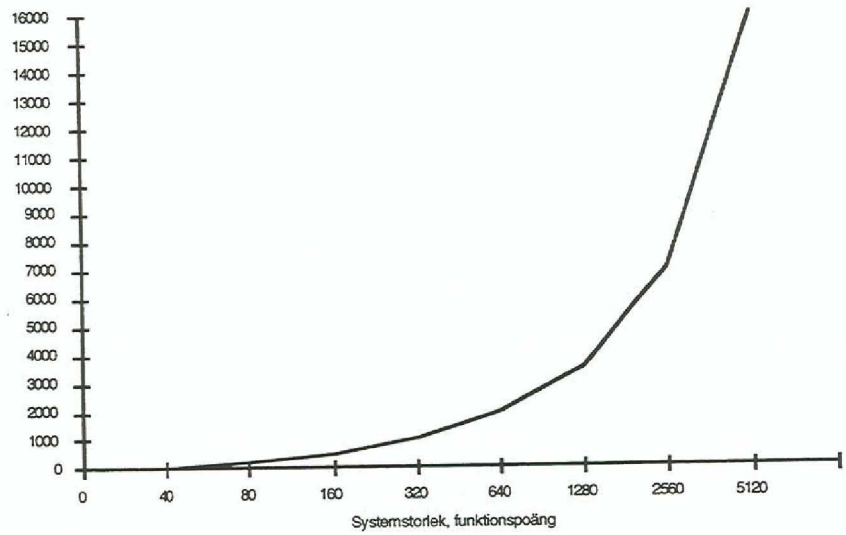
Man kan fråga sig hur rimligt det är att göra direkta produktivitetsjämförelser mellan nyutveckling och felrättning. Att utveckla något från början är inte direkt jämförbart med att reparera det. Problemställningen och arbetssituationen är helt annorlunda. Dessutom producerar felrättning ingenting, utan modifierar något som redan finns, men inte fungerar. Vem skulle komma på idén att jämföra produktiviteten 20 bilar/personmånad för en biltillverkare med produktiviteten 60 bilar/personmånad för en reparationsverkstad, och säga att verkstaden är mer produktiv. Funktionspoäng har olika betydelser i nyutveckling och i förvaltning, åtminstone vad gäller dess förhållande till resursanvändningen. Att använda produktivitetsstatistik för kalkyler och planering av nya projekt kräver att statistiken avser samma typ av projekt. Figur 8 visar hur missledande det skulle vara att använda statistiken för ändringsprojekt från figur 6 för resurs- och tidsuppskattning av ett förvaltningsprojekt av storleken 20 funktionspoäng.



Figur 8. Produktivitet för ändringar respektive felrättningar.

5.2.4. Kvalitetsrelaterad statistik

Diagrammet i figur 9 visar hur risken för fel och defekter i ett system ökar i takt med systemstorleken. Alla väsentliga felkällor såsom kravspecifikationer, designdokument, programkod, användardokumentation och dåligt utförda felrättningar är inräknade. Felidentifiering och rättning är en av de dyraste och svåraste delarna i systemutveckling, om man räknar in hela utvecklingscykeln. Det är därför mycket värdefullt att känna till potentiella fel, vilket deras ursprung är och hur de kan hittas. Det är dessvärre svårt att få fram bra statisk över felförekomster, som kan användas för analyser av potentiella fel i projekt. Det kräver att alla upptäckta fel inrapporteras med uppgifter om var och hur de upptäckts. Orsak, frekvens och karaktär måste bokföras tillsammans med uppgifter om projektets och felkällans karaktär. Med statistisk analys av historiska feluppgifter kan felpotential i ett projekt förutsägas, samtidigt som de mest defektutsatta delarna och felkällorna kan identifieras innan projektet påbörjas. Därmed får man också de bästa förutsättningarna att förebygga fel och att tidigt hitta och eliminera uppkomna fel. Ett känt faktum är att ett fel blir dyrare att åtgärda ju senare det upptäcks. Allvarliga fel som upptäcks först efter leveransen kan vara mycket dyra att åtgärda. Dessutom orsakar de problem för kunden, vilket i sin tur leder till minskad kundtillfredsställelse.

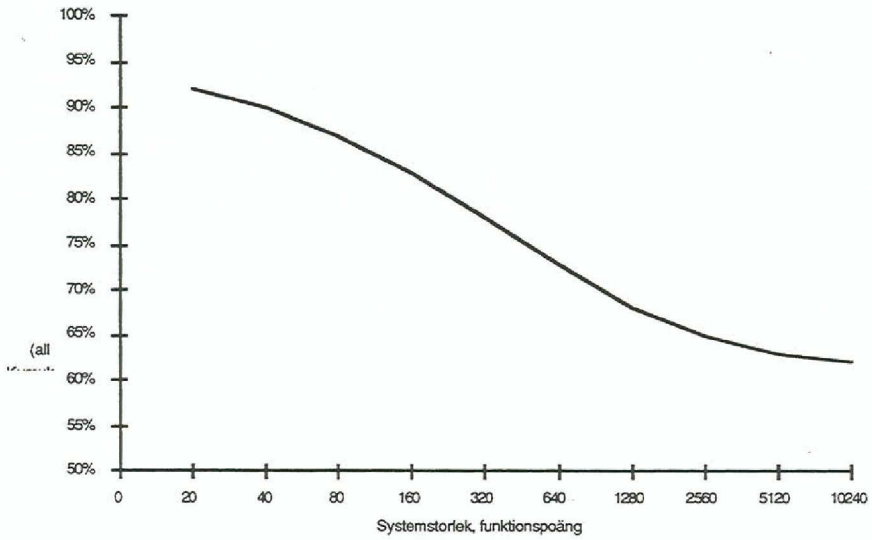


Figur 9. Antal potentiella fel när systemstorleken ökar.

Eftersom det är av stor vikt att fel upptäcks och åtgärdas i tid är felelimineringseffektiviten en viktig faktor i ett företags utvecklingsprocess. Felelimineringseffektiviten definieras här som andelen fel åtgärdade före leverans, jämfört med det totala antalet fel upptäckta under utvecklingstiden eller av kunden under systemets första år i drift.

Figur 10 visar amerikansk statistik som tyder på att närmare 95 procent av alla fel kan elimineras i mycket små projekt, medan i mycket stora projekt närmare 40 procent av alla fel riskerar att bli oupptäckta. Genomsnittlig amerikansk felelimineringseffektivitet förefaller vara ca 75 procent, vilket innebär att 25 procent av alla fel finns kvar vid leveranstillfället.

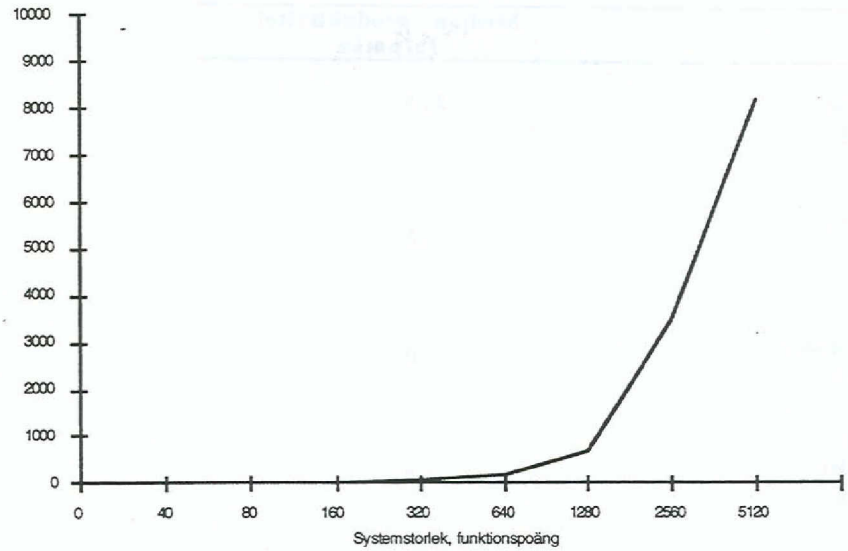
Det är intressant att notera att felelimineringseffektiviteten inte sjunker under 60 procent ens för mycket stora projekt. En orsak till detta är givetvis att system med stora felvolymer inte kan tas i drift överhuvudtaget, kunden accepterar det helt enkelt inte. Projektet läggs antingen ned, eller också måste man åtgärda felen, vilket leder till sämre produktivitet och ökade ledtider. En starkt bidragande orsak till sämre produktivitet i stora projekt (se figur 4) är förmodligen att just risken för fel, och därmed arbetsinsatsen för feleliminering, ökar med systemstorleken.



