Microsoft

Teaching Computer Science:

Creating Opportunities for Youth in Latin America

Prepared by Ignacio Jara and Pedro Hepp for Microsoft Latin America

Table of contents

1. Introduction

Digital technologies are changing the world we live in, including education. Education systems are seeking to adapt to this new context, incorporating computers and the Internet in teaching and administration and are preparing new generations to use these tools in different aspects of life. This involves an adaptation process that is permanently adjusting its course amid constant changes in the technology and social settings ripe for change.

Until now, *digital literacy* of youth has been primarily focused on being good users of technology applications. In recent years, however, a growing interest has emerged to broaden technology formation, to educate youth to understand the function and founding principles of technologies, so as to convert them into creative agents of the digital world and not just consumers. Accordingly, many developed countries are revising their school curricula to incorporate *computer science* concepts and to develop *computational thinking* in students, and various companies and nonprofit organizations have launched initiatives to promote device coding for youth, to become agents of informal *computer science* education. These skills are aimed at boosting the cognitive development of youth, facilitating employability, and broadening interest for technology careers, thereby strengthening economic growth of countries.

This movement to favor computer teaching is not new, but it has evolved. Since the 1970s, there has existed the idea that teaching children to code can be an experience that is both fruitful for their cognitive development, and encourages more youth to engage in needed professions in the technology sector. In those years, many secondary schools began to impart coding courses using accessible languages of the time, such as BASIC. For example, England, a country that has been very historically active in those subjects, launched in 1981 its first national policy called *Microputers for Schools*, focused on teaching this language (Selwyn, 2002). In the early 1980s, Seymour Papert, a South African academic and follower of Piaget, created along with his group at MIT the LOGO language, intended for children to learn mathematics in a playful manner using stringent logical reasoning to solve problems and code a computer (Papert, 1993). Papert's ideas and tools were adopted in many places during the following years and continue to have a profound influence, even today. In Latin America, for example, Costa Rica began a national policy in 1987 which still exists to teach coding in primary schools to develop logical thinking and problem-solving (Muñoz et al, 2014). Since that time, countries and schools have maintained courses in computer science, principally at the secondary level. Despite successive changes in their focus, they have maintained some level of interest in coding, at least as an elective course for the most engaged students.

In the 1990s, the focus of attention shifted toward the use of computers and the Internet as a cross-sectional aid for teaching and learning in the school curriculum. It no longer involved simply learning *about* computers but *with* computers. As a result, most existing computer science courses focused on using computer applications such as word processors, spreadsheets, databases, etc., to solve

everyday problems and ensure the basic technology skills required to use ICTs in other school subjects, leaving aside the more technical emphasis of coding. However, over the years, as children became more and more immersed in digital technology-rich environments in their own homes, computer science courses became increasingly boring and irrelevant (Cobo, 2014). At the same time, technology sector and academia circles observed with concern that the growing competitiveness of the digital economy was not accompanied by a greater interest among youth for the professions sustaining this industry, and that schools were not bringing the new generations closer to these disciplines.

In this context, the 2000s saw the idea emerge with great force that all students should be exposed to the theoretical and practical foundations of digital technologies during their educational experience, and this time in a more profound manner. In 2006, Jeannette Wing, director of the Computer Science Department of Carnegie Mellon University in Pittsburg, United States, wrote an influential article proposing that computational thinking should be considered a key skill for the 21st Century that all students should learn, opening a rich debate on the scopes and implications of this vision (Wing, 2006). In recent years, many countries have begun to revise their curricula, putting computer science at the forefront of their technology courses. For example, England's Department for Education made a radical change along these lines, and since 2014 this discipline is taught from the first years of primary. In early 2016, the U.S. President launched a campaign called *Computer Science for All* to strengthen teaching of this course in American schools. And in 2014, European Schoolnet published a study showing that the majority of European countries were already including or planning to incorporate computer programming into their curricula (EUN, 2014).

