13 **Learning Tech**
Tidsskrift for læremidler, didaktik og teknologi

LÆRE
MIDDEL
DK

Computationel Literacy

Kognitive, sociale og materielle aspekter ved teknologiforståelser i skolen

Af Roland Hachmann

Korrekt citering af denne artikel efter APA-systemet
(American Psychological Association System, 7th Edition):
Hachmann, R. (2023). Computationel Literacy. Kognitive, sociale og materielle aspekter ved teknologiforståelser i skolen. *Learning Tech – Tidsskrift for læremidler, didaktik og teknologi*, (13), 78-99. DOI: 10.7146/lt.v8i13.132969

Abstract

Denne konceptuelle artikel indlejrer sig i snitfladen mellem datalogi og didaktik. Artiklen bidrager til det danske forskningsfelt ved at rejse en begyndende diskussion om materialiteters rolle i forhold til det at lære computationel literacy som et delvist grundlag for elevernes teknologiforståelser i skolen. Computationel literacy er i artiklen defineret som færdigheder og dispositioner, der bringes i anvendelse i en teknologimedieret og situeret kontekst med det sigte at opnå et værdsat mål.

Artiklen indledes med en kort skitsering af begrebets fremkomst og derefter en udfoldelse af computationel literacy som et begreb, hvor tre sammenflettede aspekter: det kognitive, det sociale og det materielle udfoldes. Artiklen afsluttes med to eksempler på computationel literacy i en skolekontekst, og hvordan materialitet her spiller en vigtig rolle i forhold til de stof- og indholdskriterier, der er afsættet for undervisningen.

This conceptual article is situated in the cross-section between computer science and didactics. The article contributes to the research field by taking on questions regarding the role of materialities in relation to learning computational literacy in school. Computational literacy is here defined as patterned deployment of skills and dispositions that are applied in a technology-mediated and situated context with the aim of achieving a valued goal.

The article sets out with a brief outline of the emergence of computational literacy as a concept. Then moving on through a more in-depth presentation of the concept in which three intertwined aspects, the cognitive, the social, and the material, unfold. The article concludes with two examples of computational literacy in a school, and how materiality here plays an important role in relation to the subject-content and pedagogical approach.

Computationel Literacy

Kognitive, sociale og materielle aspekter ved teknologiforståelser i skolen

Af Roland Hachmann, UC SYD

Indledning

Denne artikel ønsker at skabe opmærksomhed omkring begrebet computationel literacy og herudfra en begyndende faglig og fagdidaktisk diskussion af materielle aspekters betydning i forhold til undervisning i teknologiforståelsesfagligheden i skolen. Artiklens sigte er ikke at foreskrive principper eller best practices, men i højere grad at etablere et grundlag for diskussioner af forskellige aspekter af computationel literacy. I artiklen tages der afsæt i den amerikanske uddannelsesforsker Andrea diSessas definitioner af begrebet.

Når computationel literacy her fremhæves som et brugbart begreb i forhold til andre (for eksempel computationel tænkning) er det ud fra en betragtning om, at det er en samlende betegnelse for en kommunikativ praksis, der ikke adskiller kompetencer som læsning, skrivning og regning fra hinanden. Derimod samler literacybegrebet disse og sætter dem i relation til hinanden. Derudover har literacybegrebet et blik for den sociale kontekst med et fokus på både de forudsætninger, mennesker har, samt de materialiteter, der er indlejret i de redskaber, der bringes i anvendelse i enkelte aktiviteter. Computationel literacy bliver dermed ikke en generel, overordnet kompetence, men tæt knyttet til den konkrete praksiskontekst, hvori den finder sted.

Artiklen er struktureret således, at der indledningsvist gives et kortfattet historisk oprids af begrebet computationel literacy og dets fremkomst. Dette gøres for at give læseren en fornemmelse for de uddannelsesmæssige bevægelser og tidligere forsøg på at implementere brugen af computeren i skolen – både internationalt og i en dansk sammenhæng. Der gives undervejs enkelte eksempler af spekulativ karakter, der hovedsageligt tager afsæt i kodning. Afslutningsvis gives der to konkrete eksempler fra empiriske undersøgelser i skolen (Hachmann, 2022a; 2022b), hvor computationel literacy bringes i spil.

Da der netop er tale om en konceptuel artikel med eksempler og ikke en afrapportering af empiriske fund, har artiklen ikke metode- el-

ler analyseafsnit per se, men søger gennem eksemplerne at konkretisere tekstens pointer. Eksemplerne stammer fra undersøgelser foretaget i forbindelse med et igangværende forskningsprojekt, finansieret af Det Frie Forskningsråd (DFF). Projektet, *Designing for situated computational thinking with computational things*, undersøger blandt andet computationelle tings betydning for computationel tænkning på forskellige uddannelsesniveauer.

Fra computer literacy mod computationel literacy

Både i og uden for Danmark har der siden 1970'erne været en stor entusiasme omkring computerens rolle, og hvordan den grundlæggende ville forandre måden, der undervises på i skolen. Der har eksempelvis været forsøg på at etablere et fag omkring datalære i skolen (Fischer, Frøkjær & Gedsø, 1972) samtidig med, at der har været forskellige forsøg på at udvikle programmeringssprog og -værktøjer, der kunne lære eleverne at programmere en computer (Naur et al., 2010).

Som Papert udtrykker det i sin bog *Mindstorms*, var antagelsen, at brugen af computere ville give eleverne *objects-to-think-with* (Papert, 1980; Brennan, 2015), som kunne mediere en anderledes og mere kreativ tænkning hos eleven. Både i forhold til at forstå samt løse de faglige problemstillinger, de mødte i skolen. Historisk har det dog vist sig at selv om computeren forblev i klasserummet, så forsvandt begejstringen for at lære eleverne at programmere. I stedet blev computeren mere anset som et værktøj, der skulle understøtte den eksisterende praksis gennem mere effektive og hensigtsmæssige måder at arbejde på. Frem for programmering blev fokus, at eleverne skulle have basiskompetencer og -færdigheder i forhold til at bruge computeren som et funktionelt redskab. Tilgangen blev en Computer Literacy (Molnar, 1979), der omfattede mestring af bl.a. skriveprogrammer, databaseprogrammer og regneark samt basisfærdigheder i forhold til funktioner som indhentning, opbevaring og deling af information (Haigh, 1985).

I Danmark blev det i 1990 obligatorisk at undervise eleverne i og om elektronisk databehandling (edb) i alle skolens fag. I de efterfølgende ca. 20 år blev der igangsat forskellige initiativer for at sikre elevernes computer literacy. Eksempelvis blev der i midten af 90'erne lanceret et Junior pc-kørekort som det danske svar på European Computer Driving License.