Figur 10. Andel upptäckta och eliminerade fel då systemstorlek ökar. kumulativt efter alla felelimineringsåtgärder såsom granskningar, kodinspektioner och testning.

Tillsammans med förmågan att förebygga fel, är felelimineringseffektiviteten den absolut viktigaste parametern för slutproduktens kvalitet, och därmed också en avgörande faktor för kundtillfredsställelse. Det kräver stora insatser och hårt arbete för ett företag att uppnå en genomsnittlig felelimineringseffektivitet på 95 procent, men det är möjligt. De ledande företagen inom området har uppnått en genomsnittlig felelimineringseffektivitet som till och med ligger över 95 procent. Att eliminera så många fel som möjligt är dock inte alltid lika viktigt som att eliminera de allvarligaste felen, dvs fel som leder till driftsavbrott eller allvarliga felfunktioner. Många fel leder inte ens till anmärkningsvärda kvalitetsproblem. I en studie av ett antal datasystem utförd av E. Adams från IBM [1] visade det sig att ungefär en tredjedel av alla upptäckta fel var sådana som ledde till driftsavbrott endast var 5000:e år i genomsnitt (eller ännu mera sällan). Med tanke på att ett normalt datasystems livstid sällan är längre än 15 år, är ett genomsnitt på 5000 år mellan driftsavbrott en acceptabel nivå för icke-kritiska system. Från kvalitetsperspektiv och med tanke på kundtillfredsställelse är det mest lönsamt att sätta resurserna på att hitta och åtgärda fel som med stor sannolikhet leder till driftsavbrott under systemets livstid.

Figur 11 visar kurvan för antal dolda fel vilka finns kvar då produkten tas i drift. Ett normalt system på ca 600 funktionspoäng har ca 100 dolda defekter vid leverans. Observera återigen att dessa defekter omfattar all produktion inklusive kravspecifikationer, designdokument, mjukvara och användardokumentation. Dessa fel finns alltså vid leverans och kan i värsta fall orsaka produktionsbortfall hos kunden. I takt med att felvolymen ökar, minskar användartillfredställelsen och kundrelationerna kan bli ansträngda.



Figur 11. Antal fel som finns kvar vid leverans relaterat till systemstorlek.

5.2.5. Produktivitetspåverkande faktorer

Det finns givetvis flera faktorer än bara projektstorlek som påverkar kvaliteten och produktiviteten. Andra faktorer som är relaterade till projekts inneboende natur är projekttyp (nyutveckling eller förvaltning), tillämpningsområde, problemens karaktär etc. Dessa faktorer ligger ofta utanför projektets egen kontroll och tillhör förutsättningarna. Andra faktorer, som är lättare att påverka, har att göra med själva utvecklingsprocessen. Exempel på sådana faktorer är tidsplaner, projektledning och projektorganisation, tillgång på personal och materiella resurser, kvaliteten hos metoder och verktyg, olika typer av krav på systemet såsom prestanda, svarstider och antal samtidiga användare samt personalens erfarenhet och kompetens.

Den kanske allra viktigaste faktorn för ett projekts framgång har att göra med projektorganisation och projektledning. Bra projektledning och en väl trimmad projektorganisation gör att arbetet flyter smidigt, vilket har positiva effekter på arbetsmoralen, vilket i sin tur förstärker den positiva arbetsmiljön. På motsvarande sätt kan dålig projektledning och illa anpassad projektorganisation ge upphov till förvirring och konflikter i kommunikationen mellan projektmedlemmar och projektgrupper. Arbetet försenas och tidspressen sänker moralen, vilket leder till ökad irritation och ytterligare press tills man når en nivå där produktiviteten drastiskt sjunker.

Projektledning och projekthantering är så avgörande faktorer att de måste ges första prioritet och behandlas separat från tekniska aspekter som t ex frågor om metoder och verktyg. Det är ett erkänt faktum att sättet att driva ett projekt och hantera utvecklingsprocessen kan betyda skillnaden mellan succé och katastrof. I jämförelse med detta framstår metod- och teknikaspekter som marginella. Detta innebär givetvis inte att de saknar betydelse. Tvärtom är de, tillsammans med personalens kompetens, avgörande för skillnader i produktivitet när projekthanteringen i övrigt är likadan. Det innebär däremot att det oftast lönar sig att först investera i en effektiv projekthantering, bra riktlinjer för projektorganisationen och en välutformad utvecklingsprocess, snarare än att börja med att investera i olika tekniska hjälpmedel och metoder. Den tekniska utvecklingsmiljön kommer bäst till sin rätt i händerna på en effektiv och kompetent projektorganisation.

Faktorer	Median produktivitet fp/pmån
Oerfaren personal Ostrukturerade metoder Medelmåttiga verkyg Lågnivåspråk	*2,5*
Oerfaren personal Ostrukturerade metoder ***CASE-verktyg*** Lågnivåspråk	3,5
Oerfaren personal ***Strukturerade metoder*** Medelmåttiga verkyg Lågnivåspråk	4,0
Erfaren personal Ostrukturerade metoder Medelmåttiga verkyg Lågnivåspråk	4,5
Oerfaren personal Ostrukturerade metoder Medelmåttiga verkyg ***Högnivåspråk***	5,0
Erfaren personal ***Strukturerade metoder*** ***CASE-verkyg*** ***Högnivåspråk***	***40***

Tabell 5. Median produktivitet analyserad separat för fyra produktivietsfaktorer.

Förvånande nog visar Capers Jones' undersökning (tabell 5) att det finns tekniska faktorer som betyder mera än personalens kompetens. Ett exempel på en sådan faktor är programmeringsspråket. Högnivåspråk ger större produktivitetsbidrag än personalens kompetens, m a o är projekt med oerfaren personal, men med högnivåspråk, i regel mer produktiva än projekt med erfaren och kompetent personal, men med lågnivåspråk. Detta överraskande resultat förklaras med att personalens arbetskapacitet och kompetens sugs upp av dålig språkmiljö (tiden går till att lösa programmeringsproblem), medan högnivåspråk, med större inbyggt kunskapsinnehåll och högre funktionalitet, kräver mindre av personalens kunskaper och kapacitet, som då frigörs för mer kreativa och produktiva arbetsuppgifter. Andra produktivitetspåverkande faktorer är tillämpade metoder och verktygsmiljö.

Observera den enorma produktivitetsskillnaden mellan det sämsta och bästa fallet. Väl värt att notera är också rangordningen mellan faktorerna: språk, personalens erfarenhet, metoder och verktyg. Den stora skillnaden mellan det mest gynsamma fallet och övriga gör det svårt att hävda att den tekniska miljön bara har en marginell effekt. Effekten behöver inte vara lika dramatisk som i tabellen, då faktorerna i det optimala fallet kanske är just vad som är vanligt förekommande i organisationer med kompetent projektledning och trimmad projektorganisation. I det sämsta fallet, där personal är oerfaren och metoderna ostrukturerade, kan man knappast heller vänta sig en trimmad projektorganisation.

