Kennisnet

Computational thinking in het Nederlandse onderwijs

De stand van zaken aan de hand van interviews

Laat ict werken **voor het onderwijs**

Inhoudsopgave

Inhoudsopgave

Voorwoord

Veel schoolbesturen vragen zich af hoe ze *computational thinking* (of programmeren, maar dat is eigenlijk iets anders) kunnen opnemen in hun onderwijs. Computational thinking, kortweg 'ct', is nadenken over de vraag hoe je een probleem kunt oplossen met een computer. Programmeren is een middel daarvoor.

De belangstelling om hiermee aan de slag te gaan groeit, zo blijkt uit de vragen die Kennisnet hierover krijgt. Soms worden scholen gedreven door de noodzaak om meer aandacht voor techniek te geven, of door de wil om 21e eeuwse vaardigheden te verwerken in het onderwijs, of omdat ze het gewoonweg belangrijk vinden dat ze leerlingen hiermee in aanraking laten komen.

Maar hoe pak je dit aan? Hoe kunnen we (huidige en toekomstige) leraren helpen om ct-onderwijs vorm te geven? Hoe doe je dat op een goede manier, en wat weten we daarover uit de wetenschap? Voor deze publicatie hebben we gesproken met leraren, schoolbesturen, onderzoekers en opleiders, over hun activiteiten en visie op het gebied van computational thinking in het onderwijs.

Deze publicatie is een handreiking voor het integreren van computational thinking in het onderwijs. Met ervaringen en tips uit de praktijk. We laten zien hoe (Nederlandse) scholen daarmee omgaan, en wat ze daarvan hebben geleerd. Daarnaast vertellen onderzoekers wat er bekend is vanuit de wetenschap; wat werkt en wat niet, en wat onderwijskundig gezien van belang is. Ook vanuit de pabo's schetsen we een beeld hoe zij computational thinking een plek willen geven bij het opleiden van leraren.

We hopen dat hun kennis en ervaring behulpzaam zal zijn bij het vormgeven van computational thinking in uw onderwijs.

Remco Pijpers
Strategisch adviseur digitale geletterdheid

1 Inleiding

1 Inleiding

Computational thinking (ct) en programmeren zijn hot. De termen duiken steeds vaker op in de literatuur, op conferenties, en in adviesrapporten.

Zoals:

- het eindadvies *'Ons onderwijs2032'* (Platform Onderwijs2032, 2016)
- het rapport *'Digitale geletterdheid in het voortgezet onderwijs'* (KNAW, 2013)
- het rapport *'21e eeuwse vaardigheden in het curriculum van het funderend onderwijs'* (SLO , 2014)
- de feestelijke presentatie van de *'Leerlijn programmeren in het basisonderwijs'* (PO-Raad, 25 mei 2016).

Ook wordt computational thinking voorgesteld als een van de 21e eeuwse vaardigheden in het *'Curriculum van de toekomst'*.

Staatssecretaris Sander Dekker (OCW) reageerde als volgt op het eindadvies 'Ons onderwijs2032':

> ***"Het werken en leren in de digitale wereld behoort tot de kern van toekomstgericht onderwijs [...] Dat betekent dat leerlingen ict-basiskennis opbouwen, informatievaardigheid ontwikkelen, mediawijs worden, en leren begrijpen hoe informatietechnologie werkt. Dit betreft niet alleen het gebruik van computers en ict als consument, maar ook als producent."***

De richting lijkt dus duidelijk: computational thinking gaat een steeds grotere rol spelen in het Nederlandse onderwijs. Maar wat betekent dat precies? Hoe wordt dit type onderwijs ingevuld? Wat wordt er al gedaan op dit gebied in Nederland? Wat werkt wel en wat werkt niet?

Ervaringen en visies

In dit rapport leggen we de ervaringen en visies van leraren (po en vo), schoolbestuurders, pabo-docenten en wetenschappers naast elkaar, om te zien wat de gedachten achter dit nieuwe vak zijn, wat er in de praktijk gebeurt, wat voor oplossingen er zijn gevonden, en wat er in de toekomst mogelijk is. Hoe moet je dit onderwijs inrichten? Hoe krijg je draagvlak? En hoe test je de vaardigheden en kennis van leerlingen die ct-lessen volgen? Wat kunnen we leren van de verschillende ervaringen en ideeën?

We spraken met:

Alfons ten Brummelhuis (Kennisnet/NRO): onderzoeker, strategisch adviseur, en projectleider.

Ralph Crützen (Grotiuscollege, Heerlen): wiskunde- en informaticadocent aan een openbare school voor mavo, havo, atheneum en gymnasium.

Richard Doornbos (Pabo Hanzehogeschool, Groningen): docent rekenen en wiskunde, en verantwoordelijk voor diverse programmeercursussen voor kinderen.

Gerard Dummer (Pabo Hogeschool Utrecht, Amersfoort): opleidingsdocent ict & onderwijs.

Hylke Faber (Pabo Hanzehogeschool, Groningen): junior-onderzoeker op het gebied van computational thinking-onderwijs.

Marja-Ilona Koski (NewTechKids, Amsterdam): voorheen onderzoeker bij de TU Delft, nu werkzaam bij NewTech-Kids, een buitenschoolse onderwijsorganisatie.

Sandra Legters (Oponoa, Gelderland): onderwijskundig coördinator techniek en ict. Auteur van CodeKlas, medewerker aan de 'Leerlijn programmeren in het basisonderwijs'.

Han van der Maas (Universiteit van Amsterdam): hoogleraar psychologische methodenleer, en initiatiefnemer van educatieve websites als Rekentuin en Taalzee.

Herman Rigter (Verenigde Scholen Alberdingk Thijm, Hilversum): directeur ict.

Allard Strijker (SLO, Enschede): leerplanontwikkelaar, en onderzoeker op het gebied van innovatieve ict- en media-toepassingen in het onderwijs.

Hans Vasse (Jan van Brabant College, Helmond): docent design en kunstgeschiedenis.

Mark Vrolijk (Onderwijsgroep Fier, NW-Friesland): algemeen directeur.

Computational thinking (ct) – definitie

De term 'computational thinking' (kortweg 'ct'), werd voor het eerst gebruikt in 1980, door de computerdeskundige en pedagoog Seymour Papert (uitvinder van de educatieve computertaal Logo). Maar echte bekendheid kreeg de term pas door Jeanette Wing, die in 2006 het begrip als volgt definieerde:

> ***"Computational thinking is het denkproces waarmee problemen en hun oplossingen zo worden geformuleerd dat ze kunnen worden gepresenteerd in een vorm die effectief kan worden uitgevoerd door een informatie verwerkende tussenpersoon, zoals een computer of een mens, of een combinatie van beide."***
> *(Wing 2011)*

Wings ideeën sloegen al snel aan, omdat computational thinking het (vaak verouderde) ict- en informatica-onderwijs een nieuwe, scherpere koers gaf. Niet langer stonden hardware en software centraal, maar meer het denken daarover. In *'Curriculum van de toekomst'* schrijft SLO: "Computational thinking richt zich op de vaardigheden die essentieel zijn om problemen op te lossen waarbij veel informatie, variabelen en rekenkracht nodig zijn. [...] Het is daarbij belangrijk om te begrijpen hoe informatie tot stand komt, zodat je computersystemen kunt benutten voor het oplossen van problemen, voor het denken in stappen, en daarmee in voorwaardelijkheden voor volgorde van de benodigde gegevens."

In 2010 werd computational thinking door het Amerikaanse National Research Council dan ook omarmd als nieuwe manier om de snel voortschrijdende digitalisering van de samenleving een plek te geven binnen het onderwijs. Sindsdien hebben deze ideeën zich verspreid over de hele wereld, met name in het nationale onderwijscurriculum van Groot-Brittannië en enkele andere landen (waarover later meer).

Computational thinking is meer dan programmeren

Ondanks de strakke definities van Wing, SLO en anderen, blijven er misverstanden bestaan over het begrip 'computational thinking'. Vaak wordt het in één adem genoemd met 'programmeren' en 'digitale geletterdheid', zonder een duidelijk onderscheid tussen die begrippen te maken. Dat is onwenselijk, want waarom zou je kinderen willen leren programmeren, als lang niet iedereen programmeur gaat worden?

Bill Mitchell, onderwijsdirecteur bij het British Computing Centre (BCS), en expert op het gebied van het (verplichte) Britse computing-curriculum, zegt ook uitdrukkelijk dat ct-onderwijs niet bedoeld is om van alle kinderen programmeurs te maken. Maar wel: "om het creatieve, oplossingsgerichte vermogen van kinderen aan te scherpen, en hun kennis van digitale systemen te vergroten." Computational thinking is dus meer dan alleen programmeren.

Maar programmeren is wel belangrijk. Volgens Mitchell is het de locomotief die alle andere aspecten van de digitale wereld met zich meetrekt. Door te leren programmeren, leer je ook andere aspecten van het moderne computergebruik kennen en beheersen, is het idee (dat niet alleen bij hem, maar ook bij andere onderwijskundigen leeft). Programmeren is in deze visie dus geen doel maar een middel.

Computational thinking als onderdeel van digitale geletterdheid

Computational thinking is dus breder dan programmeren. Volgens het eerder genoemde eindadvies van het Platform Onderwijs 2032 en de bijbehorende *Kamerbrief van Staatssecretaris Dekker* is computational thinking ook een onderdeel van 'digitale geletterdheid'.

Onder 'digitale geletterdheid' wordt verstaan:
- ict-vaardigheden
- informatievaardigheden
- mediawijsheid
- en computational thinking

Computational thinking als onderdeel van 21e eeuwse vaardigheden

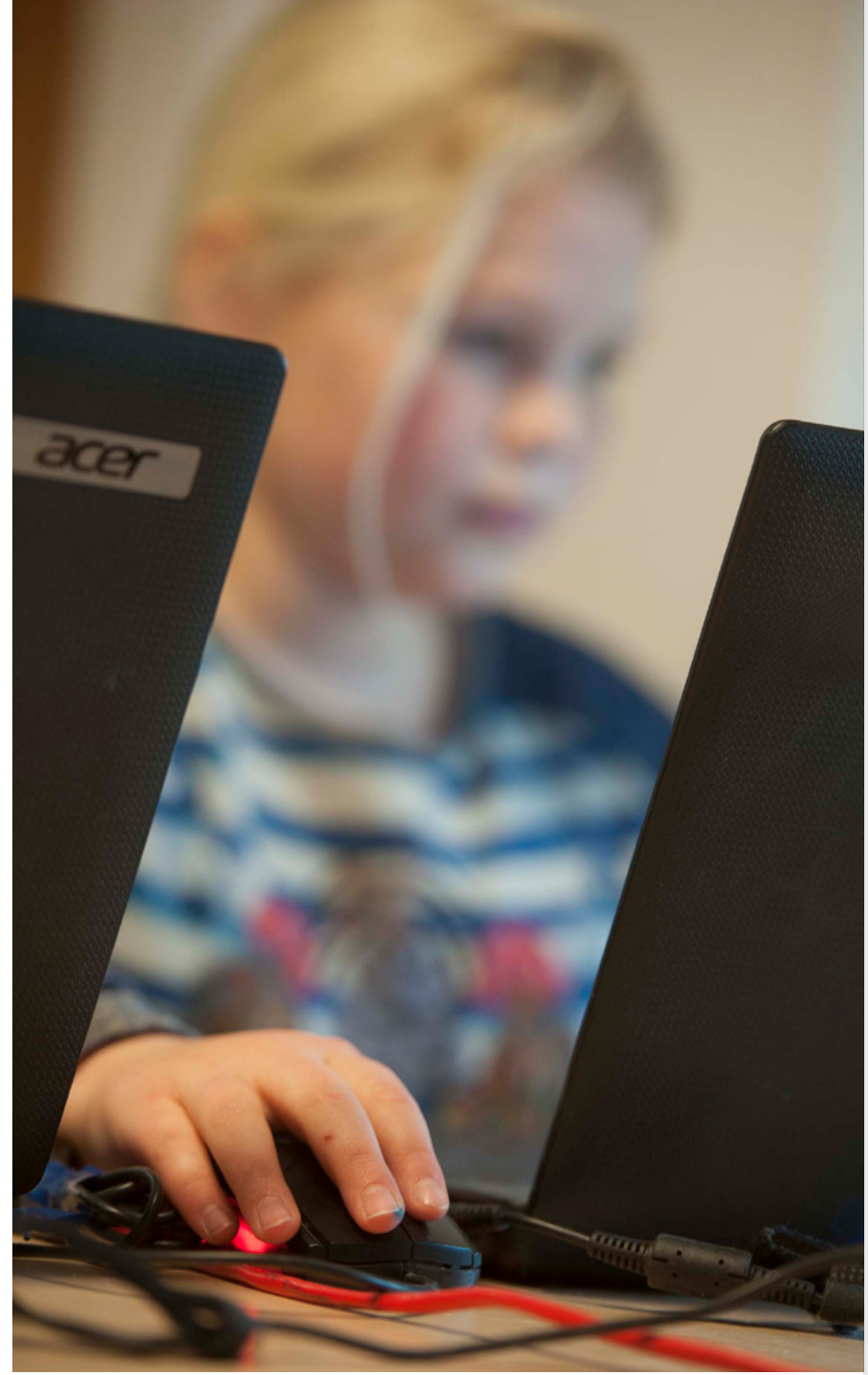
In de optiek van Kennisnet en SLO is computational thinking een van de 21e eeuwse vaardigheden (zie de afbeelding op pagina 10). Bij 21e eeuwse vaardigheden gaat het om vaardigheden die kinderen en jongeren in hun latere leven nodig zullen hebben, en die dus aangeleerd moeten worden.

