Översikt avseende forskning och erfarenheter kring programmering i förskola och grundskola

fil. dr. Susanne Kjällander, fil. dr. Anna Åkerfeldt och fil.lic. Petra Petersen

Förord och inledning

Det övergripande målet för denna översikt är att presentera och diskutera aktuell svensk forskning och pågående projekt relaterat till lärande och programmering i svensk förskola och grundskola. Detta görs i relation till internationell forskning. De reflektioner som presenteras är författarnas egna. Översikten, som ett uppdrag från Skolverket inom ramen för att ta fram förslag till nationella it-strategier för skolväsendet, har tagits fram av fil. dr. Susanne Kjällander, fil. dr. Anna Åkerfeldt och fil. lic. Petra Petersen.

Vetenskaplig grund och beprövad erfarenhet?

Inte sällan lyfts att undervisning ska vila på vetenskaplig grund och beprövad erfarenhet. Vad finns då för vetenskaplig grund och beprövad erfarenhet vad gäller programmering i förskolans och skolans verksamhet? Programmering inom den vetenskapliga disciplinen datavetenskap är ett etablerat område. Däremot är programmering inom utbildningsvetenskaplig forskning begränsad. I den här översikten har vi därför främst lutat oss mot forskning som tar sin utgångspunkt utifrån datavetenskap för att belysa programmering inom svensk skola och utbildning. Eftersom forskningsområdet är så pass nytt kommer denna översikt att ta upp svensk forskning så långt det är möjligt, men även lyfta erfarenheter och exempel från olika projekt runt om i Sveriges förskolor och skolor samt belysa ett urval av relevant internationell forskning. Skolverket skriver om hur vetenskaplig grund innebär att ”*kritiskt granska, pröva och sätta enskilda faktakunskaper i ett sammanhang samt söka efter förklaringar och orsakssamband i tillgänglig relevant forskning*”[1]. Vi tar här utgångspunkt i SKL:s skrivning om vetenskaplig grund och beprövad erfarenhet, baserat på översikten "Utmärkt undervisning" skriven av forskarna Jan Håkansson och Daniel Sundberg (2012) om samspelet mellan forskning och beprövad erfarenhet: “*För att utveckla skolan krävs både vetenskaplig grundad kunskap och kunskap grundad på praktik och erfarenhet. SKL vill därför understryka vikten av ett samspel mellan forskning och beprövad erfarenhet för att utveckla en undervisning på vetenskaplig grund.*”[2] Att särskilja vilka erfarenheter som kan anses vara beprövade blir här inte relevant, istället lyfter vi resultat av projekt som just ”erfarenheter”. Vi menar att vi kan dra lärdomar av såväl forskning som erfarenhet, men vi kommer så långt det är möjligt att skilja resultat som genererats på olika sätt åt – *forskning*

[1] http://www.skolverket.se/skolutveckling/forskning/forskningsbaserat-arbetssatt/nagra-nyckelbegrepp-1.244041
[2] http://skl.se/skolakulturfritid/skolaforskola/forskning/vetenskapliggrundbeprovaderfarenhet.728.html)

respektive *erfarenhet* – samt beskriva bakgrunden till resultaten, vilket också var ett önskemål från Skolverket. Läs mer om uppdraget i nästa stycke.

Inramning och uppdraget från Skolverket

I samband med regeringens uppdrag om en nationell IT-strategi till Skolverket[3] *Uppdrag att föreslå nationella it-strategier för skolväsendet* i september, gavs i november ett uppdrag att ta fram en översikt över svensk forskning och svensk erfarenhet om programmering i förskola och grundskola. Det fanns även ett önskemål, från såväl Skolverket som författarna, att lyfta fritidsverksamheten men ingen forskning eller erfarenhet har skrivits fram inom ramen för fritidshemmets verksamhet. Vi tror att programmering förekommer på fritidshem men att det inte ännu dokumenterats. I och med att svensk forskning om programmering i förskola och skola är begränsad önskades även en internationell utblick. Tanken är att översikten ska ligga till grund för en evidens- och forskningsbaserad motivtext för införande av en förstärkning och ett tydliggörande av kunskaper och förmågor kopplade till programmering i läroplaner och kursplaner.

Översiktens disposition

Denna översikt inleds med en metodbeskrivning följt av en bakgrund vad gäller programmering i skolan, såväl historiskt som innehållsmässigt.

Därefter redovisas svensk forskning om programmering i förskola och skola samt ett urval projekt som redovisar någon typ av resultat/erfarenhet. En internationell utblick görs med fokus på de utmaningar Sverige nu står inför. Därefter följer ett urval av relevant internationell forskning om programmering i förskola och skola.

Den sista avslutande diskuterande delen inleds med en begreppsdiskussion och fortsätter sedan med forskningsbaserade argument för, såväl som kritik mot, införandet av programmering i förskola och skola. Avslutningsvis lyfts vad som kan beaktas i och med ett införande av programmering i förskola och skola.

I det följande beskrivs kortfattat den metod som använts för att skriva denna översikt.

[3] http://www.regeringen.se/pressmeddelanden/2015/09/regeringen-ger-skolverket-i-uppdrag-att-foresla-nationella-it-strategier-for-skolvasendet/

Metod

I översikten behandlas programmering som ett *tilltänkt* ämne eller inslag i den svenska skolan. Det är således inte programmering inom den vetenskapliga disciplinen datavetenskap (computer science) som undersökts utan forskning och projekt som angränsar "skolämnet" programmering och vilken betydelse det har för elevernas lärande. Utifrån ovanstående har en systematisk genomsökning av forskning om programmering inom svensk skolkontext gjorts. Sökningar har gjorts som ramas in av digitala utbildningsmiljöer med inslag av programmering och datalogiskt tänkande i såväl förskola som grundskola samt aktiviteter på fritiden. En målsättning med översikten är att redogöra för forskning inom denna avgränsning. Eftersom programmering i skolan är ett framväxande område innebar uppdraget även att välja ut och presentera exempel på internationell forskning för att belysa den svenska.

Med hjälp av ett urval projektbeskrivningar som på något sätt redovisat någon form av resultat avser vi illustrera vad som pågår kring programmering och datalogiskt tänkande i svenska förskolor och skolor. Denna genomsökning är inte i lika hög grad systematisk utan ska ses som exempel på och ett axplock av projekt som pågår runt om i Sverige. Vissa projekt, som ännu inte presenterat några resultat, finns omnämnda i löpande text eller i fotnoter för att det kan vara projekt värda att hålla utkik efter framöver.

De källor som presenteras i denna översikt består därmed av mycket skiftande tyngd ur ett vetenskapligt perspektiv. Vissa källor är vetenskapliga granskade artiklar och doktorsavhandlingar, medan andra är projektbeskrivningar författade av såväl forskare, pedagoger som mer kommersiella aktörer. Vi har därför, så långt det varit möjligt, beskrivit de olika källorna löpande i texten.

Översikten tar sin utgångspunkt utifrån ämnena pedagogik, didaktik samt barn- och ungdomsvetenskap. Forskning om barn och elevers lärande har därför prioriterats, men även forskning om lärare finns representerade. I syfte att öka förståelsen av vad vi ramar in i en aktivitet där barn programmerar inleder vi nästa stycke med en kort bakgrund.

Bakgrund och introduktion

Nedan följer en kort översikt kring programmeringens historia[4], dess utveckling och för att ge läsaren en inblick i vad programmering kan vara.. Programmeringens historia startade

4 Det finns idag över 2 500 programmeringsspråk (http://www.levenez.com/lang/). Mediaföretaget O´Reilly har publicerat en tidslinje med 50 programmeringsspråk med början 1954-2004 och hur de hänger samman med varandra.

långt innan det vi idag kanske vanligtvis förknippar med "programmering" fanns. Denna utveckling har skett i många delar av världen och under många århundraden. 1801 uppfann Jaquard en vävstol som hade utbytbara hålkort, vilket ofta ses som en slags tidig form av programmering (se t.ex. Brookshear, 2009). I mitten av 1800-talet skrev Ada Lovelace i sina anteckningar en algoritm som ofta ses som det första datorprogrammet. 1889 utvecklade Hollerith en beräkningsmaskin som kunde läsa data i form av hålkort. Han lade också till en kontrollpanel runt 1906, som gjorde det möjligt att styra maskinen, utan att behöva bygga om den. Under andra världskriget utvecklades bl.a Colossus, en programmeringsbar dator som var designad för att knäcka tyska krypterade meddelanden. Under 1950-talet utformade Grace Hopper den första kompilatorn (eng: compiler), en slags översättare mellan olika nivåer av programmeringsspråk. Det var också nu som programmeringsspråket Fortran utvecklades av Backus vid IBM på 1950-talet, och det var först nu som det gick att använda ett datorprogram på flera olika maskiner (Brookshear, 2009). Parallellt med utvecklingen i USA och Europa, utvecklades programmering på olika sätt i det forna Sovjetstaterna, vilket på senare år uppmärksammats (se t.ex. Trogemann, Nitussov & Ernst, 2001). Margaret Hamilton var programmeraren som skrev koden som tog den första bemannade rymdfarkosten Apollo 11 till månen (Wayne, 2011). Programmeringspionjären Elsa-Karin Boestad-Nilsson, som var verksam vid Försvarets Forskningsanstalt (FOA), menar att "männen byggde maskinerna men det var kvinnor som programmerade dem" (Rose, 2015)[5]. Under 1950-talet, hade programmering ofta en låg status och Boestad-Nilsson menar att männen, som ofta var ingenjörer, inte ägnade sig åt programmering. De ville inte "befatta sig med detta tragglande med nollor och ettor" - det kunde kvinnorna göra (Rose, 2015). Detta har förändrats och programmering ses idag som en kompetens med hög status.

Idag finns det många olika programmeringsspråk som går att tillämpa i flertalet olika digitala produkter och tjänster. På senare år har det även vuxit fram programvaror där den enskilda användaren kan skapa avancerade digitala produktioner utan att kunna skriva kod till exempel *Content Managment System* (CMS) och olika författarverktyg. Här arbetar användaren istället med olika visuella representationer för att skapa till exempel en

http://archive.oreilly.com/pub/a/oreilly//news/languageposter_0504.html. För vidare läsning om programmeringens historia se till exempel Brookshear, 2009. Se även webbsidan http://www.computerhistory.org/ som drivs av Computer History Museum.

5 http://fof.se/tidning/2015/4/artikel/kvinnorna-bakom-datorernas-genombrott

webbsida eller en blogg. Wordpress[6] är ett exempel på en sådan produkt. Fler exempel redovisas i nästa stycke.

Programmeringsspråk

Programmering kan ske på olika nivåer. Den första nivån som är närmast hårdvaran, programmeras i *maskinkod* eller binärkod som består av enklare kommandon. Ett *assemblerprogram* gör en slags direktöversättning av maskinkoden där kortare kommandon representerar den binära koden. Man skulle kunna säga att det är en mer läsbar kod för människor än den binära koden. Nästa steg är *allmänna programspråk* (på engelska kallas detta "general purpose computer languages") såsom Java, C eller Pascal, där en instruktion motsvarar flertalet kommandon i maskinspråket. En *kompilator* översätter instruktionerna till maskinkod (alternativt tolkas de av en tolk). I nästa nivå beskrivs snarare vad som ska utföras, själva funktionen istället för hur det ska utföras (för mer utförlig beskrivning, se t.ex. Brookshear, 2009). Det finns även andra sätt att kategorisera programmering, t.ex. utifrån användningsområde eller programmeringsparadigm (för mer läsning om detta, se t.ex. Van Roy, 2009). Många av de programmeringsspråk som används av barn och elever i utbildningssammanhang är grafiska eller *visuella programmeringsspråk* (på engelska kallas detta "Visual Programming Languages", VPL), till exempel *Scratch*. I yngre åldrar används också ofta fysiska objekt som barn och elever kan programmera att utföra olika rörelser. De fysiska objekten som programmeras kan vara programmeringsbara klossar eller robotar, till exempel *Bee-Bot,* en insektsliknande robot.