Additionally, private initiatives driven by companies in the sector, foundations, and nonprofits have multiplied, inviting youth to learn to code as a means to complement school and expand their possibilities to learn, create, and find work. Furthermore, there is a growing interest in the *makers* movement and educational robots, offering environments conducive to the development of coding projects, whose products emerge from screens to give life to inanimate items that move, collect data on their surroundings, or control electro-mechanical devices[1]. To the surprise of many, these initiatives have achieved broad acceptance from the public. One example is the hundreds of thousands of enthusiasts who gather at the *makers* fairs organized for several years in various countries[2]; the eight million low-cost programmable *Raspberry Pi* cards that have been sold in the past four years[3]; and more than 200 million people from around the world who have used resources and participated in activities promoted by initiatives such as *Hour of Code*[4] and *YouthSpark*[5] which promote computer science teaching among children and youth. The latter initiative, which is developed by Microsoft, has a Latin American version called *YoPuedoProgramar*[6] (I Can Code), involves more than 100 partners and has reached more than 3 million youth in the region. Notably, this company, along with other large companies in the sector, have been active promoters of this movement, contributing

[1]These movements use different types of devices, electronic cards, and sensors that are connected to computers to be pro-grammed and create new artifacts and solve problems. Cards such as Raspberri Pi and Arduino are famous, among others.
[2]Visit http://makerfaire.com/
[3]Raspberry-Pi is in reality a small computer that can be programmed to execute a wide variety of tasks. Visit www.raspberrypi.org
[4]www.code.org
[5]www.microsoft.com/about/philanthropies/youthspark
[6]www.yopuedoprogramar.com/

ideas, digital resources, and seed funds to expand their reach and impact.

In the case of Microsoft, whose interest in these topics is essential to its function, its work in these fields has two aspects: on the one hand, its philanthropic efforts with civil society, particularly that focused on expanding opportunities for youth to obtain a job, become entrepreneurs, or reinsert in education, through technology, and on the other hand, its work with the education sector in various countries, especially the ministries and departments of education and ICT policies for schools. It was precisely in the junction of these two outlooks that Microsoft Latin America identified the need to collect background on the challenges of this scope faced in the region to express them in a document offering some direction for future progress. For this, it organized a series of meetings with public and private actors working in this field with the aim of gathering concerns and perspectives that could provide insight for this document[7]

Latin America is paying close attention to the intense movement toward computer science education, while at the same time it is seeking to solve its most pressing social and education problems, such as inequality, severe deficits in education quality, and the lack of opportunities of a large part of youth who do not attend school. This is in addition to the massive gaps in access to technology and digital literacy that still exist (Sunkel & Trucco, 2012). Clearly, it involves a context that is different than that of the most developed countries at the forefront of these changes, which forces us to think how these innovations occur in our realities and how we can change them. It seems appropriate, then, to hold an informed discussion to promote good decision-making and answer the various questions that could emerge from its implementation. Accordingly, the experience of initiatives participating in the meetings organized by Microsoft comprise a good foundation for learning for the future (for a list of the experts who participated in these gatherings, see the Appendix).

This document aims to present the principal elements of the movement for computer science teaching, in both its public and private components, formal and non-formal education alike, in order to provide a basic description of this landscape and its primary challenges. This document is primarily directed at public policy decision-makers who are interested in understanding this topic — those seeking guidance in the relevant decisions to be made. Accordingly, this document provides background on what is occurring in other countries, and why and how they are going about those actions. Additionally, this document is directed at those who oversee and work at private organizations such as companies, foundations, and nonprofits that are developing informal education initiatives, and to those who wish to enrich their outlook and empower their work. For them, this paper provides a regional outlook of the actors and initiatives with whom to engage and a survey of the common challenges.

This document is organized in five principal sections. Section 2 will seek to clarify exactly what we are talking about: what is proposed to be taught (what is computer science and computational thinking); a proposal

[7]Microsoft held three meetings in Argentina, Chile, and Mexico, in March and April 2016, which included the participation of doz-ens of public institutions and civil society organizations with ties to digital education. See section 4.

on how to do so (with what tools and pedagogies); and why (what rationales justify their incorporation into children's and youth education). Finally, the available evidence on the effects of these types of policies and initiatives will be discussed. Section 3 will present the main experiences of incorporating computer science teaching in education, as well as the questions that must be addressed when adopting these types of policies. Section 4 will highlight the principal initiatives undertaken by civil society to promote computer science, as well as the primary challenges faced. Finally, section 5 presents the main conclusions and recommendations emerging from the background conveyed in the preceding sections.

2. Background

This section will aim to clarify the primary concepts on computer science education for children and youth. For simplicity, this document uses the term computing to refer to the teaching of *computer science*, *computational thinking*, and/or *coding*. It refers to a relatively new term for those who do not belong to this discipline, but even among experts there is still no complete consensus on the scope of these concepts when they are used in the context of school education and the development of youth.