Voorbeelden van 21e eeuwse vaardigheden:
- kritisch denken
- creatief denken
- probleemoplossen
- ict-basisvaardigheden
- informatievaardigheden
- computational thinking

"Door te leren programmeren, leer je ook andere aspecten van het moderne computergebruik kennen en beheersen."

21e eeuwse vaardigheden

De bovenstaande infographic, die elf vaardigheden omvat, is gebaseerd op onderzoek uit 2014. Uit dat onderzoek bleek dat de genoemde vaardigheden nog niet goed aan de orde kwamen in het toenmalige onderwijs. Vooral creatief denken, probleemoplossen (naar het Engelse 'problem solving') en digitale geletterdheid waren nog weinig uitgewerkt, zo bleek. Tegelijkertijd waren dit ook de vaardigheden waarvan leraren zeiden dat ze er meer over wilden weten. "Scholen zitten te springen om materiaal waarmee ze de *21e eeuwse vaardigheden* een plaats kunnen geven in hun onderwijs", aldus Remco Pijpers van Kennisnet.



2 Wetenschappelijke achtergronden

2 Wetenschappelijke achtergronden

Een van de motieven om computational thinking in te voeren in het onderwijs is 'omdat het elders ook gebeurt'. In Groot-Brittannië (waar computing inmiddels een verplicht vak is), maar ook in Finland, Estland, Griekenland, Cyprus, Polen en Bulgarije. En buiten Europa: op de Filipijnen, in Singapore en in India.

Dat er in het buitenland veel aan ct- en programmeeronderwijs wordt gedaan, zegt Alfons ten Brummelhuis – onderzoeker bij Kennisnet en projectleider bij NRO – niet zoveel: "Er is maar heel weinig onderzoek gedaan naar de effecten van digitale geletterdheid en de opbrengst van programmeeronderwijs. We weten nog nauwelijks wat wel en niet werkt."

Ook de waarom-vraag is belangrijk. Ten Brummelhuis: "Zelfs wanneer we met z'n allen afspreken dat programmeren belangrijk is in het onderwijs, op welke gronden doe je dat dan? Doe je het omdat je vindt dat het bijdraagt aan de talentontwikkelingen van leerlingen? Voer je het in omdat het beter voorbereidt op de toekomstige arbeidsmarkt? Of vind je dat onze toekomstige burgers moeten weten hoe digitale systemen werken? Het maakt nogal wat uit, waaróm je ct- en programmeeronderwijs wilt invoeren. Bij het ene motief wil je alle leerlingen kennis en vaardigheden bijbrengen, bij een ander motief gaat het om een veel kleiner groepje. Daar moet je goed over nadenken."

Finland

Waarom er in de landen om ons heen zoveel aandacht is voor computational thinking, en in Nederland zoveel minder, was ook de vraag bij Dr. Marja-Ilona Koski. Tot voor kort was zij onderzoeker aan de TU Delft; nu is ze docent en directeur in het (buitenschoolse) computing-onderwijs bij het door haar opgerichte NewTechKids. Koski is opgeleid als theoretisch computerwetenschapper en heeft in haar moederland Finland gewerkt als *science teacher* op scholen.

Finland scoort altijd hoog als het gaat om wetenschap- en techniekkennis van leerlingen. Dit land staat bijvoorbeeld op de 6e plaats in de internationale onderwijsvergelijkingscijfers van het *OECD* (dat vooral wiskundige en wetenschappelijke kennis test bij 15-jarigen) en is daarmee het eerste niet-Aziatische land na Singapore, Hong Kong, Zuid-Korea, Japan en Taiwan. Estland is op deze lijst 7e, Nederland 9e.

Dat Finland zo hoog scoort, komt volgens Koski door de bijzondere aanpak van het Finse systeem. Een van de verschillen met Nederland is bijvoorbeeld dat alle Finse docenten universitair geschoold zijn, en veel autonomie hebben om hun eigen curriculum in te richten. Een andere reden is de nadruk die het Finse onderwijssysteem legt op het onderwijs in wetenschap en techniek.

Dat laatste is ook de focus van Koski's dissertatie 'Connecting Knowledge Domains' (2014). Koski licht toe: "In het onderwijs krijgen kinderen een eerste inzicht in de natuurlijke en artificiële wereld om hen heen. Ik wilde weten hoe dat gebeurt en hoe het eventueel te verbeteren valt."

Dat er inderdaad nogal wel wat te verbeteren valt, was een van de conclusies van haar onderzoek. "Het lijkt erop dat wetenschap en techniek geen grote rol spelen in het Nederlandse basisonderwijs," aldus Koski. "Basisbegrippen en basisprincipes uit wetenschap en techniek worden vaak niet goed uitgelegd aan leerlingen."

Dat laatste bleek ook uit het veldwerk dat ze voor haar onderzoek uitvoerde onder leraren, leraren-in-opleiding, en leerlingen. De proefpersonen – uit alle drie de groepen – bleken vaak geen verbinding te kunnen leggen tussen wetenschappelijke en technische theorie en praktijk, een substantieel gebrek aan inzicht in technische begrippen te hebben, en vaak oorzaak en gevolg niet goed te kunnen onderscheiden. "Daardoor starten Nederlandse leerlingen in het voortgezet onderwijs vaak op een lager niveau qua technisch en wetenschappelijk inzicht dan Finse leerlingen."

"Er is maar heel weinig onderzoek gedaan naar de effecten van digitale geletterdheid en de opbrengst van programmeeronderwijs."

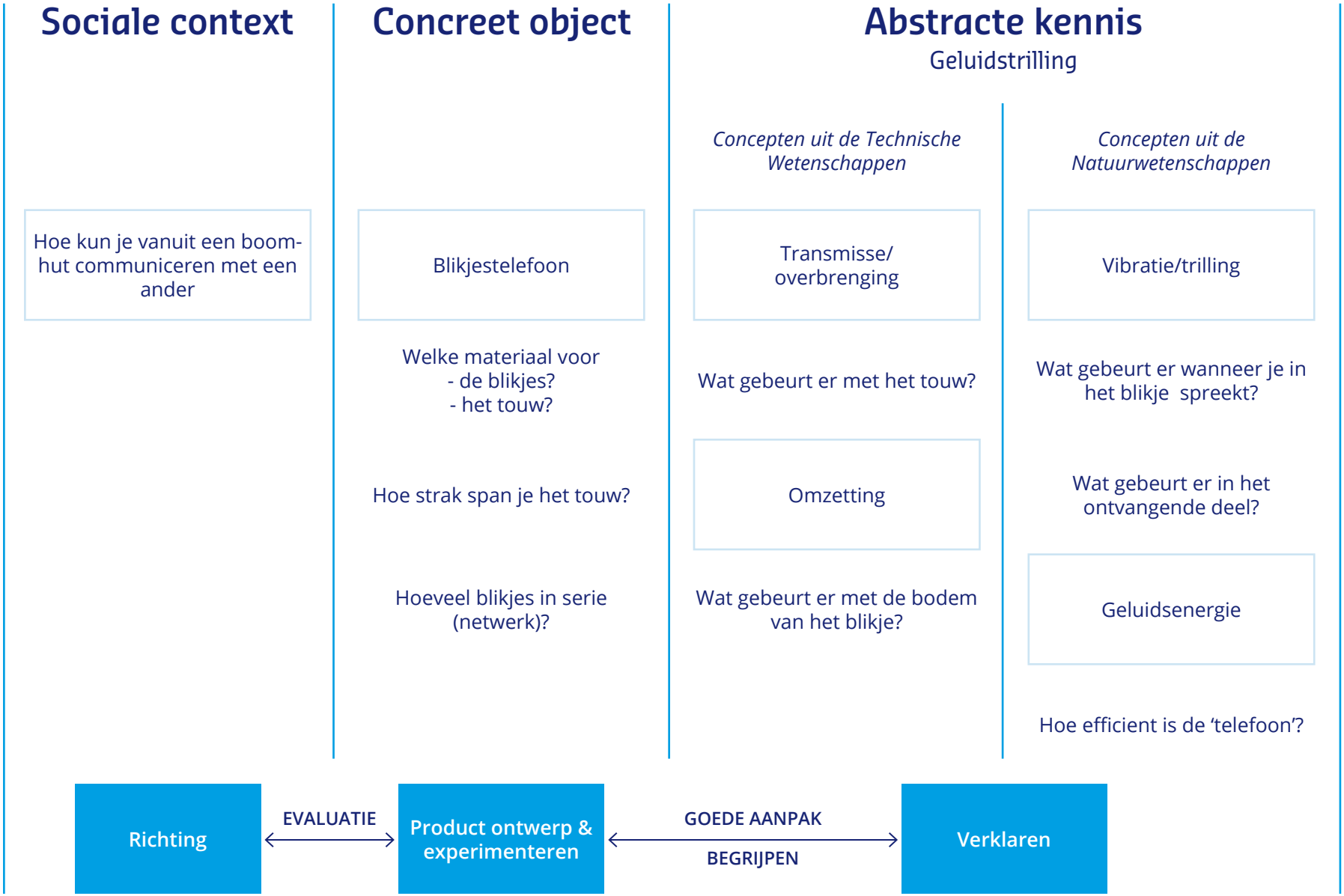
Dat wetenschap- en techniekonderwijs nog niet heel sterk doorgedrongen is in Nederland, blijkt ook uit het feit dat slechts 10% van de basisscholen voor een *techniekprofiel* heeft gekozen.

Overigens werd in mei 2013 een advies uitgebracht door SLO, om wetenschap en techniek beter vorm te geven in het basisonderwijs. Het ging daarbij om het zogenaamde *'Techniekpact'*. In 2014 volgde een gelijksoortig advies (van SLO), maar dan voor de *onderbouw van het voortgezet onderwijs.*[14]

Pedagogische onderbouwing

Koski ontwikkelde een model (zie afbeelding volgende pagina) om wetenschappelijke en technische begrippen beter te kunnen onderwijzen: "Ik wil leraren helpen om beter les te kunnen geven in techniek. Het brengt praktische en abstracte kennis bij elkaar."

wVolgens Koski is haar model een nuttig hulpmiddel om lessen over wetenschap en techniek voor te bereiden en uit te voeren. "Het brengt praktische en abstracte kennis bij elkaar in het hele onderwijsleerproces."



