Computer Science for Multilingual Students

Report from the AERA Educational Research Conference

September 22, 2021

Authors*

Sharin Rawhiya Jacob; Alison Bailey; Marina Umaschi Bers; Quinn Burke; Jill Denner; Diana Franklin; Leiny Garcia; Susan Gomez-Zwiep; Chris Hoadley; Megan Hopkins; Keith Howard; Nicol Howard; Maya Israel; Yasmin B. Kafai; Okhee Lee; Jonathan Montoya; Miranda Parker; Rose Pozos; Chris Proctor; Debra Richardson; Dana Saito-Stehberger; Bryan Twarek; Annette Vee; Sara Vogel; Hayley Weddle; Mark Warschauer

Acknowledgment: This report is the product of the Computer Science for Multilingual Students conference, held April 1-2, 2021. This research conference was supported by a grant from the Education Research Conferences Program of the American Educational Research Association (AERA). We are grateful to AERA for its support.

*Authors are listed alphabetically by last name with the exception of the first author Sharin Rawhiya Jacob, who led the development and writing of the report, and the last author Mark Warschauer, the Principal Investigator of the Computer Science for Multilingual Students Education Research Conference.

Introduction

As advances in computational technologies are changing the fabric of society, computational thinking (CT) is increasingly seen as a fundamental skill that all students should learn. While the bulk of research on CT has focused on its integration into science, technology, engineering, and math (STEM) content, there is a growing body of scholarship that focuses on the relationship between CT, language, and literacy (Bers et al., 2019; Jacob & Warschauer, 2018; Kafai et al., 2019; Vogel et al., 2020), and the roles that language and literacy play in developing students' CT skills (Bers et al., 2019; Jacob & Warschauer, 2018; Proctor & Blikstein, 2019). In 2021, approximately 50 researchers and practitioners in the fields of computer science, language and literacy, and STEM education attended the AERA conference on Computational Thinking for Multilingual Students to discuss two major associated topics: (1) computing and literacy and (2) computing and second language learning. The purpose of this conference was to develop a shared vision of the conceptual relationship of computing to language and literacy development and of evidence-based perspectives on how to support multilingual students in learning computer science. This report reflects that shared vision, focusing on three interrelated aspects: (1) the relationship between CT and language, literacy, and equity; (2) CT and the teaching of language arts and writing; and (3) methods for teaching CT to multilingual learners.

Relationship of Computational Thinking to Language, Literacy, and Equity

Computational thinking (CT) described by Jeanette Wing (2006) as "involving solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing" (p. 33) has become the common theme for computer science's move into K-12 education. Considerable debate has focused on whether this definition captures a general problem-solving skill or a skill more specific to solving computer problems (Barr et al., 2011), ignoring that such a discussion frames CT in mostly cognitive terms while leaving out other framings. Currently, three framings of CT in K-12 education are under discussion, emphasizing either (1) cognitive: skill and competency building, (2) situated: creative expression and participation, or (3) critical: social justice and reflection (for a more extended discussion, see Kafai et al., 2019). Each of these framings highlights different aspects of what learning (and teaching) CT can mean for K-12 students. A cognitive framing sees CT as a form of complex problem solving that is primarily performed by individuals (Grover & Pea, 2013), and student learning is seen as gaining competency in computational concepts such as loops, recursion, conditionals, data structures, and practices such as iteration and abstraction. Such views of CT are influenced by cognitive research theories of learning that dominated efforts to introduce programming in the 1980s (Spohrer & Soloway, 1989). A different framing draws from constructionist learning theory (Papert, 1980) and emphasizes interest-driven and peer-supported activities and thus sees CT as a vehicle for personal and creative expression and participation (Kafai & Burke, 2014). Learning key computational concepts and practices are thus situated within acts of designing complex applications of personal relevance that are shared on social networks. Finally, a third, critical framing focuses on social justice and reflection, a direction that engages students' CT with existing socio-political issues. Efforts following this direction place CT as a platform through which to address existing real-world challenges by creating original multimedia artifacts (Vakil, 2018). These three framings of CT have mostly been illustrated in STEM contexts but their connection to learning theories highlights that they easily can be applied to STEAM (i.e., STEM+Art) contexts as well as language and literacy.

Theoretical frameworks relating CT to literacy have taken asset-based approaches that aim to leverage students' existing literacy skills to develop CT and vice versa. Jacob and Warschauer (2018) proposed a three-dimensional framework for understanding the relationship between CT and literacy that (1) situates CT as a literacy in itself (i.e., computational thinking *as* literacy); (2) examines how students' literacy skills can be leveraged to develop CT (i.e., computational thinking *through* literacy); and (3) explores ways in which students' CT skills can be mobilized to develop their literacy skills (i.e., literacy *through* computational thinking (see Figure 1).