What are we talking about?

First of all, this discussion should be placed within the broader framework of ICT use in education. Normally, policies seek to take advantage of the educational potentials of new digital technologies to improve and transform the teaching and learning processes that occur in the school context, wherein they promote the didactic use of general applications, educational software, and the Internet in a cross-sector manner in all disciplines of the curriculum. The significance and final purposes of these uses may vary among countries, regions, and schools, but in general one can affirm that all active teachers in the 21st Century are being pushed to adapt their teaching and use ICTs in one way or another. This is a very gradual movement of massive scale and relevance, the final result of which nobody knows for certain, and whose consensus is inevitable.

The idea of promoting the teaching of computer science does not intend to replace or affect this broader process of adapting education to new technology contexts. Rather, it reasons that this adaptation will be incomplete as long as this new knowledge is not considered as part of the core competencies that all students should acquire during their educational experience. The use of office applications, educational software, and the Internet that students learn while they use technology in their everyday lives or during class would be obligatory yet insufficient competencies to function as fulfilled citizens of the 21st Century. Similarly, it would be insufficient what many students learn in their specialized computer science courses during secondary school, such as searching for and organizing information, working with simple databases, and having basic notions of how to code websites. All of this learning, usually referred to as *digital literacy* or technological fluency, would not be at the level of what is required in the new millennium. Nowadays, the systematic study of computing would be required. This subject would therefore be placed at the center of computer science courses and/or in the core of STEM-related disciplines (such as math or physics)[8], as well as extracurricular activities.

The computer science movement is therefore a demand to include in school curricula and youth education the competencies and skills of a new discipline, just as math or history are currently. But

[8]STEM refers to the teaching of Sciences, Technology, Engineering, and Math.

what is it exactly that is sought to be incorporated? What are these new competencies and skills? How should students be taught? Why should it hold such an important place in the education of new generations? What evidence do we have about its relevance? Although we are in a very early stage of this discipline in educational systems to provide good answers to all these questions, the following section will address these concerns in an exploratory manner.

What is proposed to be taught?

Computer Science is an academic discipline that studies what can be done using a computer and how to do so (Wing, 2006); it studies computers and algorithms, including their principles, hardware and software design, their practical applications, and their impacts on society (CSTA, 2010). Algorithms establish the step-by-step procedures that computers must follow so as to acquire, represent, structure, process, and communicate data, as well as perform calculations. All these actions are the basis of computer applications used every day. The field of *computer science* includes the study of topics such as databases, computer architecture and networks, human-computer interaction, operating systems, coding languages, IT security, data networks, distributed systems, expert systems, artificial intelligence, and robotics, among others.

Universities' *Computer Science* departments prepare IT professionals, who are responsible for designing, developing, and maintaining the IT systems in our surroundings. These professionals analyze human problems and processes, structuring and modeling the dimensions that can be addressed by computers, and design and implement hardware and software systems that give shape to IT solutions. This discipline is at the heart of the technology industry, which in recent decades has transformed the world, developing IT systems which, for example, sustain global markets, decode the human genome, organize air traffic, and operate government tenders. They also allow us to perform bank transactions from home, prompt our car to take us on the fastest route to work, keep in contact with our classmates we haven't seen in years, and request the nearest taxi from our mobile phone.