Også i fagenes styringsdokumenter kom der fokus på edb. Et blik ind i skolens læreplaner og undervisningsvejledninger viser, at der var et stort fokus på at beskrive informationsteknologiernes plads i fagene som anvendelige redskaber, der kunne understøtte træningen af faglige færdigheder. I nogle fag blev edb anset som en gennemgribende forandringsimpuls, der rejste nye spørgsmål og krav til faget. For eksempel kan man i den daværende undervisningsvejledning for dansk-faget læse:

> ”Edb er i vor tid kommet til at spille en væsentlig rolle i ethvert menneskes sproglige virkelighed med betydelige konsekvenser for både form og indhold. I forbindelse med edb er der opstået nye kommunikationsformer, og informationsteknologiens sprog og begrebsverden rejser nye spørgsmål inden for den almindelige sproglære. Alt dette gør informationsteknologien til en naturlig del af faget dansk. Datamaskinen kan desuden være et nyttigt arbejdsredskab på alle klassetrin, hvor den kan åbne nye muligheder for træning af sproglige færdigheder, for procesorienteret undervisning og for samarbejde og kreativitet i skriveprocessen. (Undervisningsministeriet, 1990)

Når netop dette eksempel fremhæves, er det, fordi det viser, hvordan *datamaskinerne* skabte et fornyet blik på kommunikative situationer og dermed også nye muligheder for nye udtryksmåder. De ansås som en formaterende kraft på menneskets almene sproglige virkelighed og skabte dermed også behov for nye færdigheder og forståelser af deres materialitet. Med andre ord kom der fokus på, at datamaterne lagde op til nye former for literacy.

Selvom det ikke er artiklens hovedfokus, er det vigtigt at nævne, at der sideløbende med edb også var et stort fokus på nye medier (elektroniske multimedier især) og deres indvirkning. Også her bliver det fra midten af 90’erne et fokus i fagene, at eleverne gennem analyser og produktioner skulle lære at forholde sig kritisk til den nye medievirkelighed. Fra 1995 og frem til 2018 kom “it og medier” til at være det dominerede fokus i skolen frem for edb. Computeren og programmering som genstandsfelt trådte i baggrunden til fordel for et fokus på multimediernes kommunikative egenskaber i og på tværs af skolens fag.

I 2016 skete der imidlertid et nybrud, da regeringen med Strategi for Danmarks digitale vækst satte fokus på ”at kommende generationer skulle blive dygtige brugere af IT gennem at forstå, udvikle og analysere IT” (Erhvervsministeriet, 2018). På denne måde ville man sikre, at den enkelte ikke blot skulle kunne deltage i fremtidens digitale samfund, men også være med til at skabe det. Der blev i forlængelse heraf iværksat en række initiativer, herunder et 4-årigt forsøgsprogram, der

skulle afprøve forskellige modeller for styrkelsen af teknologiforståelse i Folkeskolen. Med forsøgsfaget "Teknologiforståelse i Folkeskolen" (Tuhkala, Wagner, Iversen & Kärkkäinen, 2019; Wagner, Iversen & Caspersen, 2020) blev der på en række danske folkeskoler eksperimenteret med forskellige faglige tilgange gennem afprøvningen af i alt 110 prototypeforløb (Slot, Hachmann, Hjorth & Von Sehested, 2021). Disse skulle på forskellig vis give elever mulighed for at arbejde med og forstå teknologier gennem eksempelvis teknologianalyser, digital fabrikation og programmering.

Med forsøgsfaget som afsæt har robotter, mikrocomputere og programmeringssoftware igen fået særfokus, og eleverne skal nu kunne gå bag om og skabe med teknologien. Frem for blot at bruge skal de nu kunne forstå, hvordan formgivningen og indholdet er designet med specifikke intentioner. Eleverne skal således ikke kun besidde computer literacy, men også kunne gennemskue og forholde sig kritisk-konstruktivt til teknologien. De skal ligeledes, som der peges på i undervisningsvejledningen fra 1990 ovenfor, udvikle en computationel literacy, hvor de kan forstå, tænke og udtrykke sig i situerede praksisser medieret af de teknologier, de omgiver sig med.

Computationel literacy

Netop dette argumenterer Andrea diSessa i sin bog *Changing Minds* for. DiSessa peger på et skifte fra computer literacy til *computationel literacy*, og han distancerer sig fra hele idéen om computer literacy som en overvejende færdighedsorienteret brug af computeren. Samtidig udfordrer han Paperts mere individualistiske tilgang til læring, hvor computeren (an object-to-think-with) ses som en kognitiv partner, en "ting i sig selv" (Papert, 1987).

I stedet foreslår diSessa computationel literacy, hvor elevens deltagelse ses som: "a socially widespread patterned deployment of skills and capabilities in a context of material support (...) to achieve a valued intellectual goal" (diSessa, 2001, s. 19).

Ud fra denne definition argumenterer diSessa for, at computationel literacy skal forstås gennem tre sammenhængende aspekter, nemlig et kognitivt, et socialt og et materielt. Helt overordnet omhandler det kognitive aspekt, hvorledes eksternaliserede repræsentationer kan muliggøre og understøtte bestemte forståelser og idéer. Det sociale aspekt påpeger, hvorledes computationel literacy har en sociokulturel og historisk forankring, hvori bestemte normer og værdier fremtræder. Slutteligt henviser det materielle aspekt til de *ting* (Kragelund &

& Otto, 2005) vi omgiver os med, både fysiske og intellektuelle (for eksempel sproget). Selvom de tre aspekter skal ses som indfiltrede er det værd at bemærke, at det især er det materielle aspekt der i denne artikel fremhæves. Dette er ikke ensbetydende med, at de andre aspekter er fraværende, blot at de nedtones en smule. DiSessa påpeger netop, at det er gennem undersøgelser af alle tre aspekter og deres sammenhæng, der kan skabes en begyndende forståelse for, hvad og hvordan computationel literacy er et centralt perspektiv i skolen, når forskellige former for teknologier bliver centrale medaktører.

Det materielle aspekt – ting vi har, og ting vi gør

Siden den materielle vending (Hicks & Beaudry, 2010) har forskningsfelter som Science and Technology Studies (Danholt & Gad, 2021) og materielle kulturstudier (Kragelund & Otto, 2005) fremhævet materialiteters betydninger i forhold til menneskets liv og væren i verden. Materialiteter ses i disse forskningsmiljøer som et vigtigt aspekt, da de tjener som et relationelt bindeled mellem mennesker og omgivelserne. Materialitet bliver dermed et begreb, der både dækker over fysiske tings stoflighed (ting vi har), men også som de praksisser (ting vi gør), der udvikler sig i relation til både fysiske og ikke-fysiske ting.

Det materielle aspekt i computationel literacy fokuserer primært på eksternaliserede repræsentationer, herunder symboler og tegn, hvori vi som mennesker kan fastholde aspekter af vores tænkning og sociale praksisser på en måde, der gør dem reproducerbare, transporterbare og manipulerbare. Der er med andre ord tale om materialisering af dele af vores tænkning og erfaringer, således at de gennem deres konkrete og materielle form kan fastholdes, omskabes, deles og forhandles i forhold til de kommunikative praksisser, vi deltager i på forskellige tidspunkter (diSessa, 2001).

Det materielle aspekt af computationel literacy omfatter altså, at de ting, vi gør noget med, består af bestemte symbolske repræsentationer og har bestemte anvendelsesmuligheder, fortolkningsrammer, konventioner og regler.