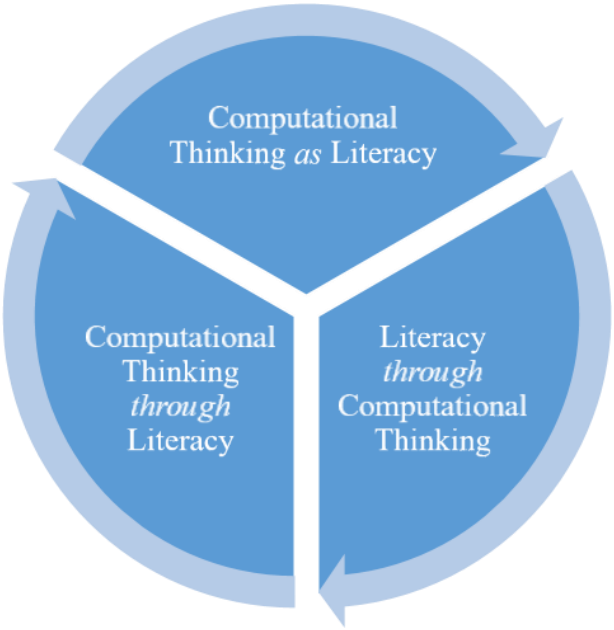

Figure 1. A three-dimensional framework for understanding computational thinking and literacy

The model first situates CT as a literacy in itself (i.e., computational thinking *as* literacy). Jacob and Warschauer (2018) define literacy as "a set of practices situated in a sociocultural context that utilize external technological media to enable expression" (p. 285). Technological media, from clay, papyrus, and wax to the printing press, computers, and the internet, have undergone paradigmatic shifts that have transformed social conceptions of literary practices (Warschauer, 1999). Programming represents an additional technological medium for communication that fosters the evolution of human expression through creative and innovative approaches to problem solving.

Second, given these multiple definitions of literacy, it is possible to leverage students' existing literacy skills as a mechanism for learning CT (i.e., computational thinking *through* literacy). Integrating CT into English language arts (ELA) content has multiple affordances for computer science learning. Narrative structures capture the semiotic process related to computing (de Souza et al., 2011). To this end, the several interlocking features of coding and literacy draw children's attention to symbol-meaning relationships. In so doing, they provide multimodal scaffolding for students to learn letters and words, and offer a highly engaging and supportive environment for children with emerging literacies to demonstrate their skills and abilities (Peppler & Warschauer, 2011).

Third, at the same time, students' CT skills can be leveraged to foster reading, writing, and language development (i.e., literacy *through* computational thinking). There is a substantial amount of work on the similarities between programming languages and traditional languages (Connolly, 2001; Pane & Myers, 2001; Vee, 2017). Jacob and Warschauer (2018) map the syntactic, semantic, and pragmatic similarities between the two. For example, teaching syntactic knowledge could involve leveraging students' knowledge of sequence to teach paragraph organization, chronological storytelling, and the writing of instructions. Students' semantic knowledge can be mobilized through meaning-based activities, such as using command cards to program robots. Finally, students can foster their knowledge of the executive function of programming commands by practicing pragmatic language functions, such as through pair programming and other collaborative activities in which they use the functional language of computer science as well as that of social interaction.

Coding as a Discourse. The relation between CT and literacy can be applied to students' participation in computing discourse (Vogel et al., 2020). While literacy has been traditionally viewed as involving the discrete skills necessary to code and decode written and spoken text, more recently, literacy has been defined as modes of interaction that characterize how individuals position themselves within specific communities of practice (Gee, 2015). K-12 computer science frameworks have outlined mechanisms for students to participate in computer science discourse in a manner that nurtures students' budding computer science identities. However, traditionally marginalized students, such as language learners, have been excluded from computer science communities of practice, thereby delegitimizing their experiences and contributions. Vogel et al. (2020) recommend translanguaging as a method for including multilingual students and other diverse learners in computing discourse by leveraging their entire repertoires for sense making. Methods that draw on students' rich and varied resources combat exclusionary practices that lead to systemic injustice and contribute to diversifying the field.

