Computation and the Reggio Emilia Approach

A provocation

Gary S. Stager, Ph.D.
Constructing Modern Knowledge

Introduction: Beyond Digital Crafts

Imagine a pair of shoes. Click the heels together, and a taxi arrives.

In most classrooms, an idea like this would end up on a poster board — a colorful sketch displayed at an "Invention Convention," celebrated for creativity, then forgotten. The technology to actually *build* it would be considered out of reach for students of any age, let alone children.

That assumption is wrong. When a group of educators at one of our Constructing Modern Knowledge summer institutes proposed exactly this idea, a fellow participant offered a simple observation: "The API for Uber is freely available." With that tip, a microcontroller, some craft supplies, a programming language, sufficient time, and a supportive culture, "Shoe-ber" was operational within just a few days.

What made the difference? Not the cleverness of the idea — fanciful ideas are everywhere. What made the difference was computation.

This is the distinction we need to name clearly. In our pursuit of meaningful learning experiences for children, we have embraced makerspaces and digital tools with genuine enthusiasm — and yet something essential is still missing. Despite the incredible affordances of computational technology, we find ourselves too often producing what amounts to "digital macaroni necklaces or shoebox dioramas." There is nothing wrong with arts and crafts, but we should be honest about what we are and are *not* teaching, while aspiring toward a world congruent with both available computational opportunities and the true competence of learners.

Even in schools with rich art classes, much of what students "make" is what might be called "fridge ready" — meaning that the culture imposes pressure on teachers to rush students to create artifacts that can be sent home to seemingly delighted parents after a 45-minute session. Such facile approaches to visual art stand in stark contrast to

settings embracing the Reggio Emilia Approach, where students invest significant time, effort, generative design, debugging, skill development, artistic fluency, and aesthetic perspective. This deep, sustained engagement produces visual expression that becomes both an artistic artifact and a visible record of their thinking and development — often of breathtaking beauty and sophistication rarely created by children of similar age in other settings. The same pressure for quick, "fridge ready" results plagues our use of computational tools, depriving children of the depth of engagement they deserve.

The time has come to distinguish between projects that are merely *digital* and those that are truly *computational* — and to understand why this distinction matters profoundly for the children in our care.

Two Worlds of Learning: Digital vs. Computational

Words matter in education, perhaps now more than ever. When we speak of technology in learning, we often conflate two fundamentally different approaches. Digital projects are those created with computers—videos, graphic images, presentations, word-processed documents, and photographs. Digital describes the form or space consumed by a product. These artifacts may be brilliant, beautiful, thoughtful, clever, or even profound. However, that leap requires an investment of talent, effort, and time rarely afforded children in school. Schoolwork tends to favor digital projects due to time constraints, teacher comfort, and their ease of sharing.

The chaotic nature of school combined with typically low expectations for children result in a celebration of far too many mediocre projects. The quality of such projects is often marred by apathy, being rushed, or iteration and revision are skipped. Quality work takes time.

Computational projects, on the other hand, require the act of computing and represent process. When a kindergartener wanted to create a robot ballerina, computation was essential. While the ballerina itself was composed of a paper napkin, pipe cleaners, magic markers, and LEGO bricks, giving it life required computation. Directing the ballerina's movements with touch sensors, changing the direction of its motors, adjusting its speed, and even playing a musical accompaniment required the active "dance" of abstract and concrete reasoning, including programming and debugging — computation.

This distinction is not academic. One could spend considerable time listing the educational objectives achieved by this little girl in a project wholly commensurate with her personality, playfulness, imagination, passion, perseverance, and ingenuity.

The formal curricular expectations for a student of her age might be "knowing left from right." Since the little girl taught those concepts formally to the computer via computation, she may now have a stronger understanding of that concept than her peers.

Computation – A Simple Definition

Stephen Wolfram defines computation as “organizing your thoughts clearly enough that you can explain them to a sufficiently smart computer.” (Kawasaki, 2019) Thus, computation becomes a way of making things – ideas, objects, computer programs. Wolfram also contends that we for any intellectual discipline, X, from archaeology or art history to zoology. There is now or soon will be a branch of that discipline called, “Computational X.” That new field represents both the intellectual and lucrative promise of the endeavor.