Det er netop disse specifikationer, der gør det muligt at sætte ydre grænser for, hvad computationel literacy er og ikke er i skolen. Computationel literacy handler ikke om blot at kunne læse, skrive eller regne på en computer, men derimod om helt bestemte praksisser og aktiviteter, hvor forskellige typer af computere eller software indgår som en del af deltagelsen.

For eksempel er kodning (ting vi gør) en computationel literacy-aktivitet i et softwareprogram (ikke-fysisk ting vi har), der afvikles på en computer (fysisk ting vi har). Fra et literacyperspektiv er et grundelement her, at der anvendes et særligt skriftsprog, der afspejler måden at udtrykke sig kodningsmæssigt på. Alt efter hvilket sprog man programmerer i, er der særlige grammatiske og syntaktiske reg-

ler, som man er nødt til at følge. Der er således et leksikalt system og et alfabet, hvor bestemte tegn, ord og termer betyder noget helt bestemt og anvendes på en måde, som adskiller sig fra andre skrivesammenhænge, eleverne indgår i.

Udover disse tekniske grundelementer er der nogle mere overordnede perspektiver, der har betydning for computationel literacy. Disse indbefatter eksempelvis kodningsgenrer og æstetiske udtryksmåder (Bers, 2021). De er i højere grad fleksible og medvirkende til at udfordre, udvide eller fastholde elevens muligheder for at udtrykke sig, når der arbejdes med kodning. Når elever bliver fortrolige med ét eller flere programmeringssprog, vil de være i stand til at ordne og opbygge deres kode efter bestemte æstetiske principper, der gør deres kode mere effektiv, det vil sige bedre læsbar og anvendelig for andre (Soon & Cox, 2021).

Som eksempel på ovenstående kan programmeringssproget Scratch, som vinder større og større indpas i skolen, bruges. Scratch bygger på en række bestemte idéer om, hvad programmering og kodning for børn er, og hvordan dette skal læres (Maloney, Resnick, Rusk, Silverman & Eastmond, 2010). Fra et læringsteoretisk perspektiv bygger Scratch på samme konstruktionistiske grundidé som LOGO (Papert, 1980). Eleven skal gennem erfaringer og engagement konstruere sin viden – ofte i et læringsmiljø sammen med andre, der kan afprøve og give feedback. Selvom Scratch kan bruges til forholdsvis kompleks programmering, er formålet at introducere programmering og kodning for elever, der ikke har tidligere erfaringer eller forudsætninger herfor. Dette har haft betydning for designet (Kafai & Resnick, 1996, s. 3), og hvordan forskellige funktioner konkretiseres i brugergrænsefladen. Grundtanken er, at det skal være nemt for eleven at udtrykke kreative idéer. Dette gøres ved, at eleverne stifter bekendtskab med kodning gennem visuelle repræsentationer og simple funktioner, der gradvist kan udbygges, når eleverne oplever et behov herfor.

Et eksempel på kompleksitetsreduktionen er måden Scratch arbejder med *variable* på. Variable er ofte noget, der er svært at forstå for nybegyndere, fordi de udfører underliggende opgaver i et program. Det kræver en relativ høj abstraktionsevne at forstå, hvordan deres indhold (værdier) hentes frem på specifikke tidspunkter i programmet. Derfor er en designfeature i Scratch, at den visuelle flade synliggør, hvad de enkelte variable indeholder (tal eller tekst) og knytter an til, når programmet eksekveres.

Fra et mere teknisk perspektiv bygger nyere udgaver af Scratch på JavaScript. Det betyder, at der er helt specifikke syntaktiske regler forbundet med, hvordan man får programmet til at afvikle kommandoer på computeren. Symboler som eksempelvis =, #, “ “, / eller { } har

en bestemt grammatisk betydning, lige som bestemte ord som for eksempel *script, loop, div, funktion, værdi* og *variable* får en helt specifik sproglig og syntaktisk betydning. I Scratch møder eleverne ikke direkte denne tekstbaserede kode. I stedet er kodningselementerne repræsenteret gennem forskellige blokke (*bevægelse, udseende, hændelser, kontrol, registrering* og så videre), der lader sig sammensætte til programmer som byggeklodser, der kan afvikle kommandoer enten virtuelt på skærmen (få for eksempel en kat til at bevæge sig) eller gennem fysiske udvidelser som Micro:bit, Lego Spike Prime, Makey Makey med flere. De forskellige blokke i Scratch har forskellige farver og former, der dels understøtter elevernes orienteringer og kodningsproces, men samtidig også er underlagte helt bestemte syntaktiske regler for, hvordan blokkene skal sammensættes for at få programmet afviklet hensigtsmæssigt. Selv om eleverne ikke ser det på overfladen, er Scratch stadig kodning og kodestrenge. Eleverne skal altså lære at følge regler, logikker og konventioner, der er indlejret i programmeringsmiljøet.

Det er her computationel literacy bliver interessant som en mønstergenkendende brug af færdigheder og dispositioner i en materielt understøttet kontekst, der sætter eleverne i stand til at udtrykke sig på en hensigtsmæssig måde, når de skal løse faglige udfordringer, der kræver kodning og programmering. Scratch er et stykke software og er dermed ikke fysisk håndgribeligt. Ikke desto mindre har det materielle egenskaber, der består i de muligheder og begrænsninger (hvoraf nogle er skitseret ovenfor) programmet tilbyder i forhold til for eksempel at skabe, dele, gemme eller manipulere repræsentationer af informationer (Dourish, 2017). På den måde bliver Scratch både en ting, vi har ved hånden, og som kan bruges til at interagere med vores fysiske verden (for eksempel få en robot til tegne), men også noget der afføder bestemte praksisser, hvor eleven gør noget bestemt i forhold til et fagligt indhold.

Det sociale aspekt og skolens fagkulturer

Som antydet indledningsvist har computationel literacy også et socialt aspekt. Det sociale aspekt implicerer forestillingen om, at literacy altid er rettet mod noget, det vil sige indlejret i en social og situeret kontekst.

diSessa understreger, at computationel literacy ikke er statisk, men altid afspejler den sociokulturelle kontekst, hvori man befinder sig. Ydermere påpeger han i sin definition med *patterned deployment* (mønstergenkendende brug), at der netop er tale om en afgrænsning, der adskiller computationel literacy fra andre former for faglige literacies.

Der er for eksempel forskel på, om der er tale om kodning i Scratch eller skrivning af tekster i Word, selvom begge foregår ved en

computer. Der er i begge tilfælde tale om en skrivepraksis, der bygger på en skriveliteracy, men de skal forstås og tilgås forskelligt og ud fra forskellige faglige anskuelser af, hvad skrivning er og kan bidrage til i bestemte kommunikative situationer (Hansen, 2018).

Ikke alle faglige udfordringer og deres løsninger er legitime fokuspunkter inden for fagenes rammer i skolen. Både udfordringer og løsninger er forankret i skolens kontekst og evalueres gennem fagene, og dermed de tilknyttede fagdiscipliner. Det er i disse sammenhænge, at indholdsområder anerkendes som vigtige og legitime eller forkastes som uvedkommende.