Computational Thinking and the Teaching of English Language Arts

While ELA and CT are often taught in different areas of the K-12 curriculum currently, their overlapping pedagogical aims suggest that they should be better integrated with each other in instructional design. Movement toward the socially just goal of more widespread distribution of CT will require embedding CT principles across the curriculum, similarly to how writing across the curriculum is implemented now. This realization that pedagogical aims of CT and other subjects, including ELA, have significant overlap is not new. Early computer educators such as Perlis (1962), and Kemeny and Kurtz (the co-developers of BASIC; Kemeny, 1983), and Papert (1980) recognized both the need and the benefits of synthesizing CT with other areas of learning. More recently, scholars interested in CT education have argued for a concept of "computational literacy" that draws on the rich pedagogical history of ELA and the social urgency and necessity of universal literacy (Kafai & Burke, 2014; Vee, 2013). Yet the ELA curriculum is still generally set apart from CT.

This curricular gap is especially present in ELA designed for multilingual students. Key to a socially just computational future is participation from diverse groups of people, including multilinguals. Moreover, CT already operates in a multilingual world, and multilinguals already live in a world infused with computation. Therefore, finding syncretic overlaps between the two

not only is reflective of current educational realities, but also may be beneficial for learning outcomes. In this section, we discuss how the instruction of CT and ELA together can provide students with the competencies that many are defining as essential in the 21st century, and how this benefits K-12 curriculum and society. Examples that model successful integration of CT and ELA in the classroom will be presented. Lastly, considerations in the development of a model for the integration of CT and ELA will be shared.

Shared Pedagogical Aims Between ELA and CT. The overlap in instructional aims of CT and ELA is one reason to support the integration of equal instruction of CT and ELA in a classroom. As lists of essential 21st century skills are being developed to guide educators to prepare the next generation to engage in the workplace and to contribute to society (National Governors Association Center for Best Practices & Council of Chief State School Officers, 2010; NRC, 2013; World Economic Forum, 2020), a core set of overlapping essential competencies are emerging that include problem solving, critical thinking, creativity, resilience, and social influence and communication. Not coincidentally, a large proportion of these core 21st century skills also coincide with key elements of the cognitive framing of CT, such as specific problem-solving processes of breaking down a problem into manageable parts and designing individual steps to solve a problem.

These concepts, practices, and perspectives are not limited to CT; they can also strengthen problem solving, decision-making, and critical thinking in other academic subjects and in everyday life. For this reason, Vee (2017) calls computational literacy a "platform literacy" upon which other skills can be built. Kafai et al. (2019) continue to describe two additional framings of CT that draw attention to contemporary educational perspectives and, many argue, are essential to emphasize with the next generation: situated computational thinking and critical computational thinking. Situated CT, which is grounded in constructionist and connected learning theories, focuses on the personal expression, motivation, sense of self and the community that are produced as students participate in developing and sharing their digital artifacts. Situated CT provides a place for humanistic inquiry, an opportunity for students to seek to satisfy their own curiosity. Critical CT emphasizes a social justice approach, in which students use the tools of CT to examine and reflect on forces that marginalize and restrict people.

Skills in the ELA domain, as articulated by the Common Core State Standards in the United States, can work together with the instruction of CT to the benefit of the 21st century workforce. Some of the ELA Common Core standards include having students be able to cite evidence in order to make a claim, to use writing as part of the research process, to work collaboratively, to see the perspectives of others, as well as to use and develop their linguistic resources (Kibler et al., 2015). These research skills could be augmented by CT concepts such as algorithmic thinking and sequence to help students plan their argument and choose salient sources for their research. The abilities to work collaboratively and to consider other perspectives align well with the situated and critical computational framings. Lastly, as students grow in linguistic sophistication across multiple languages, their process parallels computational literacy development: coding and decoding, becoming more fluent, and using coding to meet their own needs and express their own ideas (Bers, 2019).

Curricular Example. The “Coding as Another Language” (CAL) early childhood computer science pedagogical approach is a tangible example of how the instruction and learning of ELA

and CT can mutually support one another in the classroom (Bers, 2019). The development of this pedagogical approach was informed by research in early childhood literacy instruction, with particular consideration of the cognitive changes that take place as young learners begin to read and write. Another influence that has shaped this approach is the premise that coding is a literacy; just as a natural language in the form of text allows communication of ideas beyond immediate time and space, the artificial language of coding permits the same through algorithms. Bers (2019) expands: "Both activities, coding and reading and writing, involve a problem-solving dimension as well as the use and manipulation of a language, a symbolic representational system, to create a shareable, interpretable product" (p. 504). The coding and decoding of both natural and artificial languages enable users to generate unique products, not bound by time or space, that can be interpreted by others. The symbolic representational systems of text and code are tools that support the construction of unique ideas and expression.