Programmering i skolan - historisk tillbakablick

Datalära eller datakunskap återfinns i läroplaner och kursplaner redan från 1970-talet. 1975 skrev Björk (1983) att: "Skolan måste ge alla elever elementära kunskaper om vad en dator är, vad en dator kan och inte kan göra, hur problemlösning med dator inom olika områden går till samt fördelar och nackdelar med det datoriserade samhället." (Björk, 1983). På 70- och 80-talet arbetade en del lärare med programmeringsspråket BASIC på högstadiet och gymnasiet. Initiativet att införa och arbeta med programmering i skolan har framförallt varit lärarstyrt (se Björk, 1983). Från statligt håll var intresset tydligare inriktat mot att se datorn som ett verktyg för lärande som skulle stödja undervisningen. I Lgr80 infördes datalära i matematik och inom de samhällsorienterade ämnena (Farkell-Bååthe, 2000). Programmering finns med i Lgr80 som ett innehåll men intresset riktades framförallt till gymnasiet, inte till grundskolan. Eleverna skulle ges grundläggande kunskap i att

6 wordpress.com

programmera i programmeringsspråket BASIC[7]. Under 1980 och 90-talet gjordes det ett flertal satsningar på att integrera datorn som stöd i undervisningen. Fokus låg då återigen på att se datorn som ett verktyg och hur digitala hjälpmedel kunde stötta lärarna i deras undervisning och eleverna i deras lärande. Det gjordes även flertalet satsningar när det gäller utveckling av digitala läromedel för skolan som till exempel Dataprogramgruppens *Princess-projektet* och *Dpg-projektet* (Farkell-Bååthe, 2000). Internet som började användas i skolorna runt millenniumskiftet gjorde att fokus förflyttades till informationsteknik i högre grad istället för datalära eller datakunskap. I den gällande läroplanen för grundskolan, Lgr11, läggs tonvikten på att eleverna behöver lära sig handha och använda "modern teknik". Sedan Lgr80 har programmering inte nämnts i läroplanen för grundskolan utan fokus har varit på att använda datorn som ett pedagogiskt verktyg och att eleverna ska ha kunskap i hur digitala resurser kan användas som ett verktyg för lärande såsom till exempel informationssökning[8].

Svensk forskning om programmering i skolan

Forskning inom programmering som berör lärande och undervisning i svensk förskola och skola är begränsad. En stor del av den forskning som finns beskriver projekt och lyfter fram hur barn på olika sätt kan närma sig programmering. Endast en liten del fokuserar på barns och elevers lärande. Vi har valt att bredda översikten genom att till viss del beröra forskning kring lärande och undervisning i gymnasiet och här tas projekt som är knutna till högskola- och universitet upp. I avsnittet tas även forskningsresultat som inte är vetenskapligt granskade upp, såsom till exempel konferensbidrag[9].

Svenska forskningsprojekt

Mannila, Dagiene, Demo, Grgurina, Mirolo, Rolandsson och Settle (2014) har undersökt hur skolor i Sverige, Finland, Nederländerna, Italien, Litauen, och USA arbetar med programmering och datalogiskt tänkande. Författarnas syfte är att ge lärare, lärarutbildare och beslutstagare en möjlighet att ta informerade beslut om hur, när och på vilka sätt programmering och datalogiskt tänkande kan införlivas i olika delar av utbildningen. Mannila et al. (2014) redogör för en undersökning där lärares uppfattningar om datalogiskt tänkande är i fokus, samt i vilken utsträckning aspekter av datalogiskt tänkande

7 BASIC - ett pogrammeringsspråk som utvecklades framförallt för att användas i utbildningssyfte.
8 Se även https://youtu.be/uTbrERz9s_g för en historisk jämförelse med det engelska skolsystemet när det gäller undervisning relaterat till informationsteknologier.
9 Ett paper som ofta presenteras i en föreläsning på en forskarkonferens där olika forskare (ofta från olika delar av världen) möts. Ofta har ett konferensbidrag genomgått en vetenskaplig granskning av något slag.

förekommer i olika områden i skolan och på vilka sätt. Datainsamlingen har skett med hjälp av online-enkäter med sammanlagt 961 svar, där majoriteten av svaren kom från Litauen och Italien. 33 av de 961 enkätsvaren kom från svenska lärare. Majoriteten av lärarna, från samtliga länder, som svarade på enkäten var lärare i IT, matematik eller naturkunskap, vilken också kan påverka i vilka sammanhang datalogiskt tänkande rapporteras förekomma. Resultaten visar att de aspekter av datalogiskt tänkande som lärarna uppfattade var mest förekommande i sin undervisning var datainsamling, dataanalys och representation av data. De verktyg som lärarna rapporterade att de använde mest, som de uppfattade som kopplat till datalogiskt tänkande, var informationssökningar via webben och office-paketet, eller grafiska program. I mindre utsträckning användes visuella programmeringsspråk så som till exempel Scratch. I ännu mindre utsträckning användes programmeringsspråk, så som Java eller Python och bara några procent rapporterade att de använde simulering eller robotprogrammering i sin undervisning.

Heintz, Mannila, Nygårds, Parnes och Regnell (2015) har i ett konferensbidrag sammanställt olika projekt i Sverige, där barn och ungdomar på olika sätt lär sig programmering. Ett gemensamt fokus riktas av författarna på bl.a. kreativt tänkande och skaparkultur. Forskarna menar att begreppet programmering kan vara för snävt för att beskriva förmågor såsom kreativ problemlösning, att se mönster och tänka logiskt steg-för-steg. Författarna menar att dessa förmågor ingår i att kunna programmera. Författarna beskriver övergripande olika programmeringsprojekt som pågår eller är avslutade i Linköping, Lund, Luleå och Stockholm.

Konferensbidraget *Teaching programming to young learners using Scala and Kojo*, av Björn Regnell vid Lunds Tekniska Högskola och Lalit Pant från Kogics Foundation i Indien, har undersökt hur elever lär sig programmering med hjälp av applikationen *Kojo*[10] (Regnell & Pant, 2014). Elever från både Sverige och Indien har deltagit i studien och den svenska delen av studien är baserad på projektet *Programming for Everybody*, initierat av Lunds Tekniska Högskola[11]. Till skillnad från enklare applikationer eller robotar (såsom Bee-Bot där barnen ger roboten kommandon genom att fysiskt trycka på Bee-Bots rygg för att den ska utföra en rörelse) bygger Kojo på programmeringsspråket Scala där barnen skriver kommandon med hjälp av tangentbordet. Detta är enligt Regnell och Pant (2014) en styrka, eftersom barnen får en känsla av hur det är att programmera. Det är flexibelt och eleverna kan göra avancerade program. Regnell och Pant (2014) menar vidare att genom att

10 Kojo är en open source programvara som bygger på programmeringsspråket Scala.
11 Här har barn fått lära sig skapa enkla kommandon, för att få en liten sköldpadda att "rita" en bild. Idén bygger alltså på att barnen ska styra sköldpaddan så att den går i en formation och därmed ritar linjer, utefter barnens kommandon.

lära sig att programmera på det här sättet, lär sig eleverna också abstrakt tänkande - något som i denna översikt hänvisas till som en transfereffekt av programmering. Begreppet är inte allmänt vedertaget, men innebär här att man tänker sig att elever genom att programmera i skolan kan bli bättre på att till exempel tänka logiskt, räkna matte eller utveckla andra färdigheter och kunskaper som inte alls har att göra med just programmering.

Forskningsprojektet *Plattan i mattan. Didaktisk design och digitala lärplattor i förskolan* har studerat förskollärares och förskolebarns arbete med programmering (Kjällander, 2016). Projektet är avslutat och inga vetenskapligt granskade resultat har ännu publicerats. I projektet har fyra förskolors användning av digitala surfplattor studerats genom videoobservationer under drygt ett år. Inom ramen för projektet har även fokusgruppssamtal med förskollärare genomförts. Förskollärarna menar att de arbetar med programmering för att alla barn ska få möjlighet att utveckla en förståelse för att det är människor som programmerar datorernas, maskinernas och robotarnas funktioner. Barnen programmerar i olika appar såsom: *Fix the Factory*, *LightBot jr*, *Constructabot* samt *Scratch jr*. På förskolorna projiceras ofta surfplattans skärm på väggen vilket gör det möjligt för flera barn att delta i aktiviteten och inte endast barnet som sitter med lärplattan framför sig. I förskolan använder även förskollärarna robotar för att lära barnen programmering (till exempel Bee-Bot). Med hjälp av pedagogen skapas fysiska banor med tusch på papper eller med laminerade skyltar som läggs i en bana inomhus eller utomhus. På dessa fysiska banor kan sedan roboten ta sig fram på det sätt barnen programmerar den att gå på. Förskollärare menar att det fanns ett stort intresse bland barnen att styra dessa robotar och komma fram till olika lösningsförslag. Barnen provar också att gå på banorna och instruera (eller programmera som de själva säger) varandra att gå på speciella sätt. Barnen tenderar även att leka vidare olika "programmeringslekar" genom att bygga robotar i olika material, släcka och tända lampor eller dansa robotdans.

I och med att forskningen är begränsad har vi valt att ta med en studie som undersöker gymnasielärares uppfattningar om programmering. Lennart Rolandssons avhandling, *Programmed or not. A study about programming teachers' beliefs and intentions in relation to curriculum* (2015) undersöker bland annat gymnasielärares föreställningar, tankar och intentioner i förhållande till deras praktik.. Genom att under 2011 arrangera ett antal seminarium med gymnasielärare som undervisar programmering har Rolandsson samlat in det empiriska material som ligger till grund för resultatet som således baseras på lärarnas reflektioner kring deras undervisning och hur de uppfattar att undervisa i ämnet (Rolandsson, 2015).

Under seminarierna, som Rolandsson genomfört, menar lärarna att trots att lärare undervisar i programmering på olika sätt, har många elever svårt att förstå ämnet. Hans avhandling visar att programmeringsundervisning kan uppfattas som exkluderande, eftersom lärare anser att undervisning i programmering är svårt. Trots det visar studien att elever ofta förväntas lära sig själva genom att koda på egen hand. Rolandssons resultat visar också att lärarna försöker hitta strategier som gör att ”*Så många elever som möjligt skall kunna utbildas*” (Rolandsson, 2015, s 60) och för att ”*alla ska ha möjligheten tt utveckla sina förmågor till problemlösning*” (Rolandsson, 2015, s. 60). Resultat från Rolandssons studie visar även att många lärare saknar tron på att alla elever kan lära sig programmering. Istället tror lärare att endast de elever som redan har en analytisk och logisk förmåga kan lära sig programmera. Lärarna provar sig fram själva och förväntar sig ofta även att eleverna också ska prova sig fram och lära sig programmera på egen hand. Rolandsson (2015) lyfter lärarrollen som central i hur ämnet programmering utvecklas i klassrummet och stödjer sig på forskning av Pajares (1992) och Olafson och Schraw (2002) som visar att transformationen av kunskap i klassrummet visar sig bero helt och hållet på lärares epistemologiska föreställningar i relation till undervisning i programmering.

Sammanfattningsvis ser vi att ett sätt att närma sig programmering är att använda fysiska material i både förskola och skola, t.ex. robotar och olika applikationer där elever kan använda visuell programmering. Ett genomgående drag är att programmeringsspråk och programmeringsprocesserna på olika sätt visualiseras eller presenteras i nya former av representationer. Detta kan ske genom allt från bilder eller bildflöden på en skärm till tredimensionellt material, så som programmeringsbara klossar eller olika sorters robotar. Att locka och intressera flickor att lära sig programmera är centralt inom flera projekt i Sverige. Detta återfinns inte lika tydligt inom vetenskapligt granskad forskning, men berörs i kommande avsnitt om projekt som är knutna till högskolor och universitet.

Högskole- och universitetsanknutna forskningsprojekt

Lunds Tekniska Högskola har i samarbete med *Vattenhallen Science Center* anordnat programmeringsaktiviteter för barn från sju år och uppåt, i projektet *Programming for Everybody*. I projektet har målet varit att utveckla programmeringsexperiment riktade mot skolgrupper, samt att utveckla lärares kompetens inom programmering, så att de i sin tur kan lära sina elever att programmera. Vid Lunds Tekniska Högskola har man också skapat

online-resurser, med utbildningsmaterial och länkar som stöd för lärare[12]. Projektet har vid denna tidpunkt ännu inte presenterat sina resultat.