Computationel literacy i skolen vil dermed være tilknyttet særlige faglige praksisser og fænomener i undervisningen. Dette gør computationel literacy på engang mangfoldig og snæver. Snæver, fordi der ikke er tale om færdigheder, der nemt og gnidningsfrit lader sig overføre mellem kontekster og faglige praksisser. Der er tale om bestemte blikke, logikker og metoder, der skal transformeres på tværs af faglige forståelser og integreres i situerede praksisser. Det kræver en situeret parathed hos eleven at afkode disse betingelser, og hvordan de bedst imødekommes (Hachmann, 2020). Mangfoldigheden i begrebet afspejles derved i, at der er tale om fag*lighed* og ikke nødvendigvis et specifikt fag. Ligesom det ikke kun er i danskfaget, der læses og skrives eller i matematik, der regnes, bygger computationel literacy på nogle modelleringsfærdigheder, der kan anvendes på tværs af fag. Eksempelvis kan programmeringen af en robot både have et naturvidenskabeligt sigte, men også tage afsæt i mere kunstneriske og kreative fag.

På et mere overliggende niveau har computationel literacy, som andre literacies, også et situeret og dynamisk forhold til et omskifteligt samfund, hvor skole og fag er i forandring og dermed også forskellige forståelser af, hvad fag og faglighed er. Et eksempel herpå kan findes i det aktuelle forsøgsfag for teknologiforståelse i folkeskolen, der også refereres til tidligere i artiklen. En af afprøvningsmodellerne var *teknologiforståelse som integreret faglighed i de eksisterende fag*, hvor eleverne gennem de formulerede prototypeforløb blev undervist i forskellige tematikker på grænsen mellem de eksisterende fag og selvstændige fagligheder inden for design og informatik.

Helt konkret arbejder eleverne i et af prototypeforløbene *Har vi fanget et rigtigt monster?* (Kiær, Godtliebsen, Lorenzen, Nielsen & Nissen, 2020) med algoritmiske- og automatiseringsprocesser gennem flowcharts i danskfaget (1. klasse). Eleverne bliver præsenteret for udvalgte symbolske repræsentationer inden for flowchartgenren: *Terminalpunkt, proces og beslutning* (se Figur 1). Disse bruges for eksempel til beskrivelse forskellige typer af processer. Eleverne skal sidst i forløbet selv lave flowcharts som del af en instruerende tekst, der beskriver, hvordan man kan fange et monster.

Figur 1.

Uddrag af forløbsprototypen: Har vi fanget et rigtigt monster?

Terminalpunkt:	Kan forstås som begyndelsen eller afslutningen af et program-flow i dit diagram	
Proces:	Kan forstås som alle procesfunktioner (handlinger)	
Beslutning:	Kan forstås som et beslutningspunkt mellem to eller flere mulige stier i et flowchart (en forgrening)	

Flowcharts stammer oprindeligt fra ingeniørfaget og blev fra slutningen af 1940'erne en del af datalogien som en måde at beskrive og designe computerprogrammer ud fra (Goldstine & von Neumann, 1947). Uden her at udfolde en dybdegående diskussion af flowcharts som repræsentationsform er det et eksempel på en genreniche (diSessa, 2001, s. 24) i faget, der kræver, "at eleverne læser og skriver på en ny måde og genkender de regler og logikker (patterns), der er indlejrede. Eleverne kan delvist trække på erfaringer fra andre læse- og skrivepraksisser, men må samtidig også lære flowchartgenrens egen grammatik.

Hæves blikket op over et aktivitetsniveau, åbner prototypeforløbet op for en diskussion af, hvorvidt begreber som automatisering, algoritmisk tænkning og flowcharts egentlig hører danskfagets indhold og genrer til? Prototypen er naturligvis en del af et forsøgsprogram - en nyfortolkning af faget, der tilfører en anden faglig dimension. Ikke desto mindre er pointen, at det sociale aspekt computationel literacy også indbefatter, at det faglige fællesskab anerkender de repræsentations- og udtryksformer, genrer og praksisser, der følger med. Hvis disse ikke anerkendes, vil computationel literacy, såvel som andre nye literacies, ikke forblive en del af faget særlig længe. Det var netop det, der skete frem mod 90'erne, da interessen for at undervise i EDB blev til et fokus på blandt andet it-kørekort og multimedieproduktioner. Computer literacy gav på dette tidspunkt mere faglig mening for dem, der definerede fagligheden.

Det kognitive aspekt og computationel tænkning

Det kognitive aspekt af computationel literacy trækker på forestillingen om, at de ting, vi bruger og gør noget med, står i forbindelse med måden, vi tænker og opfatter verden omkring os på. Dette skal ikke forstås som deterministisk, men i højere grad at ting og tænkning ikke er adskilte, men snarere forbundet derved, at ting medierer vores tænkning på forskellig vis.

Den danske datalog Peter Naur understregede allerede i 1960'erne vigtigheden af at forstå de redskaber, der er ved hånden, og hvordan disse redskaber i specifikke situationer vil rammesætte folks tankegang samt deres opfattelse af problemer, og hvordan de løses. En central komponent for Naurs teori var derfor en symmetrisk relation mellem det, han kalder værktøjer, mennesker og problemer (Naur, 1965). Han fremhæver, at menneskers løsning af et problem vil involvere brugen af de værktøjer (fysiske, digitale eller intellektuelle), der er tilgængelige og anses for passende til opgaven. En væsentlig pointe hos Naur er, at hvis værktøjet ændres, vil problemet blive grebet anderledes an eller endda opfattet på nye eller andre måder. Værktøjers mediering betyder her både at muliggøre tænkning og handling, men også at de kan begrænse os ved at holde vores fokus på specifikke muligheder og dermed afholde os fra at se andre (Brennan, 2015). Ifølge Naur afhænger valget af et foretrukket værktøj af, hvordan et problem forstås af de involverede mennesker og hvad en ønskelig løsning kan være. Problemer eksisterer i menneskers sind, og værktøjer, der er designet til at løse problemer, som ikke er anerkendt af nogen, er meningsløse.

Et nyere begreb, der har fået en central plads i forhold til teknologiforståelse i skolen, er *computationel tænkning* (Papert, 1980; Denning & Tedre, 2019; Yadav & Bertelsen, 2022). Begrebet relaterer sig netop til kognitive aspekter af problemløsning og blev revitaliseret, da Jeanette Wing udgav sit essay *Computational Thinking* (Wing, 2006). Selvom en formel definition af computationel tænkning stadig diskuteres i forskellige forskningsfelter omkring uddannelse og skole, er der i forskningslitteraturen dog en vis enighed om at computationel tænkning dækker over nogle grundlæggende kognitive færdigheder, herunder abstraktion, analyse, problemnedbrydning, mønstergenkendelse, modellering og algoritmisk tænkning (Grover & Pea, 2013; Brennan & Resnick, 2012; Jacob & Warschauer, 2018; Dohn, Michell & Chongtay, 2021; Kafai & Proctor, 2021).