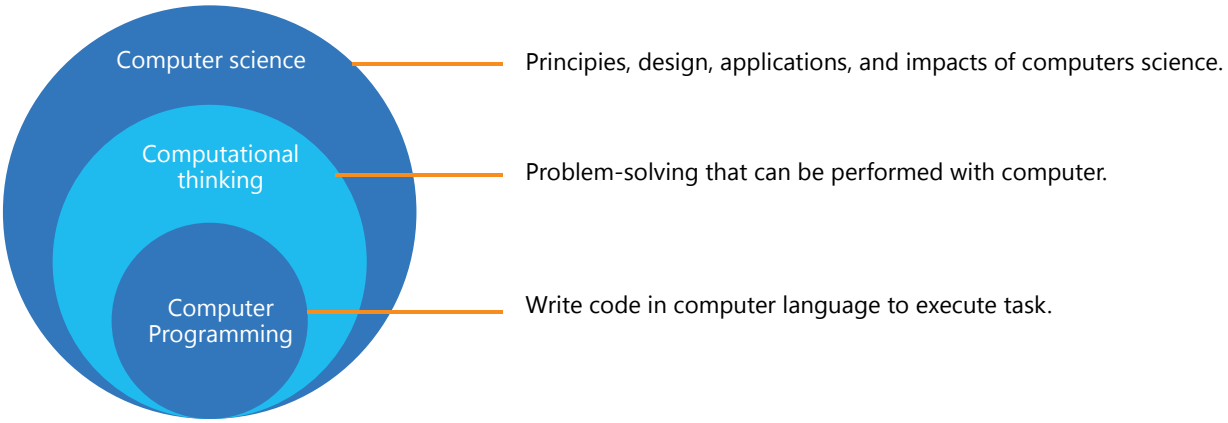
The study on the fundamentals of this science in the context of children and youth education is centered primarily on the development of *computational thinking* (which is why both concepts are used interchangeably). While the latter is not a new concept[9], there is still a lack of complete agreement among academics and educators on its scope and exact definition[10], yet its meaning can be illustrated based on the Wing's own definition: computational thinking is a mental process allowing one to formulate problems so that their solutions can be performed with computers (Wing, 2010). It involves the way of thinking of IT professionals, giving one the ability to understand the uses and limitations of computer solutions and apply computer methods to solver various problems — whether they are everyday, professional, or scientific in nature. Accordingly, computational thinking would allow youth to create a better conceptualization, analysis, and solution to complex problems, by choosing and applying strategies and tools of computer science. Computational thinking means thinking in terms of abstraction and generalization; modeling and decomposing problems into sub-problems; analyzing processes and data; and creating virtual digital and real artifacts, among others (CSTA, 2011; Selby & Woollard, 2013).

[9]The concept of Computational Thinking was used for the first time by Papert in the early 1980s.
[10]For different perspectives, see, for example, Selby & Wollard, 2013; Grover & Pea, 2013.

Sometimes, *computational thinking* is used synonymously with coding, probably because both ultimately refer to the capacities required to solve a problem by defining a set of instructions for a computer to perform a specific task. However, when talking about *computer science* and *computational thinking*, what is sought is to expand teaching beyond *coding*, including the conceptual basis of computing. In fact, one of the main results of this is the ability to code — i.e., to design algorithms and define code which carries it into practice in computer language.

Although the three concepts identified –*computer science*, *computational thinking*, and coding– are not exactly the same[11] (see figure No. 1), the three refer to providing youth with profound comprehension of the fundamentals of the digital world and providing them with tools to be creative agents and not just users and consumers of technology, as is currently promoted in education.

Figure No. 1: Relationship between the principal components of computing



To illustrate how these definitions translate into when it comes time to establish a curriculum leading them into practice, Chart No. 1 summarizes the learning objectives described for the new computer science course in England and exemplifies the type of contents it involves.

[11]It is worth mentioning that many use these three terms interchangeably without making distinctions, creating some confusion in the debate on this topic.

Chart No. 1: Computer Science Objectives in England

The objective of England's National Curriculum in Computer Science is for students to be in conditions to:

a. Understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms, and data representation.
b. Analyze problems in computational terms, and have repeated practical experience of writing computer programs in order to solve different types of problems.
c. Evaluate and apply information technology, including new or unfamiliar technologies.
d. Act as responsible, competent, confident, and creative users of information and communication technology.

Based on these objectives, the English curriculum lays out a trajectory of more specific targets throughout the four levels of school education. For example, in the first level children should be able to detect errors in simple programs, understand algorithms, and how these translate in programs in digital devices, as well as understand that programs are executed following precise sequences of instructions. In the third level, on the other hand, youth should know how to model some real-world problems; understand several key algorithms (for example, ones for sorting and searching); use simple and complex languages to make programs that use data structures (for example, lists, tables or arrays), procedures, and functions; understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems; understand how data of various types (text, sounds, and pictures) can be represented and manipulated digitally in the form of binary digits; and create applications in digital devices for a given audience, with attention to trustworthiness, design, and usability, among others.

(Cobo, 2014)

How is it taught?

There is an emerging body of literature on the development of computational thinking in education, which has fed off the experience of coding instruction in secondary and post-secondary education since the 1970s and 1980s. In that period, possibly due to the complexity of coding languages, there was more emphasis on the coding of instructions than on algorithm design, making this task seem very labor-intensive and intricate. The experience revealed it was not easy to code or learn the most abstract concepts of computing. Yet over time, the methods have matured and the tools have become cheaper and easier to use.