Computational Making (Hansen, 2024) expands the breadth, depth, and range of possible projects that may be undertaken, even by young children. Papert teaches us that “If you can make things with technology, then you can make a lot more interesting things and you can learn a lot more by making them.” (Martinez & Stager, 2018)

The Hundred and One Languages

This vision of children as powerful thinkers and creators connects deeply to the traditions we hold dear. In his famous poem "No Way. The Hundred is There," Loris Malaguzzi wrote of the child who "has a hundred languages / a hundred hands / a hundred thoughts / a hundred ways of thinking / of playing, of speaking." He warned us that "the school and the culture / separate the head from the body" and that "they steal ninety-nine" of these languages from children. The poem ends with the child's defiant assertion: "No way. The hundred is there."

For decades, educators committed to the Reggio Emilia Approach have worked to preserve and honor these hundred languages—to resist the forces that would diminish children's ways of knowing and expressing themselves. Today, we face a new challenge and a new opportunity: to recognize that computation represents not a theft of these languages, but a powerful addition to them. A language that, far from separating "the head from the body" or telling children “That work and play / reality and fantasy / science and imagination" do not belong together, actually reunites what schools have divided.

From Fantasy to Reality: The Power of Computational Thinking

Shoe-ber did not succeed because the idea was clever. It succeeded because computation closed the gap between imagination and reality. This is what computation does, reliably and repeatedly, in the hands of children given the appropriate time, tools, trust, and high expectations.

Consider how that gap appears in school. A child has an idea — vivid, specific, genuinely their own. The teacher, working within real constraints of time and curriculum, must find some way to honor it. The result is almost always a representation of the idea rather than a realization of it: a drawing, a model, a presentation, a brochure. These schoolified artifacts are not the thing itself, and children know the difference.

Computation changes this calculus. When a kindergartener wanted to create a robot ballerina, no paper and pencil or even digital substitute would do. The ballerina herself was built from a paper napkin, pipe cleaners, magic markers, motors, gears, sensors, and LEGO bricks. Giving the creation life required computation. Directing her movements with touch sensors, reversing her motors, adjusting her speed, and playing a musical accompaniment required the active "dance" of abstract and concrete reasoning: programming, testing, debugging, revising. The child was not representing a ballerina. She was building one.

The educational significance of this distinction is easy to underestimate. One could spend considerable time listing the formal learning objectives achieved in that single project — objectives wholly commensurate with this child's personality, playfulness, imagination, passion, perseverance, and ingenuity. The formal curricular expectation for a student her age might be simply "knowing left from right." Because she taught those concepts to a computer through computation, she may now understand them more deeply than any worksheet could have produced.

This is what Papert meant when he argued that computation expands not just what children can make, but what they can know. "If you can make things with technology," he observed, "then you can make a lot more interesting things and you can learn a lot more by making them."

Computation is not a tool for dressing up old projects in new clothes. It is rocket fuel expanding the breadth, depth, and range of project possibilities, and in doing so, raises the ceiling on how children are capable of surprising us.

Making Thinking Visible: The Reggio Connection

Our colleagues in Reggio Emilia do not subscribe to a strident reflexive rejection of technology in the lives of children. *Loris Malaguzzi and the teachers: Dialogues on collaboration and conflict among children,* offers evidence of the ways in which computers have been used for decades as a medium for creative expression, communication, and inquiry in Reggio Emilia. That recently translated book includes a transcript of educators discussing their documentation of five-year-old children programming a physical Logo turtle in 1990. The sophistication and creativity of the children is only matched by the curiosity, thoughtfulness, and wisdom of the educators.

This connection between computation and the Reggio approach runs deeper than mere documentation of technology use. Both approaches share fundamental beliefs about learning, documentation, and the capacity of children. Consider Malaguzzi's insistence that children possess "a hundred ways of listening / of marveling of loving / a hundred joys / for singing and understanding." These are not abstract poeticisms—they are pedagogical principles. And computation, properly understood, embodies these same principles. It offers children ways to marvel at pattern and possibility, to understand through creating, to express joy through making things that sing, move, and think.