Wings nyere definition af computationel tænkning som: "… the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Wing, 2010, s. 20)

påpeger ønsket om et sæt kognitive færdigheder, der er generaliserbare og overførbare på tværs af forskellige fagdiscipliner.

Wings argumenter om, at elever i skolen skulle lære at tænke som en datalog (computer scientist) genoplivede diskussionerne, der tidligere var rejst af både Papert, diSessa, Naur og andre omkring det computationelles rolle og bidrag til problemløsning på tværs af forskellige fagdomæner i skolen og menneskelivet (Jacob & Warschauer, 2018). Hos Wing er der modsat eksempelvis Papert og Naur dog fokus på tænkemåder *der leder frem* mod en for eksempel automatiseringsproces frem for selve det *at udføre* automatiseringsprocessen selv med forskellige værktøjer ved hånden. Med andre ord adskiller Wing sig fra Papert ved at have fokus på "learning to compute" frem for "compute to learn" (Caspersen, Iversen, Nielsen, Hjorth & Musaeus, 2018; Dohn, 2021).

Wing (og mange andre) er hovedsagelig fokuseret på kognitive strategier i forhold til problemløsning. Fra et computationel literacy-perspektiv er der dog mere og andet til det kognitive aspekt end problemløsningsstrategier. At udtrykke idéer er ikke nødvendigvis kun forbundet med problemer og deres løsning. Derimod kan der i lige så høj grad være tale om at udtrykke fantasi, kreativitet, håb eller kritisk stillingtagen. Det handler om at dele sine idéer og tanker med andre gennem de computationelle ting, fysiske og ikke-fysiske, der er til rådighed. Problemformuleringer og -løsninger kan her naturligvis også være et delelement, men ikke et mål i sig. Som Bers (2021) argumenterer for, må forholdet mellem det at kunne formulere og problemløse på den ene side, og på den anden side det at udtrykke sig selv ses som et kontinuum, hvor eleven på forskellige vis bevæger sig mellem problemløsning og udtryk, når de eksempelvis koder på computeren.

Scratch kan altså bruges til at undervise eleverne i problemløsningsstrategier som for eksempel debugging, problemnedbrydning, sekvensering, mønstergenkendelse, automatisering og så videre. Men Scratch er også et programmeringsmiljø, hvor elever kan udtrykke sig kreativt og skabe indhold, der afspejler forståelser, forundring, følelser, identitet og fantasi (Kafai & Proctor, 2021; Bers, 2021; Kafai, Peppler & Chapman, 2009). Dette kan både ske gennem programmering af små spil, men også andre former for fortællinger som eksempelvis animationsfilm, interaktive fortællinger eller fremstilling af e-tekstiler.

Kodning i Scratch kræver, at eleverne dels forstår programmets syntaks og dels kan omsætte deres tanker og idéer computationelt. Der er således tale om en dobbelthed. En elev kan godt være fortrolig og dygtig til at kode i Scratch, men det er ikke nødvendigvis det samme som at kunne modellere og omskabe kreative idéer til noget, der meningsfuldt kan udtrykkes i Scratch. Det samme gør sig gældende

den modsatte vej, hvor eleven kan have virkelig gode idéer, men ikke formår at bruge programmet til at udtrykke disse gennem kodning. Sagt med en anden analogi: En elev der er god til at skrive, er ikke nødvendigvis en god forfatter, ligesom en elev der er god til sudoku, ikke nødvendigvis er god til matematik – og omvendt. Fra et kognitivt aspekt stilles der altså særlige krav til elevernes dispositioner i forhold til at afkode og modellere fænomener, således at de kan udtrykkes eller fortolkes gennem brugen af computationelle ting.

Computationel literacy i skolen og eksempler fra praksis

Computationel literacy består som uddybet ovenfor af bestemte måder at tænke og handle på i en situeret kontekst, der medieres (understøttes og begrænses) af ting og deres materialiteter. Det at undervise i og gennem computationel literacy i skolen bør derfor bygge på et perspektiv, der indlejrer alle tre aspekter.

Lu og Fletcher (2009) påpeger, at programmering er kognitivt abstrakt og kræver et bredt fundament og dyb forståelse for symbolmanipulation og kodning. Programmering kan i denne optik sammenlignes med bevisførelse i matematik eller avancerede analyser af litteratur i danskfaget. I begge tilfælde kræver det, at eleverne har en literacy, faglige færdigheder og -forståelser inden for de respektive fagområder.

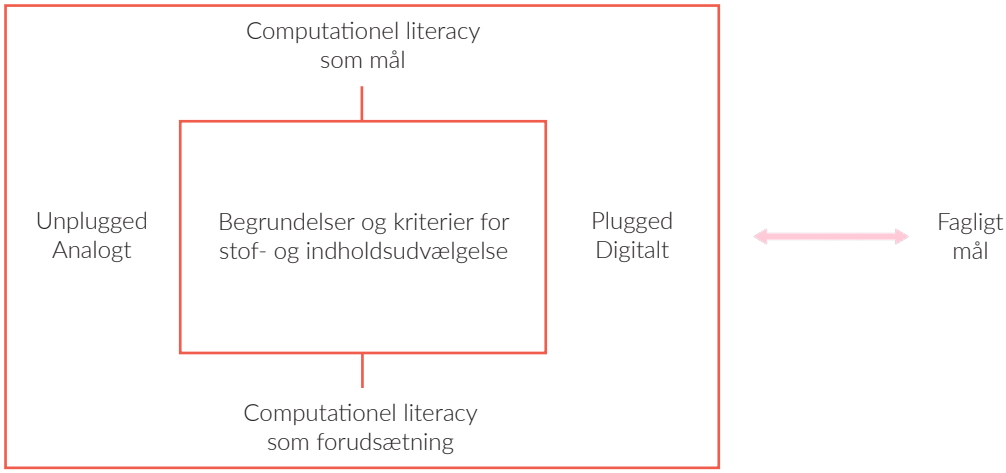
Computationel literacy i skolen udvikles over tid gennem mange forskellige tilgange (Kafai & Proctor, 2021), der har fokus på elevernes forudsætninger og dispositioner i forhold til en gradvis progression. Det kunne eksempelvis være undervisning, der tog afsæt i, at eleverne, både direkte og indirekte, møder de begreber og kompetenceområder, der knytter sig til computationel literacy i og på tværs af fagene. Kodning i Scratch, som har været det gennemgående eksempel i artiklen, kunne være én tilgang, men langt fra den eneste.

Den grundlæggende styrke ved kodning gennem en eksperimenterende og undersøgende tilgang er, at eleverne gives mulighed for på forskellig vis at møde komplekse idéer, der er logisk organiseret og for eksempel gør brug af abstraktionsevner og repræsentationsformer gennem meningsfulde projekter og aktiviteter, som eleverne selv er med til at definere.