Despite this progress, it is still too early to have a solid, systematized body of knowledge on the teaching and learning of this discipline among children. There is still a major challenge about who can teach computing to youth and how it should be done.

Coding is the principal territory where computational thinking is applied and developed, and it is precisely that area where there is the most experience. Coding computers can be very stimulating, but it is never something simple and it cannot be learned quickly. Since the 1970s, there have been efforts to facilitate

coding education — developing environments and tools, and applications and languages for coding — specifically focused on this purpose, but the selection of these tools, as well as their corresponding teaching methodologies, continues to be a field full of complex decisions encompassing multiple factors, such as the characteristics and skills of learners, learning objectives, and available resources, among others.

In computer science teaching, there is a common practice to use methodologies related to project-, problem-, or challenge-based learning — which are also increasingly used in science learning in general — where students can apply the knowledge they have acquired. Ideally, these problems should be real and authentic, and the focus of work should be on processing information and rewarding student reflection (Lye & Koh, 2014). In the framework of curriculum time restraints and school schedules, it is common practice to allow students to follow step-by-step instructions to carry out the first stages of project creation, but these strategies undermine the possibility of stimulating children's creativity, which is best achieved when they are allowed to use their own strategies to create or adapt projects to solve a problem.

The problems that are addressed when coding is learned occur in concrete digital contexts. For example, game design is a frequently used type of problem as it is particularly attractive to children and youth. Another appealing type is the creation of applications for mobile devices — especially smartphones. Similarly, the construction of tangible artifacts, such as programmable cards or robots[12] — whose physical nature provides an inspiring experience — is also stimulating (Garneli et al, 2015).

Graphic coding languages such as Scratch, Kodu, and SNAP[13] which allow the user to code by moving blocks of instructions on the screen instead of writing code, are the most commonly used in coding teaching, especially in early stages, as they are easy to use and allow users to focus their attention on the design and creation of solutions, avoiding details of syntax of the more sophisticated languages such as Python or Java[14]. While graphic languages are simpler, they have proven to be sufficient to develop the most important concepts of *computational thinking* (Garneli et al, 2015).

There is also a growing number of virtual resources available on the Internet to support teaching and for youth to learn to code independently. Examples of these resources are those developed by initiatives such as the aforementioned *Hour of Code and YouthSpark*. The massive scope these resources have had confirms their usefulness and ease of use.

According to Grover and Pea (2013), the environments and tools that best foster computational thinking have a "low floor" and a "high ceiling" — i.e., they allow users to use them on a basic level, while at the same they can carry out complex projects; they have the right architecture to support the user; they allow transfer to other contexts; and they are equitable and systematic.

[12]There are a variety of programmable cards used in schools, including common ones such as Arduino and GogoBoards. The same is true for robotics kits, with LegoMindstorm being the most popular.
[13]There are many graphic languages, and perhaps the most-used are Scratch (a descendent of LOGO), Alice, GameMaker, and Kodu, among others (Grover & Pea, 2013). SNAP (http://snap.berkeley.edu) and Blockly (https://developers.google.com/blockly/) are other recent examples, which have continued to enrich and simplify coding environments, allowing even young children and pre-readers to use them.
[14]More professional languages, such as Python, Java, and Scheme, among many other options, can be used by more advanced students, as they allow for more authentic and real problems to be approached.

Why teach computing to children and youth?

Different arguments are used to justify the need to educate youth in *computer science*, *computational thinking*, *and/or coding*. The most common ones refer to the benefits they would have for people to have a better and deeper understanding of the digital world in which we are immersed, as well as to develop problem-solving skills while taking advantage of the potentials of digital technologies. For example, it would seem reasonable for everyone to have knowledge on the minimum notions of the internal functioning of computers and the Internet so as to understand their possible scopes and limitations.

This knowledge would allow us to understand and influence the digitalized, computerized, and programmable world of our surroundings. An understanding of the fundamentals of computing would allow people to innovate in the various environments in question, based on the design of new solutions that would improve everyone's quality of life. Without this new knowledge, our full participation in the 21st Century society and economy would be at risk.

From this viewpoint, computational thinking is seen as a basic skill that fosters problem-solving strategies that are useful regardless of the area of study or work, beyond technology itself, as suggested by the growing intersection of this discipline with other disciplines of science and engineering. For example, computing simulations are essential to discover the fundamental rules governing the myriad range of systems, from how ants gather food, to how global markets behave, as well as the detection and treatment of diseases such as cancer, made possible thanks to the development of computing methods enabling simulation and understanding of the genetic mutations involved (CSTA, 2011).