Het drie-domeinen-model van Koski (voorbeeldtoepassing)

Zelf gebruikt Koski deze aanpak in haar Amsterdamse academie 'NewTechKids', waar basisschoolkinderen via buitenschoolse onderwijs aan deelnemen. Tijdens de lessen – die zowel in het Nederlands als in het Engels gegeven worden – leren ze niet alleen programmeren, maar ook de fundamentele concepten achter computertechnologie te begrijpen. Er is veel aandacht voor algoritmisch denken, creatief oplossend vermogen en systeemdenken.

NewTechKids werkt als volgt:

- **jongere leerlingen (4-6 jaar)** krijgen *unplugged* lessen (zonder beeldscherm), bijvoorbeeld over het concept van de systematische herhaling (*loop*);
- **iets oudere kinderen (7-9)** gebruiken Lego Mindstorm en leren daar bijvoorbeeld over de verschillende soorten *loops* die er bestaan (while, repeat, for, etc.);
- **oudere kinderen (10-12)** gaan daarmee verder, en leren ook over de theorie en achtergronden van herhaling binnen systemen.

Vrijwel alle lessen draaien om fysieke objecten, zoals een robot die herhalende bewegingen moet maken. Koski: "Zo'n concreet, fysiek object is belangrijk, dat is het draaipunt waarbij de theorie en de praktijk met elkaar verbonden worden."

Maar wat dóet dit onderwijs nu eigenlijk? In hoeverre vergroten dit soort lessen daadwerkelijk 'creativiteit', 'probleemoplossend vermogen', 'abstracte denken' en 'analytische vaardigheden'? "Daar is nog heel weinig wetenschappelijk bewijs voor," geeft Koski toe. "Ook al omdat computing en programmeren als onderwijsvak natuurlijk nog niet zo lang bestaan. Je kunt de hogere-orde-denkvaardigheden nu nog niet testen want daarvoor moet je het denkvermogen en de prestaties van kinderen over langere tijd volgen."

Het is dus voor een kwestie van 'geloof'. Maar ondanks het gebrek aan wetenschappelijke onderbouwing gelooft Koski er heilig in dat ct-onderwijs van groot belang is. "Kinderen moeten technologisch geletterd worden," zegt ze kordaat. "We leven in een technologische tijd en dat zal alleen maar sterker worden. Als kinderen volwassen worden, hebben ze vaardigheden nodig om de problemen van hun tijd op te lossen. Daarvoor moeten ze logisch kunnen denken, technisch geletterd zijn, op een creatieve wijze problemen kunnen oplossen, en over 21e eeuwse vaardigheden beschikken."

Geen wetenschappelijk bewijs maar *best practices*

Professor Han van der Maas (Uva), hoogleraar psychologische methodenleer en initiatiefnemer van de online oefenprogramma's Rekentuin en Taalzee, vindt het niet zo vreemd dat er nog geen overtuigend wetenschappelijk bewijs is dat computational thinking doet wat het zou moeten doen.

Hieronder gaat Van der Maas in discussie met Alfons ten Brummelhuis (Kennisnet, NRO). Kort samengevat zijn hun posities als volgt:

- Van der Maas is een *believer*, die erkent dat er geen harde bewijzen zijn (en ook nauwelijks mogelijk zijn) voor de effectiviteit van ct-onderwijs, maar er desondanks wel in gelooft;
- Ten Brummelhuis is een *scepticus*, die toch wel graag wat onderzoek zou willen hebben, en zoveel mogelijk getoetst zou willen zien.

"Wat is de psychologische definitie van creativiteit?" vraagt Van der Maas retorisch. "En hoe kun je dat meten? Als je dat zou willen doen, moet je een lang lopend *double blind* onderzoek uitvoeren, zoals bij farmaceutisch onderzoek, waar heel veel geld, energie en tijd mee gemoeid is. Maar dat geldt voor het hele onderwijs. Als je er goed naar kijkt, is ons hele onderwijssysteem niet *evidence based*. Dat kan eigenlijk ook niet. Daarvoor ontbreekt het niet alleen aan geld en de tijd, maar er bestaan geen ook deugdelijke meetmethodes. Hoe meet je of kinderen over langere tijd echt iets hebben begrepen en geleerd dat ze niet op een andere manier hadden kunnen leren?"

"En toch hebben we een van de beste onderwijssystemen ter wereld. Dat zie je aan de cijfers," zegt Van der Maas. De manier waarop het onderwijs zich ontwikkelt, is volgens hem een systeem van *best practices:* docenten en scholen experimenteren met nieuwe onderwijsvormen, kijken of het wat oplevert, en als het goed gaat nemen anderen het over. Maar: "Je moet je wel realiseren dat elke onderwijsvernieuwing de eerste jaren voorspoedig verloopt. Iedereen is namelijk enthousiast en zet zijn beste beentje voor, zowel docenten als leerlingen. Als je dan gaat evalueren, lijkt het al snel alsof het een geweldige vooruitgang is."

Ten Brummelhuis reageert: "Er zijn inderdaad veel mensen die prachtige dromen hebben over programmeer- en ct-onderwijs. Maar als je die dromen niet toetst, heb je er weinig aan. We weten nog nauwelijks wat dit type onderwijs oplevert, en hoe je dat het beste kan geven."

Van der Maas is desondanks een voorstander van onderwijs in computational thinking: "Al heb je geen hard bewijs, daarom hoeft het nog steeds geen slecht idee te zijn. Soms weet je het niet helemaal zeker; dat is nu eenmaal eigen aan het onderwijs. Maar ik hoef alleen maar om mij heen te kijken om te zien hoe belangrijk het is om dit soort denken te beheersen, en enige programmeervaardigheid te hebben."

"Er zijn veel mensen die prachtige dromen hebben over programmeer- en ct-onderwijs. Maar als je die niet toetst, heb je er weinig aan."

Daar is Ten Brummelhuis het gedeeltelijk mee eens: "Ook al kun je niet voor de volle 100% wetenschappelijk bewijs vinden voor de vraag wat de precieze waarde van ct-onderwijs is, en hoe je het zou moeten inrichten; toch kun je door goed en gedegen onderzoek uit te voeren in ieder geval stap voor stap meer houvast krijgen."

Volgens Van der Maas is computational thinking een kernvaardigheid voor bijna elke (academische) opleiding: “Je moet creatief kunnen omgaan met een computer, want dat is de grondstof van onze informatiemaatschappij.” Van der Maas geeft het voorbeeld van zijn eigen vakgebied, de psychologie: “Kern van ons vak is het bouwen van experimenten. Daarvoor wordt *simulatie* steeds belangrijker, en daarvoor moet je iets weten van programmeren. En vervolgens moet je de resultaten kunnen interpreteren met behulp van statistische programma’s. Je moet bijvoorbeeld clusters in data kunnen opsporen. Om dat goed te kunnen doen, moet je gewoon op redelijk niveau kunnen programmeren. En dat is inmiddels bij veel universitaire studies zo.”

“Als je een paar uur per week spelenderwijs leert programmeren, dan heb je het later veel makkelijker. In vrijwel alle beroepen.”

Daarom vindt Van der Maas dat computational thinking, inclusief programmeren, een kernvak in het funderend onderwijs zou moeten zijn. “Als je een paar uur per week spelenderwijs leert programmeren, dan heb je het later zoveel makkelijker. In vrijwel alle beroepen.”

Ten Brummelhuis tekent aan: “Als je de rapporten leest van 30 jaar geleden, toen computers en ict voor het eerst opkwamen, dan kom je daar dezelfde argumenten en formuleringen tegen als nu: we móeten vanwege informatietechnologie van alles in het onderwijs invoeren en aanpassen, want anders missen we de boot. Maar we hebben toen allerlei dingen ingevoerd zonder dat we precies wisten wat het was, en lieten docenten grotendeels aan hun lot over. Daardoor kwamen informatica-onderwijs en informatiekunde nooit echt goed van de grond, totdat er uiteindelijk gezegd werd: we moeten er maar mee stoppen. De les die je uit die tijd kunt trekken, is dat je nooit iets overhaast grootschalig moet invoeren zonder dat je de opbrengsten ervan hebt onderzocht.”

Van der Maas ziet computational thinking vooral als een praktische vaardigheid: “Het gaat mij vooral om creatief denken voor het inzetten van digitale tools om een probleem op te lossen. Een voorwaarde daarvoor is het leren van een programmeertaal. Dat draagt bij aan het begrip van de mechanismen achter technieken en apparaten.”

Ten Brummelhuis vindt dat ook. Maar: “Je moet het alleen zorgvuldig invoeren, stapje voor stapje, en er op een systematisch manier naar kijken. Anders maak je dezelfde fouten als vroeger. Je zult moeten toetsen wat werkt, en wat de beste weg is. Wat wij merken bij Kennisnet, is dat scholen die behoefte wel hebben. Die willen graag dat de nieuwe onderwijsvormen die ze hebben opgezet, onderzocht en getoetst worden. Daar ligt nog een belangrijke taak.”

3 Praktijkervaringen

3 Praktijkervaringen

Basisonderwijs

Computational thinking is relatief nieuw maar toch gebeurt er al veel. Deels binnen het reguliere onderwijs en deels daarbuiten. Buitenschools bijvoorbeeld in de vorm van codeclubs, digilabs, fablabs, makerclubs, codeklassen en code-uren *(zoals NewTechKids.)*

Tijdens de Europese CodeWeek, in Nederland georganiseerd door ECP (een samenwerkingsverband van overheid en bedrijfsleven), wordt er meegedaan door tientallen Nederlandse basisscholen. Met commerciële partners als Randstad, Microsoft, Bol.com en Ziggo. Belangrijker dan deze incidentele programmeerklassen en eenmalige codeweken zijn de pogingen van diverse scholen om computational thinking structureel op te nemen in het curriculum.[1]

Het merendeel van de scholen doet echter niets met *programmeren*, volgens een steekproef uit 2015. Daaruit bleek dat maar 30% van de basisscholen en iets minder dan 50% van de scholen in het voortgezet onderwijs 'iets' aan programmeren doet. De meeste scholen die wél iets aan programmeren doen, geven dit vak 1 uur of minder per week (86% po, 68% vo).

De belangrijkste redenen om dit soort onderwijs te geven, waren:
- het beter voorbereiden op werk of opleiding (70% in het po, 57% in het vo);
- het vergroten van het probleemoplossend vermogen (61% po, 36% vo).

Het verst gevorderd zijn de scholen die aangesloten zijn bij:
- Stichting Oponoa (16 scholen);
- Onderwijsgroep Fier (17 scholen);
- Openbaar Onderwijs aan de Amstel (Ooada, 22 scholen).

De besturen van deze onderwijsgroepen stelden speciale leraren aan, en dienden een versnellingsvraag[2] in bij de PO-Raad, om uit te laten zoeken hoe een goede leerlijn eruit zou moeten zien. Dat resulteerde in de *'Leerlijn programmeren in het basisonderwijs'* die in mei 2016 werd gepresenteerd. Daarbij gaat het om de basisprincipes van programmeren, niet om programmeertools of -talen. Hiervoor is bewust gekozen om te voorkomen dat de leerlijn te snel veroudert.

1 Bijvoorbeeld: De Kleine Reus (Amsterdam), De Lorentzschool (Leiden), De Twaalfruiter (Vleuten), Shri Saraswatie School (Rotterdam), Shri Vishnu School (Den Haag), Driemaster (Lelystad), Dom Helder Camara (Groningen), Wjukslach (Langezwaag), Willemsparkschool (Den Haag), PCBO (Meppel), en IJpleinschool (Amsterdam-Noord).

2 'Versnellingsvragen' zijn ondersteuningsverzoeken van schoolbesturen, voor het ontwikkelen of implementeren van ict in het onderwijs. De PO-Raad en Kennisnet helpen deze vragen te beantwoorden en blokkades weg te nemen, zodat scholen niet meer afzonderlijk het wiel hoeven uit te vinden.

Onderwijsgroep Fier (NW-Friesland) – vernieuwing in krimpregio

Als algemeen directeur van de onderwijsgroep Fier in Noord-West Friesland, was Mark Vrolijk betrokken bij de bovengenoemde leerlijn. Sinds augustus 2013 heeft hij nadrukkelijk geïnvesteerd in de vernieuwing van het onderwijsaanbod.