Som Tannert, Lorenzen & Berthelsen (2022) peger på, kan det computationelle i skolen både være et primært fagligt fokus, men også

noget sekundært. Som primært fokus bliver det at lære computationel literacy et mål i sig selv, og det stof og indhold, eleverne møder, understøtter netop dette formål. Som sekundært fokus bliver det i højere grad en forudsætning for at kunne løse andre faglige udfordringer gennem inddragelse af forskellige computationelle strategier. Ydermere fremhæves det i den fremtrædende forskningslitteratur, at der er et behov for at arbejde både gennem *plugged* og *unplugged* tilgange (Caeli & Yadav, 2019) i undervisning, således at der arbejdes med kognitive og kropslige perspektiver gennem brugen af både *analoge* og *digitale* ting (Brennan & Resninck, 2012; Grover & Pea, 2013, Mikkonen, 2021; Dohn, 2021). I Figur 2 nedenfor præsenteres en simpel grafik, der illustrerer, hvordan begrundelser for stof- og indholdsudvælgelse (hvad og hvordan) i et computationelt literacy-perspektiv er forankret i de ovenstående opmærksomhedspunkter:

Figur 2.
Begrundelser for stof- og indholdsudvælgelse i et computationel literacy-perspektiv.



Selv om der her ikke er tale om en egentlig didaktisk model, kan denne forsimplede illustration være med til at skabe opmærksomhed på sammenhængen mellem indholdsopfattelser, aktivitetsformer og computationel literacy i undervisningen. Lærerens rammesætning af aktiviteter gennem eksempelvis kodning og programmering af spil i Scratch kan afspejle et stof og indholdsudvælgelse, der hovedsageligt lægger vægt på anvendelse af computationel literacy i digitale omgivelser. Hvorimod det tidligere eksempel fra prototypeforløbet med brugen af flowcharts afspejler en tilegnelse af computationelle færdigheder gennem en unplugged tilgang.

Escape Puzzles som vej til at lære om Micro:bit

Et eksempel på en kropslig tilgang til programmering med computationel literacy som mål i sig selv er tidligere beskrevet og udfoldet hos Hachmann (2022a; 2022b). Elever deltager i et scenariebaseret Escape Puzzle (se Figur 3) som en del af et længerevarende forløb om Micro:bit. Aktiviteten har til hensigt at give eleverne indsigt i de forskellige sensormuligheder, der er indlejret i en Micro:bit og deres anvendelseslogikker. Eleverne skal gennem aktiviteten fysisk manipulere med forskellige træbokse, hvori en Micro:bit med en specifik sensoregenskab er gemt. Undervejs i forløbet får eleverne hints fra en af boksene i form af ord på et display som "Elsker metal", "Lyset" eller "Fart". Disse skal hjælpe dem med afkodningen. Kun ved at aktivere alle sensorer på én gang, kan eleverne komme i mål med den stillede opgave. Som håndgribelig og fysisk repræsentation har træboksene den egenskab, at Mikro:bit'ens sensoregenskaber kræver fysiske, kropslige handlinger hos eleven.

Figur 3.
Foto fra empirisk undersøgelse før og under aktiviteten (Hachmann, 2022a).

Som Mikkonen (2019; 2021) har vist gennem undersøgelser af *Body-gramming,* hvor universitetsstuderende gennem kropsbaserede og kollaborative aktiviteter simulerer computerens algoritmiske processer, er kropslige erfaringer med sådanne abstrakte fænomener en vigtig forudsætning for at forstå det computationelle perspektiv i forhold til eksempelvis at programmere en mikrocomputer. Mikkonen understreger, at kropslig leg og sociale aktiviteter illustrerer forskellene mellem perspektiver og abstraktionsformer for henholdsvis en computer og brugeren af den. Forståelsen af disse forskelle gør det derfor nemmere for eleverne at lære programmering, fordi de forstår de perspektiver og abstraktionsformer, som programmeringen adresserer. På samme måde er Escape Puzzle aktiviteten hovedsageligt baseret på, at eleverne gennem en kropslig oplevelse opdager bestemte funktioner og muligheder hos Micro:bit'en som en materiel ting. De skal analogt manipulere med boksene frem for at kode i for eksempel Scratch, og de anvender strategier baseret på samarbejde, problemnedbrydning, mønstergenkendelse og algoritmisk tænkning, der er indlejret i de kognitive aspekter af computationel literacy. Forløbet afspejler et fagsyn (det sociale aspekt), der lægger vægt på, at eleverne lærer grundlæggende begreber og færdigheder gennem en legende og eksperimenterende tilgang. Trækassernes materielle beskaffenhed og

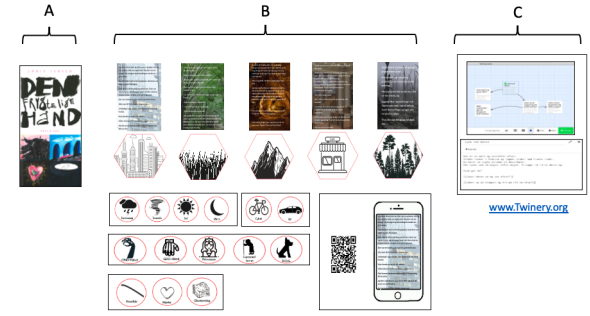
benævnelse som "MysteryBoxe" gør aktiviteterne til både analyser, interpretationer og fælles refleksioner i forhold til nedbrydning (dekomposition) og løsningen af opgaven. Det langsigtede mål er elevernes tilegnelse af en computationel literacy, men det gøres her ud fra idéen om en sekventiel progression, hvor eleverne trinvist og i en bevægelse fra konkret mod abstrakt forståelse lærer grundbegreber, teknikker og strategier, der er basisfaglige og grundlæggende for at kunne bruge Micro:bit'en som værktøj på et mere abstrakt niveau senere i forløbet.

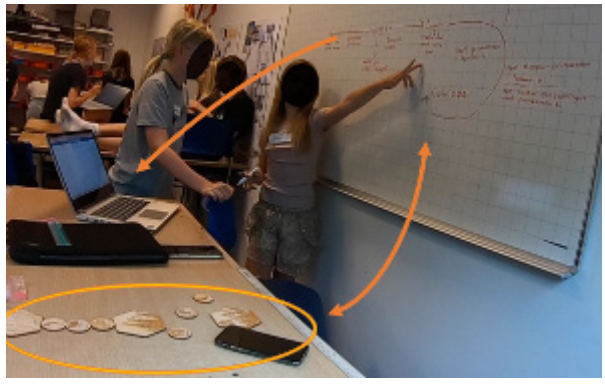
Den frygtelige hånd – skriveundervisning gennem kodning
Et andet eksempel har fokus på computationel literacy som forudsætning og inddrager både plugged/analoge og unplugged/digitale perspektiver. Eksemplet tager afsæt i skriveundervisningen på mellemtrinnet, hvor elever i 5.-6. klasse undervises i procesorienteret skrivning og medforfatterskaber af en fantastisk fortælling. Hertil udvikles et didaktisk design (se Figur 4), hvor eleverne først får oplæst et uddrag af *Den frygtelige hånd (Jensen, 2001),* der er en roman inden for genren fantastisk fortælling (A). Herefter skal eleverne ved hjælp af en række fysiske træbrikker (B) idégenerere og strukturere deres fortælling, der er en fortsættelse og afslutning på den oplæste tekst. Bag på fem store brikker sidder en QR-kode, der giver eleverne mulighed for at læse udvalgte dele af Den frygtelige hånd som inspiration. De mindre og runde brikker repræsenterer forskellige elementer, personer eller genstande fra historien. Som afslutning på processen skal eleverne skabe interaktive fortællinger i onlineapplikationen Twine (C). Her skal eleverne skabe en interaktiv hypertekstfortælling, hvor læseren er hovedpersonen og undervejs gennem hyperlinks selv er med at vælge, hvad hovedpersonen gør.