Another important argument refers to the advantages countries would have by attracting more youth toward technology professions. This argument holds that youth must be drawn closer to these topics during — not after — their academic experience so as to increase their interest in pursuing technology-related degrees required by all organizations and industries, especially the IT development sector. Of particular concern is that despite the technology sector's increasing importance in the drive for innovation and economic growth, there are fewer professionals specialized in this area. If this trend is to consolidate, over the long term countries could miss out on technological progress, without the possibility to produce or have control over the systems they would be forced to consume. That is why it seems urgent to reverse this trend, to make school education draw students to the fundamentals of the technological revolution.

In Latin America, where many countries share the aforesaid concerns, there are additional considerations. In particular, this region exhibits high social risks due to the percentage of youth who leave secondary school to work in low-qualifying activities or be exposed to unemployment, drugs, and crime. For them, having technology skills as advanced as possible may be a factor that would increase their employment opportunities, or even better, prevent school dropout rates, amid a context of growing demand for technology skills in the job market. Additionally, given that computer science arouses interest in youth, it seems to be a good vehicle to draw them toward technology and the sciences. Even for the youth of the most vulnerable sectors, technology tends to be appealing and motivating, and is a means where they can unwind, even if at a minimum.

Unfortunately, in our region there are still many countries where access to technology and the most basic skills are beyond the reach of significant sectors of the population. In these contexts, the challenges are even greater, but there are similarly enormous advantages to investing in initiatives that take seize upon youth's interest in technology to integrate them

into education and employment. Hence the value of the efforts by institutions that work with the least-favored sectors in this continent, providing access, basic digital skills, and notions of coding, some of which will be observed in section 4.

In all, the teaching of computer science now carries the promise of an education in tune with the needs of the new century, enabling new generations to acquire skills to not only understand and exploit the digital world, but also to be creative, innovative, and entrepreneurial; skills considered motors for growth and development of different countries. Perhaps it is too early to know if the teaching of computing may respond to this great aspiration, but at the same time it seems even more difficult to hold that its incorporation into the educational agenda could harm it.

What evidence exists?

The promises of computer science education are based on the idea that this discipline develops high-level cognitive skills, such as logical reasoning and problem-solving in the context of computing solutions. As indicated, these skills and knowledge would be valuable by themselves, and transferrable to the most diverse areas of life.

There have been attempts to gauge the impact of coding education in the cognitive development of children since the 1970s, many of which have put a spotlight on problem-solving skills and their transfer to other contexts. However, the evidence gathered is still insufficient to confirm that impact.

Notably, computing education, as well as nearly all technology policies and initiatives in education, have provided a wealth of circumstantial evidence on their impact in people's lives, primarily through records of beneficiaries' testimonies, but when verifying the results in a systematic manner or on a large scale, the evidence tends to be scarce and controversial. While there are studies confirming good results, other studies deny them and others claim there is not enough rigor in none of the preceding ones or that in reality something else must be assessed. Finally, it is impossible to claim with certainty that the promised impacts have been achieved.

This situation is best illustrated with an example. At the beginning of the 1980s, Papert's constructionist theory (1980) was in vogue, which proposed student-centered teaching and the construction of knowledge through design and creation of artifacts that were significant for those who design and construct them. It proposed an education around project development of the entire person: Their intellect, emotions, and motor skills, as well as their passions and preferences. Papert proposed that computer coding was a scenario conducive to putting this idea into practice and he developed the LOGO coding language for this purpose. In this context, great interest was generated in researching the benefits of teaching coding as part of childhood cognitive development. For example, Clement and Gullo (1984) designed an experiment to measure the impact of coding in the style and cognitive development of students and reached the conclusion that there existed significant differences in cognitive style (children are more reflexive), in meta-cognition and divergent thinking, but not in cognitive development in general. Clement's and Gullo's design was supposed to surpass the limitations they had found in prior studies, which they judged to lack rigor: Studies that indicated some of the effects of coding on problem-solving (Billings, 1983; Milner, 1973; Soloway, Lochhead & Clement, 1983; Statz, 1974); as well as others indicating that the transfer of skills developed through coding are strongly dependent on the context, so it is difficult to obtain conclusive results (Pea, Hawkings & Sheingold, 1983). However, the majority of the studies from that period have been questioned as they presented serious problems with validity as they were not rigorously designed (see, for example, Milojkovic's critique, 1984); and in particular, the tests used by Clement and Gullo have

been repeatedly criticized.