Vrolijk: "Wij willen een onderwijsorganisatie zijn die op innovatieve wijze kansen biedt voor kinderen." Die wens hangt samen met de opdracht die hij meekreeg van het bestuur bij zijn aantreden. "Dit is krimpgebied, en daarom moeten we saneren. De leerlingenaantallen in dit gebied zijn al jaren aan het dalen. Mijn opdracht was om deze krimp te begeleiden, samenwerking te zoeken, en tegelijkertijd de onderwijsorganisatie toekomstbestendig te maken."

"Ook in een krimpregio kun je nieuwe kansen scheppen met innovatieve onderwijsoplossingen."

Fier koos ervoor om lesprogramma's te ontwikkelen rondom ict en 21ste eeuwse vaardigheden, en een fysieke leeromgeving in te richten: het *Community Learning Center* (CLC). Docenten werden uitgeroosterd, er werd samenwerking gezocht met andere scholen en instanties (zoals de Oponoa-scholen en SLO), en er werd hardware en software aangeschaft voor de inrichting van het CLC. Vrolijk: "Vorig jaar zomer legden we het plan voor aan de scholen, en vroegen we wie mee wilde doen om de lessen uit te proberen. Dat bleken er in eerste instantie drie te zijn. Die zijn in de pilot gestapt."

Alle leerlingen van deze scholen gaan vanaf 2016 een aantal keren per jaar naar het CLC, waar ze bijvoorbeeld robots leren programmeren en digitale opdrachten uitvoeren. "Op alle scholen bieden we verder de programma's van Snappet, Rekentuin, Taalzee en Got-it aan op tablets, zodat verschillende leerlingen op maat kunnen worden bediend. Door de krimp moeten we nu eenmaal meer met combinatiegroepen werken, en om leerlingen toch op hun eigen niveau te kunnen bedienen, hebben we geïnvesteerd in dit soort middelen." In de zelfstandige werkruimtes van de scholen kunnen leerlingen met behulp van de programma's en de tablets in hun eigen tempo aan de slag met hun weektaken, die zich aanpassen aan het niveau van de leerling.

De scholen van de Onderwijsgroep Fier die meededen aan het pilot rond het Community Learning Center (CLC) zijn allemaal enthousiast, zegt Vrolijk. "Sommige andere scholen wilden eerst de kat uit de boom kijken, maar die horen zulke enthousiaste verhalen dat ze nu zeggen: dat wil ik ook!" Ook de ouders zijn geestdriftig, en zeker de leerlingen. "Zo zie je maar dat je ook in zo'n krimpregio nieuwe kansen kunt scheppen met innovatieve onderwijsoplossingen."

Alberdingk Thijm ('t Gooi) – creatieve technologie als speerpunt

De Verenigde Scholen J.A. Alberdingk Thijm omvat 18 basisscholen en 6 middelbare scholen in Het Gooi, met 'creatieve technologie' en 'tweetaligheid/internationalisering' als speerpunten. 3 jaar geleden werd het vak 'programmeren' ingevoerd. In het zogenaamde *IT-lab* (bestaande uit twee speciaal ingerichte lokalen), worden om de week op dinsdagmiddag de beginselen van programmeren en computational thinking onderwezen, waarbij gebruik wordt gemaakt van robotjes, Lego Mindstorm, computers, en Scratch. Ook specifieke vaardigheden komen aan de orde, zoals de omgang met Office365, stop-motion filmpjes maken, en werken met een elektronische microscoop.

Er doen 80 geselecteerde (basisschool)leerlingen mee; voor de helft afkomstig uit de onderbouw, en voor de andere helft uit de bovenbouw. "Het zijn leerlingen die een middag van het reguliere onderwijs kunnen missen," zegt Herman Rigter, ict-directeur van de scholengroep. "De leerkrachten bepalen wie dat zijn, waarna in overleg met de ouders en de schoolleiding wordt beslist wie de lessen kunnen volgen."

Vaak wordt er in groepjes gewerkt aan opdrachten, en meestal aan de hand van een specifiek probleem. Bijvoorbeeld: hoe je een robot een route laat afleggen over een bepaald parcours. Rigter: "We vinden het een waardevolle aanvulling op ons onderwijsaanbod en vinden dit een kans voor leerlingen die een middag in de week kunnen missen."

Impliciet wordt daarbij ook aandacht besteed aan computational thinking. Rigter: "Aan het eind van elke les wordt er vaak even stilgestaan bij principes als algoritmes, foutopsporing en herhaling, maar ook over mogelijke toepassingen en gevaren. Daarmee proberen we gestructureerd, kritisch en creatief denken te bevorderen. Maar het gaat vooral om te durven experimenteren, en daarbij hoort ook dat je af en toe kunt falen. Geen jongere zou meer moeten schrikken van techniek, is onze opvatting."

"Dat ict bij vernieuwingen vaak een belangrijke rol speelt, is secundair. Uiteindelijk wil ik het gesprek aanzwengelen: wat is nu eigenlijk goed onderwijs?"

Naast deze speciale dinsdaglessen, krijgen alle basisschoolleerlingen van het Alberdingk Thijm (van groep 5 tot 8) ongeveer 6 ict/programmeer-lessen per jaar. "Die worden gegeven door hun eigen leraren, aan de hand van een handleiding die we zelf hebben laten maken."

De vraag kwam vanuit de scholen zelf. Op de middelbare scholen van de scholengroep was al veel expertise aanwezig over de inzet van ict in het onderwijs, inclusief de bijbehorende curriculumontwikkeling. Mede op basis van deze kennis werd het lesprogramma voor de basisscholen opgezet. Daarbij werd er samengewerkt tussen leraren uit het basisonderwijs en het voortgezet onderwijs.

Bij Alberdingk Thijm dient het programmeer- en ct-onderwijs ook een breder doel, namelijk permanent onderzoek naar onderwijsvernieuwing in het algemeen. Ict-directeur Rigter: “Dat ict bij vernieuwingen vaak een belangrijke rol speelt, is secundair. Uiteindelijk wil ik het gesprek aanzwengelen: wat is nu eigenlijk goed onderwijs? En hoe kunnen we dat met elkaar, en met behulp van technologische vernieuwingen, realiseren?”

Oponoa (Achterhoek) – de rijdende leraar

Sandra Legters is onderwijskundig coördinator techniek en ict bij Stichting Openbaar Primair Onderwijs Noord Oost Achterhoek (Oponoa, 16 basisscholen). Sinds 2010 rijdt zij rond in de Achterhoek, en bezoekt ze elke dag een school om programmeerlessen te geven. Legters: “Op 7 van de 16 scholen krijgen kinderen die een extra uitdaging nodig hebben een uur per week les in programmeren”. Om het overzichtelijk te houden zijn de groepen niet groter dan 8 kinderen. “Anders kun je niet genoeg aandacht schenken aan iedereen.”

Omdat het een langlopend traject is, is er binnen Oponoa een doorlopende leerlijn ontwikkeld in het kader van de versnellingsvragen van de PO-Raad (met steun van SLO en Kennisnet), samen met onder andere de eerdergenoemde Onderwijsgroep Fier uit Noord-West Friesland. Deze leerlijn zal ervoor zorgen dat de lessen op elkaar aansluiten en op elkaar voortbouwen. Zo leren kleuters aan de hand van unplugged-lessen op een speelse manier inzichten over ct-principes als herhaling, sorteren, procedures en algoritmes. Later werken ze met de Bee-Bot, een programmeerbaar robotje in de vorm van een bij.

“Programmeren is niet alleen voor plusklassen of naschoolse lessen.”

Omdat Legters en Oponoa al enige jaren werken met programmeren in het onderwijs, hebben ze al vele hulpmiddelen en methodes toegepast. Maar inmiddels werkt Legters vooral met *Code Studio* van Code.org, een website die aansluit op de recent ontwikkelde leerlijn. Legters: “Die site ontdekte ik in het voorjaar van 2014. Ik zag dat dit platform mooi paste bij onze ideeën, met prachtig voorbeeldmateriaal en een heel actieve groep gebruikers, die snel vragen konden beantwoorden en problemen konden oplossen.”

Legters is een groot voorstander van een brede aanpak, waarbij alle leerlingen en leraren kunnen meedoen. “Dus niet alleen plusklassen of naschoolse lessen.”

Delfzijl en omgeving (Noord-Groningen) – projectlessen van Google

Soms moet je werken met korte trajecten, zoals in Noord-Groningen gebeurde, bij 20 scholen in en rond Delfzijl, die elk 5 programmeerlessen van 2 uur aangeboden kregen van speciaal getrainde studenten van de Pedagogische Academie in de stad Groningen. Deze projectlessen waren een idee van het internetbedrijf Google, dat een datacenter bouwt in de Eemshaven, in het noorden van Groningen. Google vond het volgens manager Thomas Rulmont belangrijk om 'een goede buur' te zijn, en de jeugd enthousiast te maken voor programmeren.

Bij de Pedagogische Academie van de Hanzehogeschool waren docent Richard Doornbos en toenmalig student en huidig onderzoeker Hylke Faber inhoudelijk verantwoordelijk voor de lessen. Zij betrokken op hun beurt de Stichting Groningen Programmeert bij het project. Faber: "Alle lessen waren unplugged. Ze waren afkomstig van www.codekinderen.nl (van Kennisnet – red.) of werden door ons zelf ontwikkeld."

Een les over 'variabelen'

Faber haalt een doos met koffiebekertjes tevoorschijn, en legt uit dat een van de bekertjes de variabele is. "Doe je er een fiche in, dan krijgt die variabele een waarde. De variabele krijgt een naam door een sticker op het bekertje te plakken." Ander voorbeeld: werken met 'voorwaarden'. "Als je een zwarte kaart uit een stapeltje speelkaarten trekt, moet je een stap vooruit. Zo kun je een parcours afleggen." Bij lessen over 'herhaling', moest een cirkel in achten worden verdeeld en moest telkens dezelfde bewerking worden gedaan.

Doornbos: "Deze lessen waren unplugged omdat er op veel scholen nog niet voldoende computers waren. Wel gaven we links naar websites, zoals code.org, waar allerlei materiaal te vinden is, onder andere over programmeren in Scratch, zodat kinderen en leraren zelf verder aan de slag konden."

"Kinderen pikken heel snel iets op, soms op een niveau dat je vooraf niet verwacht."

Wat Faber en Doornbos – die zelf ook les gaven op de Noord-Groningse basisscholen – opviel, was dat de leerlingen niet alleen enthousiast waren, maar ook behoorlijk veel aankonden qua opdrachten en ideeën. Doornbos: "Kinderen pikken heel snel iets op, soms op een niveau dat je vooraf niet verwacht. Zo leerden we ze in korte tijd iets over binaire getallen en daarin bleken ze behoorlijk goed. Daardoor konden we zelfs binaire getallen laten vertalen in cijfers, daarna in letters, en vervolgens in toetsen van het toetsenbord. We twijfelden we erover of ze dat wel zouden begrijpen, maar het bleek voor hen allemaal heel goed te doen."

De lessenserie wordt geëvalueerd door studenten van de Groningse pabo. Maar nu al is er een plan voor volgend jaar: "Volgend jaar willen we de basisscholen zelf de lessen laten geven. Wij lopen dan alleen rond ter ondersteuning. In het derde jaar kunnen ze dan zelf zo'n project beginnen en de lessen geven," aldus Doornbos.

Voortgezet onderwijs

Ook in het voortgezet onderwijs zijn al veel scholen bezig met programmeren en computational thinking. Hieronder een aantal praktijkervaringen.

Grotius College (Heerlen) – elke maandag een MakerKlas

Wiskundeleraar Ralph Crützen geeft informatica op het Grotius College in Heerlen. Hij organiseert daar elke maandag een MakerKlas, waarin niet alleen de beginselen van programmeren aan bod komen, maar ook robotica en 3D-printen (zie Bijlage). Daarnaast geeft hij informatica – als keuzevak – op havo- en vwo-niveau.