I Vinnova-projektet *Trippel Helix - Nationell samling för skolans digitalisering*, görs en satsning på att öka samarbetet mellan skola, näringsliv och akademi. Målet är att formulera en "*gemensam konkret och genomförbar agenda som driver på tankemässig och verksamhetsmässig förändring av framtidens skola baserat på digitaliseringens möjligheter och de kunskapskrav som följer med den*".[13] Vidare lyfts kritiskt tänkande, kreativitet och nyfikenhet bland barn och unga, och det finns inom projektet en bred syn på vad digitalisering av skolan kan och bör innefatta. Projektet har ännu inte publicerat några resultat som går att hänvisa till i det här sammanhanget.

Det finns också flera projekt runt om i Sverige som arrangeras för barn och ungdomar av olika organisationer och/eller ideella verksamheter. Dessa beskrivs i nästa stycke.

Programmeringsaktiviteter inom och utanför svensk förskola/skola

I följande avsnitt presenteras och beskrivs några projekt som pågår, eller nyligen är avslutade, där programmering mer eller mindre ingår som en central del av verksamheten. I avsnittet tas både skolrelaterade projekt upp och aktiviteter för barn och unga som anordnas utanför en skolkontext (till exempel aktiviteter anordnade av den ideella organisationen *Geek Girl Meetup*).

Inom ramen för en skolkontext finns det flera enskilda lärare som arbetar med programmering med sina elever. Till exempel i Kyrkenorumskolan i Stenungsund[14] där eleverna i åk 3 arbetar i mindre grupper och skapar interaktiva berättelser med hjälp av programmet *Scratch*[15]. Läraren Christina Löfving menar att genom att de arbetar med programmering i klassen har elevernas intresse för matematiken ökat då de i programmeringsarbetet ofta diskuterar matematiska begrepp. Även läraren Karin Nygårds[16] har arbetat med applikationerna Scratch och Kojo[17] med sina elever. Hon menar att eleverna behöver lära sig att hjälpa och dela kunskap med varandra vilket de delvis

12 http://www.lth.se/programmera/programmering-i-skolan/
13 http://swedsoft.se/wp-content/uploads/sites/7/2015/09/Projektbeskrivning-digitalisering-f-framtidens-skola.pdf
14 http://www.lararnasnyheter.se/lararnas-tidning/2015/03/20/har-lar-3c-programmera
15 https://scratch.mit.edu. Scratch (se även Scratch Junior) är en applikation där användaren med hjälp av visuella block kan programmera ett objekt istället för att skriva kod.
16 Karin Nygårds har byggt upp webbsajten Teacherhack http://teacherhack.com med stöd av Internetfonden.
17 Kojo använder programmeringsspråket Scala.

tränar när de programmerar[18]. Karin Nygårds har arbetat med programmering i skolan sedan 2013[19].

På Lugnets Skola i Hammarby Sjöstad pågår pilotprojektet *IT-matematik*[20] i samarbete med Claes Johnson, professor i tillämpad matematik vid Chalmers Tekniska Högskola och Kungliga Tekniska Högskolan. I projektet används appen *Codea*. Johnson visar hur 10-åringar på Lugnets skola kan använda samma programmeringsspråk som studenter på Chalmers. Barnen utforskar programmering och prövar sig fram och kan med viss handledning såväl programmera spel, förstå spel och spela dem. Johnson tolkning är att detta beror på att datalogiskt tänkande ligger nära barns lek och dataspelande (Intervju, 2015-12-01). Tillsammans med andra forskare har Johnson skrivit fram ett förslag för ett nytt kärnämne där programmering integreras med matematik: IT-matematik[21].

Det finns även lokala projekt på kommunnivå inom ramen för förskoleverksamhet som organiserar aktiviteter där barn på olika sätt lär sig att programmera. Till exempel i Halmstads förskolor och även i Lysekils kommun[22] arbetar pedagogerna med programmering. I båda kommunerna används programmeringsbara robotar som t.ex. *Bee-Bot*. Med hjälp av den insektsliknande roboten Bee-Bot kan barnen programmera insekten att gå dit de vill genom att trycka in en sekvens av kommandon på biets rygg. Ett annat sätt som förskolor arbetar med programmering är via olika applikationer. På Katamaranen på Strannegårdens förskola i Kungsbacka använder pedagogerna applikationen *Lightbot*[23], där barnen programmerar en robot att röra sig på olika sätt i applikationen. Det finns också exempel på hur pedagoger inom förskolan använder ett mer analogt material för att skapa programmeringsaktiviteter för barn. Pysslingenförskolan *Västra Allé* i Helsingborg har arbetat med att låta barn och pedagoger skypa med varandra och med att låta barnen instruera varandra att gå framåt, åt höger etc. för att komma fram till önskat mål. Pedagogerna har också använt analogt material såsom *Plus-plus* och *BeyBlades*, där barn får instruera andra barn, att bygga, steg för steg. Här talar man om att "leka in kommandon och algoritmer"[24].

18 http://www.skolverket.se/skolutveckling/resurser-for-larande/itiskolan/sa-arbetar-andra/overgripande/de-programmerar-for-att-forsta-sin-samtid-1.222467
19 http://viprogrammerar.se Se även http://claesjohnson.blogspot.se/
20 https://matematikit.wordpress.com
21 Johnson menar att detta är bland annat för att revolutionera samt "rädda" matematikämnet som befinner sig i kris (http://www.nyteknik.se/asikter/debatt/article695620.ece).
22 http://iktsidan.com/2014/11/07/programmering-i-forskolan/
23 https://lightbot.com/
24 http://www.pysslingen.se/skolor/2015/02/19/programmering-pa-vastra-alles-forskola/

Det pågår olika sorters projekt och initiativ utanför skolan i Sverige där programmering utgör den centrala aktiviteten på ett eller annat sätt. Det är lokala initiativ eller projekt där olika former av organisationer anordnar aktiviteter, där barn och unga ges möjligheten att testa att programmera på olika sätt. Det finns även olika organisationer, såväl ideella l som vinstdrivande företag, som på olika sätt verkar för att barn och ungdomar ska lära sig att programmera. En del av dessa är speciellt inriktade på att uppmuntra och stödja flickors intresse att börja programmera. Ett exempel är *Tjejhack*[25], som håller workshops och arrangerar klubbverksamheter som riktar sig till flickor i olika åldrar, som är intresserade av datorspel och programmering. Festivalen *Tekla* är en teknikfestival där flickor mellan 11 och 18 år får prova robotprogrammering, speldesign, digital visualisering och 3D-printing. *Girl Tech Sweden* är ett annat initiativ med syfte att öka flickors intresse för programmering. Ytterligare ett exempel på en organisation som vill uppmuntra tjejer att börja programmera är *Geek Girl meetup*[26] (se även Geek Girl mini[27]) som även anordnar olika aktiviteter kring programmering. Liknande organisationer som ovan men som vänder sig till både flickor och pojkar är till exempel *CoderDoja*[28] där bland annat Linköpings Universitet, Institutionen för datavetenskap varit delaktiga i att arrangera olika aktiviteter riktade till barn och unga. Internetstiftelsen i Sverige (IIS) står bakom *Barnhack*[29] som anordnar programmeringaktiviteter på olika platser runt om i Sverige där barn i åldrarna 6-13 år får möjlighet att programmera i Scratch[30]. IIS har även gett ut en skrift under samma namn. Den ideella föreningen *Kodcentrum* arbetar både med programmeringsaktiviteter på fritiden men även inom ramen för skolan. Under åren 2016-2017 har organisationen fått finansiering av Vinnova för att utarbeta "...en modell för hur lärare i årskurs 4-5 ska kunna föra in programmering och digitalt skapande i undervisningen."[31]. Kodcentrum erbjuder programmeringsaktiviteter efter skoltid där barnen kan få handledning i programmering av programmerare eller av programmerarstudenter[32].

Avslutningsvis kommer vi nedan att ta upp några exempel på aktiviteter som ofta benämns som *Maker-kultur* eller *MakerSpace* (på engelska benämns det Maker Movement).

25 http://tjejhack.se/
26 http://geekgirlmeetup.com/ se även http://railsgirls.com/, http://tjejhack.se/
27 https://www.internetfonden.se/geek-girl-mini
28 https://coderdojo.com/attend. CoderDojo, är en ideell organisation som grundades 2013 av James Whelton med syfte att uppmuntra barn och unga i åldrarna 7-17 att träffas, få stöd av mentorer och lära sig mer om programmering. I dag finns det 14 CoderDoja i Sverige (2016-03-08).
29 https://www.iis.se/vad-vi-gor/barnhack/
30 Scratch är en mjukvara framtagen av MIT - Massachusetts Institute of Technology för att introducera barn och ungdomar för programmering.
31 http://www.kodcentrum.se/kod-i-skolan
32 www.kodcentrum.se

MakerSpace är nära anknutet till programmering och det förekommer ofta programmeringsaktiviteter inom ramen för benämningen men är oftast inte skaparden primära aktiviteten utan ska ses utifrån ett vidare perspektiv. Ofta står kreativitet och skapande tydligt i centrum. Detta kan förstås utifrån att under de senaste åren har olika teknologier, både programmeringsverktyg och mer fysisk teknologi såsom 3D-skrivare, blivit mer lättillgängliga och billigare. Det har också gjort att det blivit möjligt för en bredare massa att programmera, designa och skapa olika produkter, utifrån sina egna intressen. Peter Parnes, Professor vid Luleå universitet, menar att skaparkultur/makerkultur (maker culture) är kopplat till både programmering i skolan och kreativt skapande[33].

Det finns några exempel på hur man även i skolan börjat anordna MakerSpaces. I Sollentuna pågår projektet *Makerspace i skolan*, i samarbete med Göteborg, Partille, Kungälv, Interactive Institute Swedish ICT, KTH, Chalmers och Dataföreningen. Projektet är finansierat med hjälp av medel från Vinnova[34] och syftet är att: "skapa en fysisk och virtuell testmiljö för att systematiskt forska, testa och utvärdera koncept, digital kompetens och lärande inom programmering och makerkultur. Syftet är att skapa bättre förutsättningar för lärande och digitala arbetssätt i bryggan mellan den fysiska världen och den digitala"[35]. Ett annat exempel på makerprojketet är *Skaepiedidh*[36], som är ett pågående Vinnova-projekt mellan Luleå Tekniska Högskola och Luleå kommun som arbetar med kreativt skapande där barn och ungdomar lär sig att använda programmering och 3D-skrivare för att skapa olika saker. Till exempel har barnen skapat lysande led-smycken.

På flera science center och museum finns *MakerSpaces* där programmeringsaktiviteter tydligare står i fokus. Ett exempel är *Tom Tits Experiment*[37] i Södertälje, där det finns ett MakerSpace där barn i alla åldrar kan prova på att använda de programmeringsbara robotarna *Bee-Bot*, *BlueBot*, *Quirkbots* och *Dash & Dot*. Besök på Makerspace-aktiviteter på Tom Tits kan ske både av förskola, skola och fritidsverksamhet och även av

33 http://www.parnes.com/blog/index.php/2015/11/23/makerkultur-som-drivkraft-for-kreativt-larande-i-skolan-internetdagarna-20151123/
34 En statlig myndighet under Näringsdepartementet, med syfte att stärka Sveriges innovationskraft för hållbar tillväxt och samhällsnytta
35 http://makerspaceiskolan.tumblr.com
36 Skaepiedidh betyder skapa på sydsamiska. Se mer http://www.skapa.how/#!/
37 Verksamheten i Tom Tits Makerspace, används som exempel i denna översikt, för att visa några praktiska exempel på hur aktiviteter med programmering för förskole- och skolbarn kan se ut. Detta ska se som ett exempel, bland många pågående programmeringsprojekt i runt om i Sverige.

privatpersoner. Ylva Skillberg och Niclas Ekholm[38] som arbetar på Tom Tits och med MakerSpace i samband med skolbesök menar att de ser vissa tendenser i sin verksamhet. En sådan tendens är att barn lär sig att programmera och att dela upp problem i olika steg. En annan att barn samarbetar, tar olika roller samt använder sitt verbala språk för att tillsammans komma fram till olika lösningar när de programmerar. Samtidigt understryker de att dessa processer hela tiden sker i samspel med en lärare eller pedagog. Vidare lyfter de att barnen använder sin rumsuppfattning och spatiala förmåga när de programmerar en robot att röra sig på ett visst sätt i rummet och följa en bana som barnen själva skapat.