Twine gør brug af html-kodning, og eleverne skal kunne gennemskue dette programmeringsformat. For eksempel skal de for at lave nye passager bruge firkantede parenteser omkring en tekst eller bruge html-kode for at indlejre billeder, ændre tekstfarven og så videre.

Figur 4.

Grafisk illustration af det didaktiske design (øverst) og dets realisering i en 5. klasse (nederst).





Fra et computationel literacy-perspektiv udgør træbrikkerne en afgrænsning i forhold til elevernes idégenerering (fra A→B). Samtidig muliggør brikkerne gennem deres materielle beskaffenhed, at eleverne anvender en sekventiel og algoritmisk tænkning, der giver dem mulighed for at dekomponere deres fortælling i mindre dele og arrangere indholdet på forskellig vis i forholdt til de mønstre, de genkender i uddragene fra den oprindelige historie. Dette forarbejde træner elevernes computationelle strategier samtidig med, at det muliggør en vis grad af kropslighed og tingsliggørelse i forhold til strukturering af narrativer i deres fortælling.

Bevægelsen fra B→C i Figur 4 implicerer også, at eleverne omskaber tegnsystemer. Dette betegnes hos Selander & Kress (2012) som en transduktionsproces, hvor eleverne gennem forskellige former for symbolmanipulationer omskaber én repræsentationsform til en anden.

Brikkerne kan i sig selv skabe en vis form for strukturel og visuel repræsentation, der lægger op til meningsforhandlinger og kreative idéer, men som det ses på Figur 4, har eleverne brug for at omskabe, fastholde og uddybe disse med skrift og ord (på whiteboardet), før deres idéer bliver omsat til computerkode.

Eksemplet viser, at eleverne gennem processen trækker dels på andre literacyformer fra eksempelvis danskfaget, men også at der er brug for en specifik computationel literacy og modelleringsevne i forhold til at skabe deres endelige fortællinger i Twine, der som sagt gør brug af html-kodning.

Afsluttende bemærkninger

Som der indledningsvist blev påpeget, er denne artikel et forsøg på at etablere et grundlag for fremadrettede diskussioner af begrebet computationel literacy i skolen. Det, der har været i fokus her, er materialiteter og deres betydning i forhold til begrebet. Samtidig er det forsøgt gennem eksempler at skabe en ydre afgrænsning for, hvornår der kan tales om *computationel* literacy i forhold til andre former for literacy. Der er i den danske skole ikke en tradition for, på samme måde som i udlandet, at have fokus på datalogiske aspekter som kodning og programmering. Med forsøgsfaget teknologiforståelse som afsæt for et forsøg på at etablere nye fagligheder i skolen virker et blik på computationel literacy uundgåeligt.

Et begreb som computationel literacy vil kunne skabe et sprog omkring integrationen af computationelle perspektiver i både nye og eksisterende literacy-praksisser. Det er dog også en præmis, at en sådan literacy bør undersøges, diskuteres og udvikles i relation til skolens formål. Ét argument ville være, at digitalisering, data og teknologi er integrerede dele af elevernes liv i og uden for skolen. Selvom der i de sidste 50 år har været et fokus på it i skolen, er der i de seneste 5-10 år sket en acceleration i digitaliseringen og fokus på, hvad algoritmer, automatisering og data har af betydning for vores liv. Hvilket giver skolen grund til at gentænke dannelsesopgaven. Skolens opgave er i denne sammenhæng ikke at uddanne eleverne til dataloger eller ingeniører, men at give eleverne en grundforståelse og en demokratisk handlekompetence i forhold til det samfund, de møder udenfor skolen. Dette implicerer blandt andet, at de får en grundforståelse for, hvad eksempelvis algoritmer og automatisering er, og hvordan de skabes med forskellige formål, der i sidste ende influerer på deres liv. Dette kan gøres både direkte og indirekte gennem fokus på problemløsning, som det er

tilfældet i eksemplet *Cybervåbnet*, eller kreative skabelsesprocesser, der er en del af eksemplet: *Den frygtelige hånd*. I sidstnævnte er en lige så vigtigt pointe, at der fokuseres på elevernes muligheder for at udvikle fantasi og være kreativt udtrykkende.

Som med andre literacies tager det tid at udvikle, og lige som læsning, skrivning og regning kræver det, at man gør det i mange forskellige sammenhænge og på mange forskellige måder. Ideelt ville eleverne gøre brug af computationel literacy ofte og i forskellige sammenhænge på tværs af fag – gennem kodning, unplugged aktiviteter, spiludvikling, digital leg, og med forskellige formål som problemløsning, innovation, meningsskabelse, æstetiske oplevelser og så videre.

Feltet er nyt, og der er mange spørgsmål af både empirisk og teoretisk karakter, der står ubesvaret hen. Ikke desto mindre er der, som vist i artiklen, en historik og et forarbejde at bygge videre på. Dette muliggør at diskutere computationel literacy som et begreb og genstandsfelt for videre forskning i skole- og uddannelsessammenhænge.

Referencer

Brennan, K. (2015). Objects To Think With. *Constructivist Foundations*, *10*(3), 313-314.

Brennan, K. & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking. Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vol. 1, Vancouver, 13-17 April 2012, s. 25.

Bers, M. U. (2021). *Coding as a playground: Programming and computational thinking in the early childhood classroom* (2. Udg.). Routledge. DOI:10.4324/9781003022602

Caeli, E. N. & Yadav, A. (2019). Unplugged Approaches to Computational Thinking: A Historical Perspective. *TechTrends 64*, 29-36. DOI:10.1007/s11528-019-00410-5

Caspersen, M. E., Iversen, O. S., Nielsen, M., Hjorth, A. & Musaeus, L. H. (2018). *Computational Thinking – Hvorfor, hvad og hvordan?* It-vest – samarbejdende universiteter.

Danholt, P. & Gad, C. (2021). *Videnskab, teknologi og samfund: En introduktion til STS*. Hans Reitzel.

Denning, P. J. & Tedre, M. (2019). *Computational thinking.* The MIT Press. DOI:10.7551/mitpress/11740.001.0001

DiSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy* (1. udgave paperback). The MIT Press. DOI:10.7551/mitpress/1786.001.0001

Dohn, N. B., Mitchell, R. & Chongtay, R. (Red). (2021). *Computational thinking: Teoretiske, empiriske og didaktiske perspektiver*. Samfundslitteratur.