5.2.6. Risk för misslyckande och nedläggning av projekt

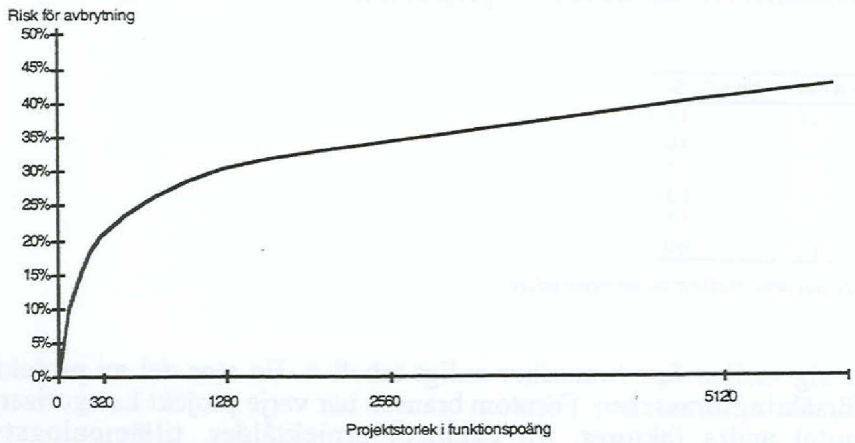
Ett projekt som misslyckas med att leverera resultat och läggs ned bidrar inte till företagets (leverantörens) produktionsvolym. Däremot bidrar det till att binda resurser och generera kostnader. Den kanske viktigaste långsiktiga effekten av misslyckade projekt är återverkningarna på leverantörens anseende och förmåga att knyta till sig nya uppdrag. En kund som förgäves investerat stora belopp och resurser i ett datasystem som inte kunnat

levereras, blir mindre benägen att anlita samma leverantör i andra sammanhang. I de fall då en produkt skall utvecklas för den öppna marknaden, kan ett misslyckat projekt betraktas som en investering som inte lett till de förväntade intäkterna. Misslyckade projekt har stor inverkan på ett företags lönsamhet och konkurrensförmåga.

Risken att misslyckas är en strategisk faktor. Som redan tidigare påpekats i samband med taktisk och strategisk mätning, skall alla insatser, inklusive misslyckade och nedlagda projekt, vägas in vid strategisk mätning av till exempel produktiviteten. Eftersom misslyckade projekt bidrar till resursförbrukning utan motsvarande bidrag i produktionsvolym, har de en direkt negativ effekt på ett företags strategiska produktivitet (företagets sammanlagda årliga produktionsvolym per sammanlagd resursförbrukning). Produktionsvolym genererar intäkter (om man utgår från att allt kan säljas) och därmed även lönsamhet, medan resursförbrukning genererar kostnader. En hög produktivitet i *framgångsrika* projekt räcker inte för att garantera hög strategisk produktivitet. Företagets lönsamhet är alltså direkt relaterad till den strategiska produktiviteten. Konkurrenter med ungefär samma kostnad per resursenhet, men med bättre strategisk produktivitet, är mer lönsamma, och har också bättre förutsättningar att konkurrera prismässigt.

Slutsatsen är att riskanalys är en strategisk framgångsfaktor även vid systemutveckling. Det är viktigt att veta vilka uppdrag man kan åta sig med en rimlig riskbild. Risk för att misslyckas i ett stort projekt innebär också en risk för lönsamheten och på längre sikt en risk för konkurrensförmågan.

Figur 12 visar hur risken för misslyckande och nedläggning förhåller sig till projektstorleken. Närmare hälften av alla mycket stora projekt avbryts. Andra oberoende amerikanska undersökningar visar att för stora system- och försvarsprojekt är andelen misslyckanden generellt över 25 procent. Informationssystemprojekt inom näringslivet är oftast mycket mindre och löper därmed mindre risk att misslyckas. Å andra sidan är privata företag oftast utsatta för hård konkurrens och har andra ekonomiska utgångspunkter, vilket leder till att även lägre risknivåer ter sig oacceptabla.



Figur 12. Samband mellan projektstorlek och risk för misslyckande och avbrytning av projekt.

Det finns flera skäl till att ett projekt läggs ned. En faktor som oftast ligger utom projektets kontroll, är förändringar i verksamheten. Det kan gälla fusioner och företagsuppdelningar, eller att en avdelning, ett områdeskontor eller ett dotterbolag läggs ned. Den vanligaste interna faktorn för nedläggning, brukar vara allvarligt överskridande av budget- och tidsramar. I visionära eller banbrytande projekt är orsaken ofta att projektets ambitioner legat utanför vad som är rimligt eller ens möjligt med tillgänglig teknik eller utanför gränserna för projektpersonalens tekniska kunnande.

5.3. Statistik från Laturi-databasen

Följande avsnitt presenterar statistik som bygger på en finskt projektdatabas som är kopplad till verktyget Laturi. Laturi ger funktionspoängbaserat metod- kalkylstöd för projektmätning och projektuppskattning. Databasen innehåller för närvarande 117 projekt från Finland och sträcker sig över tidsperioden 1985-1992. Projekten täcker flera branscher och tekniska miljöer.

Alla produktivitetstal anges som funktionspoäng per personmånad. Personmånad räknas här som 150 persontimmar. Det förutsätts vidare att all tid som ägnas åt arbete i projektet också bokförs på projektet (dessvärre finns här en uppenbar risk för slarvig eller bristfällig bokföring i projektmaterialet). Alla medelvärden är aritmetiska medelvärden av motsvarande värden för enskilda projekt om inte annat uttryckligen anges.

Systemstorlek enligt Laturis variant av funktionspoäng är inte direkt jämförbar med IBM:s version som tillämpats i den amerikanska undersökningen. Laturi ger färre funktionspoäng för små och enkla system och fler funktionspoäng för system med stora och komplexa funktioner. Omräknat i IBM-funktionspoäng blir storleken för Laturi-projekt generellt 15-20% mindre. All statistik från Laturi-databasen som presenteras i denna skrift uttrycks i Laturi-funktionspoäng. Om någorlunda jämförbarhet med IBM-funktionspoäng och den amerikanska undersökningen önskas, bör ca 20% dras av från storleks- och produktivitetsangivelserna i Laturi-statistiken. Vi uppmanar dock till försiktighet vid direkta jämförelser mellan den amerikanska undersökningen och Laturi-materialet, på grund av olikheterna i underlagen och osäkerheten i deras korrekthet och representativitet för systemutveckling i allmänhet. Eftersom projektmaterielat i Laturi-databasen inte bygger på statistiskt urval och eftersom antalet projekt ännu är ganska litet, kan man inte dra några hållbara generella slutsatser.

5.3.1. Översikt av Laturi-databasen

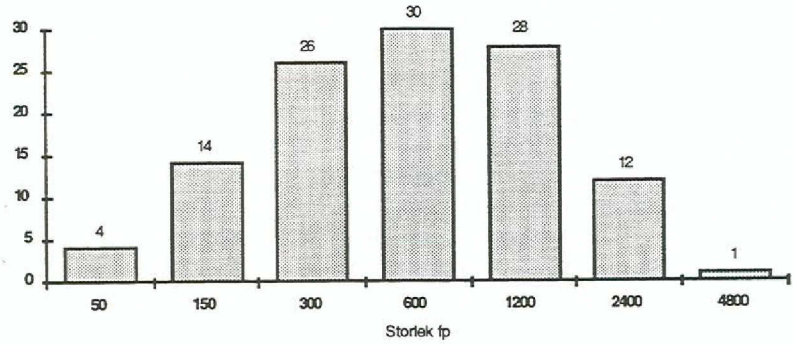
Laturi-databasen innehåller för närvarande 117 projekt från mer än 30 företag.

Projekt typ	Antal	Andel %
Bank	16	14
Försäkring	47	40
Handel	15	13
Industri	27	23
Offentlig sektor	12	10
Totalt	117	100

Tabell 6. Fördelning av projekt mellan olika branscher

Projekten fördelar sig mellan fem branscher enligt tabell 6. En stor del av projekten härstammar från försäkringsbranschen. Förutom bransch har varje projekt kategoriserats enligt ett stort antal andra faktorer, till exempel projektålder, tillämpningstyp, metodanvändning, verktygsstöd, maskinmiljö, projekthanteringsmetoder och personalens erfarenhet inom olika områden. Utöver dessa finns uppgifter om projektstorlek, systemens fördelning i olika typer av funktioner, funktionernas storlek, projektens resurs- och tidsförbrukning per utvecklingsaktivitet (analys/kravspecifikation, konstruktion, tillverkning, testning och införande) samt bedömning av ett antal produktivitetspåverkande faktorer.

5.3.2. Systemstorlek



Figur 13. Fördelning av projekt mellan olika projektstorlekar (uttryckt i Laturi-funktionspoäng).