Det finns också flera internationella initiativ för att lära ut programmering till barn och ungdomar, både i skolan, fritidsverksamhet och på fritiden. Dessa drivs av olika slags organisationer, både ideella och vinstdrivande företag. *Hour of code*[39] arrangeras av organisationen code.org som, enligt dem själva, verkar för jämställdhet inom programmering och pekar på vikten av att alla barn har rätt att delta i sådana aktiviteter. Vidare menar de att barnen också lär sig problemlösning, logiskt tänkande och får använda sin kreativitet. Eu Code Week[40] är ytterligare ett exempel på initiativ för att lära barn att programmera som är finansierat av Europeiska kommissionen, för att öka intresset och kunskapen om programmering hos fler människor.

Utifrån ovanstående genomgång framgår att det finns begränsade erfarenheter kring vad barn lär sig utöver att lära sig programmera. Vad som både Nygårds och personalen på Tom Tits menar utifrån sina personliga erfarenheter och observationer är att barnen lär sig att samarbeta när de programmerar. Det finns ett flertal initiativ och projekt som pågår i Sverige som är intressanta att studera vidare utifrån lärandeperspektiv, för att skapa ökad kunskap om vilken betydelse programmering har för barn och ungas lärande i skolan.

Internationell utblick med England och Finland som utgångspunkt

Valet att använda England och Finland som utgångspunkt är att båda länderna, har eller ska implementera programmering i sina respektive läroplaner (England, hösten 2014 och Finland, hösten 2016). De två länder vi valt som utgångspunkt har många likheter med den svenska skolan, varför vi här anser att vi kan dra lärdomar utifrån deras erfarenheter - så långt det är möjligt.

38 Petra Petersen, en av författarna till översikten, besökte och intervjuade Ylva Skillberg och Niclas Ekholm i samband med arbetet av översikten. Intervjun genomfördes den 5 feb. 2016.
39 https://hourofcode.com/us
40 http://codeweek.eu

Rapporten *Shut down or restart* (The Royal Society, 2012) skapade underlag för införandet av programmering i Englands läroplan. Den skrevs fram som en källa till evidens som krävdes för att underbygga beslut som togs för att programmering infördes i läroplanen. Rapporten inleds med att hävda att: ”*Computing is of enormous importance to the economy*.” (2012, s. 3). Argumentet är att i vårt dagliga liv omges vi av olika tekniska system som vi använder men sällan vet hur de fungerar eller kan påverka hur de ska designas. Genom att lära eleverna en grundläggande förståelse för programmering och hur system designas och fungerar utifrån tekniska aspekter finns förhoppningar om att det ska kunna ge en grund för att nästa generation kan påverka och ställa krav på system och i högre grad bli medproducenter, istället för enbart konsumenter, av teknologier[41]: “*Advances in most major societal challenges require significant Computer Science skills and input*” (The Royal Society, 2012, s 25). I sin framställning ger översikten fyra utbildningsrelaterade huvudskäl (The Royal Society, 2012, s. 29) till varför datavetenskap och/eller programmering (computing science) ska införas i skolans läroplan:

1. *Computation is a fundamental part of our World.* Programmering anses vara och förbli en fundamental del av vår värld.
2. *Computer Science develops key thinking skills.* Datavetenskap utvecklar nyckelförmågor vad gäller bland annat tänkande såsom logiska resonemang, abstrakt tänkande och problemlösning.
3. *Creation and Creativity*. Skapande och kreativitet anges som ett skäl eftersom eleven har en frihet och kan bestämma vad och hur den ska skapa samt realisera sina idéer och därefter förbättra dem.
4. *Computational thinking is invading every other discipline*. Datalogiskt tänkande kommer in i alla andra discipliner på så sätt att det är ett övertygande verktyg samt att det faktiskt kan förändra själva disciplinerna.

Finland underbygger inte sitt beslut att införa programmering i skolan på empirisk forskning utan, enligt forskaren Linda Mannila, snarare på: “... *en kombination av faktorer och*

41 Inom interaktionsdesign har forskare under längre tid arbetat med ett designbaserat förhållningssätt (Design based research - DBR). Design, konstruktion, implementering och analys är centralt inom det DBR (Barad & Squire, 2004; Mor, 2010) som har utvecklats som metod inom interaktionsdesign för att inte bara studera utan även aktivt använda forskningen som ett redskap i designprocessen av produkter eller tjänster. Ytterligare en aspekt som är central inom DBR är att designen av produkter sker genom en iterativ process i nära samarbete med deltagarna för att skapa en första skiss av produkten som sedan revideras utifrån de erfarenheter som görs (Barad & Squire, 2004) vilket gör att produkten utvecklas nära den praktik där den kommer att användas. Se även Mishra och Yadav som argumenterar för att eleverna ska tränas i datalogigt tänkande för att de ska förflyttas från att vara konsumenter av teknologier till att utveckla nya former av uttryck och bygga teknologiska verktyg men även att gynna kreativitetet (Mishra & Yadav, 2013 i Voogt et al., 2015).

föregicks av nationella och internationella diskussioner. Det handlar om att programmering, eller datalogiskt tänkande, numera anses vara en basfärdighet på samma sätt som att läsa, skriva och räkna." (Björkman, 2015[42]). Mannila lyfter även digital kompetens som en viktig del av en mångsidig kompetens som elever behöver utveckla. Skäl som lyfts i Finland är framförallt att barn ska kunna förstå teknikens betydelse i vardag och samhälle och att de ska kunna använda teknik i sitt lärande.

Enligt forskaren Linda Mannila vid Åbo Akademi blir den största utmaningen för Finland att hinna fortbilda alla lärare och ta fram undervisningsmaterial[43]. Detta lyfter även Dorling[44]fram som en av de största utmaningarna i England just nu. Detta lyfts även fram i andra publikationer skrivna av svenska och internationella forskare[45]. Peter Parnes (2015) identifierar också ett flertal utmaningar då makerkulturen integreras i skolans undervisning såsom moderniseringen av lärarutbildningen och lärares kompetensutveckling. Eftersom programmering hittills inte varit något betydande inslag i undervisningen i förskola och skola på internationell nivå så saknas det forskning och utvärderingar om detta. Enligt rapporten "Computing our future" (European Schoolnet, 2014) är det få länder som utvärderat införandet av inslag av programmering i skolan. Det finns ett fåtal utvärderingar i några länder såsom Tjeckien, Danmark, Ungern, Israel, Malta och Spanien, men det är inget vi väljer att ta upp här.

Programmering som skolämne

Arton av sexton medlemsländer i EU har infört programmering i läroplanen på lokal, regional eller nationell nivå (European Schoolnet, 2014). Vissa länder som infört programmering har gjort det som ett eget skolämne på schemat, medan andra infört det i några utvalda ämnen - och då oftast inom ämnena matematik eller naturvetenskap. Ytterligare några länder har valt att introducera programmering som ämnesöverskridande. Utifrån en analys återfinns två tydliga spår när det gäller hur olika länder har valt att införa programmering i skolan. Det ena spåret lyfter fram att det borde bli ett eget ämne medan det andra lyfter att det ska integreras i redan befintliga skolämnen – och då främst

42 http://www.lararnasnyheter.se/origo/2015/03/09/finland-uppgraderar
43 Finland uppgraderar. Intervju med Linda Mannila i Lärarnas Nyheter. http://www.lararnasnyheter.se/origo/2015/03/09/finland-uppgraderar
44 Intervju via Skype, gjord av Susanne Kjällander 2016-03-01
45 Runeson, P., Andersson, S., Heintz, F., Mannila, L. & Rolandsson, L. Hur skall det svenska utbildningssystemet möta framtidens utmaningar? Helhetssyn på svensk utbildning om digitalisering och programmering. http://swedsoft.se/wp-content/uploads/sites/7/2015/05/2015-01-Utbildning-white-paper.pdf

matematik (The Royal Society, 2012). I Japan har man sedan 2013 års revision av läroplanen lyft in programmering i grundskolan. I en japansk studie visar resultaten att grundskolebarn som fick programmera, med hjälp av robotar, blev positivt inställda till programmering (Yamanishi, Sugihara, Ohkuma och Uosaki, 2015). Estland, Kina och Vietnam har redan infört programmering i skolan och i Storbritanniens nya läroplan ska programmering läras ut från fem års ålder. I de flesta av dessa länder är programmering (här även beskrivet som "information technology") inkluderat i andra skolämnen under skolans tidigare år och används som redskap över ämnesgränserna. Programmering (Information technology/Computer Science) som ett eget ämne uppkommer oftast först i de senare åren i skolan – och då varierar det om det är ett obligatoriskt eller valbart ämne. I några länder, såsom England, har programmering införts som ett eget ämne i grundskolan.

När det gäller vid vilken ålder olika länder valt att börja undervisa i programmering finns det stor variation. Som några exempel kan nämnas Japan där programmering lärs ut från det att eleverna är 12 år, USA (Massachusetts, som ligger i framkant av amerikanska stater avseende programmering i skolan) introduceras programmering vid 14 års ålder, Singapore introduceras programmering först när eleverna är 16 år och i England har man valt att introducera programmering redan när barnen börjar i skolan, alltså när de är 5 år. I vissa länders styrdokument där just programmering är inskrivet som ett eget ämne finns även programmeringsspråken inskrivna medan det i andra styrdokument finns mer flexibilitet kring programmeringsspråken (The Royal Society, 2012,). I höst (2016) får Finland en ny läroplan för grundskolan. Programmering kommer att ingå i kursplanerna för matematik och slöjd och blir därmed inte ett eget skolämne. Programmering kommer att skrivas in i kursplanerna för matematik, årskurserna 1–9, och slöjd , årskurserna 3–9. Det finns ännu inga tillgängliga utvärderingar kring detta.

Exempel på internationell forskning om programmering i urval

I följande avsnitt kommer endast ett litet urval av internationell relevant forskning om programmering i förskola och skola att presenteras. I en metastudie från Singapore, har data från 27 studier, från olika länder, om datalogiskt tänkande[46] i förskola, grundskola, gymnasium och högre utbildning sammanställts och analyserats. Författarna vill, liksom många andra (se t.ex. Heintz et al., 2015) lyfta programmering som något mer än kodning, och använder begreppet datalogiskt tänkande; "*Programming is more than just coding, for, it exposes students to computational thinking which involves problem-solving using computer science concepts*

46 Eng:computational thinking

like abstraction and decomposition." (Lye & Koh, 2014, abstract). Flertalet av de undersökningar som studien analyserar, bygger enligt författarna på en syn på kunskap och lärande, där man ser att de deltagande barnen och ungdomarna skapar något när de programmerar och att detta i sin tur befäster vad de lärt sig (Lye & Koh, 2014). Samtidigt som studien baseras på flera studier, efterlyser författarna ett mer empiriskt underlag, som kan ligga till grund för införande av programmering i utbildningssammanhang.

I en interventionsstudie med för- och eftertester har Kazakoff, Sullivan och Bers (2013) undersökt yngre barns användning av robotar för programmering, i förskolan i Harlem, New York. De använde CHERP, där barnen både använder en digital applikation och analogt material i form av klossar, för att programmera. Efter en veckas intensiv robotprogrammering, fick barnen (samt en kontrollgrupp) göra ett test med så kallade sekvens-kort[47]. Det visade sig att barnen i robotprogrammerings-gruppen fick signifikant högre resultat avseende att berätta historier. I en amerikansk studie där förskolebarn får använda ett fysiskt material och programmera en robot, visar resultaten att barn som deltagit i en veckas intensiv robotprogrammering förbättrar sin förmåga att minnas hur olika förlopp hänger ihop jämfört med en kontrollgrupp (Kazakoff, Sullivan & Bers, 2013). Detta utvärderas med hjälp av ett före-och eftertest med kort med olika bilder, som barnen kunde sätta i ordning, eller i en sekvens.