The CAL approach carefully considers developmental learning trajectories that young learners experience when engaged with specific curriculum. Drawing on established stages of literacy development, the CAL approach guides students through six coding stages: emergent, coding and decoding, fluency, new knowledge, multiple perspective, and purposefulness. The CAL curriculum provides problem-solving challenges and a personally meaningful computational project for each of these six stages.

The CAL curriculum that embodies the pedagogical approach described above is designed for 4–7-year-olds and integrates the instruction of both ELA and CT to enhance one another. It has been aligned with both Common Core ELA/Literacy Framework (for kindergarten) as well as the Computer Science Framework. There are four units in the CAL curriculum, each of which explores storybooks, such as *Where the Wild Things Are* by Maurice Sendak, and includes learning activities based on them. Each of the four units is comprised of 12 lessons that focus on a "powerful idea of computational thinking and literacy." For example, the unit on sequencing teaches students about hardware/software, algorithms, and representation, as well as summarizing/retelling, the sequence of a story, and descriptive language in writing. The unit on debugging teaches students about debugging in CT as well as editing and awareness of the audience in writing. The CAL approach serves as a model of CT and ELA integration that may be particularly applicable to multilingual learners.

Future Research. There are many other examples of successful integration between CT and ELA, including Unfold Studio, an interactive story platform using the programming language Ink (Proctor & Blikstein, 2019); Storygame, which uses Twine to teach high schoolers about coding and writing (Mike Sell, Indiana University of Pennsylvania); and textbooks that support ELA and CT integration such as Nick Montfort's *Exploratory Programming for the Arts and Humanities*. Research on the relative efficacy of these approaches for CT and ELA development, student engagement, teacher participation, and practicality of implementation will be helpful as educators continue to develop innovative ways to integrate CT and ELA. Given that any ELA instruction in American schools happens in multilingual settings, specific attention to the skills that multilingual learners bring to ELA, as well as any unique challenges they may have, will be critical for ELA and CT integration research generally. Finally, any research agenda on CT and ELA should include meaningful engagement and collaboration with teachers involved in this work.

Teaching Computational Thinking to Multilingual Learners

In the years since development of the Common Core State Standards and the Next Generation Science Standards, there has been increased emphasis on the communicative aspects of the disciplines and implications for education of students acquiring English in US schools (Bailey & Carroll, 2015; Council of Chief State School Officers, 2012). This has placed the learning of language and content in tandem and at the front and center of K-12 multilingual learner instructional improvements (see, for example, the Understanding Language initiative, https://ell.stanford.edu/). The integration of language instruction with content instruction is touted as the most effective approach to instruction for students learning English in school for a host of reasons, not least because it prevents students from falling behind in their content knowledge acquisition as they develop English language proficiency. Rather than viewing language exclusively as an object of instruction, language, it is argued, is "best learned as a medium of content rather than as the focus of instruction" (Potowski, 2004, p. 95).

While content-based ESL has been around as an approach to English language instruction for several decades (for reviews, see Snow, 1998, and Lyster, 2017), its primary objective is most often to support English language development through instruction in the content areas such as mathematics, science, or social studies, rather than to specifically promote learning in these disciplines. Consequently, the rigor of the content with which language learning is paired may be compromised through techniques to simplify and "shelter" the content. Alternatively, the focus on content may come at the cost of effective strategies for language development (Tedick et al., 2011). Furthermore, students who are new speakers of English may be taught separately from their English-speaking peers in specialized classes, not the general education classroom. Recent pedagogical strategies have taken a more inclusive approach to content instruction with multilingual learners and have focused educators' efforts on making rigorous, on-grade academic content accessible through multiple semiotic means. These include translanguaging practices that leverage all linguistic resources available to students including the use of the L1 and that do not exclusively rely on students' knowledge of English as they acquire new knowledge of the disciplines (e.g., National Academies of Sciences, Engineering, and Medicine [NASEM], 2018).

We use the term multilingual to refer to students who speak more than one language and may be learning English. Multilingual students are extremely diverse across several dimensions including but not limited to their cultural backgrounds, languages spoken, immigration status, and time residing in the US (Menken, 2013). Despite their differences, these students bring a shared positionality within educational institutions. For example, all multilingual students bring a wealth of cultural, linguistic, semiotic, and embodied resources to the classroom that can be leveraged for equitable participation (Jacob et al., 2020, 2021; Vogel et al., 2020).