Another recent example is the case of robotics, which have been proposed as a learning scenario to boost computational thinking and develop cross-cutting skills and specific, multi-disciplinary knowledge. Benitti (2012) made a compilation of the effects of incorporating robotics into learning, but as in other aforesaid compilations, it underscores the fact that there are few rigorously designed investigations to draw conclusive results.

Currently, the focus of research on the impact of computational teaching is more centered on computational thinking. Yet there is still a lack of conclusive evidence on the effect of coding in the development of computational thinking or of it on the other cognitive capacities of children. For example, Grover and Pea (2013), indicate there exists a great debt with empirical research from learning sciences, to determine the impact on socio-cultural, emotional, and cognitive aspects, in the possibilities of using integrated computational thinking in other areas and in how to improve learning itself from coding.

There are other promises associated to the study of computing which have not been systematically confirmed. For example, some hold that it is a privileged vehicle to develop creativity, an area where traditional schools tend to be stifled within their structures and demands, which aim for students to prepare unique correct answers while offering few degrees of freedom for disruptive approaches to problems and questions. On the other hand, in the exercise of coding, there are multiple, diverse, fairly elegant, and always-personal solutions that are correct, and children have space to explore and create solutions to problems that are interesting to them, with the single guiding framework -- which works -- being the criterion of reliability. While there are numerous testimonies that reflect the development of child and youth creativity, there exists scarce evidence that has been rigorously measured.

Another attractive aspect of computer science education would be its capacity for inclusion, especially for women and people with special educational needs, as a means to expand the opportunities for employment and progress of these groups which, for various reasons, are often marginalized, especially in Latin America. Testimonies show that all children, of different gender, capacity, and social origin, can code and exhibit high degrees of creativity, presenting surprising and original results. This has been possible even in contexts of high vulnerability working with all types of children and youth, producing excellent results. In the case of girls, testimonies show the possibility of using this method to draw them closer to the study of science and technology, as both part of their academic formation and preparation for the job market. This has also been observed in the case of children and youth with special educational needs (disabilities), which marks a new and hopeful opportunity for inclusion.

The teaching of coding, like other educational uses of technology in and outside of classrooms, also holds the promise of engaging and motivating children and youth in a very particular manner. Proof of this, for example, are the millions of participants in online activities of initiatives such as Hour of Code and YouthSpark, even in Latin America. There are multiple testimonies in this sense, both inside and outside of school. For many, this motivation provides an opportunity to refresh parts of students' daily school activities and at home, especially those who are the most vulnerable, as an alternative to recreation, drugs, or crime.

Finally, it is worth nothing that since the beginning of the digital revolution, the main motor of integration of technology in education has been a clear vision on the strategic role that technologies will play in the future construction of our societies. Recent history seems to confirm these predictions, as well as the importance of a technological education that strengthens our

ability to make the most of this revolution for our benefit. At times, this conviction can seem to be sufficient an argument to support the teaching of computer science to youth and children, and would therefore need no further empirical evidence. However, it is always useful to have evidence on the results of these policies and initiatives, in order to introduce benefits, adapt their course, and convince the skeptics. Unfortunately, as has been indicated previously, it is still not possible to have consistent evidence in this respect, but experience and ongoing research are expected to shed light on this important educational field (Hubwieser et al, 2015).

3. Public policies

Many countries are placing greater emphasis on teaching computer science, computational thinking, or coding in education. This section will review some of these experiences, and the primary questions to be addressed when it comes to making decisions in this field will be outlined.

3.1. Country experiences

To illustrate the trend of public policies in relation to the teaching of computing, we have chosen some examples from countries noted for their tradition and experience in this field, as well as the influence in our region. Globally, Europe's situation will be examined, with England's experience standing out. We will also comment on the progress of the United States, Israel, and New Zealand. In Latin America, the key experience of Costa Rica will be approached, as well as the attempts by Uruguay and Chile to connect coding to the school population. To clarify, this selection of experiences should be understood only as examples to illustrate trends, and in no case are these countries the only ones doing something in this matter[15].