Bers (2010) beskriver samma arbetssätt som används av Kazakoff, Sullivan och Bers (2013), men lyfter särskilt hur barns arbete med robotar innefattar mycket mer än att bara bygga fysiska artefakter och menar att; *"Bringing robots to "life" involves computer programming. Thus, children learn to create computer programs—algorithms or sequences of instructions that allow robots to move and to sense and respond to their environment."*[48]

Ett genomgående drag, både i svensk och internationell forskning, är att programmeringsspråken och programmeringsprocesserna på olika sätt visualiseras. Detta kan ske genom allt från bilder eller bildflöden på en skärm till tredimensionellt material, så som programmeringsbara klossar eller olika sorters robotar. Kojo (se Regnell & Pant, 2014) bygger på det *visuella* programmeringsspråket *Scala*, vilket gör att barnen lättare kan lära sig programmering. Lego *Mindstorms* är ett annat exempel där barnen programmerar fysiska lego-robotar (se t.ex. Christensen, Fogh och Lund, 2014). För de yngsta barnen används ofta *Bee-Bot* eller *LightBot*, som är insektsliknande robotar, som barnen kan

47 Sekvens-kort innebär här att barnen fick fyra olika bilder att sätta i ordning, så att det blev en historia.
48 http://ecrp.uiuc.edu/v12n2/bers.html

programmera genom en serie knapptryckningar på robotens rygg (se t.ex. Kazakoff et al. 2013). Denna visualisering och mer fysiska del av programmering kan kopplas samman med den förekommande praktiken i Sverige, där programmering ofta sker i form av fysiskt material i både förskola och skola, med t.ex. robotar och olika visuella applikationer. I och med denna utveckling av programmeringsspråk som tydligare bygger på visuella principer (till exempel Scala) blir det enklare för framförallt elever i de lägre åldrarna att lära sig att programmera än det var tidigare.

Transfereffekter

Papert (1980) menade att det fanns många möjligheter för barn att tänka kreativt och logiskt på ett nytt sätt, när de fick programmera med hjälp av Logo turtle. Papert's idéer ifrågasattes när inga empiriska studier kunde finna någon transfereffekt och det gick inte att visa på att programmeringsaktiviteter påverkade barns förmåga att tänka abstrakt (se Kurland, Pea, Clement & Mawby, 1986; Pea, 1983). Samtidigt genomfördes studier vars resultat visade att barns problemlösningsförmåga förbättrades med hjälp av Logo-programmeringsaktiviteter, dock inte andra kognitiva förmågor (Clements & Gullo, 1984). Paperts (1980) arbete med att lära barn programmera i LOGO var delvis ett sätt att ge barnen möjlighet att reflektera kring sitt eget lärande. Denna kunskap skulle sedan kunna användas i andra sammanhang och inom andra ämnen. Studier som har genomförts för att studera transfereffekten är inte enhälliga att eleven kan överföra kunskap mellan olika domäner när de har arbetat med programmeringsaktiviteter (se Voogt et al., 2015). I översikten gjord av Voogt et al. (2015) tas en annan studie upp som delvis kan förklara de motstridiga resultaten kring transfereffekten. Salomon och Perkins (1989) erbjuder en annan förklaring då de undersöker "high-road and low-road transfer" (Voogt et al., 2015. s. 717). High-road transfer är de kunskaper som har högre abstraktion av koncept, processer och idéer kring ett område som eleverna ska lära sig. Om eleverna får möjlighet att reflektera kring sina egen kunskaper och hur dessa kan användas i andra sammanhang i undervisningssituationen kan en transfereffekt möjligtvis uppnås. *"... the high-road transfer is heavily dependent on the instructional practices of the educator, which leads to question whether it is the programming experience that makes the difference or the methodology of the instructors."* (Voogt et al., 2015. s. 718).

Det finns internationella studier med resultat som visar på vissa transfereffekter. I en studie om hur 12–14-åringar i grundskolan i Turkiet programmerar dataspel med hjälp av Scratch, visar eleverna signifikant högre förmåga till att lösa sannolikhetsproblem (Akpinar & Aslan, 2015). Här används både ett före-och eftertest och exempel ur barnens dataspel. Det finns

dock ingen kontrollgrupp i denna studie och det fanns element i barnens spelprogrammering som innefattade sannolikhetslära, vilket gör att det kan diskuteras om det verkligen rör sig om transfereffekter i detta fall. I studier som undersöker datalogiskt tänkande och programmering finns en visst *förgivettagande* att transfereffekter förekommer. Bl.a Kafai och Burke (2013) och Bers, Flannery, Kazakoff och Sullivan (2014), utgår från att programmering i skolan kommer att leda till att barnen förbättrar sin förmåga till problemlösning, kreativitet, logiskt tänkande, såväl som att en förändring gällande jämställdheten mellan kön.

Genus och programmering

Ett flertal av de projekt och initiativ som presenteras i denna översikt (se tidigare avsnitt) finns ett starkt fokus på att väcka intresse hos flickor att lära sig programmering. Initiativtagare utgår ofta ifrån att det finns genusskillnader som på något sätt kan överbryggas genom att flickor uppmuntras att lära sig programmera. Det finns begränsat med forskning kring programmering utifrån ett genusperspektiv. Vi har ändå valt att belysa frågan med hjälp av några studier som angränsar området.

Frågan om kvinnors roll inom datavetenskap har lyfts ur flera perspektiv, bl.a. inom teknofeminismen (se t.ex. Wajcman, 2007). Wajcman menar att trots att kvinnors användning av digitala verktyg är de underrepresenterade som producenter, till exempel som programmerare av tekniska system. För att det ska ske en förändring, behöver fler kvinnor komma in och vara aktiva skapare av tekniska system menar Wajcman (2007). Som jämförelse kan nämnas att vid KTH är endast en tredjedel av de antagna till höstens tekniska program kvinnor[49]. Sullivan och Bers (2013) har undersökt om det finns några skillnader i hur pojkar och flickor i en amerikansk förskola löser olika uppgifter som är kopplade till programmering, bl.a. i form av robotprogrammering. Sullivan och Bers (2013) hypotes är att dessa skillnader planas ut hos förskolebarn, eftersom dessa barn inte hunnit bli lika "kulturellt indoktrinerade" till de stereotyper som finns i vårt samhälle (Bers & Sullivan, 2013). Resultaten visar att skillnaderna var små mellan hur pojkarna och flickorna lyckades lösa olika programmeringsrelaterade uppgifter i förskolan. Det område där det ändå fanns en viss skillnad var bl.a. att pojkar var bättre på att sätta ihop robotdelar på ett korrekt sätt. Men flickor och pojkar var ändå lika bra på att bygga en stadig robotbil och att sedan programmera den. Sullivan och Bers (2013) drar slutsatsen att programmeringsaktiviteter i förskolan skapar möjligheter att föregå socialisering i

49 DN, http://www.dn.se/ekonomi/jobb-karriar/fler-kvinnliga-ingenjorer-men-det-gar-langsamt/

stereotypa roller. På så sätt kan flickors självförtroende och förtrogenhet när det gäller programmering förbättras och bidra till mer jämlika förutsättningar menar Sullivan & Bers (2013).

I en studie av Howland och Good (2014) studerades 29 stycken 12–13-åriga flickor och något färre pojkar i tre klasser när de för första gången undervisades i programmering. Interventionen skedde under två lektioner i veckan under åtta veckor. De undervisades i programmeringsspråket *Flip* och undervisningen hade inslag av sagoberättande. De flesta eleverna kunde genom att programmera skapa interaktiva moment i de spel som de skapade. För- och eftertester visade en signifikant förbättring i hur de kommunicerade datalogiskt efter att ha använt Flip. Artikeln lyfter framocksåtrenden att flickorna skrev mer komplexa kommandon än pojkarna.

Flera studier såsom till exempel Kelleher et al. (2007) och Denner et al. (2012) visar just att flickor blir i högre grad mer motiverade att lära sig programmera när aktiviteten innefattar ett berättande. Resultat från en för- och efterenkätstudie (Robertson, 2013) som besvarades av 225 elever visade att flickorna uppskattade programmering i form av spelkonstruktioner, men fortfarande visade resultatet att pojkarna var mer entusiastiska kring programmering. Flickorna uppgav i post-enkätstudien att de dessutom var mindre benägna att studera datavetenskap i framtiden. Robertson (2013) menar att resultaten indikerar att mer forskning behövs kring varför flickor inte blev mer intresserade av att fortsätta med programmering trots att de var intresserade och tyckte om att arbeta med att skapa dataspel (Robertson, 2013, s. 79). Det finns dock studier som visar på det motsatta, där elever blir mer intresserade och positiva till att lära sig mer om programmering. Nedan presenteras några av dessa studier.

Kreativitet, motivation och intresse för programmering

När det gäller motivation och intresse för programmering och datavetenskap och/eller teknik, finns det flera studier som visar hur t.ex. dataspelsprogrammering kan öka elevers intresse för programmering. Robertson och Howells (2008) har genomfört fältstudier under en åttaveckors period där en klass 10-åringar fick skapa egna dataspel. Studien bestod av bland annat intervjuer, fallstudier av ett litet antal barn, enkäter, fältanteckningar och klassrumsobservationer. Deras resultat indikerar att eleverna visade motivation och entusiasm för lärande då de programmerade. Eleverna var bestämda över att de ville nå höga resultat och visade förmågor för såväl självständigt lärande som lärande i grupp. Forskarna menade att eleverna kopplade och använde sitt lärande i nya situationer, men det

framgår inte i vilka nya situationer. Författarna lyfter även att eleverna som skapat dataspel uttrycker positiva uppfattningar om spelprogrammering.

I Grekland har Sapounidis och Demetriadis (2013) jämfört programmering med hjälp av fysiska objekt med visuell programmering, i olika åldersgrupper. Utifrån intervjuer och observationer visar författarna hur särskilt de yngsta deltagarna (5-8 åringarna) och flickor var mest positiva till och hade lättast att använda fysiska objekt när de arbetade med programmering. Samtidigt lyfter författarna att många i åldersgruppen 11-12 tyckte att mer grafiska arbetssätt var enklare att använda, särskilt om barnen hade tidigare erfarenhet av digitala resurser och programmering med hjälp av fysiska objekt.

Yamanishi, Sugihara, Ohkuma och Uosaki (2015) har undersökt hur japanska grundskolebarn fått lära sig att programmera en microrobot. De har sedan intervjuat eleverna och ställt frågor både om deras förståelse av programmering och om deras uppfattningar samt inställning till programmering och robotteknik. Resultaten visade att barnen var mer positivt inställda till programmering och barnen ansåg också att de hade fått en större förståelse för hur programmering fungerar. Eftersom detta är en intervjustudie och frågan om vad de lärt sig enbart byggde på självskattningar och ja/nej-frågor, kan denna studie inte visa vad barnen lärt sig om programmering, men däremot kan den visa barnens uppfattning om vad de lärt sig och deras attityder kring programmering[50].

Med stöd i det som hittills presenterats i översikten följer nu en forskningsreflektion kring programmering som inslag i den svenska förskolan och grundskolan.

Reflektion och avslutande kommentarer

Nedan följer sammanfattande reflektioner och kommentarer kring den forskning och den erfarenhet om programmering som presenterats i översikten. I arbetet med översikten har vi tillämpat en vid syn på hur begreppet programmering kan förstås. Detta för att det till stor del saknas forskning om programmering i skolan utifrån ett lärandeperspektiv. Detta är i sig inte förvånande med tanke på att programmering i svensk grundskolas styrdokument inte har funnits med sedan 1980-talet. Något som tydligt framkommit såväl i Sverige som internationellt, är att det saknas en samstämmighet kring begrepp som används för att diskutera barns och elevers programmering i förskola och skola. Vi vill därför inleda med att diskutera begreppen programmering och datalogiskt tänkande.