Storleken på projekten i Laturi-databasen varierar från några tiotal funktionspoäng till över 3000 funktionspoäng. Figur 13 illustrerar fördelningen av projektstorlek i ett antal storleksklasser. Observera att storleksaxeln i diagrammet inte är linjär utan att storleken fördubblas för varje steg. Som synes i diagrammet, är den vanligaste projektstorleken ungefär 600 Laturi-funktionspoäng (eller ca 480 IBM-funktionspoäng). Det är intressant notera att den vanligaste projektstorleken i den amerikanska undersökningen är mellan 40 och 80 funktionspoäng.

En viktig förklaring till denna stora skillnad är att projekten i Laturi-databasen är nyutvecklingsprojekt medan en stor del av det amerikanska underlaget består av förvaltningsprojekt (inklusive systemförbättringar), vilka ju i regel är små. Ytterligare en bidragande orsak kan vara att de flesta av de amerikanska projekten är mycket äldre än Laturi-projekten. Även om det inte framgår i någon statistik, är det troligt att projekt från 1950-1970 är mindre än dagens projekt. Enbart dessa två skäl kan dock knappast förklara den stora skillnaden i projektstorlek. Sannolikt spelar projektens typ och ursprung också roll. Till skillnad från den amerikanska undersökningen täcker Laturi-databasen endast några få branscher och systemtyper.

5.3.3. Resursnyttjande, ledtider, personal och produktivitet

Tabell 7 visar ett resultat från en analys av Laturi-databasens innehåll. Projekten har delats upp i storleksklasser på så sätt att ett projekt hamnar i den storleksklass som är närmast projektets egen storlek i funktionspoäng. Dessutom har ett antal projekt i varje storleksklass angivits för att läsaren själv skall kunna ta ställning till de redovisade siffrornas representativitet. Det framgår klart att projektunderlaget inte är tillräckligt omfattande för några pålitliga analyser och generella slutsatser. Det är dock intressant att studera trenderna och jämföra dem med det amerikanska materialet.

Storlek fp	Antal	Resurser pers.mån	Ledtid mån	Personal heltids ekv	Arbets till-delning fp/pers	Produk-tivitet fp/pmån
50	4	5	5	1	56	11,3
150	14	12	8	1,5	103	12,1
300	26	19	10	1,8	165	15,8
600	30	42	20	2,1	291	14,0
1200	28	88	25	3,6	337	12,7
2400	12	168	32	5,3	450	11,8

Tabell 7. Resursanvändning, ledtider och personalstorlek, arbetstilldelning och produktivitet för olika projektstorlekar för projekt i Laturi-databasen.

Laturis variant av funktionspoängmetoden ger generellt sett cirka 20 procent fler funktionspoäng är IBM:s version som använts för den amerikanska statistiken. För någorlunda jämförbarhet bör alltså projektstorlek och alla siffror som utgår från projektstorlek justeras nedåt med 20 procent. Tabell 7 visar resultatet då projektstorleken nedjusterats och tabellen räknats om med samma storleksklasser som för den amerikanska statistiken i tabell 4 . Tyvärr redovisar inte Capers Jones hur han har delat in projekten i storleksklasser. Här har projekten tilldelats den storleksklass (40 fp, 80 fp, 160 fp etc) som ligger närmast projektets egen storlek. Ett annat alternativ hade varit att betrakta storleksklasserna som övre intervallgränser. På så sätt skulle storleksklass 80 fp omfatta projekt från 41-80 fp, storleksklass 160 skulle omfatta projekt från 81-160 fp osv. En kontrollräkning visar dock att siffrorna för resursförbrukning, ledtider och produktivitet och trender blir mycket lika oberoende av vilket alternativ som väljs. Alternativet för storleksklasser har ingen större betydelse för de analyser och jämförelser som görs här.

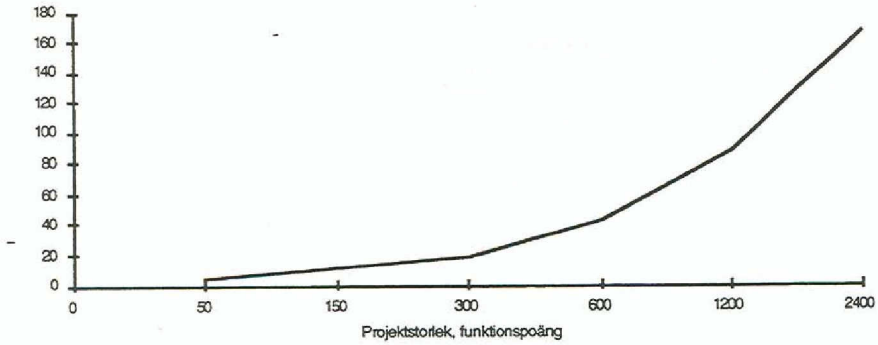
Storlek fp	Antal	Resurser pers.mån	Ledtid mån	Personal heltids ekv	Arbets-till-delning fp/pers	Produkt-ivitet fp/pmån
40	2	7	6	1,2	25	5,3
80	7	14	8	1,6	56	7,6
160	17	12	9	1,4	119	13,6
320	27	30	14	2,1	150	10,8
640	25	42	20	2,1	285	13,4
1280	33	118	28	4,2	305	9,4
2560	3	223	32	6,9	420	10,1

Tabell 9. Siffrorna i tabell 7 omräknade så att storleken och resten av värdena anpassats till den funktionspoängmetod och de storleksintervaller som använts för den amerikanska statistiken i tabell 4.

För små och mycket små projekt är siffrorna högst osäkra på grund av det ringa antalet projekt. Trendskillnaderna är att stora finska projekt relativt sett har mycket mindre resursförbrukning och bemanning än motsvarande amerikanska projekt. Projekt i den amerikanska undersökningen i storleksklassen 1280 funktionspoäng har flera gånger så stor resursförbrukning som motsvarande projekt Laturi-databasen: 405 personmånader mot 118 personmånader. Detta ger givetvis återverkningar i produktivitetssiffrorna, som för Laturi-projekt överlag ligger en bra bit över de amerikanska.

Med tanke på de stora skillnaderna i resursförbrukning, bemanning och arbetstilldelning är ledtiderna förvånansvärt lika för USA och Laturi-projekt. Trots sämre produktivitets-siffror tycks amerikanerna kunna hålla ledtiderna nere genom att tilldela projekten ordentligt med resurser.

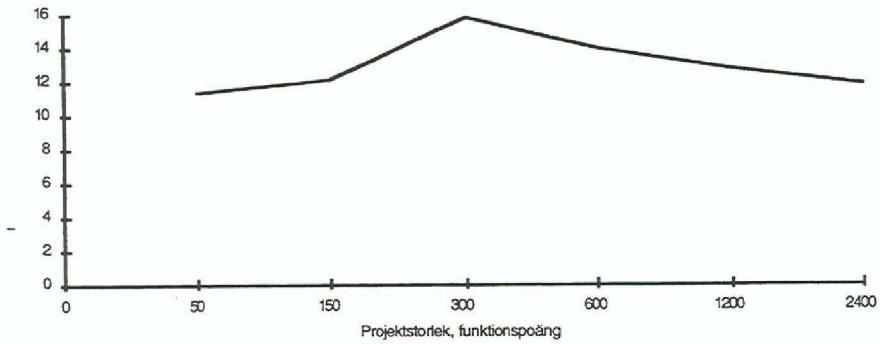
I följande material kommer alla storleksangivelser och till dessa relaterade mått att anges i originalform, dvs enligt Laturis funktionspoängmetod om inte annat anges. Orsaken till detta är helt enkelt att en schablonmässig omräkning till IBM-funktionspoäng ytterligare bidrar till siffermaterialets osäkerhet.