De leerlingen die voor informatica kiezen, krijgen les in een apart computerlokaal waar 32 Windows-computers (laptops en desktop-pc's) staan opgesteld. Het aantal lesuren verschilt per leerjaar en schooltype. Plus eventueel een extra uur voor de MakerKlas.

Crützen: "Er is geen centraal schriftelijk voor dit vak, maar we geven wel schooltoetsen en praktische opdrachten die meetellen voor het eindcijfer". Hij gebruikt de lesmethode *'Informatica-Actief'* en af en toe maakt hij ook zelf lessen. Die zijn gebaseerd op *'EduScrum'*, een educatieve variant van 'Scrum' (software-ontwikkeling door middel van samenwerken in multidisciplinaire teams).

Jan van Brabant College (Helmond) – programmeren in de brugklas

Op het Jan van Brabant College in Helmond wordt vanaf augustus 2016 standaard lesgegeven in programmeren en computational thinking aan alle brugklassers van alle niveaus. Gedurende een half jaar zal er 2 uur per week worden lesgegeven door Hans Vasse, docent 'design en kunstgeschiedenis', en twee van zijn collega's (een docent techniek, en een docent natuurkunde).

De leerlingen werken op hun eigen Chromebook, in hun eigen tempo en op hun eigen niveau. Vasse : "We hebben een leerlijn ingericht die het mogelijk maakt voor snelle leerlingen om extra oefeningen te doen, zodat ze zich niet hoeven te vervelen. En de leerlingen die wat meer moeite hebben met de materie, kunnen in hun eigen tempo de stof tot zich nemen zonder dat ze zich hoeven te haasten."

Deze aanpak lijkt een beetje op die van Groot-Brittannië: *verplichte standaardlessen voor alle leerlingen.* "Bij ons is het verplicht," zegt Vasse. "Waarbij het vooral gaat om het verhogen van het probleem-oplossend vermogen."

Centraal in het ct-onderwijs van het Jan van Brabant College staat een MOOC (massive open online course): een online cursusomge-

ving voor iedereen. Leerlingen en docenten kunnen hiervan altijd en overal gebruik maken, zolang ze maar online zijn. “Op die manier laat je de leerlingen veel meer hun eigen weg zoeken,” zegt Vasse.

De inspanningen van het Jan van Brabant College hangen samen met het feit dat het een *BrainPort-school* is. BrainPort-scholen bevinden zich in een internationaal georiënteerde regio met een zeer hoog kennisniveau, werken samen met bedrijven in de omgeving, en proberen het onderwijs meer aan te laten sluiten bij de latere beroepspraktijk. Vasse: “Wat we van die bedrijven hoorden, was vooral dat we kinderen moeten leren om zich flexibel op te stellen. Ze moeten oplossingen bedenken voor echte problemen, bijvoorbeeld als de parameters veranderen tijdens het ontwerpproces. Dat soort denken wordt in het traditionele onderwijs veel minder geleerd, vandaar dat we ons heel erg hebben gericht op computational thinking-onderwijs.”

“Wat we van die bedrijven hoorden, was vooral dat we kinderen moeten leren om zich flexibel op te stellen.”

“De leerlingen hoeven echt niet allemaal programmeur te worden,” zegt Vasse. “Maar ze moeten wel de denktrant begrijpen. Daarbij gaat het niet om de vraag hoe een computer precies werkt, maar hoe je de bijbehorende problemen kunt oplossen. We proberen dat als basisvaardigheid aan te leren bij onze leerlingen.”

Pabo

Het voorbereiden van leraren (po) op lesgeven in computational thinking en programmeren is een taak voor de pabo's. Hoe gaan de opleidingen daarmee om? Hieronder twee voorbeelden: de Hogeschool Utrecht in Amersfoort, en de Hanzehogeschool in Groningen.

Hogeschool Utrecht – ontwerpend leren

"In elk studiejaar is er aandacht voor ict in de lespraktijk," zegt Gerard Dummer, opleidingsdocent ict & onderwijs aan de pabo van de Hogeschool Utrecht in Amersfoort. "Onze studenten leren hoe ze hun onderwijs kunnen verrijken met de inzet van ict. Van instructies op het digibord, tot creatieve verwerkingsopdrachten. We werken daarvoor samen met de verschillende vakdocenten."

Het curriculum is als volgt ingericht:

- **1e jaar:** introductiecursussen
- **2e jaar:** keuzevak 'wetenschap en techniek'
- 3e jaar: Beebots, Scratch, en unplugged-lessen
- **4e jaar:** programmeer- en ict-projecten

Bij de projecten kan het bijvoorbeeld gaan om het maken van een digitaal educatief product. Dummer: "De studenten hadden de keuze uit: het maken van een educatieve video, een WikiKids-Portaal, een educatieve website, of een Geocache. Het was de bedoeling dat ze een onderwerp zouden kiezen dat dicht bij hun belevingswereld zou staan, maar ook dicht in de buurt van hun eigen school of omgeving: een lokaal onderwerp, waarover nog niet zo veel materiaal beschikbaar was."

Er wordt gewerkt volgens het principe van 'ontwerpend leren', vooruitlopend op de invoering van het techniekonderwijs op de basisschool (geplande invoering: 2020). Volgens Dummer vergt het wel enige creativiteit om een ontwerpprobleem te formuleren. Is het probleem te breed, dan is het niet haalbaar om het op te lossen. En is het te gesloten, dan lokt het probleem geen creativiteit uit. "Belangrijkste van het model is dat je kinderen probeert grip op de wereld te geven aan de hand van een concreet probleem," zegt Dummer. "Daar biedt dit model goede handvatten voor."

Speciale aandacht voor ict en computational thinking is volgens Dummer nodig omdat beginnende leraren zich vaak nog niet voldoende voorbereid voelden op de inzet van ict in het onderwijs. "We proberen onze studenten daarom zo goed mogelijk voor te bereiden op de toekomst, ook al omdat vanaf 2020 alle basisscholen verplicht aandacht moeten besteden aan wetenschap en technologie. Ons eigen onderwijs in programmeren, ict en computational thinking sluit daarbij aan."

Hanzehogeschool Groningen – *work in progress*

De pabo van de Hanzehogeschool Groningen verkeert nog in de onderzoeksfase. “Computational thinking is op dit moment nog geen vast onderdeel van het curriculum,” zegt docent rekenen en wiskunde Richard Doornbos (die eerder in dit rapport al vertelde over zijn betrokkenheid bij de projectlessen op basisscholen in Delfzijl en omgeving). “Dit jaar hebben we voor het eerst een introductiecursus gegeven voor een kleine groep belangstellende studenten. En verder willen we dat je in de toekomst voor je minor iets in de richting van computational thinking kunt doen. Bijvoorbeeld programmeerlessen volgen als je het keuzeprogramma ‘science’ doet,” aldus Doornbos.

Doornbos: “We laten de studenten bijvoorbeeld zien hoe ze een *kahoot* via het digibord kunnen organiseren: een responssysteem waarbij de leraar vragen aan de leerlingen stelt via het digibord, die ze op hun eigen smartphone, tablet of notebook kunnen beantwoorden. Vervolgens worden de resultaten op het digibord geprojecteerd en kan er een discussie ontstaan.”

Samen met docent Doornbos onderzoekt junior-onderzoeker Hylke Faber de mogelijkheden van het nieuwe vakgebied. In dat kader ging Faber onder andere naar Finland, waar hij sprak met Linda Liukas van *‘Hello Ruby’*, een veelgebruikte methode (annex boek). Hello Ruby is een spannend en avontuurlijk verhaal dat jonge kinderen digitale vaardigheden bijbrengt en creatieve oplossingen leert zoeken.[25]

“We laten de studenten bijvoorbeeld zien hoe ze een kahoot via het digibord kunnen organiseren.”

Doornbos en Faber betwijfelen of programmeren een apart vak op de pabo van de Hanzehogeschool moet worden. “We zien veel meer in het integreren van dit soort ideeën en vaardigheden in de bestaande vakken,” zegt Doornbos. “Misschien dat we in de toekomst gaan beginnen met een paar programmeerlessen, maar dat we het daarna laten terugkomen bij taal, rekenen, aardrijkskunde. De principes van computational thinking zoals het opdelen van problemen in stapjes kun je ook toepassen bij het rekenen. De ct-vaardigheden die je leert, kun je op allerlei momenten toepassen binnen verschillende vakken. Een voorbeeld daarvan is een wiskundig raadsel over een slak. Een slak kruipt over een elastiek naar een muur toe. Het elastiek wordt na elke minuut een meter uitgerekt. Haalt de slak de muur? Ik heb dat raadsel niet op een wiskundige manier opgelost, maar in Excel. Dan maak je gebruik van je ct-skills. Dat willen we ook aan de studenten meegeven.”

“Er moet nog veel onderzocht worden,” besluit Faber. “En dat gaan we de komende jaren doen.”

4 Transfer

4 Transfer

Bij onderwijskundige kwesties wordt altijd de vraag gesteld hoe de transfer, oftewel 'de overdracht', geregeld is. Maar wat wordt daar eigenlijk mee bedoeld? Wat wordt overgedragen naar wat?

Hieronder worden drie gevallen van transfer bij 'computational thinking' beschreven. De eerste blijkt niet van toepassing te zijn, en de overige twee roepen ernstige vragen op. Al met al is de transfer van computational thinking dus eigenlijk nihil.

Soorten transfer

Bij **'transfer'** kan het om drie vormen van overdracht gaan:

- de overdracht *van* **datgene wat je geleerd hebt** (theoretische kennis) **naar de praktijk;**
- of andersom: de overdracht **van praktijkervaring naar** (theoretische) **kennis;**
- de overdracht van **kennis uit het ene domein** (zoals wiskunde) **naar het andere domein** (zoals natuurkunde of psychologie).

We gaan dieper in op de derde transfer-variant: de overdracht van kennis vanuit het ene domein naar het andere. In hoeverre kan ct-onderwijs leiden tot betere prestaties op andere fronten? Die vraag blijkt moeilijk te beantwoorden.

We spraken met hoogleraar Han van der Maas (Uva), die bekendheid verwierf met zijn onderzoek naar de transfer van schaken naar wis- en natuurkunde. Omdat Van der Maas zelf ooit programmeerlessen gaf op een basisschool, kent hij het transfer-probleem van dichtbij.

Liever schools dan zandbak

Van der Maas: "Als je dit soort lessen geeft, zit je vaak in een spagaat. Je hebt het constructivistisch idee van de zandbak ('laat ze maar lekker experimenteren') maar de kans is dan groot dat ze foute technieken gaan toepassen, en vast blijven zitten in allerlei misconcepties. Je zou dus een wat schoolsere aanpak moeten volgen, waarin je ze stap voor stap dingen leert."

"De vraag is altijd: waar is het redelijke midden? Daar moet iedere docent zelf achter zien te komen, want geen onderwijssituatie is hetzelfde. Maar je moet als docent wel durven ingrijpen, en af en toe een duwtje durven geven. Want kinderen hebben de neiging om steeds hetzelfde te blijven doen, en steeds dezelfde trucjes te blijven herhalen die ze al goed kunnen. Dat moet je zien te doorbreken."

Transfer valt vaak tegen

Wat leerlingen uiteindelijk opsteken van dit type onderwijs, behalve de durf om apparaten te leren ontdekken en te programmeren, is volgens Van der Maas moeilijk te zeggen. In hoeverre worden ct-principes als algoritmes, abstract denken en probleemoplossen bijgebracht, als je leert hoe je een robotje een parcours kunt laten afleggen?

Van der Maas: “Je moet goed beseffen hoe lastig het is om de principes van het programmeren naar andere vakken over te brengen. In het algemeen blijkt uit onderzoek dat die transfer vaak erg tegenvalt.”

Transfer expliciet maken

Bij schaken zie je volgens Van der Maas hetzelfde: de algemene kennis die je daarbij kunt opsteken, pas je vaak niet toe op andere domeinen, omdat het een *impliciete transfer* is.