50 http://onlinelibrary.wiley.com/doi/10.1002/cae.21582/epdf

Programmering och datalogiskt tänkande

Vad som avses med begreppet programmering inom ramen för utbildning skiljer sig delvis åt i de texter som tas upp i översikten. Som ett exempel kan nämnas att i vissa sammanhang kan programmering vara endast skrivande av kod medan i andra sammanhang tas ett vidare perspektiv på programmering där även problemformuleringen, val av lösning, själva kodningen, testning och även dokumentationen räknas in i programmeringsaktiviteten (Björk, 1983; Mannila et al., 2014; Heintz et. al. 2015). Oftast kallas denna vidare syn på programmering datalogiskt tänkande. Nygårds, Parnes och Regnell (2015) diskuterar frågor om datalogiskt tänkande i relation till programmering och de pekar på att fokus inte bara ska ligga på programmering eller kodning, utan på en bredare uppsättning egenskaper såsom förmågan till kreativ problemlösning, logiskt tänkande, strukturellt arbetssätt och att göra generaliseringar. Jeannette Wing (2008) gör en distinktion mellan datalogiskt tänkande och programmering: "Computer science is not computer programming. Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction." (Wing, 2008, s. 35). På samma sätt menar Heintz et al. (2015) att synen på programmering behöver vidgas där fokus inte bara är på att lära barn och elever att skriva kod, utan att betona förmågor såsom problemlösning, abstrakt och analytiskt tänkande som både tränas och behövs, när barn och elever programmerar. Liknande resonemang förs internationellt: "Computational thinking is typically associated with coding and computer programming, but it is also more than that, involving solving problems, designing systems, and understanding human behaviour" (European Schoolnet, 2014, s. 7). Mannila et al. (2014) har använt en definition av datalogiskt tänkande som formulerades av ISTE (International Society for Technology in Education) och CSTA (American Computer Science Teachers Association). Nio grundläggande begrepp används[51] datainsamling, analys av data, representation av data, dekonstruktion av problem, abstraktion, algoritmer, automatisering, parallellisering och simulering. När det gäller problemlösning innefattas här förmågan att formulera problem och skapa villkor; att på ett logiskt sätt organisera och analysera data; att kunna

[51] 51 "The CSTA and ISTE have proposed a definition of CT suitable for use in K-12 education, identifying nine essential concepts: data collection, data analysis, data representation, problem decomposition, abstraction, algorithms, automation, parallelization and simulation." (Mannila et al., 2014, s. 2).

representera något genom abstraktioner så som modeller och simuleringar; att automatisera lösningar genom att tänka utifrån algoritmer; att kunna identifiera, analysera och implementera olika lösningar på det mest effektiva sättet. Vidare menar Mannila et al. (2014) att dessa färdigheter när det gäller problemlösning även kan vara användbara i andra ämnen och inte bara i matematik och naturorienterande ämnen.

Begreppet datalogiskt tänkande har kritiserats (bland annat av Jones, 2011) för att vara för abstrakt och för att det inte avgränsas tillräckligt utan beskrivs som att det kräver tänkande på multipla nivåer av abstraktion (se till exempel Wing, 2006; 2008). Jones (2011) menar att det därmed blir oklart hur datalogiskt tänkande skiljer sig från andra typer av kognition och eftersöker därmed en jämförelse och tydligare skillnad vad som åsyftas med "datalogiskt tänkande". Forskning som lyft begreppet har sammanfattats i en artikel av Grover och Pea (2013). Kritiska röster har även höjts kring att sambandet mellan att introducera programmering i skolan därmed skulle öka intresset för programmering och att fler elever sedan skulle välja att studera inom området datavetenskap är långt ifrån belagt. Istället kan vi delvis skönja det motsatta (se till exempel Robertson, 2013; Voogt et al., 2015).

I linje med de studier vi tagit del av (Wing, 2008; Mannila et al., 2014; Heintz et. al. 2015; Voogt et al., 2015) kan vi i föreliggande översikt se att det finns en konsensus kring att datalogiskt tänkande är mer än enbart programmering men att programmering kan användas som ett verktyg för att utveckla datalogiskt tänkande (Voogt et al., 2015, s. 726). Vi kan även skönja ett växande fält där det framåt kommer behövas ytterligare forskning som studerar hur ämnet programmering i skolan kan förstås och beskrivas som ett eget ämne eller integrerat med andra ämnen.

Vi vill också betona och lyfta fram att tidigare studier visar bland annat att undervisningen generellt blir mer inriktad på problemlösande aktiviteter, informationssökning och kreativt skapande när digitala verktyg används i högre grad i skolarbetet (se till exempel: Binkley, et al., 2012; Griffin et al., 2012; Fleischer, 2013; Åkerfeldt, 2014). Skillnaden eller förhoppningen är, som vi tolkar det, att genom att lära eleverna att programmera och träna sig i datalogiskt tänkande kan den kunskapen "spilla över" på andra ämnen och aktiviteter i skolan vilket skulle gynna elevernas lärande i flera ämnen. Denna så kallade transfereffekt mellan olika ämnen finns det ännu inga tydliga enhälliga resultat om. Tvärtom är resultaten motstridiga (se nedan).

Argument för att programmering skulle kunna bli ett inslag i förskola/skola

Det finns flera återkommande argument för att införa inslag av programmering i skolan. Den pågående debatten är inte nyanserad på denna punkt och de flesta röster som hörs menar att programmering ska in i skolan. Här vill vi lyfta några av de argument som vi bedömer vara baserade på forskning och erfarenhet, om än i liten utsträckning. Vi vill därför börja med att peka på:

- Det finns begränsad *vetenskaplig granskad empirisk forskning* om programmering i svensk skola. Att införa programmering i förskolan och skolan i nuläget kan därför inte sägas vila på vetenskaplig grund utifrån en svensk skolkontext.
- Det finns begränsad *beprövad erfarenhet* av programmering i svensk skola. Att införa programmering i förskolan och skolan kan därför inte sägas vila på svensk beprövad erfarenhet.
- Det finns en del värdefulla erfarenheter av programmering i svensk skola som kan tjäna som goda exempel.

Trots ovanstående punkter kommer vi nedan att lyfta argument för inslag av programmering i skolan utifrån det som framkommit i arbetet med översikten.

Demokratiaspekten

Översikten visar att det i dagens samhälle är av stor vikt att barn och elever förstår hur datorer och tekniska system fungerar[52]. Det är viktigt att de förstår hur programmeringsspråk kan användas för att designa system samt hur dessa system påverkar deras vardag.

Regeringens digitaliseringskommission (2015) framför att programmering som ett tydligt inslag i grundskolan kan ge eleverna grundläggande kunskaper för att kunna hantera sin digitala vardag och använda digitala verktyg samt att träna logiskt tänkande. Införandet av programmering handlar också om att skapa ökade möjligheter för både lärarna och eleverna att få tillgång till ett tekniskt språk som gör att de i högre grad kan kommunicera med teknisk kunnig personal. Att få tillgång till en teknisk språklig domän gör det möjligt för lärare och elever att i högre grad ställa krav på tekniska system (se Voogt et al., 2015).

52 Som till exempel Internet, lagring av filer i molnet, sökmotorer.

Ett motiv som ofta lyfts är att det är viktigt att alla ska få samma möjlighet att lära sig programmering. Detta går i linje med hur Heintz, Mannila, Nygårds, Parnes och Regnell (2015) ser på programmering som en demokratisk rättighet i ett samhälle som till stor del bygger på olika tekniska system och där även en framtida arbetsmarknad kan komma att domineras av att handha och designa teknologiska system. Regeringens digitaliseringskommission (2015) menar att programmering i skolan kan väcka både pojkars och inte minst flickors intresse och lust för tekniska frågor för att med tiden få fler att söka sig till tekniska utbildningar. Dock är detta samband långt ifrån klarlagt som tidigare nämnts.

En tydlig slutsats som ändå kan dras från denna översikt är att programmering kan ses som en del av ett större arbete kring att höja och förstärka undervisningen och elevernas kunskap kring användningen av digitala verktyg och hur de påverkar deras vardag idag och i framtiden.

Genusaspekten

När det gäller frågor om programmering utifrån ett genusperspektiv råder det brist på kartläggning och forskning inom området. Trots att man inom de forskningsrelaterade projekten (se t.ex. Heintz, et al, 2015) lyfter frågan om vikten av ett jämställt lärande i programmering, bygger aktiviteter för flickors programmering på privata, ideella eller företags initiativ, så som Tjejhack, Girl Tech Sweden och Tekla. Här har man ännu inte studerat på vilket sätt dessa aktiviteter påverkar deltagarnas intresse för programmering över tid och i andra sammanhang.

Det finns ingen forskning som visar genusskillnader vad gäller kompetens att programmera men det finns tendenser i forskning som tyder på att flickor tidigt positioneras eller positionerar sig som mindre intresserade av att programmera varvid en introduktion av programmering i förskolan skulle kunna förespråkas. Andra orsaker till att kvinnor inte väljer IT som yrke kan vara att branschen uppfattas som manligt kodad, att kvinnor uppfattar att de har bristande kännedom om yrket, att det saknas kvinnliga förebilder samt att de inte fått adekvat undervisning i tidig ålder i skolan (Insight intelligence, 2015). Här finns ännu inga vetenskapliga enhälliga resultat som pekar i den riktningen. Med stöd i internationell forskning menar vi att flickors intresse för teknik kan gynnas på olika sätt när programmering används för kreativa och skapande verksamheter, såsom till exempel design av spel där berättande ingår som en central del av arbetet (se till exempel Robertson, 2013; 2012). Å andra sidan finns forskningsresultat som visar att flickor efter att ha

programmerat själva sade sig vara mindre benägna att studera datavetenskap i framtiden (Robertson, 2013). Det är viktigt att i högre grad skapa utrymme för flickor att lära sig och skapa intresse för teknik och programmering, både i arbetslivet och i högre utbildning. Utifrån ovanstående genomgång menar vi att det finns möjligheter att skapa ett större intresse för teknikorienterade ämnen genom att introducera programmering i förskola/skola. Det finns dock ingen empirisk vetenskapligt granskad forskning i en svensk kontext som stödjer dessa resonemang.

Transferaspekten

När det gäller sambandet mellan att eleverna programmerar och så kallade transfereffekter inom och mellan andra ämnen, finns de få undersökningar som stödjer att det skulle ske. Transfereffekter lyfts som ett argument för att införa programmering i skolan, men forskningsresultaten är få och motstridiga. Vi upplever att det finns en, ibland ogrundad, tro på att programmering i sig ska göra att barn lär sig mer och på andra sätt. Det finns få empiriska studier som understödjer detta samband. I denna översikt har internationell forskning vars empiriska resultat visar på möjliga transfereffekter lyfts fram, där man tänker sig att barns lärande i andra ämnen gynnas av att de undervisas i datalogiskt tänkande och genom att barnen får engagera sig i programmering. Vad det däremot kan sägas finnas någon form av enighet kring är att om transfereffekter ska uppnås behöver undervisningen iscensättas så att detta blir möjligt. Det sker inte automatiskt (se till exempel Voogt et al., 2015; Rolandsson, 2015).

Konkurrensaspekten

Det är inte ovanligt att programmering i skolan lyfts i ett nationalekonomiskt perspektiv där tillväxt, ekonomi, hållbarhet, konkurrens och Sveriges framtid diskuteras. Detta har inte varit vår inriktning i översikten som fokuserar på barns lärande, men argumentet är värt att nämna. Rolandsson (2015) menar att programmering i läroplaner varierar från land till land men han poängterar också att man kan förvänta sig att alla elever behöver gå kurser i programmering: "...as we are living in a world built on ideologies pushing for a democratic progress, it could be expected that all pupils/students need to attend courses in computing." (Rolandsson, 2015, s 24).