Computers used wisely in classrooms can be computational ateliers. Open-ended, constructive, computational software environments in the spirit of Logo and its descendants are ateliers of the mind in which powerful ideas are tinkered with, hypotheses are tested, and knowledge is constructed much as one learns in a physical workshop.

Debugging as an Essential Life Skill

Debugging—the act of identifying, understanding, and rectifying errors—is an essential element of computer programming and producing a computational result. The spirit of debugging is also integral to the project work undertaken in Reggio environments. Most attempts to learn or create something are plagued by misguided efforts, poor planning, unanticipated events, temporary setbacks, faulty logic, haste, or inattention to detail. This makes the development of debugging skills imperative.

Seymour Papert viewed debugging as both a critique of schooling and more poetically as a way of navigating life. Setting aside fears of being wrong and asking if either a computer program or a life problem is fixable is key. Being stuck requires the

development of strategies for getting unstuck or charting a different path, perhaps even to a new destination. (Martinez & Stager, 2018)

When a child is trying to build the tallest block tower and it crumbles, they laugh and build back better. That is unless an adult is watching them. Then fear of judgement turns to tears. Instead of "embracing failure," both Papert and the educators of Reggio Emilia shift the focus of education away from assessment and towards the act of learners learning.

This rejection of punitive assessment aligns perfectly with Malaguzzi's critique of schools that "tell the child: to think without hands / to do without head / to listen and not to speak / to understand without joy." Debugging insists on thinking with hands, doing with head, speaking about problems, and understanding through the joy of making things work. It refuses the separation of "work and play / reality and fantasy / science and imagination" that conventional schooling imposes.

Documentation That Serves Learning

In Reggio environments, documentation isn't a form of assessment in search of deficiencies, but a way of making thinking visible in a manner beneficial to the entire community of practice—teachers, parents, and learners themselves. As Rinaldi stated, "We place the emphasis on documentation as an integral part of the procedures aimed at fostering learning and for modifying the learning-teaching relationship. Documentation is a tool for helping teachers and children reflect on prior experience; listen to each other's ideas, theories, insights, and understandings; and then make decisions together about future learning paths."

The beauty of computational projects is that the computer program or computational notebook used in the construction of a project is in itself a form of documentation making the thinking of a child or group of children visible to all to understand, appreciate, interpret, and build upon. Children working in environments like *Logo, Snap!, MakeCode, Scratch, TinkerCAD,* and *Wolfram Notebook Assistant* naturally produce documentation of their thinking. Their creations narrate the story of their thinking. Code reveals epistemological pluralism. Product and process are indistinguishable.

Transparency and Democracy

Computational projects benefit from transparency. Little is hidden or mistaken for magic. There are no tricks up your sleeve. This has serious implications for the advancement of science. Computation accelerates the ease and speed of collaboration, dissemination, replication, validation, or refutation of an experiment. This democratizes the scientific process and expands the universe of scientists.

Kids can not only assess the results of their peers but also interrogate their thinking process by reading their code. Such transparency allows for children to be inspired by the accomplishments of their peers and put their knowledge into use. Reading another person's code and modifying (remixing) it, is a popular and powerful form of self-scaffolding by learners.

Checking another person's work, verifying data, looking under the hood, attributing sources, and maintaining transparency are also essential to the preservation of democracy. We must prepare learners to think critically, anticipate multiple consequences, expect transparency, and develop what has been called a highly tuned BS detector. (Postman, 1969)

More than a quarter century apart, Stephen Wolfram and Seymour Papert discuss this transparency in the context of learning. Papert is more specific in referring to students engaged in Logo programming.