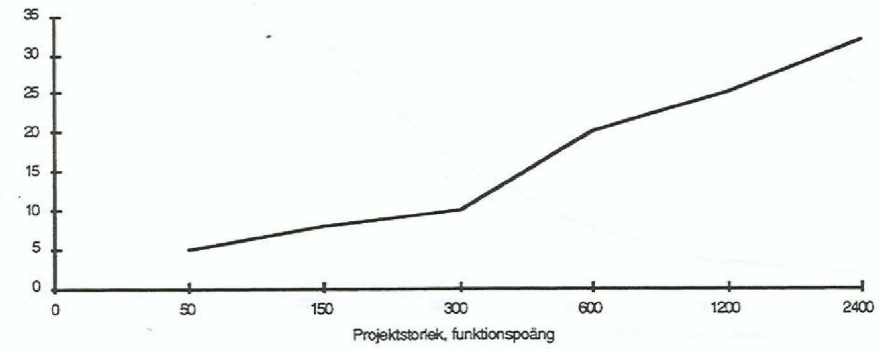
Figur 14. Sambandet mellan projektstorlek i funktionspoäng och resursanvändning i personmånader.

Diagrammen i figur 14, 15 och 16 visar hur resursförbrukning, produktivitet och ledtid förhåller sig till projektstorleken (uttryckt i Laturi-funktionspoäng). Som synes liknar trenderna de för amerikanska projekt, även om resursförbrukningen inte stiger lika våldsamt för stora projekt. Med tanke på att Laturi-projekt gäller nyutveckling, är det intressant att notera att produktivitetskurvan markant skiljer sig från amerikansk nyutveckling. Snarare liknar den kurvan för amerikanska ändringsprojekt (se figurerna 5 och 6). Det verkar finnas en optimal projektstorlek ur produktivitetsperspektiv. Optimum ligger kring 300 Laturi-funktionspoäng enligt diagrammet (eller någonstans mellan 160 och 320 IBM-funktionspoäng enligt tabell 7).

En undersökning utförd i England av C R Symons [11] bekräftar tesen att det finns en optimal systemstorlek. Även Symons kom fram till att optimum för nyutveckling ligger någonstans kring 300 funktionspoäng. Dessa upptäckter ger anledning att, vid initiering av nya projekt, begrunda lämplig projektstorlek och uppdelning av system till lämpliga, någorlunda självständiga, delsystem som kan utvecklas i delprojekt. Lämplig storlek för system och delsystem är en viktig faktor även ur kvalitetsperspektiv. Den amerikanska undersökningen visar att risken för fel vid leverans ökar kraftigt med systemstorleken, från några tiotal fel vid ca 300 funktionspoäng till flera tusen fel vid ca 2500 funktionspoäng (figur 11).



Figur 15. Sambandet mellan projektstorlek och produktivitet.



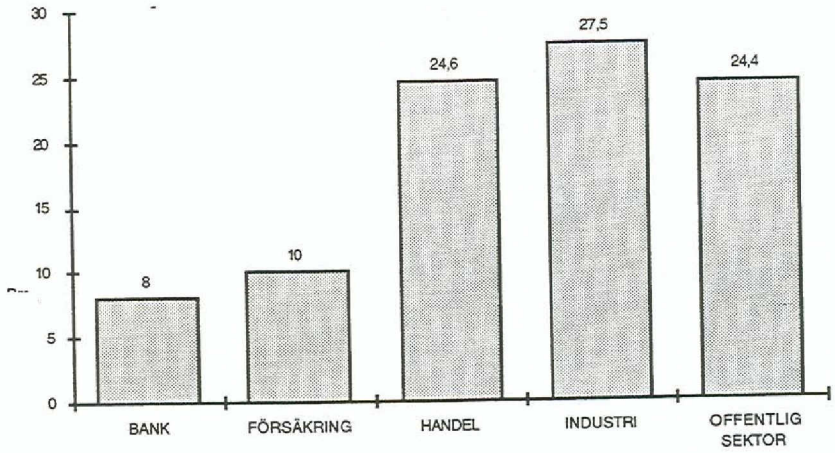
Figur 16. Sambandet mellan projektstorlek och ledtid.

Produktiviteten för stora projekt sjunker, men inte alls så brant som i den amerikanska statistiken. En viktig orsak till den flackare kurvan för finska projekt finner vi i skillnaderna i projektbemanning. Med ökad projektbemanning ökar också koordineringssvårigheterna och därmed risken för kommunikationsproblem mellan individer och projektgrupper. Genom att hålla bemanningen nere och projektgrupperna små, ökar man möjligheterna för bättre produktivitet. Låg bemanning förefaller inte heller medföra ökade ledtider, om man utgår från att trenderna med avseende på ledtider i den amerikanska respektive finska statistiken stämmer.

5.3.4. Produktivitet för olika utvecklingsmiljöer

I det följande redovisas produktivitetstal med avseende på faktorer vilka kan antas påverka produktiviteten i systemutvecklingsprojekt. Redovisningen är uppdelad i två delar: Faktorer som främst berör utvecklingsmiljön och faktorer som har att göra med systemets målmiljö.

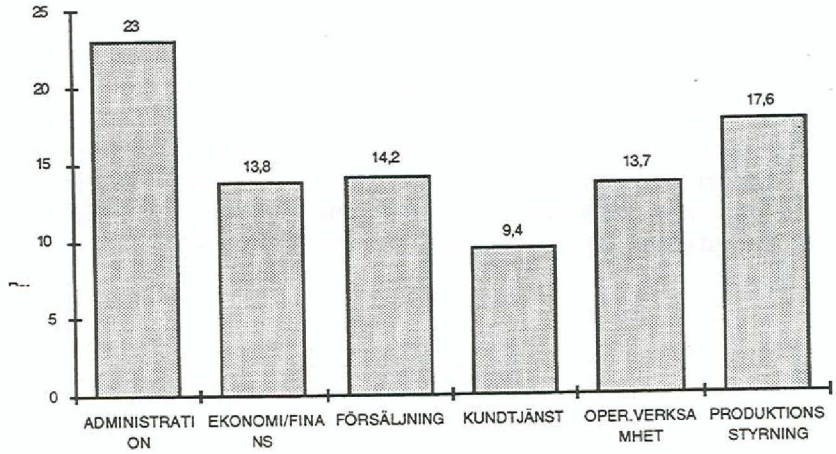
Det är inte tillrådligt att utan vidare extrapolera generella produktivitetstal på miljöer med speciella förutsättningar. Figur 17 bekräftar detta. Det finns stora skillnader i produktivitetsnivåer för projekt från olika branscher. Därmed inte sagt att vissa branscher är i motsvarande grad *sämre* än andra. Exempelvis har banker och försäkringsbolag höga säkerhets- och kvalitetskrav, omfattande testverksamhet med realistiska testdata, hundratals, ibland tusentals systemanvändare, enorma transaktionsvolymer etc. Sådant har givetvis återverkningar på produktiviteten. Men som diagrammet visar är skillnaden ändå förvånansvärt stor. Måhända spelar även utvecklingsmiljön roll, typiska datormiljöer är stordatorer, DB/2 och COBOL. I alla avseenden förefaller banker och försäkringsbolag ha stor potential till produktivitetsvinster. Den offentliga sektorn har ofta beskyllts för dålig produktivitet i många sammanhang, men i systemutveckling verkar det inte stämma (däremot tar projekten ofta mycket lång tid).



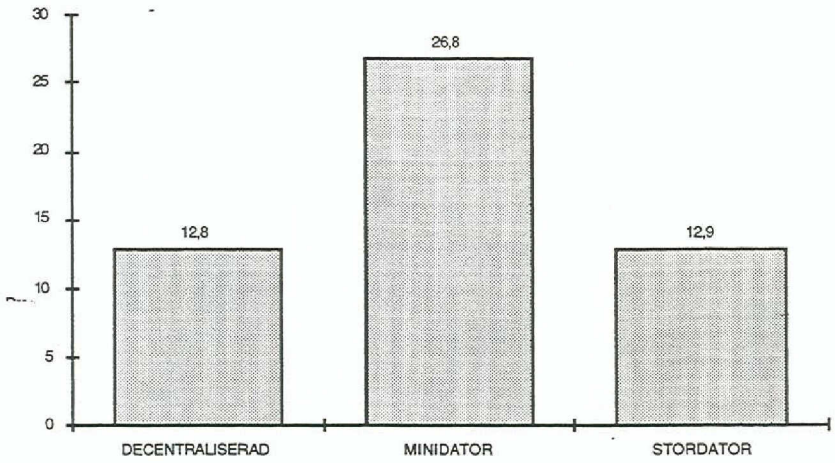
Figur 17. Medelproduktivitet i olika branscher i funktionspoäng per personmånad.