Kritiska aspekter kring införandet av programmering i förskola/skola

Det råder för närvarande i Sverige såväl som internationellt en relativ samstämmighet kring införandet av programmering, såväl som datalogiskt tänkande, i skolan varvid en nyanserad debatt delvis saknas. I samband med genomsökningen av forskning och erfarenhet har

dock några utmaningar framkommit som delvis kan tolkas som kritik mot att införa programmering i skolan. Ett antal utmaningar lyfts fram i såväl internationell forskning som av erfarenhet när det gäller införandet av programmering i skolan. Vi anser att dessa kan vara av värde att kortfattat presentera eftersom den svenska skolan nu kan tänkas stå inför dessa utmaningar.

Brist på forskning

Denna översikt visar på bristen av forskning om programmering i skolan och det i sig skulle kunna tolkas som ett skäl att avvakta med att skriva in programmering i kursplanerna, eftersom undervisningen ska bygga på vetenskaplig grund.

Det finns vissa brister när det gäller tydlig koppling till forskning i flera av de dokument som ligger till grund för satsningar och beslut om införandet av programmering i skolan. European schoolnet[53] (2014) har inga empiriska data, endast rapporterade uppgifter om hur och i vilken utsträckning programmering förekommer i olika europeiska länder. Här hänvisas till experter som menar att programmering leder till andra värdefulla förmågor, användbara även om man inte ska arbeta med programmering. European Schoolnet lutar sig inte på empiri avseende eventuella effekter eller lärande i samband med programmeringsaktiviteter i förskola och skola. Programmering beskrivs likväl som en nyckelkompetens: ”Coding is becoming increasingly a key competence which will have to be acquired by all young students and increasingly by workers in a wide range of industries and professions. Coding is part of logical reasoning and represents one of the key skills which are part of what is now called “21st century skills” (2014, s 4). Genomgående tas ett perspektiv där behovet av framtida arbetskraft sätts i centrum.

Frågan blir i nuläget om man i Sverige bör avvakta med att införa programmering som ett inslag i förskola/skola tills mer forskning genererats och publicerats. Hur denna forskning ska kunna genereras i en svensk skolkontext om programmering inte införs blir då nästa fråga.

Brist på lärare som kan undervisa i programmering

Översikten har vid upprepade tillfällen fört fram att Sveriges pedagoger ännu inte har den kompetens, utbildning och erfarenhet som kan krävas för att undervisa i programmering.

53 European Schoolnet finansieras och organiseras delvis i samarbete med marknadskrafter såsom Grand Coalition for Digital jobs och eSkills for Jobs 2014. Detsamma gäller flera aktörer kopplade till implementeringen av programmering i Storbritannien; exempelvis delfinansieras Barefoot Computing av Google och IBM.

Rolandsson (2015) som belyser vikten av att programmering som ämne ska bygga på lärares ämneskunskaper såväl som didaktiska kunskaper menar att: "*Som expert på det ämnesdidaktiska ämnesområdet vill jag därför rekommendera den svenska regeringen att skapa möjligheter för lärare att fördjupa sina ämnesdidaktiska kunskaper inom datavetenskap.*" (Rolandsson, 2015, s 60). Även Voogt et al., (2015) lyfter frågan kring att lärare behöver kompetensutvecklas inom området om de ska undervisa i programmering.

Enligt rapporten "Computing our future" (European schoolnet, 2014) är det viktigt att lärare fortbildas och får stöd i sin undervisning i programmering och datalogiskt tänkande så att elevers förståelse och lärande utvecklas. I England pågår för närvarande massiva fortbildningsinsatser för befintliga lärare i engelska grundskolor. Heintz et al. (2014) lyfter också problematiken med att lärare till stor del saknar kompetens i programmering. De efterlyser också fler möjligheter inom lärarutbildningen där lärare får lära sig mer om datalogiskt tänkande, datavetenskap och programmering (Heintz et al., 2014). När det gäller lärarfortbildning kring programmering, finns det olika aktörer, både företag, stiftelser och andra organisationer som tillhandahåller kurser, workshops och onlineutbildningar, som till exempel Teacherhack[54] och webbstjärnan[55]. Idag återfinns det få inslag av programmering inom Sveriges lärarutbildningar[56].

Här vill vi framhålla att det blir viktigt att avsätta resurser för fortbildningsinsatser för förskollärare och lärare som eventuellt kommer att undervisa i programmering. Vi vill även poängtera att lärarutbildningarna också måste förändras och även lärarutbildare behöver kompetensutvecklas.

Brist på digitala resurser

I Finland ser man det som en stor utmaning att ta fram material för programmering i ämnena matematik och slöjd. I England har man löst detta genom att anlita *Barefoot Computing*[57], British Telecom, som tillhandahåller material och workshops. Eftersom programmering är nytt råder det brist på läromedel, men vi kan i översikten se att samma program eller appar återkommer i såväl empiri, i forskningen som i projektbeskrivningar.

54 http://teacherhack.com/programmering

55 http://kurs.webbstjarnan.se

56 Samtliga lärarutbildningar har ett gemensamt mål när det gäller arbetet med IKT inom lärarutbildningen. Studenterna ska "visa förmåga att säkert och kritiskt använda digitala verktyg i den pedagogiska verksamheten och att beakta betydelsen av olika mediers och digitala miljöers roll för denna." (Examensordningen SFS 2010:541).

57 http://barefootcas.org.uk/

Inte bara verktygen för lärande kan tänkas bli en utmaning för den svenska skolan, utan även verktygen för bedömning av elevers programmering. Forskare vid Universitetet Rey Juan Carlos har identifierat och presenterat hinder för införandet av programmering i skolan med stöd i ett forskningsprojekt i Navarra. Ett sådant hinder är att det saknas verktyg för att bedöma elevers programmeringsprojekt (Calao, L.A., Moreno-León, J., Ester Correa, H. & Robles, G., 2015).

När och hur kan programmering införas i förskola och skola?

Med stöd i det som presenterats i översikten vill vi här lyfta följande:

- Med stöd i forskning och erfarenhet som presenteras i denna översikt föreslås att programmering införs ämnesövergripande istället för som ett separat ämne i skolan.
- Översikten presenterar projekt där barn och elever i olika åldrar programmerar. Programmering kan alltså även ingå i förskolans verksamhet.
- Översikten presenterar många ideella och vinstdrivande organisationers initiativ för barn och ungas, och särskilt flickors, programmering utanför skolans kontext. En fråga blir här om, och i så fall hur, skolan kan dra nytta av dessa initiativ.

Programmering ligger nära matematik, men även estetiska ämnen och samhällskunskap diskuteras som möjliga inramningar i de projekt som lyfts fram i den här översikten. I en Trippel Helix workshop[58] presenteras en enkätundersökning som visar hur lärare anser att programmering kan anses passa inom ramarna för ämnena teknik, matematik och naturorienterande ämnen. Vi menar att programmering bör införas ämnesövergripande i ett urval skolämnen och inte som ett enskilt ämne. Det blir därmed viktigt att skapa ett tydligt innehåll för de olika ämnena och samsyn kring vad som ska ingå för att se en progression i ämnet. Som framgår ovan har det skett en förskjutning av hur datakunskap, datalära och senare digital kompetens skrivits fram i olika läroplaner. Vi vill i det här sammanhanget lyfta att när ett område är ämnesövergripande, såsom digital kompetens har blivit, finns det en risk att ingen har ett ansvar för att driva området framåt vilket kan göra att det inte i lika hög grad prioriteras i undervisningen. Därför är det viktigt att den erfarenhet som finns och skapas idag i de lokala, privata och ideella sammanhangen tas

58 Trippel Helix Workshop. Input till Skolverkets arbete med de nationella IT-strategierna för skolan #TrippelHelix. Göteborg, 5 februari 2016. WiFi U: ChSRAB-C L: ChalmersKonferens http://www.ida.liu.se/~frehe08/TrippelHelix-Intro-20160205.pdf

tillvara i en skolkontext. Det är också viktigt att det byggs upp samarbeten på lokal nivå mellan skolämnen där programmering möjligtvis kan komma att ingå. Det finns många initiativ, drivna av företag, med syfte att inspirera barn och unga att lära sig mer om programmering. Eftersom det i nuläget inte finns några statliga riktlinjer, stöd eller plattformar som kan stödja pedagoger och elever inom området, är de ideella, privata och företagens initiativ de stöd som står att finna. Här ser vi ett behov av nationella riktlinjer och stöd till förskollärares och lärares grundutbildning och kompetensutveckling. Vi ser även behov av att skapa hjälpmedel, inspiration och handledning, baserad på aktuell forskning och utan kommersiella vinstintressen.

En liten del forskning och en hel del erfarenheter från projekt kring yngre barns programmerande har presenterats i översikten. I rapporten "Computing our future" (European Schoolnet, 2014) omnämns att i Navarra i Spanien har yngre barn studerats när de programmerar. Resultatet visar att barn är förberedda att programmera i unga åldrar samt vilken effekt det kan ha på lärande i andra skolämnen. Resultaten som tyder på transfereffekter på yngre barns lärande i andra ämnen omnämns, men återfinns ännu inte i någon vetenskapligt granskad publikation.

Vad bör beaktas i samband med en implementering av programmering i förskola/skola?

Diskussionen kring syftet med att införa programmering i skolan bör tas på alla nivåer: vad vill regeringen, kommuner, skolan och läraren åstadkomma med att elever ska lära sig programmera?

Rolandsson (2015) hänvisar till Donaldson och Knupfer (2001) och Pelgrum (2001) som menar att det länge har funnits en spänning inom programmering mellan didaktiska och tekniska frågor där IT-teknikaliteter dominerar den didaktiska designen av undervisningen. Därför blir det viktigt att lyfta behovet av att diskutera och utveckla metodfrågor (Cuban, 1986, 2001). Rolandsson efterlyser vidare en ämnesdidaktisk diskussion mellan lärare, forskare och lärarutbildare för att alla elever ska få en möjlighet att lära sig programmera och att undervisningen behöver utvecklas.

Utifrån den genomförda översikten förordas att programmering förstås i vid bemärkelse och i linje med begreppet datalogiskt tänkande. Här behöver ytterligare samordning göras med hur skolan arbetar med elevers digitala kompetens i sin helhet.

Avslutande kommentarer

Att införa programmering, eller som vi förordar i den här översikten datalogiskt tänkande, i förskola och skola är rimligt. Det är nödvändigt för elever att lära sig hur tekniska system fungerar för att de ska kunna delta i ett samhälle där människor i allt högre grad använder digitala verktyg såväl i arbetslivet som på fritiden.

Vad som är kritiskt och av stor vikt att diskutera vidare är didaktiska aspekter kring varför, vad, hur och på vilket sätt programmering/datalogiskt tänkande ska implementeras i skolan. Tidigare erfarenheter kring implementering av digitala verktyg i skolan är att det krävs framförallt ett tydligt ledarskap och kontinuerlig kompetensutveckling av lärare (Grönlund, 2014; Åkerfeldt et al., 2013).

Att införa programmering i förskola och grundskola kan i dagsläget inte sägas vila på vetenskaplig grund eftersom vetenskapligt granskad forskning saknas i Sverige[59]. I översikten kan vi konstatera att vi endast har hittat en vetenskapligt granskad text baserad på empiriska resultat. Att införa programmering i förskola och grundskola kan i dagsläget inte heller sägas vila på beprövad erfarenhet. Vi har hittat och beskrivit ett flertal projekt men med tanke på att det finns ca 250 tusen lärare[60] är det inte många som idag ägnar sig åt programmering i sin undervisning. I och med att ämnet är relativt nytt i en skolkontext kan man inte heller förvänta sig att det ska finnas gedigen forskning och beprövad erfarenhet att ta stöd i. Forskning och beprövad erfarenhet tar tid och i ett digitaliserat samhälle är förändringstakten hög. I brist på vetenskapligt granskad forskning och beprövad erfarenhet skapas här ett utrymme för företag och ideella organisationer att vara med och framförhandla förskolans och skolans utformning av programmering som inslag i undervisningen. I Sverige, liksom i de andra länder vi haft som utgångspunkt i denna översikt, finns därför många marknadskrafter och frivilliga och ideella organisationer som är aktörer i frågan kring barns och elevers programmering.