Van der Maas: “Als je wilt dat de kennis en de inzichten die je bij het ene vak hebt opgedaan, ook in een ander vak toegepast kunnen worden, dan moet je dat actief bewerkstelligen. Dan moet je het bij programmeren over logische operatoren hebben, en laten zien hoe dat in de wiskunde werkt. Je moet de transfer dus expliciet maken. Maar zelfs dan is de opbrengst vaak gering.”

5 De uitvoerders

5 De uitvoerders

Wie geeft het vak?

Computational thinking kan door verschillende personen gegeven worden:

- op de basisschool zijn het vaak vrijgeroosterde leraren. Dit is onder andere het geval bij de eerder beschreven scholen van Oponoa, Fier en Alberdingk Thijm;
- of er worden programmeer-leraren van buiten aangetrokken, zoals in Noord-Groningen;
- op de middelbare scholen zijn het vaak leraren wiskunde- of natuurkunde die informatica of informatiekunde geven (Grotius, Alberdingk Thijm).

Allard Strijker (SLO): “Op veel basisscholen waren het tot nu toe vooral degenen die enthousiast waren voor het vak. Maar als je computational thinking echt een eigen plek in het onderwijs wilt geven, zul je ook de andere docenten erbij moeten betrekken. Maar dat wil niet iedereen, en kan ook lang niet iedereen. Nederland is Engeland niet en hier is het niet verplicht om les te geven in computational thinking of programmeren. Gelukkig is er nu de eerste leerlijn programmeren in het basisonderwijs. Daarmee kunnen ook de andere docenten een kijkje nemen in dit nieuwe vak.”

SLO is overigens van plan verder te gaan met een leerlijn die niet alleen op programmeren is gericht, maar vooral ook op computational thinking.

Wat moet je ervoor kunnen?

Wat zijn de eisen die je aan een computational thinking-docent mag stellen? Marja-Ilona Koski (NewTechKids) formuleert drie vereisten. “Zo’n docent moet in ieder geval begrijpen hoe programmeren werkt en wat de verschillende computational thinking-begrippen inhouden. Daarnaast moet hij of zij minimaal één keer een eigen programmeerproject hebben afgerond. Je moet weten hoe zoiets werkt en wat er bij komt kijken. Tenslotte moet een docent zijn eigen curriculum kunnen samenstellen, gericht op het niveau van zijn leerlingen.”

Han van der Maas, hoogleraar psychologische methodenleer aan de UvA, is het hier grotendeels mee eens. “Je moet een beetje kunnen programmeren en wat met dit soort programma’s geëxperimenteerd hebben.” Voor de rest is het voor Van der Maas tamelijk eenvoudig: vakkennis. “Je kunt het gewoon vergelijken met wiskunde en natuurkunde. Voor 90% is het vakkennis: begrippen, theorieën en ideeën. Je moet computational thinking en programmeren dan gewoon als vak benaderen.”

Volgens Van der Maas zouden uitgevers daarom met goede leermethodes moeten komen. “Je moet als docent niet voortdurend zelf lesjes hoeven maken. Dat schiet niet op.”

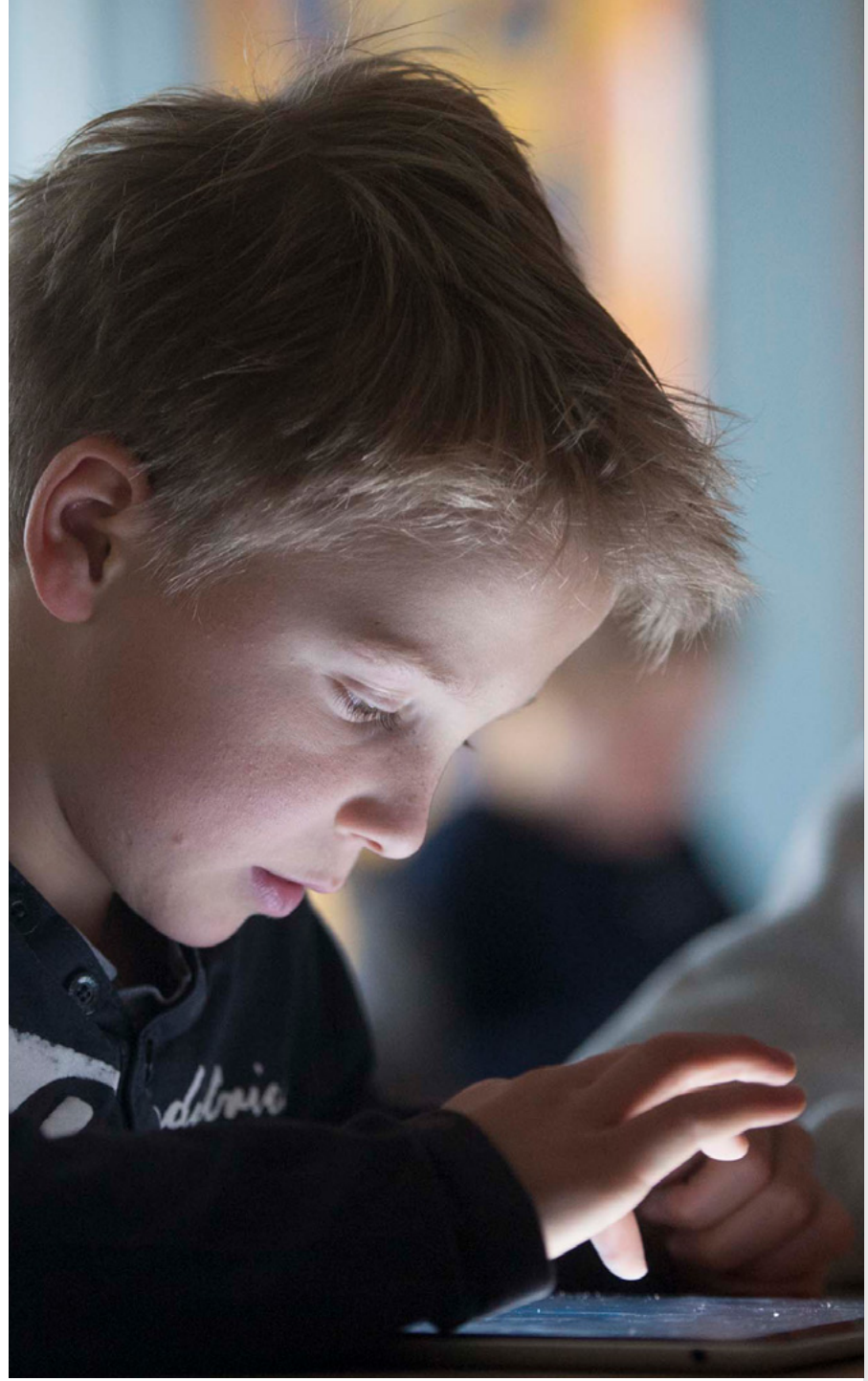
Marja-Ilona Koski (NewTechKids) vult aan dat docenten geen computerwetenschap op universitair niveau hoeven te beheersen, maar

dat ze wel moeten begrijpen waar het in computational thinking over gaat. En dan is er volgens nog een iets andere *mindset* nodig. “In dit vak moet je durven experimenteren en daarvoor moet je ook durven falen. Dat geldt voor de docenten en voor de leerlingen. Af en toe mislukt er iets. Dat is onderdeel van het proces.”

“Je moet als docent niet voortdurend zelf lesjes hoeven maken. Dat schiet niet op.”

Ook aan de pabo’s stelden we de vraag aan welke eisen een ct-docent moet voldoen. Volgens Gerard Dummer (Hogeschool Utrecht) moet je in ieder geval een beetje kunnen programmeren, het ideeën- en begrippenapparaat van computational thinking beheersen en zelfs ooit iets hebben geprogrammeerd in een taal als scratch. Dummer zou graag de lijn van *Mitch Resnick* willen volgen, die een onderscheid maakt tussen computationele *concepten* (zoals loops en data), computationele *vaardigheden* (zoals testen, debuggen en hergebruiken), en computationele *perspectieven* als expressing en questioning.[26]

Hylke Faber (Hanzehogeschool Groningen) is daar iets milder over. “Ze moeten inderdaad iets weten over computational thinking, maar ze hoeven geen rekenmachine te leren programmeren. Wel is het belangrijk dat ze kennismaken met een aantal methodes en ideeën en dat ze die kunnen wegen.”



6 Lessen voor alle betrokkenen

6 Lessen voor alle betrokkenen

Lessen voor onderzoekers

- Laat de scholen niet aan hun lot over, maar flankeer hun ct-/programmeeronderwijs met onderzoek. Onderzoek wat de scholen doen en wat de opbrengsten zijn.
- Sluit zoveel mogelijk aan bij de wensen van de scholen, vooral als het gaat om de waarde van ct-onderwijs. Ga met dit soort innovatieve scholen in gesprek, en bepaal gezamenlijk welk soort onderzoek van belang is.
- Onderzoek wat de opbrengsten zijn van dit soort onderwijs voor de leerlingen. Welke inzichten verwerven zij, die ze op een andere manier niet hadden kunnen verwerven?
- Onderzoek vooral ook de toetsingsmogelijkheden. En: welke vormen van feedback hebben het beste resultaat?

Lessen voor schoolbestuurders

- Begin niet lukraak met ct-/programmeeronderwijs, maar formulier een visie en maak een strategisch beleidsplan.
- Kijk naar de samenhang tussen het ct-/programmeeronderwijs enerzijds en de bestaande vakken anderzijds. Betrek ook de 21e eeuwse vaardigheden daarbij.
- Bepaal vooral waaróm je dit soort onderwijs zou willen: om de arbeidsmarktpositie van de leerlingen te verbeteren, of om hun creativiteit en probleemoplossend vermogen te verbeteren? In het laatste geval moet je eerder beginnen.
- Maak een keuze: moet het een apart vak worden, of integreer je ct in andere vakken?
- Praat met andere scholen die langer ervaring hebben met ct-onderwijs en leer van hun fouten en oplossingen. Gebruik daarbij instanties als Kennisnet als verbindingsofficier.
- Zorg dat je genoeg kennis in huis haalt om goede plannen te kunnen ontwikkelen en goed te kunnen beslissen over ct-onderwijs.
- Overweeg een gelijksoortig initiatief als de Fier Academy (*www.fier.nl/fier-academy*) om de deskundigheid van je leraren te bevorderen.
- Gebruik mogelijkheden als de NRO Kennisrotonde (*www.nro.nl/kennisrotonde*) om doelgerichte vragen te stellen over de inrichting van dit soort onderwijs.

Lessen voor leraren

- Gebruik een degelijke leerlijn, zoals de ‘Leerlijn programmeren in het basisonderwijs’ (PO-Raad, SLO, Kennisnet 2016). Lessen van internet plukken of zelf bedenken kan ook, maar is vaak minder gestructureerd dan een pedagogisch verantwoorde leerlijn.
- Wil je niet meteen dure apparatuur aanschaffen? Overweeg dan om apps (voor tablets of smartphones) te gebruiken die de echte apparatuur simuleren. Bijvoorbeeld: Beebot-app, Cargo-Bot, Hopscoth, Lego Trains, Lightbot One Hour Coding, Robologic, Robot Bros of ScratchJr.
- Bedenk dat er veel unplugged-lessen zijn, waarbij je geen computers en schermen nodig hebt. Je vindt ze onder andere bij CodeUur en Smartlife.
- Laat de leerlingen hun programmeerproblemen verwoorden; aan elkaar of aan de computer. Zo vinden ze vaak het begin van een oplossing.
- Snelle leerlingen kunnen hun tragere klasgenoten helpen, maar zorg dat ze niet gaan voorzeggen. En laat ze vooral niet andermans toetsenbord overnemen. Leer hen vooral vragen te stellen en hints te geven.

7
Meest gestelde vragen

7 Meest gestelde vragen

Als een school computational thinking wil invoeren, kunnen zich allerlei vragen voordoen. Hieronder de vragen die het meest worden gesteld.

Wat zijn de struikelblokken?