Det finns också skillnader i produktivitet beroende på vilken typ av system man betraktar. En uppdelning av system utifrån deras ändamål ger en produktivitetsbild enligt figur 18.

Kundhanteringssystem har lägst produktivitet, vilket kanske inte är förvånande eftersom kundbegreppet i sig är svårt, och för att det finns mycket varierande krav på vad man vill veta om sina kunder. Administrativa datasystem köps ofta in som standardsystem för att sedan anpassas. Dessutom är den administrativa världen relativt homogen, vilket kanske förklarar den höga produktiviteten.



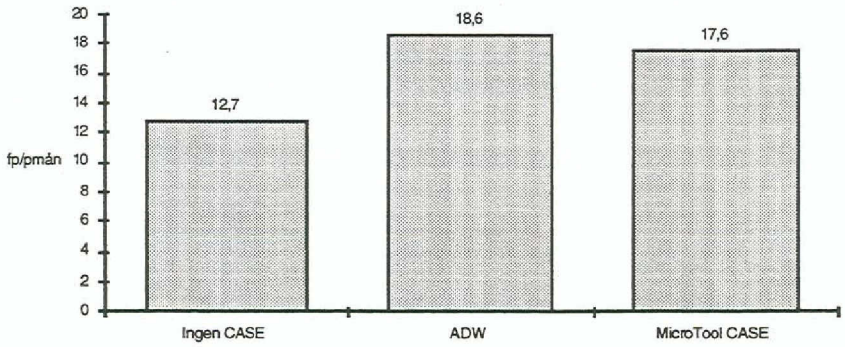
Figur 18. Medelproduktivitet för olika tillämpningstyper i funktionspoäng per personmånad.



Figur 19. Medelproduktivitet för olika datormiljöer i funktionspoäng per personmånad.

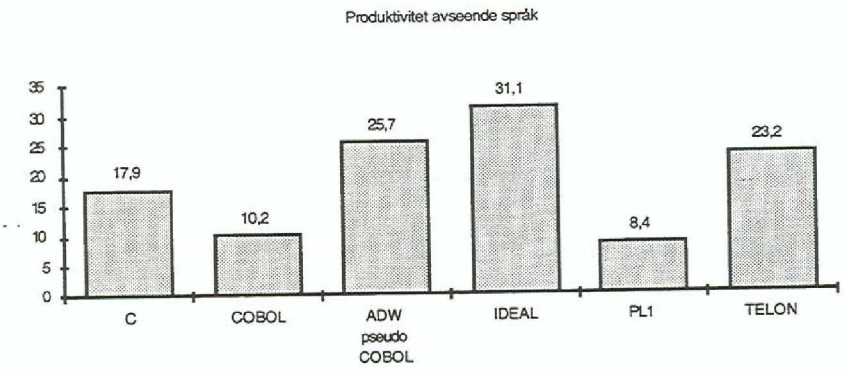
Det påstås ofta att produktiviten i persondatormiljöer är högre än i stordatormiljöer. Diagrammet i figur 19 ger en fingervisning om hur olika maskinarkitekturer påverkar produktiviteten. Tyvärr var antalet persondatorprojekt för litet för att kunna presenteras här, men de projekt som fanns registrerade i Laturi-databasen hade en genomsnittlig produktivitetsnivå på drygt 40 funktionspoäng per personmånad. Trenden är enligt Laturi-materialet att stordatorer inte förefaller befrämja hög produktivitet, medan en persondatormiljö eller eventuell något i mellanklassen ("minidatorer") presterar bättre. Decentralisering, inklusive nätverksarkitekturer får i denna undersökning nästan lika dålig produktivitet som traditionella stordatormiljöer, kanske p g a den ökade komplexiteten en sådan miljö ger.

Det är dock viktigt att ta hänsyn till att det inte enbart är maskinmiljön som orsakar dessa produktivitetstal. Det finns en tydlig korrelation till andra faktorer. Det är ofta vissa typer av företag och branscher som har vissa typer av maskinmiljöer. Även typen av system som utvecklas och de utvecklingshjälpmedel som används i dessa miljöer påverkar. Det gäller alltså att betrakta produktiviteten ur flera perspektiv om man vill komma fram till korrekta slutsatser. I Laturi-materialet är det till exempel företrädesvis banker och försäkringsbolag med sina stora och komplexa system som använder stordatorer, DB/2 och COBOL, varför det är omöjligt att avgöra enskilda faktorers betydelse.



Figur 20. Medelproduktivitet med eller utan CASE-användning - i detta fall när ADW eller MicroTool CASE använts.

En majoritet av stordatorsystem utvecklas mestadels i COBOL medan PC-baserade system i regel tas fram med modernt CASE- och 4GL-stöd. Figur 21 bekräftar misstanken att den dåliga produktiviteten i stordatormiljöer inte bara beror på stordatorarkitekturen. Språket spelar tydligen också en roll.



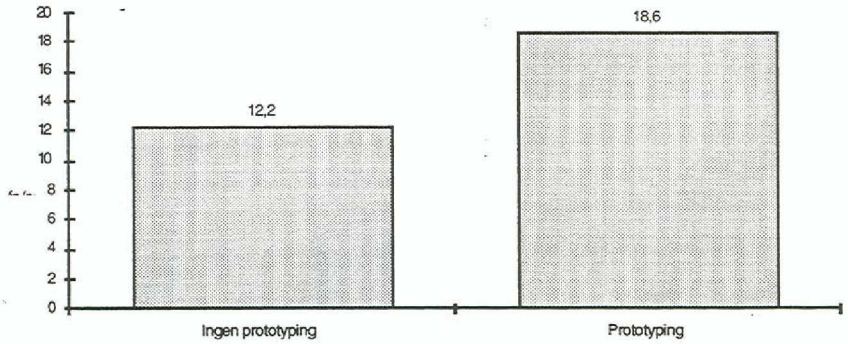
Figur 21. Medelproduktivitet för olika programmeringsspråk i funktionspoäng per personmånad.

Föga överraskande är högnivåspråk mer produktiva, ju högre nivå desto bättre. CASE-verktyget ADWs pseudo-COBOL och 4GL-språken Telon och IDEAL visar på höga produktivitetstal jämfört med lågnivåspråken COBOL och PL1. Värt att notera är att moderna högnivåspråk oftast används för utveckling i just PC-datorer, arbetsstationer och minidatorer vilka enligt ovan har högre produktivitet.

Systemutveckling är inte bara programmering, utan i allt ökande grad analys, specifikation och systemkonstruktion, särskilt vad gäller stora projekt. CASE-verktygen har marknadsförts som produktivitetshöjande hjälpmedel. En del skribenter och även forskare har på senare tid ansett att CASE-verktygen inte riktigt infriat sina löften om ökad produktivitet, även om kvaliteten och dokumentationen har förbättrats. I Laturi-databasen finns ca 30 CASE-projekt. Diagrammet i figur 20 ge en bild av CASE-verktygens betydelse för produktiviteten. Bara två verktyg finns med här eftersom de andra verktygen hade använts i för få projekt.

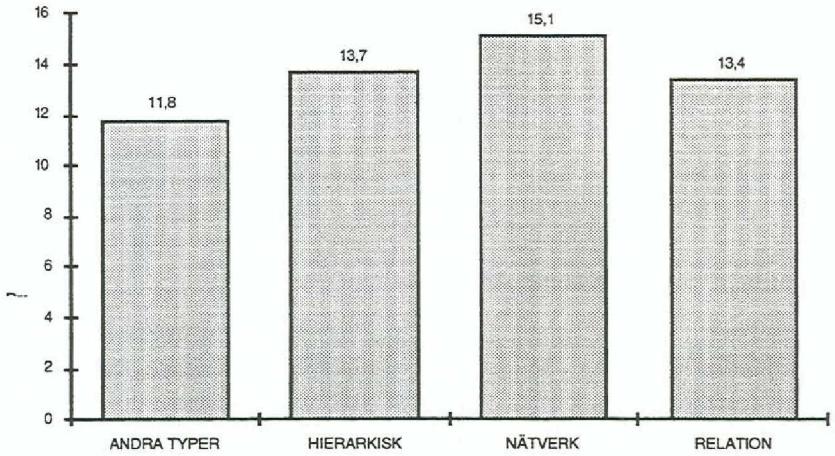
Av diagrammet att döma ger användning av CASE-verktyg en produktivitetsökning, här över 40%. En bidragande faktor kan dock vara att CASE-användning ofta hänger ihop med metodanvändning. Att tillämpa metodiskt arbetssätt och standarder i dokumentation har i sig en produktivitetsfrämjande effekt vilket CASE-verktygen sedan förstärker.