> “Computational thinking provides a framework that makes things more transparent and easier to understand. When you formulate something computationally, everyone can try it out and explicitly see how it works. There’s nothing hidden that the student somehow has to infer from some comment the teacher made.” (Wolfram, 2016)

> "Children want to get together with others engaged in similar activities because they have a lot to talk about. And what they have to say to one another is not limited to talking about their products: LOGO is designed to make it easy to tell about the process of making them.” (Papert, 1980)

The Mathematical Revolution We're Missing

The 1989 National Council of Teachers of Mathematics Standards made a stunning claim that "50% of all mathematics has been invented since World War II." Common sense suggests that percentage has only increased in the intervening years. This overlooked statement is no mere factoid, but rather a challenge, and urgent plea to

reinvent the life of the classroom. Educators should possess an insatiable desire to find new things for more kids to know in new ways.

New mathematical disciplines, including cellular automata, fractal geometry, number theory, information theory, combinatorics, cryptography, data science, as well as practical applications of computation like robotics, computer programming, physical computing, and artificial intelligence are now accessible to children. Their newness does not relegate them to post-graduate study. One need not endure 12-16 years of an asparagus diet before enjoying dessert.

This is the powerful idea behind Seymour Papert's metaphorical Mathland, a "place" where mathematical fluency, power, meaning, and relevance comes as naturally as learning to speak French by growing up in France. Computation allows children to be mathematicians, rather than be taught a rigid sequence of tricks called "Math."

Seymour Papert introduced a distinction between instructionism and constructionism. Instructionists believe that learning is the result of being taught and that educational progress is based on a treatment model – a new textbook, piece of software, teaching technique applied to a student. Constructionists view the world differently. They center the learner and believe that knowledge is constructed by the learner, alone or in a group, as a consequence of experience. Asserting one's personal stance vis-à-vis learning profoundly shapes the educational environment and resulting praxis.

Regardless of a school's philosophy, Math has often been the shame of progressive education and the place where instructionism rules. Ask any progressive school leader and they are likely to confess that their vision is harder to realize when it comes to teaching Math. Papert argued that no matter how constructionist a school or classroom is, there likely is a time of day, or part of the week, when coercion is reintroduced into the system, teachers become insecure, or students feel deficient. That time is commonly known as "Math."

Papert told me that such coercion isn't just deleterious for mathematics learning, but that is ultimately corrosive for the entire system of schooling. In other words, it undermines every other learner-centric principle.

In his writing and personal conversations, Papert shared how learner-centered, project-based approaches to teaching Language Arts, Social Studies, the Arts, and even Science have been well established for perhaps a century or more, but that Math instruction rejects constructionism most of all. However, Papert was optimistic that the computer and computational environments make hands-on/minds-on mathematical learning possible for the first time for a great number of students.

Beyond Cute: Seeking Megachange

Educational rhetoric has long been rife with obfuscation, euphemism, and verbal inflation. Very little educational "change" truly earns the appellation of revolutionary. One of the most overused superlatives of our era is "revolutionary." Declaring a person, product, or action as revolutionary is a very big claim—one worthy of scrutiny, or at least skepticism.

Too often, schools head fake towards computation while merely using digital technology to create a patina of innovation. The result is a curriculum left intact and zero evidence of progress. Districts declare STEAM Day and have families color blackline illustrations of scientists or instruct kids to make propagandistic posters. Such efforts waste time, deprive kids of richer experiences, reinforce the status quo, and insult our intelligence.

Revolutions are seldom cute. Any educational revolution should add surplus value—leave the classroom better than we found it. The democratization of computation offers genuine revolutionary potential, bringing us closer to a vision where mathematics is useful, relevant, beautiful, playful, and comprehensible.

A New Standard for Learning

The project should be a teacher's smallest unit of concern. All teaching should be focused on creating rich experiences for children, recognizing the ideal that "knowledge is a consequence of experience." Rather than concerning ourselves with grading, ranking, or sorting students, our focus should be on creating productive contexts for learning in which rich provocations and materials exist for all children to engage in personally meaningful project work.

Teaching is most rewarding when students surprise us. I suggest we aspire to a new standard: that 10% of students create something "interesting" and 5% do something original that never occurred to the teacher or would astound curriculum writers. This benchmark is much more attainable when computation is involved.