Vi önskar därför avsluta med en förhoppning om att pedagoger i förskola, fritidsverksamhet och skola i högre grad börjar dokumentera sitt arbete med programmering med barn och elever. Detta för att skapa goda exempel för andra pedagoger att inspireras av. En merpart av all forskning som presenteras i denna översikt finns inom disciplinerna datavetenskap och systemvetenskap. I många fall har dessa forskare inlett samarbete med pedagoger som arbetar med programmering. Vi ställer oss

59 Vilket inte är konstigt i sig då programmering inte har funnits i skolan sedan 1980-talet och då endast på gymnasiet.
60 http://www.lararnasnyheter.se/chef-ledarskap/2014/03/31/bristen-pa-larare-hur-stor-blir

frågan var forskare i ämnena barn- och ungdomsvetenskap, didaktik och pedagogik står i förhållande till programmering och datalogiskt tänkande i förskola/skola. Vi hoppas därför att didaktisk, pedagogisk och barn- och ungdomsvetenskaplig akademisk forskning iscensätts i såväl förskola som skola för att det inom en snar framtid ska kunna bedrivas undervisning i programmering som vilar på vetenskaplig grund i Sveriges förskolor, skolor och fritidsverksamheter.

Referenser

Akpinar, Y., & Aslan, U. (2015). Supporting Children's Learning of Probability Through Video Game Programming. *Journal of Educational Computing Research, 53*(2) 228–259.

Barab, S. & Squire, K. (2004). Design-Based Research: Putting a Stake in the Ground. The *Journal of the Learning Sciences*. 13(1), 1–14

Bers, M.U. (2010). The TangiableK robotics program: Applied computational thinking for young children. *Early Childhood Reasearch and Practice , 12* (2), 1524-1539.

Bers, M.U., Flannery, L., Kazakoff, E.R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education archive 72*, 145-157.

Björk, L-E. (1983). Datorers intåg i svensk skola. *Nämnaren*. Nr. 1 s. 32. http://ncm.gu.se/pdf/namnaren/3235_83-84_1.pdf

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining Twenty-First Century Skills. I P. Griffin, B. McGaw, & E. Care, (Red.), *Assessment and Teaching of 21st Century Skills* (ss. 17-66). Dordrecht: Springer Netherlands.

Björkman, K. (2015). *Finland uppgraderar*. Intervju med Linda Mannila i Lärarnas Nyheter. http://www.lararnasnyheter.se/origo/2015/03/09/finland-uppgraderar

Brookshear, J.G. (2009). *Computer science: an overview*. (10. ed.) Boston: Pearson Addison-Wesley.

Calao, L. A., Moreno-León, J., Correa, H. E., & Robles, G. (2015). Developing Mathematical Thinking with Scratch: an Experiment with 6th grade students. *Design*

for Teaching and Learning in a Networked World (pp. 17-27). Springer International Publishing.

Christensen, D.J., Fogh, R., & Lund, H.H. (2014). Playte, a tangible interface for engaging human-robot interaction. *Robot and Human Interactive Communication*, RO-MAN: The 23rd IEEE International Symposium, 56-62.

Clements, D.H., & Gullo, D.F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, *76*(6), 1051-1058.

Cuban, L. (2001). *Oversold and underused: computers in the classroom*. Cambridge, Mass.: Harvard University Press.

Cuban, L. (1986). *Teachers and machines: the classroom use of technology since 1920.* New York: Teachers college Press.

Denner, J., L. Werner, L. &Ortiz E. (2012) Computer games created by middle school girls: can they be used to measure understanding of computer science concepts? *Computers and Education*, 58 (1) (2012), pp. 240–249

Digitaliseringskommissionen. (2015). *Digitaliseringens transformerande kraft – vägval för framtiden.* (SOU 2015:91). Stockholm. https://digitaliseringskommissionen.se/wp-content/uploads/2015/11/SOU-2015_91_till-webb.pdf

Donaldson, J. A. & Knupfer, N.N. (2001). Education, learning, and technology. In P. L. Rogers (Red.). *Designing instruction for technology-enhanced learning.* Hershey, PA: Idea Group Pub.

European Schoolnet. (2015). *Computing our future – Computer programming and coding Priorities, school curricula and initiatives across Europe.* http://fcl.eun.org/documents/10180/14689/Computing+our+future_final.pdf/746e36b1-e1a6-4bf1-8105-ea27c0d2bbe0

Farkell-Bååthe, S. (2000). *Datorn som pedagogiskt hjälpmedel: effekter och erfarenheter av datorstöd i matematik*. Stockholm: Institutionen för individ, omvärld och lärande, Lärarhögskolan.

Fleischer, H. (2012). What is our current understanding of one-to-one computer projects: A systematic narrative research review. *Educational Research Review*, 7(2), 107-122.

Griffin, P., McGaw, B., & Care, E. (Red.). (2012). *Assessment and Teaching of 21st Century Skills.* Dordrecht: Springer Netherlands.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher, 42* (1), 38-43.

Grönlund, Å. (2014). *Att förändra skolan med teknik: bortom ”en dator per elev”.* Örebro: Örebro univ. http://www.skl.se/MediaBinaryLoader.axd?MediaArchive_FileID=c4756896- 6797 4b66-957f-f653dfe7e1f9&FileName=Bok+och+antologi+Unos+Uno+- +Att+f%c3%b6r%c3%a4ndra+med+teknik.pdf

Heintz, F., Mannila, L., Nygårds, K., Parnes, P., & Regnell, B. (2015). Computing at School in Sweden - Experiences from Introducing Computer Science within Existing Subjects. 8th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives I *Informatics in Schools. Curricula, Competences, and Competitions /Lecture Notes in Computer Science and General Issues* 9378, 118-130. Publikation hämtad 151215: https://www.ida.liu.se/divisions/aiics/publications/ISSEP-2015-Computing-At-School.pdf

Howland, K., & Good, J. (2015). Learning to communicate computationally with Flip: A bi-modal programming language for game creation. *Computers & Education.* Volume 80, January 2015, Pages 224–240

Håkansson, J. & Sundberg, D. (2012). *Utmärkt undervisning: Framgångsfaktorer i svensk och internationell belysning* (1ed.). Stockholm: Natur och kultur.

Insight Intelligence. (2015). *Unga kvinnor och IT. Unga kvinnors syn på IT som bransch och yrke 2015.* Insight Intelligence AB. https://www.iis.se/docs/Unga_kvinnor_och_IT.pdf

Jones, E. (2011). *The Trouble with Computational Thinking.* Engl. 890J, Professors Buell and Cooley. University of South Carolina. http://csta.acm.org/Curriculum/sub/CurrFiles/JonesCTOnePager.pdf

Kafai, Y.B., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan, 95* (1), 61-65.

Kazakoff, E.R., Sullivan, A., & Bers, M.U. (2013). The Effect of a Classroom-Based Intensive Robotics and Programming Workshop on Sequencing Ability in *Early Childhood. Early Childhood Education Journal*, 41 (4), 245-255.

Kelleher, C., Pausch, R. & Kiesler, S. (2007). Storytelling alice motivates middle school girls to learn computer programming Proceedings of the SIGCHI conference on Human factors in computing systems (2007), pp. 1455–1464

Kurland, D.M., Pea, R.D., Clement, C., & Mawby, R. (1986). A study of the development of programming ability and thinking skills in high school students. *Journal of educational computing research*, *2*(4), 429-458.

Kjällander, S. (2016). *Plattan i mattan – digitala lärplattor och didaktisk design i förskolan 2013-2016.* www.uppsala.se

Lye, S.Y., & Koh, J.H.L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior 41*, 51–61.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L. & Settle, A. (2014). *Computational Thinking in K-9 Education.* ITiCSE '14 Proceedings of the 2014 conference on Innovation & technology in computer science education, 1-29.

Mor, Y. (2010). Embedding Design Patterns in a Methodology for a Design Science of e-Learning. I Christian Kohls & Joachim Wedekind, (Red.), *Problems Investigations of E-Learning Patterns: Context Factors Solutions*, Information Science Publishing, Hershey, PA. (preprint draft).

Nygårds, K. (2015). *Koden till digital kompetens*. Stockholm: Natur & Kultur.

Olafson, L. & Schraw, G. (2006). Teachers' beliefs and practices within and across domains. *International Journal of Educational Research* 45 (1-2), 71–84.

Pajares, M. F. (1992). Teacher's beliefs and educational research: Cleaning up a messy construct. *Review of Educational Research* 62 (3), 307–332.

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas.* New York: Basic books.

Parnes, P. (2015). *Skapande och skaparkultur som drivkraft för kreativt lärande i skolan!* Luleå Tekniska Universitet.
http://www.parnes.com/blog/2015/02/25/Skapande%20och%20skaparkultur%20som%20drivkraft%20f%C3%B6r%20kreativt%20l%C3%A4rande%20i%20skolan%20-%20Prof.%20Peter%20Parnes%2020150225.pdf

Pea, R.D. (1983). Logo Programming and Problem Solving. [Technical Report No. 12.]. American Educational Research Association Symposium: Chameleon in the Classroom: Developing Roles for Computers. Montreal, Canada, April 1983.

Pelgrum, W. J. (2001). Obstacles to the integration of ICT in education: results from a worldwide educational assessment. *Computers & Education* 37, 163–178.

Regnell, B., & Pant, L. (2014). *Teaching programming to young learners using Scala and Kojo.* LTHs 8:e Pedagogiska inspirationskonferens. http://lup.lub.lu.se/record/4780249

Robertson, J., & Howells, C. (2008). Computer game design: Opportunities for successful learning. *Computers & Education, 50* (2), 559-578.

Robertson, J. (2012). Making games in the classroom: benefits and gender concerns. *Computers & Education.* Volume 59, Issue 2, September 2012, Pages 385–398

Robertson, J. (2013). The influence of a game-making project on male and female learners' attitudes to computing *Computer Science Education*, 23 (1) (2013), pp. 58–83

Rolandsson, L. (2015). *Programmed or not. A study about programming teachers' beliefs and intentions in relation to curriculum*. KTH, School of Education and Communication in Engineering Science.

Rose, J. (2015). Kvinnorna bakom datorernas genombrott. *Forskning och Framsteg*. [Tidningsartikel] http://fof.se/tidning/2015/4/artikel/kvinnorna-bakom-datorernas-genombrott

Sapounidis, T., & Demetriadis, S. (2013). Tangible versus graphical user interfaces for robot programming: Exploring cross-age children's preferences. *Personal and Ubiquitous Computing*, 17(8), 1775-1786.

Sullivan, A., & Bers, M.U. (2013). Gender differences in kindergarteners' robotics and programming achievement. *International Journal of Technology and Design Education, 23* (3), 691-702.

Trogemann, G., Nitussov, A.Y., Ernst, W. (red.) (2001). *Computing in Russia: the history of computer devices and information technology revealed*. (1. ed.) Braunschweig: Vieweg.

The Royal Society. (2012). *Shut down or restart. The way forward for computing in UK schools.* Excellence in Science. The Royal Academy of Engineering.

Van Roy, P. (2009). Programming paradigms for dummies: What every programmer should know. *New computational paradigms for computer music, 104*, 9-47.

Voogt, J., Fisser, P., Good, J., Mishra, P. & Yadav. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies* 20, Issue 4, pp 715-728.

Wajcman, J. (2007). From women and technology to gendered technoscience. *Information, Communication & Society*, *10* (3), 287–298.

Wayne, T.K. (2011). *American Women of Science since 1900*. ABC-CLIO Interactive.

Wing J. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–36.

Wing, J. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society* A, 366, 3717–3725.

Yamanishi, T., Sugihara, K., Ohkuma, K., & Uosaki, K. (2015). Programming instruction using a micro robot as a teaching tool. *Computer Applications in Engineering Education*, *23*(1), 109-116.

Åkerfeldt, A., Karlström, P., Selander, S., & Ekenberg, L. (2013). *Lärande i en digital miljö.* Observation av 1:1 (No. 13-007 ISSN 1101-8526). Stockholm: Data- och systemvetenskap, Stockholms Universitet.

Åkerfeldt, A. (2014). *Didaktisk design med digitala resurser: en studie av kunskapsrepresentationer i en digitaliserad skola.* Diss. (sammanfattning) Stockholm: Stockholms universitet, 2014. Stockholm.