Under det senaste decenniet har prototyping ofta diskuterats och dagens metoder har i många fall anpassats för det arbetssätt som prototyping kräver. 4GL- och CASE-verktygen underlättar prototyping även om principen också går att tillämpa med lågnivåspråk . Prototyping, eventuellt i kombination med lämpligt verktygsstöd påvisar här en radikal effekt på projektproduktiviteten, vilket diagrammet i figur 22 visar. Prototyping-projekt har ca 50% högre produktivitet än projekt som inte tillämpar prototyping. Trots osäkerheten i det underliggande projektmaterialet är produktivitetsskillnaden så stor att man med fog kan påstå att prototyping har en klar produktivitetshöjande effekt.

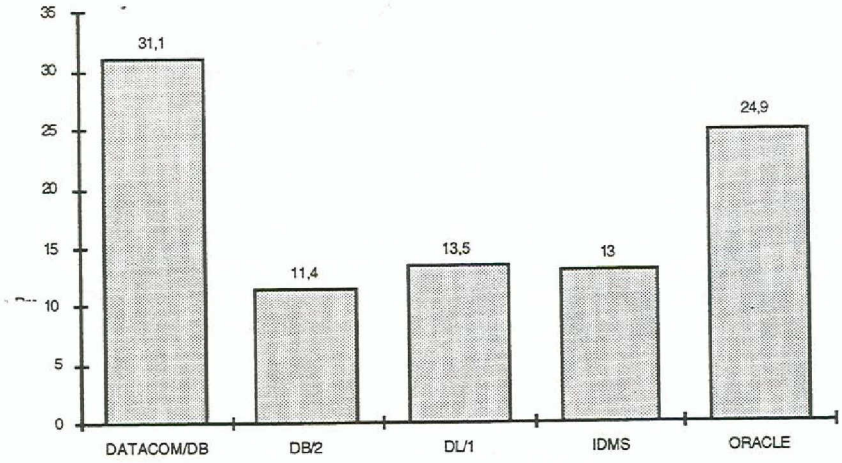


Figur 22. Medelproduktivitet för projekt med och utan prototyping.

Databashanteraren är en viktig faktor för varje informationssystem. Med den följer också en databasarkitektur: hierarkisk, relation etc. Databasarkitekturen avgör till stor del hur informationen skall struktureras och hanteras. Databashanteraren skall underlätta och även automatisera en del av hanteringen, så att programmeraren slipper koda in alla detaljer. I dag är relationdatabashanterare kanske vanligast, men det finns fortfarande en hel del hierarkiska databaser i användning. Figurerna 23 och 24 ger Laturi-materialets produktivitetsnivåer för olika databasarkitekturer och databashanterare.



Figur 23. Medelproduktivitet för olika databasarkitekturer i funktionspoäng per personmånad.



Figur 24. Medelproduktivitet för olika databashanterare i funktionspoäng per personmånad.

Förvånande nog, är alla arkitekturer nästan likvärdiga ur produktivitetsperspektiv. Särskilt kan man notera att relationsdatabaser, åtminstone inte här, framstår som mer produktivitetsfrämjande än andra. Däremot finns det en tydlig skillnad mellan databashanterare. Ytligt sett förefaller det till och med vara så att DB/2 drar ner betyget för relationsarkitekturen i allmänhet, åtminstone om man vill se Ingress och Oracle som representativa företrädare för relationsdatabashanterare. Med litet eftertanke inser man dock att det är något förhastat att göra en sådan bedömning, eftersom just DB/2 tillsammans med COBOL är standard i stordatormiljöer (se påverkan av datormiljö och språk ovan) och, som vi tidigare sett i branscher med låg produktivitet. Om den låga produktiviteten främst beror på DB/2 eller på någon annan faktor eller på en kombination av vissa faktorer går dock inte att bedöma utifrån Laturi-materialet.

6. Slutord

Denna rapport har behandlat mätning av systemutveckling som ett medel för ökad effektivitet och produktivitet, minskade ledtider och bättre kvalitet i produkter och utvecklingsprocess. Framgångsrik mätning kräver en etablerad utvecklingsprocess, dvs en ordentligt beskriven mängd av aktiviteter, metoder och arbetssätt som företaget använder för att utveckla och förvalta programvara, men också tillhörande produkter tillsammans med en beskrivning av hur dessa förhåller sig till varandra i praktiskt arbete (observera att någon enhetlig och konsistent beskrivning knappast kan göras av en ad hoc-mässig utvecklingsverksamhet utan praxis och gemensamma riktlinjer). Mätning uppmuntrar även till att förbättra utvecklingsprocessen genom att lyfta fram dess styrkor och svagheter.

Den ökade kvaliteten i utvecklingsprocessen har flera orsaker. Den kanske viktigaste kvalitetshöjande faktorn är en ökad medvetenhet om vilka fel som finns och vad som orsakar dem. Denna medvetenhet har som omedelbar effekt att de som utvecklar fäster sin uppmärksamhet på sådant som bidrar till brister i kvaliteten. En annan faktor är att medvetenhet tillsammans med kunskap från analyser och jämförelser av olika praxis hjälper till att bygga in förebyggande och eliminerande åtgärder i utvecklingsprocessen. Effekten av dessa åtgärder kan enkelt följas upp genom fortsatt mätning.

På motsvarande sätt bidrar mätning till ökad produktivitet. Konkret information om produktiviteten i egna projekt, tillsammans med information om vilka faktorer som påverkar produktiviteten, och vilka av dessa som projektdeltagarna själva kan påverka, gör att utvecklarna själva kan ta itu med produktivitetshämmande förhållanden. En annan effekt av mätning är att en företagsspecifik mätdatabas kan byggas och utnyttjas för djuplodande analyser av faktorer och förhållanden som påverkar den egna produktiviteten. Med hjälp av lämpligt verktygsstöd kan mätdatabasen utnyttjas för att analysera hur produktiviteten varierar beroende på projektförutsättningar. Denna kunskap gör det enkelt att öka precisionen i projektkalkyler och att identifiera och minska potentiella risker. Den bidrar även vid styrning av projektet under arbetets gång. Med andra ord, utvecklingsprocessen blir mera förutsägbar och styrbar.

Ofta hör man uttalandet att ökad kvalitet medför högre produktivitet och minskade ledtider. Men det är att dra förhastade slutsatser om orsak och verkan. Det är sant att företag som satsat på kvalitet också fått ökad produktivitet och minskade ledtider. Det innebär emellertid inte att det är den ökade kvaliteten i sig som ger de andra förbättringarna. I stället har effekterna i kvalitet, produktivitet och ledtider samma grundorsak: Mätning och den kunskap som mätning ger. Kvalitetssatsningar börjar med mätning av feltyper och felfrekvenser, dvs fel upptäckta under utvecklingsarbetet och inrapporterade fel efter leverans. Denna information ger sedan kunskap om potentiella fel och om felelimineringseffektiviteten i utvecklingsarbetet. Härmed är grunden lagd för förebyggande och eliminering av fel. Fel kan undvikas tack vare av kunskap från mätningar eller elimineras i tidigt skede eftersom mätningar ger indikationer om var fel av olika typer kan hittas. Detta spar tid och resurser. Sparad tid betyder minskade ledtider. Sparade resurser innebär ökad produktivitet. Färre fel i slutprodukten betyder bättre kvalitet. Om mätningen sedan utvidgas till att även omfatta andra betydelsefulla faktorer, inte bara felfrekvens (vilket alla företagen med framgångsrika kvalitetssatsningar fortsatt med), kan produktivitet och ledtider - och även kvalitet - förbättras ytterligare. Det är mätning som ger ökad kvalitet, ökad produktivitet och minskade ledtider.