In 2015 vroeg Kennisnet aan scholen waarom ze (nog) *niets aan programmeren of computational thinking deden.*
Dat leverde het volgende beeld:

	po	vo
Meer duidelijkheid nodig over wat te doen of wat we willen	17%	45%
Meer professionalisering nodig	23%	40%
Meer of betere lesinhoud nodig	29%	37%
Meer steun/visie/overtuiging van bestuur of directie nodig	34%	34%
Meer tijd/ruimte in rooster/curriculum nodig	17%	20%
Betere apparatuur en/of devices nodig	15%	19%
Stabielere en/of snellere internetverbinding nodig	7%	14%
Het past niet in het programma	5%	10%
Het is geen onderdeel van eindtoets of centraal examen	38%	0%

Waarom doet uw school (nog) niets aan programmeren? (Kennisnet, 2015)

Hieronder geven deskundigen commentaar op de belangrijkste problemen. Marja-Ilona Koski (NewTechKids) vat het samen in een notendop: “Je moet in ieder geval een deugdelijk onderwijsplan hebben. Het liefst een degelijke leerlijn die het leren van dit soort kennis en vaardigheden in stapjes opbouwt. Maar daarmee ben je er nog lang niet. Je hebt vooral ook een goede docententraining nodig.”

Planmatige aanpak – Volgens ten Brummelhuis (Kennisnet en NRO) moet je je vooral afvragen waaróm je leerlingen computational thinking zou willen bijbrengen. Wat is je motivatie, als docent, als school, en als samenleving? “Het maakt nogal wat uit of je het doet om leerlingen een betere positie in de latere arbeidsmarkt te geven, of dat je in het algemeen een bepaald soort denken wilt aanleren. In het eerste geval kun je je richten op een veel kleinere groep leerlingen, en kun je misschien ook op een iets hogere leeftijd beginnen met programmeerlessen.”

Leerlijn – Allard Strijker (SLO) bevestigt het belang van een deugdelijke leerlijn. Hij vindt daarom de nieuwe leerlijn *‘Programmeren in het basisonderwijs’* (PO-Raad, 25 mei 2016) een stap in de goede richting. “Dat is een redelijk laagdrempelige leerlijn, waarin je als docent niet alleen het materiaal leert gebruiken, maar ook iets meer te weten kunt komen over de achtergronden en doelen van dit nieuwe vak. Vergeet niet dat docenten natuurlijk met een enorme inhaalslag bezig zijn op het gebied van computational thinking. In Engeland is het niet alleen verplicht vak, maar hebben ze ook een grote organisatie als Computing at School (CAS) achter zich staan die allerlei materiaal aanbiedt, cursussen en bijscholen geeft, en de boel organiseert.”

Professionalisering – Alfons ten Brummelhuis (Kennisnet en NRO) benadrukt de rol van de uitvoerende professionals: “De waarde van dit soort onderwijs zit altijd in de competentie van de docenten,” zegt hij. “Het tempo en de snelheid waarin onderwijsvernieuwingen een plek krijgen binnen het bestel, heeft altijd te maken met de mogelijkheden en de motivatie van degenen die het moeten uitvoeren, en dat zijn de docenten. Soms gaat het heel snel, zoals we zagen bij het toepassen van digiboards (die veel sneller werden ingevoerd dan beleidsmakers hadden kunnen bedenken) en soms gaat het heel langzaam. Maar de kern is: als een professional er niet van overtuigd is of de onderwijsinnovatie bijdraagt aan de kennisvermeerdering en vaardigheidsverbetering van de leerlingen, dan houdt hij er vroeg of laat mee op.”

Individueel of in groepjes?

Po – Bij de Gelderse Oponoa-scholen werken de kinderen af en toe in groepjes, maar doen ze ook veel individuele opdrachten. “Het hangt van het soort les af,” zegt Sandra Legters (onderwijskundig coördinator techniek/ict, die als reizend gastdocent 7 scholen bedient). “Bij unplugged-lessen doen we vaak groepsopdrachten, maar als de kinderen uit de hogere groepen echt gaan programmeren, doen ze dat veel individueel achter hun scherm. Toch raad ik ze altijd aan om met elkaar samen te werken.”

Vo – Bij het Grotius College in Heerlen werkt de MakerKlas meestal met projectonderwijs, vertelt wiskunde- en informaticadocent Crützen, waarbij de leerlingen bijvoorbeeld in groepjes een game maken. Zulke opdrachten duren verschillende weken en Crützen loopt rond om vragen te beantwoorden, aanwijzingen te geven en theorie uit te leggen. De rollen die de kinderen in het groepje spelen – leider, ontwerper, programmeur, debugger – moeten ze zelf bepalen.

Tip: Laat anderen meekijken

“Als er iets is wat we met deze lessen aan de kinderen bij kunnen brengen, dan is het dat je dingen mag uitproberen en niet te veel moet twijfelen,” zegt Legters. “Laat zien wat je aan het doen bent, en laat anderen met je werk meekijken.”

Hoe leg je verbinding met andere vakken?

Po - In de ct-lessen van Legters (Oponoa) wordt regelmatig verbinding gezocht met vakken als taal en rekenen: “Een voorbeeld daarvan is een project over geometrische figuren. Zulke opdrachten kun je oplossen met programmeren, maar vertellen ook iets over goniometrie. Kinderen begrijpen dat.”

En wat de verbinding met ‘taal’ betreft: “In sommige lessen leer ik de kinderen structuren te herkennen en goed begrijpend te lezen. Dat begint al bij de opdrachten. Om die goed te kunnen uitvoeren, moet je de context begrijpen. Goed lezen is daarvoor een vereiste.”

Vo – Op het Jan van Brabant College in Helmond wordt verbinding gelegd met het vreemde-talenonderwijs, met name Frans. Zo wordt de programmeertaal Scratch gebruikt om de leerlingen quizjes te laten maken over Frankrijk. Docente en initiatiefnemer Ihsane Beyd

(Frans) deed inspiratie op bij een onderwijsbeurs in Engeland, en zocht samenwerking met *Codecult*, een Amsterdamse organisatie die workshops en cursussen programmeren aanbiedt aan scholen en particulieren. In de tweetalige vwo-3-klas van Beyd maakten de leerlingen zelf quizzen over de *Franse cultuur en politiek*, die door de andere leerlingen weer konden worden gespeeld. “Je ziet dat leerlingen erg enthousiast zijn,” zegt Beyd. “En ze leren nog Frans ook.”[30]

Hoe meet je de voortgang en hoe beoordeel je de resultaten?

Po - Sandra Legters (Oponoa) hanteert verschillende methodes om de voortgang van de leerlingen te meten, afhankelijk van de school waar ze lesgeeft. “Op een van de scholen werken ze met een dossier. Ze houden bij hoe hun projecten verlopen en rapporteren daarover.” Elders werkt ze met de mogelijkheden van Code.org. “Als de kinderen een account hebben dat door de leerkracht is aangemaakt, kun je als leerkracht in het dashboard de vorderingen zien.”

Maar nog belangrijker vindt ze dat er een constante dialoog is met de leerlingen over wat ze hebben gedaan en wat ze hebben geleerd. “Ik geef geen cijfers, maar houd wel in de gaten hoe ze groeien, dat kan ik als leerkracht goed beoordelen. Wel krijgen ze een certificaat van *Code Studio* als ze bepaalde onderdelen hebben afgerond.”

Vo onderbouw – Bij de brugklassen van het Jan van Brabant College in Helmond wordt niet met individuele cijfers gewerkt, vertelt docent Hans Vasse. “Binnen de MOOC [online cursusomgeving – red.] kun je *badges* krijgen, een beetje vergelijkbaar met judobanden. En als er toch gevraagd wordt om een cijfer, dan geven we die niet individueel maar per groep.”

> *“Toetsing is een belangrijk onderdeel van het leerproces.”*

Vo bovenbouw – Op het Grotius College in Heerlen wordt gebruik gemaakt van een (digitaal) *scrumbord*[3]. Daarmee kunnen de leerlingen overzicht houden op de structuur van hun project, en kunnen ze hun eigen voortgang bijhouden. “Het mooie is dat je in zo’n scrumbord ook goed kunt zien wat de voortgang is van de groepen,” zegt Ralph Crützen, (wiskunde en informaticadocent). “Aan het eind van de lessen rolt er een grafiek uit, die laat zien hoe ze gewerkt hebben.” Evalueren doen de leerlingen zelf, aan de hand van discussies. “Wel geef ik ze cijfers, dat moet ook, maar dat doe ik per groep. Maar ik houd natuurlijk ook, als docent, een oogje in het zeil wie wat doet.”

3 Een scrumbord of sprintbord in een hulpmiddel voor teams, zoals (software-)ontwikkelingsteams, maar ook bruikbaar voor schoolklassen, waarmee je de structuur en de voortgang van een project kunt visualiseren. Bijvoorbeeld: een whiteboard waarop je kolommen tekent, en met Post-its de status van de activiteiten (sprints) aangeeft. Er zijn ook digitale varianten.

Onderwijskundig perspectief – Volgens Alfons ten Brummelhuis (onderzoeker en adviseur bij Kennisnet en NRO) is het heel belangrijk om de toetsing goed op te pakken: "Toetsing is een belangrijk onderdeel van het leerproces. Door terugkoppeling kun je leerlingen feedback geven, of hints. Die terugkoppeling moet direct plaatsvinden, zodat leerlingen naar hogere niveaus kunnen komen. Een van de kenmerken van effectieve feedback is dat het vrij snel na de handeling plaats moet vinden. Vandaar dat je ook hierbij goed gebruik moet maken van de middelen die ict aanreikt. Daar is een breed spectrum aan mogelijkheden voor, in deze vorm van adaptief leermateriaal."

Hoe realiseer je kennisuitwisseling?

Een belangrijke methode om de kwaliteit van het onderwijs (en de onderwijsgevenden) te verhogen, is het uitwisselen van kennis. Dat kan op verschillende manieren.

Via landelijke organisaties – zoals *Computing at School* (CAS) in Groot Brittannië, of Kennisnet in Nederland. Het onderhavige rapport, met al zijn interviews en praktijkvoorbeelden, is daarvan een voorbeeld.

Via sociale media – zoals Facebook en Twitter. Op die manier blijft Ralph Crützen (Grotius College, Heerlen) op de hoogte van datgene wat zijn informaticacollega's bij andere scholen doen. "Zo wisselen we ideeën uit."

Via collega's – waarbij echte interesse wel een vereiste is natuurlijk. Sandra Legters (Oponoa, Gelderland) ziet dat een deel van haar collega's nieuwsgierig is naar haar programmeerlessen. "Die zien soms wat we doen, en wat de kinderen maken, en staan er versteld van wat je kunt maken met leerlingen van groep 5." Voor andere collega's is het een lastig onderwerp. "Bij een deel van hen bestaat koudwatervrees; die blijven kijken vanaf een afstandje," constateert Legters.

Via workshops – Ralph Crützen (Grotius College, Heerlen) geeft af en toe workshops aan collega-docenten, bijvoorbeeld over de Arduino (een goedkope ontwikkelcomputer). Toch valt de interesse van zijn collega's wel een beetje tegen, geeft hij toe. "Er komen er nooit veel, terwijl het erg interessante ontwikkelingen zijn."

Via meetings – Allard Strijker (leerplanontwikkelaar, SLO) : "Scholen krijgen nu door dat computational thinking en programmeren een serieus iets is, wat erbij is gekomen. Ik zou ze daarom aanraden om bijeenkomsten te organiseren, en aan het docentencorps, de ouders en de bestuurders uit te leggen wat computational thinking is en wat je daarmee kunt bereiken. Daarbij zou je ook voorbeelden kunnen laten zien, zoals de nieuwe 'Leerlijn programmeren in het basisonderwijs'."

Via andere scholen - Allard Strijker (leerplanontwikkelaar, SLO) vindt dat scholen ruimte moeten zoeken in hun roosters, en te rade moeten gaan bij scholen die al wat verder zijn. "Dit is de tijd om je te verdiepen en om samen te praten."

Hoe creëer je draagvlak?