Unfortunately, there are a number of factors that help to explain the chronic lack of representation for linguistically diverse students. First, there is little to no data about multilingual students in computer science education (Schildkamp & Kuiper, 2010). Without basing decisions on data, educators and stakeholders may make biased assumptions about these students or ignore them entirely. To exacerbate this issue, schools with 12% or greater numbers of students designated as English learners offer half as many computer science courses as other schools (Martin et al., 2015). Issues of access are compounded by pervasive stereotyping in the field which is perpetuated through media representation. Students do not report seeing computer

scientists in the media who look like them, which sends messages about who does and who does not do computer science (Code.org, CSTA, & ECEP Alliance, 2020). Finally, the traditions and values of culturally and linguistically diverse students and their families are not reflected in much of the computer science curricula implemented to date (Margolis et al., 2012). Purposefully tailored instruction can address these issues by leveraging the wealth of knowledge and resources students bring to the classroom and fostering agency in multilingual students to shape disciplinary practices, provide meaningful critique of the field, and become agents of change in their communities (Jacob et al., 2020, 2021; Vogel et al., 2020).

Engaging Multilingual Students in Computer Science. Too often educators assume that multilingual learners are unable to tackle seemingly advanced academic content, such as CS, until they first catch up in English language and literacy development (see discussion in Lee & Stephens, 2020). We reject this deficit view and instead highlight the affordances and opportunities for these students and for our nation in fully including multilingual learners of all ages to participate in CS learning.

We find, for example, that CS expands options for communicating ideas in ways anticipated a quarter century ago by the New London Group (Cazden et al., 1996). The multimodal nature of CS (e.g., creating artifacts with code) affords multilingual students a range of communicative strategies and ways to engage in social interactions with their peers. For one, collaborating on CS projects encourages multilingual students to interact and jointly produce complex ideas through making and problem solving, ideas which they then communicate to others. These projects also have built-in ways for multilingual students to give and receive feedback, such as identifying bugs in their peers' code, gaining new ideas about the use of code from peers, or offering suggestions to a peer who is struggling. Thus, participation in CS provides opportunities for cognitive engagement and interaction in rich disciplinary language. This affordance is valuable both for young learners who are developing literacy (K-5) as well as for secondary students who may be transferring literacy skills from one language to another.

Given the ability of CS to offer forms of communication that are not dependent upon English proficiency, CS provides an opportunity to connect the formal school curriculum to multilingual students' interests, communities, and funds of knowledge. In bringing their informal computing and computational thinking expertise to the classroom, multilingual students challenge the misconception that they are not able to do CS because they have not sufficiently mastered English.

Viewing multilingual students as valued and knowledgeable participants in CS brings new perspectives and ways of knowing to the field. Expanding notions of who participates in CS and for what reasons transforms visions of CS education (Vogel et al., 2017) to consider positive community impacts beyond economic benefits, such as equipping communities to more fully engage with civic life and social reform efforts.

Synopsis of Projects on Computer Science for Multilingual Students. The following four initiatives provide valuable examples of diverse approaches toward engaging multilingual students in computer science.

PiLa-CS (https://www.pila-cs.org/) is a partnership with NYC DOE teachers and district staff that aimed to implement and study pedagogical and professional development approaches to CS education that build on multilingual students' diverse language practices and involve them in meaningful conversations at the intersection of computing, school disciplines, and their communities. Teachers participating in PiLa-CS use translanguaging to draw upon multilingual students' varied and rich resources (linguistic, cultural, semiotic, embodied) to engage them in computing. The project frames coding as a discourse and argues that marginalized students such as multilingual learners are systematically excluded from this discourse. By building on multilingual students' diverse language practices, teachers increase students' participation in computing and provide them with rich opportunities to actively shape the CS discipline.

Elementary Computing for All (https://www.elementarycomputingforall.org/) brings together researchers and practitioners in the University of California, Irvine, the University of Chicago, Santa Ana Unified School District, and Chicago Public Schools to iteratively develop and evaluate a curriculum targeted at the needs of multilingual students in elementary schools. The curriculum adopts a structured inquiry approach and language-based scaffolding to involve students in creative coding, and helps them develop STEM identities. The curriculum is rooted in five effective practices for engaging multilingual students in STEM including: (1) engaging students in disciplinary practices, (2) encouraging rich classroom discourse, (3) building on students' multiple meaning-making resources, (4) encouraging students to use multiple registers and modalities, and (5) providing explicit focus on how language functions in the discipline (NASEM, 2018).

Computing for the Social Good (https://www.etr.org/about-us/our-projects/computing-for-social-good/) is a partnership that includes the non-profit organization, ETR; Santa Cruz City Schools; Santa Cruz Education Foundation; University of California, Santa Cruz; and Stanford University. Funded by the National Science Foundation, the partnership aims to use computer science as a lever to address social and academic inequities between White and Latinx students and their families in the Santa Cruz City Schools. To address these challenges, the team developed a three-part strategy to (1) integrate CS into core K-8 curriculum, (2) engage families in computing, and (3) create a local CS advisory committee to ensure the long-term sustainability of the effort. Guiding the work is a justice-oriented framework for computational thinking integration that brings together an understanding of language as dynamic communicative processes that change and grow over time and sociocultural pedagogy that sees language and disciplinary content learning as inextricably linked, and positions students as agentic learners who bring a wealth of resources to learning.