Dohn, N.B. (2021). Computational Thinking – indplacering i et landskab af it-begreber. I: N. B. Dohn, R. Mitchell & R. Chongtay (Red). *Computational thinking: Teoretiske, empiriske og didaktiske perspektiver*. Samfundslitteratur.

Dourish, P. (2017). *The Stuff of Bits: An Essay on the Materialities of Information*. MIT Press. DOI:10.7551/mitpress/10999.001.0001

Erhvervsministeriet. (2018). *Strategi for Danmarks digitale vækst*. Erhvervsministeriet.

Fischer, C., Frøkjær, E. & Gedsø, L. (1972). *Datalære i skolen – Om data og edb i samfundet*. G. E. C. Gads Forlag.

Hachmann, R. (2020). *Didactic Design for Transformations of Subject-content Knowledge: An investigation of student teachers' transformations of subject-content knowledge between professional education and practice* [PhD Dissertation]. University of Southern Denmark.

Hachmann, R. (2022a). The Cyber Weapon: Decomposing Puzzles in Unplugged Computational Thinking Practices with Computational Objects. *KI – Künstliche Intelligenz*, *36*, 59-68. DOI:10.1007/s13218-022-00756-8

Hachmann, R. (2022b). Cybervåbnet: computationel praksis i 8. klasse. I: S. S. Fougt, J. Bundsgaard, T. Hanghøj & M. Misfeldt (red.), *Håndbog i scenariedidaktik* (s. 489-501). Didaktiske studier Nr. 8. Aarhus Universitetsforlag. DOI:10.2307/j.ctv33jb48k.37

Grover, S. & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38-43. DOI:10.3102/0013189X12463051

Haigh, R. W. (1985). Planning for Computer Literacy. *The Journal of Higher Education, 56*(2), 161-171. DOI:10.2307/1981664

Hansen, J. J. (Red.). (2018). *Digital skrivedidaktik*. Akademisk Forlag.

Hicks, D. & Beaudry, M. C. (Red.). (2010). *The Oxford handbook of material culture studies*. Oxford University Press.

Jacob, S. R. & Warschauer, M. (2018). Computational Thinking and Literacy. *Journal of Computer Science Integration*, *1*(1). DOI:10.26716/jcsi.2018.01.1.1

Jensen, L. (2001). *Den frygtelige hånd*. Høst & Søn.

Kafai, Y. & Resnick, M. (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Lawrence Erlbaum Associates, Inc.

Kafai, Y. B., Peppler, K. A. & Chapman, R. N. (Red.). (2009). *The Computer Clubhouse: Constructionism and creativity in youth communities*. Teachers College Press.

Kafai, Y. B. & Proctor, C. (2021). A Revaluation of Computational Thinking in K–12 Education: Moving Toward Computational Literacies. *Educational Researcher, 51*(2), 146-151. DOI:10.3102/0013189X211057904

Kiær, K., Godtliebsen, A., Lorenzen, R.F., Nielsen L. & Nissen, A. (2020). *Har vi fanget et monster?* Lokaliseret 9. Juni 2022 på, https://xn--tekforsget-6cb.dk/wp-content/uploads/2020/09/fanget-et-monster-1.kl_.-dansk-15.09.20.pdf

Kragelund, M. & Otto, L. (Red.). (2005). *Materialitet og dannelse: En studiebog* (1. udgave). Danmarks Pædagogiske Universitets Forlag.

Lu, J. J. & Fletcher, G. H. L. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, *41*(1), 260-264. DOI:10.1145/1539024.1508959

Maloney, J., Resnick, M., Rusk, N., Silverman, B. & Eastmond, E. (2010). The Scratch Programming Language and Environment. *ACM Transactions on Computing Education*, *10*(4), 1-15.

Mikkonen, J. (2019). Bodygramming. Embodying the computational behaviour as a collective effort. *The Design Journal,* 22, 1423-1437. DOI:10.1080/14606925.2019.1594967

Mikkonen, J. (2021). Kropsbaseret Computational Thinking. I: N. B. Dohn, R. Mitchell, & R. Chongtay (Red.), *Computational Thinking – Teoretiske, empiriske og didaktiske perspektiver*. Samfundslitteratur.

Molnar, A. R. (1979). The Next Great Crisis in American Education: Computer Literacy. *Journal of Educational Technology Systems*, *7*(3), 275-285. DOI:10.2190/TM2Q-LLGE-AFJ8-UYBP

Naur, P. (1965). The Place of Programming in a World of Problem, Tools, and People. *Information processing 1965*, 195-199.

Naur, P., Vinter, B., Hansen, K., Mogensen, T. Æ., Erleben, K., Pisinger, D., Nielsen, M., Kringelbach, M., Pedersen, E. W., Blume, P., Helles, R., Andersen, T. O. (red.), Bansler, J. P. (red.), Clausen, H. R. (red.), Jensen, I. H. (red.) & Zachariasen, M. (red.) (2010). *Den digitale revolution: fortællinger fra datalogiens verden*. Datalogisk Institut, Københavns Universitet.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.

Papert, S. (1987). Information technology and education: Computer criticism vs. technocentric thinking. *Educational Researcher, 16*(1), 22-30

Selander, S. & Kress, G. (2012). *Læringsdesign – I et multimodalt perspektiv*. Frydenlund.

Slot, M. F., Hachmann, R., Hjorth, M. & Von Sehested, M. (2021). Teknologiforståelse – en sammenhængende faglighed: En beskrivende analyse af 110 undervisnings forløb. *Learning Tech*, *10*, 296-322. DOI:10.7146/lt.v6i10.125600

Soon, W. & Cox, G. (2021). *Aesthetic programming: A handbook of software studies*. Open Humanities Press.

Tannert, M., Lorentzen, R. F. & Berthelsen, U. D. (2021). Computational Thinking as Subject Matter. I: A. Yadav & U. D. Berthelsen. (Red). *Computational Thinking in Education*. Routledge. DOI:10.4324/9781003102991-5

Tuhkala, A., Wagner, M.-L., Iversen, O. S. & Kärkkäinen, T. (2019). Technology Comprehension – Combining computing, design, and societal reflection as a national subject. *International Journal of Child-Computer Interaction*, *20*, 54-63. DOI:10.1016/j.ijcci.2019.03.004

Undervisningsministeriet (1990). *EDB i folkeskolens fag: Dansk og EDB.* Undervisningsvejledning for folkeskolen.

Wagner, M.-L., Iversen, O. S. & Caspersen, M. E. (2020). Teknologiforståelsens rationale: På vej mod coputational empowerment i den danske grundskole. *Unge Pædagoger*, *81*(1), 6-14.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. DOI:10.1145/1118178.1118215

Wing, J. M. (2010). Computational Thinking: What and Why? Link Magazine, *6*. Lokaliseret på, https://www.cs.cmu.edu/~CompThink/papers/TheLinkWing.pdf

Yadav, A. & Berthelsen, U. D. (Red.). (2022). *Computational thinking in compulsory education: A pedagogical perspective* (First Edition). Routledge. DOI:10.4324/9781003102991