Educators should always be on the lookout for opportunities to take projects to the next level. Sometimes, all that is required is asking simple questions like, "Why does that fall down?" or "What would you like it to do next?" Increasingly, computational power matches the imagination of children. This in turn supercharges their projects and when simple things become easy to do, complexity becomes possible.

Conclusion: a hundred hundred hundred more

Malaguzzi's poem warns us that schools tell children "to discover the world already there" rather than to invent and dream their own worlds. Computation is fundamentally about invention and creation—about giving children the power to bring into being the hundred worlds they imagine. When a kindergartener programs a robot ballerina or when students invent shoes that summon a taxi, they are exercising precisely those languages that school too often steals away.

Malaguzzi's poem also reminds us that children possess "a hundred ways of thinking / of playing, of speaking." Computation honors this plurality. It does not impose a single correct answer or a single path to understanding. Instead, it reveals what has been called epistemological pluralism—the recognition that there are many valid ways of knowing and problem-solving. When we read a child's code, we see their unique way of thinking made visible and concrete. The playful evocative Logo turtle remains one of the most durable objects-to-think-with ever conceived. It is both a product of computation and a vehicle for computational making.

We need a language of computation not because technology demands it, but because children deserve it. When we provide students with computational tools and experiences, we honor their intellectual capacity, support their natural curiosity, and prepare them for a future where computational thinking is not optional but essential.

The convergence of computational power with progressive educational principles offers unprecedented opportunities for learning. By embracing computation as more than a technical skill—by recognizing it as a fundamental literacy for making sense of and acting upon our world—we can create learning environments where children's voices are heard, their thinking is made visible, and their potential is truly unlimited.

This is not about replacing what we value in education but expanding it. It is not about abandoning the hundred languages of children but adding to them—honoring Malaguzzi's vision by ensuring that "the hundred is there" and that we add one more: the language that transforms "a hundred worlds to invent" and "a hundred worlds to dream" into a hundred worlds made real.

Malaguzzi knew that "the child says: No way. The hundred is there." Our task as educators is to stand with children in that defiance, to resist the forces that would steal their languages, and to offer them every tool—including the powerful language of computation—to discover, invent, and dream their hundred worlds into being.

References

Edwards, C., Gandini, L., & Nimmo, J. (2014). *Loris Malaguzzi and the teachers: Dialogues on collaboration and conflict among children, Reggio Emilia 1990*. Lulu.com.

Hansen, S. (2024). What's worth making? with Prof. Hal Abelson [Audio podcast episode]. In *Chalk Radio*. https://chalk-radio.simplecast.com/episodes/whats-worth-making-with-prof-hal-abelson/transcript

Kawasaki, G. (2019). Stephen Wolfram [Audio podcast episode]. In *Remarkable People*. https://guykawasaki.com/stephen-wolfram-remarkable-people/

Malaguzzi, L. (n.d.). *The hundred languages of children* [Webpage]. Reggio Children. https://www.reggiochildren.it/en/reggio-emilia-approach/100-linguaggi-en/

Martinez, S., & Stager, G. (2018, August 1). Around the world with the 8 big ideas of the constructionist learning lab. *Invent to Learn*. http://inventtolearn.com/around-the-world-with-the-8-big-ideas-of-the-constructionist-learning-lab/

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

Papert, S. (1990). The Perestroika of Epistemological Politics. In A. McDougall & C. Dowling (Eds.), *Computers in Education: Proceedings of the IFIP TC 3 Fifth World Conference on Computers in Education, WCCE 90, Sydney, Australia, July 9-13, 1990* (pp. 3). North Holland.