6.1. Slutsatser från statistiken

Trots den påpekade osäkerheten i underlaget i de statistiska undersökningar som presenterats i denna rapport, är det möjligt för oss att dra vissa slutsatser. Slutsatserna bygger inte på några absoluta tal från dessa undersökningar utan snarare på trender och allmänna mönster i det mycket omfattande projektmaterialet.

För det första finns det ett tydligt samband mellan projektstorlek och projektproduktivitet. För nyutveckling är systemstorlek samma som projektstorlek, medan i förvaltning projektstorlek är storleken på de ingrepp som görs i ett system. Det tycks finnas en optimal projektstorlek. För nyutveckling är en projektstorlek mellan 200-400 funktionspoäng en lämplig storlek, medan produktiviteten för systemändringar är högst vid en projektstorlek på 40-160 funktionspoäng . Felrättning är mest effektivt då ingreppen sker endast på några enstaka ställen. Enligt amerikansk statistik är de mest effektiva ingreppen i storleksordningen 5-20 funktionspoäng. Därefter ökar komplexiteten i arbetet och ingreppen riskerar att orsaka nya fel vars eliminering drar ned produktiviteten. Ett större system med många fel som upptäcks strax före leverans eller med många dolda fel kräver stora felrättningsingrepp under det första året. Eftersom produktiviteten för sådana ingrepp i regel är mycket låg, blir felrättningsarbetet dyrt. Det stör dessutom andra utvecklingsaktiviteter och försenar leveransen.

Den slående skillnaden i resursförbrukningen och produktiviteten och, som kontrast, likheten i ledtider mellan amerikanska projekt och Laturi-projekt motiverar en andra slutsats. Laturi-projekt har mycket mindre resursförbrukning och mindre projektpersonal, särskilt i stora projekt. Trots detta är ledtiderna nästan identiska. Alltså har den bättre produktiviteten i Laturi-projekt lett till motsvarande minskning i ledtider. Orsaken till detta förhållande finns med stor sannolikhet i arbetssättet. Traditionella metoder och sätt att driva systemutvecklingsprojekt tillåter inte minskning av ledtider mer än till viss nivå. Orsaken är att existerande projektmodeller och metoder bygger på uppdelning av projekt i sekventiella etapper och aktiviteter. Om en dramatisk minskning av ledtider eftersträvas, måste helt andra sätt att strukturera projekt och att organisera arbetet anammas. En mycket lovande och aktuell paradigm inom detta område är Concurrent Engineering, som just utgår från maximal parallellisering av aktiviteter, även i situationer där en aktivitet är beroende av resultat från en annan. Principer enligt Concurrent Engineering har redan tillämpats i systemutveckling av några företag för att få ned ledtiderna och erfarenheterna hittills är mycket lovande.

En tredje slutsats är att utvecklingsmiljön har stor betydelse för produktiviteten. Kompetent projekthantering och rätt situationsanpassad projektstruktur är givetvis den viktigaste enskilda famgångsfaktorn, men då projekthanteringen fungerar, blir metoderna och den tekniska miljön mycket viktiga produktivitetspåverkande faktorer. Capers Jones undersökning om produktivitetspåverkande faktorer visar att en bra teknisk miljö, med bra språk och verktyg, är en mer signifikant framgångsfaktor än enbart personalens kompetens. Orsaken till detta torde vara att mycket av den detaljkunskap som behövs, finns inbyggd i verktygen och i högnivåspråken, samtidigt som kunskapskrävande och mödosamma aktiviteter till stor del automatiseras. Verktygen och högnivåspråken kompenserar bristen på vissa typer av detaljkunskap, samtidigt som de gör att annars tidskrävande arbetsmoment snabbt avverkas . Även Laturi-materialet indikerar den tekniska miljöns stora betydelse.

6.2. Införande av mätning

Syftet med denna rapport har varit att ge en inblick i vad mätning av systemutveckling är, vad som kan mätas samt, med hjälp av statistik från genomförda mätningar, ge exempel på den kunskap som mätning ger upphov till. Att sätta egna mål för produktivitet, kvalitet och ledtider och att utifrån dessa etablera lämpliga mått och en fungerande mätverksamhet kan vara nog så svårt utan bra handledning.

Det finns dock erfarenheter och en hel del vägledande litteratur inom området som kan vara till hjälp. I sin mycket läsvärda rapport "Bättre programvara till lägre kostnad" [13] berättar Lars Wiktorin om satsningar, resultat och erfarenheter från flera företag i Sverige, Europa och USA. Wiktorin ger också en kort översikt av aktuella forskningsinsatser inom området samt sammanfattar några allmänna råd för den som står i begrepp börja mäta den egna utvecklingsprocessen.

Användbar och relativt detaljerad vägledning för införande av mätning finns i boken "Software Metrics - a practioners guide to improved product development" av K.H. Möller och D.J. Paulish. Boken bygger på resultat från, och arbete utfört i Esprit-projektet PYRAMID. Boken ger också analyserande exempel på konkreta mätsatsningar i industrin och sammanfattar erfarenheterna från dessa.

Även Capers Jones behandlar införande av mätning i sin bok "Applied Software Measurement" [5]. Här avhandlas tämligen översiktligt *hur* man inför mätning, medan detaljerade exempel ges på *vad* som kan mätas och hur resultaten kan bokföras och rapporteras.

7. Litteraturförteckning

[1] Adams, E. : "Optimizing preventive service of software products", IBM J. Research & Development 28(1), 1984, 2-14.

[2] Chappel C., Stevenson I.: "Concurrent Engineering - Then Market Opportunity", Ovum rapport från Ovum Ltd, 1992, England.

[3] Dreger, J.B.: "Function Point Analysis", Prentice Hall, Englewood Cliffs, New Jersey, 1989, ISBN 0-13-332321-8.

[4] Fenton, Norman E: "Software Metrics, A Rigorous Approach", Chapman&Hall, London, 1991, ISBN 0-412-40440-0.

[5] Jones, Capers: "Applied Software Measurement, Assuring Productivity and Quality", MacGraw-Hill Inc, New York, 1991, ISBN 0-07-032813-7.

[6] Jones, Capers: "Software Metrics and Total Quality Management", artikel i CASE OUTLOOK 1992, vol. 6, nr 4.

[7] Karlöf, B. och Östblom, S.:"Benchmarking, vägvisare till mästerskap i produktivitet och kvalitet", Svenska Dagbladets förlag AB, 1993, ISBN 91-7738-329-X.

[8] Paulish D.J., Möller K.H.: "Software Metrics, A Practitioners Guide to Improved Product Development", Chapman&Hall, London, 1993, ISBN 0-412-45900-0.

[9] Rook P (ed): "Software Reliability Handbook", Elsevier N. Holland, 1990.

[10] Shepperd Martin: "Software Engineering Metrics, Volume 1: Measures and Validation", MacGraw-Hill Intern., London, 1993, ISBN 0-07-707410-6.

[11] Symons, C.R.: "Software sizing och estimating", John Wiley, 1991.

[12] Watts, Richard: "Measuring Software Quality", NCC Publications (National Computing Centre), England, 1987, ISBN 0-85012-557-X.

[13] Wiktorin, Lars: "Bättre programvara till lägre kostnad - Att mäta och styra utvecklingsprocessen. Nuläge och trender", Fakta rapport i serien Informationsteknologi, nr 3, maj 1993, Sveriges Verkstadsindustrier, ISSN 1003-7008.

Svenska Institutet för Systemutveckling, SISU, bedriver forskning, följer utvecklingen och förmedlar kunskap om informationsteknologins tillämpning på informationsanvändning och informationsförsörjning i företag, myndigheter och andra organisationer. Institutet verkar inom detta område som ett opartiskt nationellt kompetenscentrum.

SISU

Electrum 212, 164 40 Kista
Isafjordsgatan 26
Telefon 08-752 16 00 Telefax 08-752 68 00