Het creëren van draagvlak is eigenlijk nog het minste probleem. Het enthousiasme van staatssecretaris Dekkers, de inspanningen van de industrie-lobby, de overvloed aan rapporten, blogs en artikelen, en de peptalk van Neelie Kroes lijken hun doel niet gemist te hebben. Mediawijsheid is uit, programmeren is in.

Alle betrokkenen die wij spraken, ervoeren veel draagvlak voor computational thinking (vaak *geframed* als 'programmeren'):

- "Schoolbestuur en ouders waren meteen enthousiast," zegt Ralph Crützen (docent Grotius College, Heerlen). "Het bestuur maakte meteen tijd en ruimte vrij voor onze plannen. Ook de meeste collega's vonden het een goed idee."
- "Draagvlak krijgen voor het plan en de totstandkoming van het IT-lab was niet moeilijk," zegt Herman Rigter (ict-directeur Alberdingk Thijm Scholen, Hilversum). Omdat zowel de ouders als het schoolbestuur computational thinking belangrijk vinden. "Ouders moeten vaak zelf hun kinderen naar het IT-lab brengen, en dat doen ze graag, hoor ik. Het is natuurlijk ook bijzonder als je kind is geselecteerd om hieraan mee te doen."
- Hoe concreter hoe beter. Mark Vrolijk (Onderwijsgroep Fier, NW-Friesland) constateerde dat ouders niet zozeer gericht zijn op de hele schoolorganisatie, maar op de eigen school van het kind. "Daar moet je je communicatie op richten."

Maar: pas wel op voor de valkuil dat je 'computational thinking' niet versmalt tot 'programmeren', vindt Allard Strijker (SLO). "Daar gaat het uiteindelijk niet om bij dit soort onderwijs. Het gaat erom dat je een probleem leert oplossen met behulp van een computer. Computational thinking is veel breder. Dat gaat ook over de vraag hoe je

voor je werkstuk in Google kunt zoeken en hoe zo'n zoekmachine eigenlijk werkt en aan zijn resultaten komt. Het gaat ook over het slim gebruiken van Excel, of hoe je effectief kunt zoeken en vervangen in Word. Ons advies daarbij is om steeds de bredere context van computational thinking bij je lessen te betrekken. Hoe kun je een probleem analyseren, opbreken in hapklare brokjes, en oplossen met behulp van een computer?"

"Hoe kun je een probleem analyseren, opbreken in hapklare brokjes, en oplossen met behulp van een computer?"

Is sponsoring verantwoord?

In hoeverre kunnen scholen een samenwerking aangaan met commerciële partijen, zoals gebeurde in Noord-Oost Groningen, waar Google op 20 basisscholen programmeerlessen aanbood?

Doornbos en Faber (pabo Hanzehogeschool Groningen) hebben geen bezwaren tegen samenwerking met partijen als Google. Doornbos: "Er was geen sprake van sponsoring, want er was geen geld mee gemoeid. De mensen van Google kwamen alleen in het begin van de les praten over het Google datacenter en wat het betekent om te kunnen programmeren. De lessen hebben we helemaal zelf gemaakt, zonder bemoeienis van hun kant."

Meestal is de invloed van sponsors op het onderwijs nihil (of in ieder geval gering). In de interviews die we afnamen, en de research die we deden, kwamen we geen beïnvloeding tegen. Overigens werden het fysieke *IT-lab* (2 lokalen, Alberdingk Thijm, 't Gooi) en het eveneens fysieke *Community Learning Center* (aparte ruimte, onderwijsgroep Fier, NW-Friesland) niet gesponsord maar uit het eigen middelen bekostigd.

Mark Vrolijk, algemeen directeur van onderwijsgroep Fier uit Noord-West Friesland, is in principe niet tegen samenwerking met partners, zolang ze maar passen bij de algehele strategie. "Dan maakt het verder niet zoveel uit of dat organisaties als SLO zijn of bedrijven, zolang wij maar kunnen bepalen wat voor onderwijs we willen geven." (Fier wordt niet gesponsord door bedrijven.)

Het Jan van Brabant College (Helmond) werkt als BrainPort-school samen met partners uit het lokale bedrijfsleven, maar dat gaat vooral over de vraag welke kennis en vaardigheden belangrijk zijn als kinderen later op de arbeidsmarkt terechtkomen. Hans Vasse (docent design en kunstgeschiedenis): "Dat is belangrijke input voor ons, maar uiteindelijk beslissen wij zelf wat belangrijk is en hoe wij het onderwijs inrichten. Daar hebben bedrijven geen zeggenschap over."

8 Samenvatting

8 Samenvatting

Stand van zaken – Twee derde van de Nederlands basisscholen en ruim de helft van de middelbare scholen doet nog niets aan computational thinking of programmeren maar de aandacht ervoor groeit. Er komen steeds meer (onderwijs-)experimenten die de weg wijzen naar wat er allemaal mogelijk is. Sommige scholen geven een of zelfs meerdere uren per week programmeerles, andere organiseren projecten van enkele weken, of hebben naschoolse codeclubs opgericht waar kinderen vrijwillig aan kunnen deelnemen.

Visie – Scholen moeten vooraf bepalen waaróm ze computational thinking of programmeren willen onderwijzen. Doe je het om de arbeidsmarktkansen van je leerlingen te vergroten of om een bepaald soort denken aan te leren of te verscherpen? In het laatste geval zul je een veel breder type onderwijs moeten geven en op een veel jongere leeftijd moeten beginnen.

Mogelijke aanleidingen – ct-onderwijs kan gebruikt worden om een school aantrekkelijker te maken. Er kan ook sprake zijn van 'toeval', zoals een leraar of een bestuurslid die het een interessant onderwerp vindt en zich inspant om dit vak van de grond te krijgen. Bij basisscholen kan de overweging een rol spelen dat ze over een paar jaar (vermoedelijk in 2020) verplicht aandacht moeten besteden aan wetenschap en technologie.

Draagvlak – Meestal is het vrij gemakkelijk om draagvlak te krijgen. Schoolbesturen, directies, ouders en leerlingen zijn vaak snel enthousiast. Moeilijker kan het zijn om voldoende leraren mee te krijgen.

Apart vak of niet? – Aan de ene kant zijn er deskundigen als Han van der Maas (Uva) en Maria-Ilona Koski (NewTechkids) die vinden van wel. Omdat de maatschappij zich steeds sneller ontwikkelt in de richting van een digitale informatiesamenleving, en omdat het vervolgonderwijs en de arbeidsmarkt erom vragen. Aan de andere kant zijn er leraren en vooral pabo's, die ct-onderwijs en programmeren liever integreren met andere vakken, om de leeropbrengst te vergroten.

Continuïteit – Hoe ga je verder met ct-onderwijs als de codeweken voorbij zijn, de projecten zijn afgerond en de (project)subsidie ophoudt? Inbedding van dit type onderwijs in het reguliere programma is dus van groot belang, evenals het bij- of nascholen van leraren. En bedenk: als de leraren zelf niet enthousiast zijn voor dit soort onderwijs, zal het nooit goed van de grond komen.

Uitvoerders – Op basisscholen worden de lessen meestal gegeven door leraren die zijn vrijgeroosterd of worden ingehuurd. Op middelbare scholen zijn het veelal wiskunde- en natuurkundedocenten die de lessen geven, vooral in de bovenbouw.

Locatie – Een aparte ruimte is lang niet altijd nodig. Sommige scholen experimenteren weliswaar met IT-labs, Makerlabs, Community Learning Centers of andere ruimtes die speciaal zijn toegerust, maar bij andere scholen worden de lessen gewoon in de klas gegeven. Dat kan heel goed als er bijvoorbeeld al tablets en een digibord beschikbaar zijn. Bovendien kunnen veel lessen (vooral in het basisonderwijs) zonder computer gegeven worden; de zogenoemde *unplugged* lessen.

Lesinhoud – Er is al heel veel lesmateriaal beschikbaar, onder andere via websites als Code.org, Codeuur.nl en Codekinderen.nl. Daarnaast zijn er inmiddels volledige leerlijnen, zoals de nieuwe 'Leerlijn programmeren in het basisonderwijs' (PO-Raad, SLO, Kennisnet, 2016). Ook educatieve uitgeverijen als DaVinci ontwikkelen leerlijnen voor computational thinking. Tenslotte ontwikkelen veel leraren ook eigen materiaal.

Didactiek – Er is een breed spectrum van methodes, variërend van de zandbakmethode (experimenteren en uitproberen) tot een gestructureerde aanpak waarin stap voor stap kennis, begrippen en principes worden overgebracht. Het ideaal ligt waarschijnlijk in het midden ('guided discovery learning').

Transfer – Het is nogal lastig om de opgedane kennis over te brengen naar andere vakken. Uit onderzoek blijkt dat die overdracht vaak erg tegenvalt. Je moet de transfer in ieder geval expliciet maken. Maar zelfs dan is de opbrengst vaak gering.

Beoordeling – Inmiddels zijn er diverse assessment-tools ontwikkeld, zoals *Eduscrum* en *Computing Progression Pathways*, maar ook elektronische portfolio's als OBS (Open Broadcaster Sofware) waarin de leerlingen zelf hun resultaten bijhouden. Discussies voeren en vragen stellen zijn bij dit type onderwijs van het grootste belang, want pas door gerichte vragen te stellen en te praten over projecten kan een leraar inschatten of een leerling de ideeën en processen achter een opgave of project begrijpt.

Leeropbrengst – Er is geen wetenschappelijk bewijs dat computational thinking en programmeren de creativiteit en het probleemoplossend vermogen verbetert. Dat bewijs zal er ook niet snel komen. We zullen het moeten doen met *best practices*.

Literatuur

Geraadpleegde bronnen

- **Bloemink, Sanne** (2015): *'Google Klas - 21st century skills in het onderwijs: revolutie of hype?'* In: *De Groene*, 10 juni 2015.
- **Brennan, K, Balch, C., Chung, M.** (2015): *Creative Computing,* Harvard School of Education.
- **Fisser, P.H.G.** (2016): *Computational Thinking.* SLO, mei 2016.
- **Grgurina, N.** (2013): 'Computational Thinking in Dutch Secondary Education'. In: *Informatics in schools : local proceedings of the 6th International Conference ISSEP 2013*; selected papers ; Oldenburg, Germany, February 26–March 2, 2013.
- **Katz, J.** (2015): *'Jong geleerd, oud gedaan: later wordt zij uitvinder'.* In: *Forum*, 08-10-2015.
- **PO-Raad, SLO, Kennisnet** (2016): *Leerlijn programmeren in het basisonderwijs.*
- **Pijpers, R.** (2015): *Wat wij kunnen leren van computing-onderwijs in de Britse praktijk,* Kennisnet, 10 oktober 2015.
- **Platform Onderwijs2032** (2016): *Ons onderwijs2032. Eindadvies,* januari 2016.
- **Wing, J.** (2011): *Research Notebook: Computational Thinking - What and Why?,* Carnegie Mellon, Pittsburgh, 2011.

Colofon

Kennisnet

Het rapport ‘Computational thinking in het Nederlandse onderwijs’ is een uitgave van Kennisnet.

Datum
Oktober 2016

Concept en coördinatie
Wietse van Bruggen (Kennisnet)

Adviezen
Remco Pijpers (Kennisnet)

Interviews en tekst
Louis Stiller

Eindredactie
Henk Boeke

Vormgeving
Gloed*communicatie*

Fotografie
Reyer Boxem, Kornak Kersten, Etienne Oldeman, Dirk-Jan Visser, Hollandse Hoogte

Over Kennisnet

Kennisnet is de publieke organisatie voor onderwijs en ict. We zorgen voor een landelijke ict-basisinfrastructuur, adviseren de sectorraden en delen onze kennis met het primair onderwijs (po), het voortgezet onderwijs (vo) en het middelbaar beroepsonderwijs (mbo).

Kennisnet
Paletsingel 32
2718 NT Zoetermeer

T 0800 321 22 33
E support@kennisnet.nl
I kennisnet.nl

Postbus 778
2700 AT Zoetermeer