Postman, N. (1969, November 28). *Bullshit and the art of crap-detection* [Paper presentation]. Annual Convention of the National Council of Teachers of English, Washington, DC. Retrieved April 7, 2026, from https://www.fairshake.net/wp-content/uploads/2021/08/Bullshit-and-the-Art-of-Crap-Detection-Postman-1969.pdf

Stager, G. (2025a). *The case for computation*. The Language of Computation – Constructing Modern Knowledge in Reggio Emilia. https://reggio.constructingmodernknowledge.com/the-case-for-computation

Stager, G. (2025b). *Computation makes learning visible*. The Language of Computation – Constructing Modern Knowledge in Reggio Emilia. https://reggio.constructingmodernknowledge.com/computation-makes-learning-visible

Stager, G. (2025c). *CUTE*. The Language of Computation – Constructing Modern Knowledge in Reggio Emilia. https://reggio.constructingmodernknowledge.com/cute

Stager, G. (2025d). *Digital computational*. The Language of Computation – Constructing Modern Knowledge in Reggio Emilia. https://reggio.constructingmodernknowledge.com/digital-computational

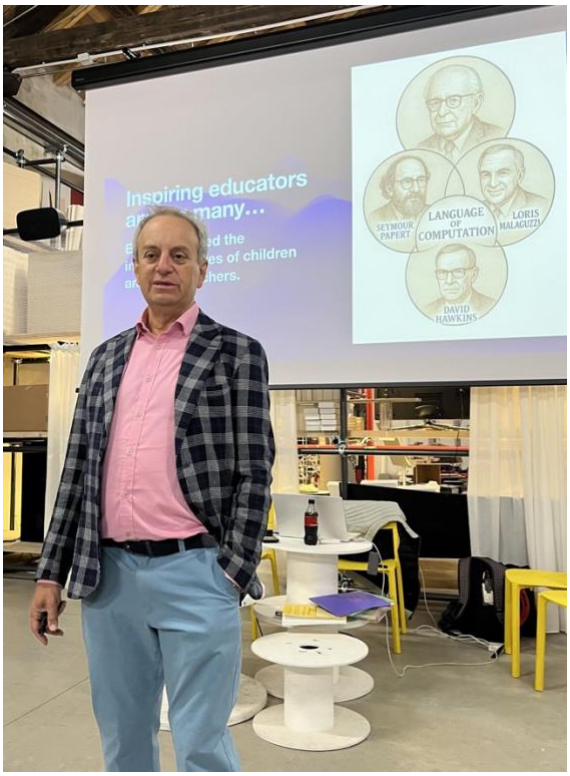
Stager, G. S. (2006). *An investigation of constructionism in the Maine Youth Center* [Doctoral dissertation, University of Melbourne]. Melbourne.

Turkle, S., & Papert, S. (1991). Epistemological Pluralism and the Revaluation of the Concrete. In I. Harel & S. Papert (Eds.), *Constructionism* (pp. 161-191). Ablex Publishing Corporation.

Wolfram, S. (2016). How to teach computational thinking. *Stephen Wolfram Blog*. https://writings.stephenwolfram.com/2016/09/how-to-teach-computational-thinking/

About the Author

Dr. Gary Stager is a veteran teacher educator with more than four decades of experience helping schools makes sense of an increasingly complex and technologically sophisticated world. He played a substantial role in realizing 1:1 computing in schools, project-based online learning, and the book he co-authored, Invent To Learn: Making Tinkering, and Engineering in Classroom, has been called "the bible of the maker movement in schools." That book has now been translated into ten languages.