CSforEL (https://www.csteachers.org/305564/Page/Show?ClassCode=Page&Slug=%2Fcsforel) is a federally funded initiative by the U.S. Department of Education designed to engage English learners in AP Computer Science Principles (AP CSP) throughout Arizona, New Mexico, San Diego County, and Orange County, CA. The project includes teacher professional development, district outreach, and program evaluation. The goals of the project are to (1) increase enrollment in AP CSP courses for English learners; (2) increase scores on the AP CSP exam for marginalized students, including English learners; (3) increase English learners' grades in AP CSP; and (4) increase English Language Arts proficiency in students who are designated as English learners.

Table 1 identifies the stated or implied targets of assessment or measurement in the four multilingual learner focused computer science projects. Additionally, current and intended evidence of change is also highlighted. The four projects have some commonalities in key goals such as the authenticity of student discourse and identity relating to coding tasks and the engagement of families or the wider school community. There was some variation in what is currently being assessed or is feasible to assess, ranging from measuring student participation to measuring the efficacy of curricular changes and to identifying resources that can promote access to computer science. Evidence relies primarily on teacher reports of student engagement and teacher reports of benefitting from curricular changes (e.g., integration of CS in core curriculum), as well as analyses of school policies and on occasion reports from students. Although acknowledged by at least one of the projects as challenging to document, suggestions for future targets of assessment are growth in students' CS knowledge and development of a language repertoire for the CS discipline. Additionally, observations of changes to teacher behaviors and student engagement can complement the reliance on teacher report.

Though all four projects seek to draw on the resources of multilingual learners to promote more equitable CS education, they vary in their approaches, with some projects emphasizing provision of language scaffolding so that students can master disciplinary language and others focusing more on challenging existing linguistic norms. While the authors of this report have diverse perspectives on these issues, we are in agreement that there are multiple ways to support multilingual learners, and that those working toward this goal would do well to expand goals to include not only computer science knowledge and academic language proficiency, but also creativity, participation in authentic computing communities, and use of code and computing for expression and learning. We also agree that a wide range of stakeholders, from school districts to communities, deserve a voice in shaping the future of computer science education.

Conclusion

There is a unique opportunity for multilingual students to learn computer science knowledge, skills, and attitudes through peer-to-peer interaction that facilitates sense making through the use of rich discourse. Integrating computational thinking and language can be particularly challenging as English language development instruction tends to crowd out STEM instruction, rather than complement it, especially in elementary school grades (Dorph et al., 2011). Computational thinking and literacy instruction hold promise for bringing computer science to all students. The affordances that media rich programming environments such as Scratch have for storytelling naturally fit within the literary genre. Practically, the integration into ELA mitigates time constraints related to limited time for STEM instruction in elementary grades. Though there is much work that needs to be done, we hope that the concepts and approaches summarized in this report can help scholars and practitioners better serve multilingual students in developing their CS knowledge, skills, and attitudes. Doing so will empower these students to become active creators of CS content who can provide meaningful critique of new technologies and act as change agents in their communities.

Table 1. Targets of assessment and documented evidence of change across four multilingual learner focused CS projects

Project	**Institutions/ Presenters**	**Approach**	**Key Goals**	**Current/Possible Targets of Assessment/ Measurement**	**Current/Future Evidence of Change**
Participating in Literacies and Computer Science (*PiLa-CS*)	NYU/CUNY/ NYCDOE et al., Chris Hoadley, Jasmine Ma, Sara Vogel (NYU) [Laura Ascenzi-Moreno, CUNY Brooklyn College]	RPP	"Build on and sustain the language practices, identities, and communities of learners." "Bi/multilingual learners use their language and code to make meaning, express, critique, and contribute to meaningful conversations."	The ***learning environments that teachers and researchers co-design*** should: 1) Reflect the diverse communities of bi/multilingual learners 2) Show how they promote the participation of bi/multilingual learners	● Teachers report (or observed) an increase in noticing, welcoming, and expanding students' language repertoires. ● Documented incorporation of code into conversations of students and their communities (home, online, disciplinary, neighborhood, etc.). ● Teachers report (or observed) providing multiple and flexible entry points for students to use language flexibly to communicate about, with, and through code.
Elementary Computing for ALL (ECforALL)/ IMPACT Curriculum	UCI, Dana Saito-Stehberger [Mark Warschauer, UCI]	Structured Inquiry Approaches	"STEM knowledge and language developed through interaction and regular participation in profession-like activities. Discourse and argumentation to support the creation of knowledge."	Using adapted NASEM (2018) instructional practices: Efficacy of IMPACT Curriculum components (Teacher's Guide, Student Workbook, Lesson Slide Decks, & Online Resources)	• Teachers report (or observed) engaging students in disciplinary practices. • Engaging students in productive discourse and interactions with others (e.g., Pair Programming culminating projects). • Utilizing and encouraging students to use multiple registers and multiple modalities (e.g., simulations, End of Unit Reflections). • Leveraging multiple meaning-making resources (e.g., Responsive Storybooks). • Providing some explicit focus on how language functions in the discipline.