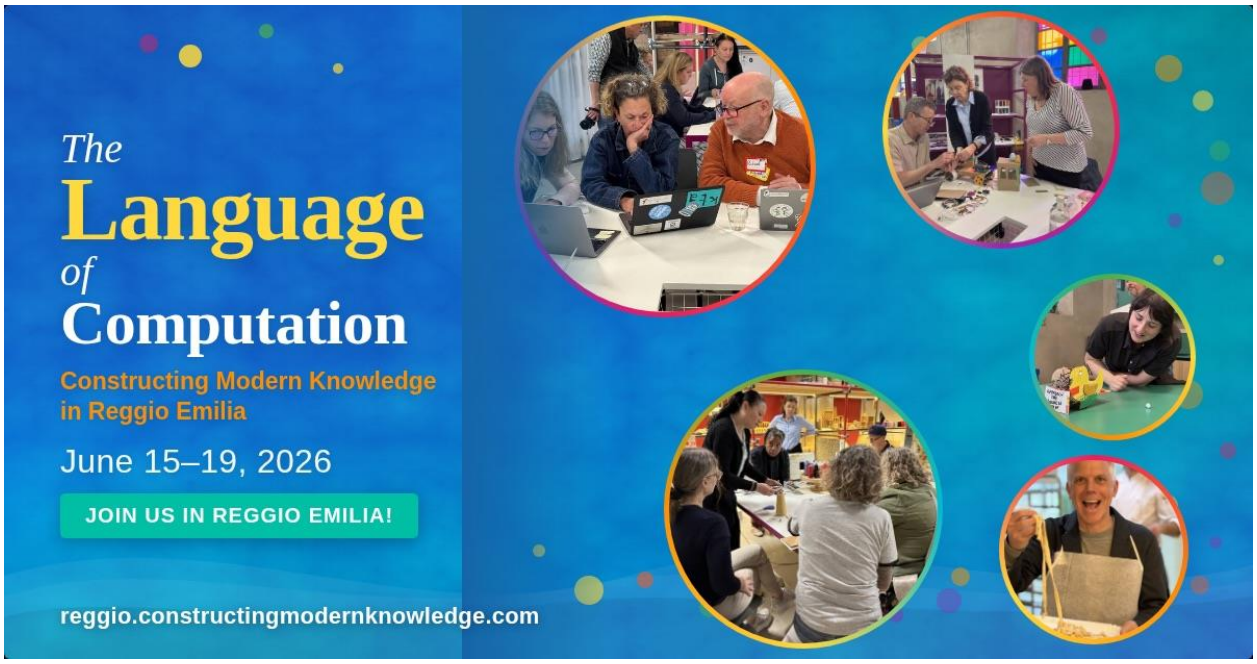


A preschool and primary school teacher by training, Dr. Stager has taught students from preschool through doctoral level.

Gary earned his PhD in Science and Mathematics Education at the University of Melbourne and worked closely with the father of educational computing, Dr. Seymour Papert for decades. Dr. Stager is a popular keynote speaker at many of the world's leading conferences and his most recent project includes a collaboration with the remarkable educators of Reggio Emilia, Italy to create The Language of Computation – Constructing Modern Knowledge in Reggio Emilia.

Learn more about Gary, explore his publications, media appearances, and learn how to collaborate at professorgarystager.com.

You're Invited!



The Language of Computation

Join us in Reggio Emilia, home of remarkable pedagogical practices, for a hands-on exploration of powerful ideas for educators inspired to make the world a better place for children.

"The Language of Computation" is rooted in the belief that the future is computational, children are natural mathematicians, and that computation enriches the learning possibilities for all.

When

Monday - Friday 15 - 19 June, 2026
Reggio Emilia, Italy

Who should participate?

Pk-12 educators & teacher educators interested in learning to combine the powerful ideas of the Reggio Emilia Approach®, project-based learning, and computation in meaningful ways for the future. Learn more at reggio.constructingmodernknoweldge.com

Hansen, S. (2024). Chalk Radio [Audio podcast episode]. In *What's worth making? with Prof. Hal Abelson*. https://chalk-radio.simplecast.com/episodes/whats-worth-making-with-prof-hal-abelson/transcript

Kawasaki, G. (2019). Remarkable People [Audio podcast episode]. In *Stephen Wolfram*. https://guykawasaki.com/stephen-wolfram-remarkable-people/

Martinez, S., & Stager, G. (2018). *Around the world with the 8 big ideas of the constructionist learning lab*. Retrieved august 1, 2018 from http://inventtolearn.com/around-the-world-with-the-8-big-ideas-of-the-constructionist-learning-lab/

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

Postman, N. (1969). Bullshit and the art of crap-detection. annual convention of the National Council of Teachers of English, Washington, DC Retrieved September,

Wolfram, S. (2016). How to teach computational thinking. *Stephen Wolfram Blog*.