Computing for the Social Good	Stanford/UCSC/SCCS et al., Rose K. Pozos (Stanford)	RPP Piloted Integrated CT Lessons in Core Content	"Prepare students and their families to be global citizens, critical thinkers and communicators through the use of equity-oriented Computer Science."	1) Integrate computational thinking into core curriculum 2) Build family engagement with computing 3) Create a Local CS Advisory Committee	• Teachers report higher motivation, preparation, and support for their integration of equity-oriented CS into core curriculum. • Increase in enrollment in the K-8 pathway from students and families from across the district. • Documented family engagement and digital competence by family uptake in computer literacy and leadership activities. • Success of Local CS Advisory Committee measured by increase in financial support for CS from school & community and creation of long-term vision and sustainability initiatives.
CSforEL	UCSD/CSTA et al., Megan Hopkins (UCSD) (BT Twarek, CSTA)	Classroom-Focused Intervention (PLCs - PD for AP CSP teachers/ lesson study); School-Focused Intervention (Collaborative equity audit)	Address opportunity gaps in CS for EL students through classroom (improve instruction) and whole school intervention (improve access).	Classroom-Focused Intervention and School-Focused Intervention: Success of equity audit to identify resources to promote EL students' access to CS.	• Documented transformation of systems and daily practices to expand learning opportunities. Success determined by analysis of school policies; course enrollment data; and teacher-, staff-, and student-reported experiences and outcomes.

References

Bailey, A. L., & Carroll, P. E. (2015). Assessment of English language learners in the era of new academic content standards. *Review of Research in Education, 39*(1), 253-294.

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, *38*(6), 20-23.

Bers, M. U. (2019). Coding as another language: A pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, *6*(4), 499-528.

Cazden, C., Cope, B., Fairclough, N., Gee, J., Kalantzis, M., Kress, G., ... & Nakata, M. (1996). A pedagogy of multiliteracies: Designing social futures. *Harvard Educational Review*, *66*(1), 60-92.

Code.org, CSTA, & ECEP Alliance. (2020). 2020 State of Computer Science Education: Illuminating Disparities. https://advocacy.code.org/stateofcs

Connolly, J. H. (2001, July). Context in the study of human languages and computer programming languages: A comparison. In *International and Interdisciplinary Conference on Modeling and Using Context* (pp. 116-128). Springer.

Council of Chief State School Officers. (2012). *Framework for English language proficiency development standards corresponding to the Common Core State Standards and the Next Generation Science Standards.* CCSSO.

de Souza, C., Garcia, A., Slaviero, C., Pinto, H., & Repenning, A. (2011). Semiotic traces of computational thinking acquisition. End-User Development, 155-170. https://doi.org/10.1007/978-3-642-21530-8_13

Dorph, R., Shields, P., Tiffany-Morales, J., Hartry, A., & McCaffrey, T. (2011). High Hopes--Few Opportunities: The Status of Elementary Science Education in California. Strengthening Science Education in California. *Center for the future of teaching and learning at WestEd*.

Flores, N., & Rosa, J. (2015). Undoing appropriateness: Raciolinguistic ideologies and language diversity in education. *Harvard Educational Review, 85*(2), 149-171.

Flores, N. (2020). From academic language to language architecture: Challenging raciolinguistic ideologies in research and practice. *Theory into Practice, 59*(1), 22-31.

Gee, J. P. (2015). Discourse, small d, big D. *The International Encyclopedia of Language and Social Interaction*, 1-5.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

Jacob, S., Garcia, L., & Warschauer, M. (2020). Leveraging multilingual identities in computer science education. In *Technology and the psychology of second language learners and users* (pp. 309-331). Palgrave Macmillan, Cham.

Jacob, S. R., Vogel, S., Pozos, R., Ordonez Franco, P., & Ryoo, J. (2021). Leveraging multilingual students' resources for equitable computer science instruction. IEEE Annual International Conference on Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT'21).

Jacob, S. R., & Warschauer, M. (2018). Computational thinking and literacy. *Journal of Computer Science Integration*, *1*(1).

Kafai, Y. B., & Burke, Q. (2014). *Connected code: Why children need to learn programming*. MIT Press.

Kafai, Y. B., Proctor, C., & Lui, D. A. (2019). Framing computational thinking for computational literacies in K-12 education. In *Weizenbaum Conference* (p. 6). DEU.

Kemeny, J. G. (1983). The case for computer literacy. *Daedalus*, *112*(2), 211–230.

Kibler, A. K., Walqui, A., & Bunch, G. C. (2015). Transformational opportunities: Language and literacy instruction for English language learners in the Common Core era in the United States. *TESOL Journal*, *6*(1), 9-35.

Lee, O., & Stephens, A. (2020). English learners in STEM subjects: Contemporary views on STEM subjects and language with English learners. *Educational Researcher*, *49*(6), 426–432. https://doi.org/10.3102/0013189X20923708

Lyster, R. (2017). *Content-based language teaching.* Routledge.

Margolis, J., Ryoo, J. J., Sandoval, C. D., Lee, C., Goode, J., & Chapman, G. (2012). Beyond access: Broadening participation in high school computer science. *ACM Inroads*, *3*(4), 72-78.

Martin, A., McAlear, F., & Scott, A. (2015). Path not found: Disparities in access to computer science courses in California high schools. Retrieved from https://eric.ed.gov/?id=ED561181

Menken, K. (2013). Emergent bilingual students in secondary school: Along the academic language and literacy continuum. *Language Teaching*, *46*(4), 438.

National Academies of Sciences, Engineering, and Medicine. (2018). *English learners in STEM subjects: Transforming classrooms, schools, and lives.* National Academies Press.

National Governors Association Center for Best Practices & Council of Chief State School Officers. (2010). *Common Core State Standards*. Authors.

National Research Council. (2013). *Next Generation Science Standards: For states, by states*. The National Academies Press. https://doi.org/10.17226/18290.

Pane, J. F., & Myers, B. A. (2001). Studying the language and structure in non-programmers' solutions to programming problems. *International Journal of Human-Computer Studies*, *54*(2), 237-264.

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

Peppler, K. A., & Warschauer, M. (2011). Uncovering literacies, disrupting stereotypes: Examining the (dis) abilities of a child learning to computer program and read. *International Journal of Learning and Media*, *3*(3), 15-41.

Perlis, A. (1962). The computer and the university. In M. Greenberger (Ed.), *Computers and the world of the future*. MIT Press.

Potowski, K. (2004). Student Spanish use and investment in a dual immersion classroom: Implications for second language acquisition and heritage language maintenance. *The Modern Language Journal*, *88*(1), 75-101.

Proctor, C., & Blikstein, P. (2019). Unfold Studio: Supporting critical literacies of text and code. *Information and Learning Sciences*.

Schildkamp, K., & Kuiper, W. (2010). Data-informed curriculum reform: Which data, what purposes, and promoting and hindering factors. *Teaching and Teacher Education*, *26*(3), 482-496.

Snow, M. A. (1998). Trends and issues in content-based instruction. *Annual Review of Applied Linguistics*, *18*, 243-267.

Spohrer, J. C., & Soloway, E. (1989, January). Simulating student programmers. In *IJCAI* (Vol. 89, pp. 543-549).

Tedick, D. J., Christian, D., & Fortune, T. W. (Eds.). (2011). *Immersion education: Practices, policies, possibilities.* Multilingual Matters.

Vakil, S. (2018). Equity in computer science education. *Harvard Educational Review*, *88*(1), 26-53.

Vee, A. (2013). Understanding computer programming as a literacy. *Literacy in Composition Studies*, *1*(2), 42–64.

Vee, A. (2017). *Coding literacy: How computer programming is changing writing*. MIT Press.

Vogel, S., Hoadley, C., Castillo, A. R., & Ascenzi-Moreno, L. (2020). Languages, literacies and literate programming: Can we use the latest theories on how bilingual people learn to help us teach computational literacies? *Computer Science Education, 30*(4), 420-443.

Vogel, S., Santo, R., & Ching, D. (2017, March). Visions of computer science education: Unpacking arguments for and projected impacts of CS4All initiatives. In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (pp. 609-614).

Warschauer, M. (1999). *Electronic literacies: Language, culture and power in online education*. Routledge.

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

World Economic Forum. (2020). *The future of jobs report 2020.*