Europe

A study by European Schoolnet published in 2014 on the teaching of coding in schools of that continent proves it is a focus of interest in most of its school systems (EUN, 2014). The study included 20 countries, including Belgium, Bulgaria, Cyprus, the Czech Republic, Denmark, Estonia, Spain, Finland, France, Greece, Holland, England, Ireland, Italy, Lithuania, Luxembourg, Norway, Portugal, Poland, and Turkey.

The study reveals that 12 of the 20 countries surveyed already have coding in their curricula (Bulgaria, Cyprus, Czech Republic, Denmark, Estonia, Greece, England, Ireland, Italy, Lithuania, Poland, Portugal) and seven more plan to integrate it. The exception is Norway, but it has a course called "Technology in Practice," in which professors are allowed to teach coding on an individual basis if they desire.

Most of these countries, whether or not they have already implemented coding in their curricula, believe it is very important to develop this competency[16] because, in addition to being valuable by itself, it would contribute to students' critical thinking and

[15]For reviews of other countries, see CSTA (2005), CAS (2011) and Hubweiser, et al (2015).
[16]Not all European countries refer to the competencies to be developed in the same manner: while the majority call it computer coding, some refer to it as computational thinking or algorithmic thinking.

problem-solving skills. Similarly, half the countries believe that developing this competency is vital to attract more students to computing careers and to promote employability in the technology sector.

Nearly all the countries that have already integrated coding into their school systems do so through a special computer science or information technology course, although there are variations on the portion of this curriculum that concern the development of this discipline. Most of these countries consider coding in the secondary level and very few from primary (four of the 12). In general, the compulsory nature of this course is reserved for the highest levels of education, being optional in the other levels, especially in primary. The only country that has established the teaching of computer science as mandatory starting from primary is England, which because of this and other reasons, will be looked at with more attention below.

England

The case of how England came to integrate computing into its K-12 curriculum tends to be taken by many as an example to follow, both because of the radicalism of change and the articulation of different actors in civil society -- academics and industries in the technology field -- with schools, teachers, and policies driven by authorities (Brown et al., 2013).

England has had an information science course since the 1980s when students were taught to code in BASIC, but in the following decade its curriculum was primarily focused on the use of Office tools. However, since the mid-1990s, some began to raise the question that that approach was not yielding the expected results (Stevenson Committee, 1997). A decade later, the situation continued without changes, even when voices from education, the industry, and the government showed growing dissatisfaction amid the way in which the topic was being addressed (Anderiesz, 2014).

In 2007, Simon Peyton-Jones, a researcher from Microsoft Research, began meeting with computer science teachers of schools, and he diagnosed that the curriculum in the subject was failing to instill the required digital skills in students. In fact, it seemed to distance them from computers. From this initial group emerged the association *Computing at Schools* (CAS), which in little time and with the affiliation of hundreds of members, prepared in 2009 the document *Computing at School*: *The state of the Nation*, explaining in detail the problems with technology courses and demanded profound changes to the curriculum addressing the teaching of *computer science*.

When the new government took power in 2010 and began efforts to revise work with ICT in schools, CAS and other organizations wielded a decisive influence in consideration of placing the teaching of *computer science* at the forefront of the new technology course. For example, in 2011, CAS, working with the *British Computer Society* (BCS), launched the basis of a curricular proposal to materialize the proposed changes, and in early 2012 the *British Royal Society* (BRS) published another influential report titled *Shutdown or Restart?: The way forward for computing*

[16]No todos los países europeos se refieren de la misma forma a las competencias por desarrollar: mientras la mayoría la denomina como programación de computadores, algunos las refieren como pensamiento computacional o pensamiento algorítmico.

in UK schools. Similarly, major technology companies, such as Google, Microsoft, and BritishTelecom, began to promote the formation of teachers, a key actor for these types of transformations. Finally, the Department for Education established that the new curriculum, whose specific objectives were embodied in new programs of study[17], should be implemented in all English schools beginning in 2014 (Cobo, 2014).

The principal objective of England's new curriculum is to equip pupils to use computational thinking and creativity to understand and change the world, and to use technology to create programs, systems, and contents, and become *digitally literate* at a level suitable for the world of the future[18].

The particularity of England's case involves the entire country making a complete shift from an Office-centered curriculum to computer science in a mandatory course starting in the first years of primary school. While England had teachers specially prepared to impart information technology courses, the new curriculum implies an enormous challenge in terms of formation of and support for teachers, as well as the creation of orientations and the provision of support materials (texts, assessment instruments, specialized software, etc.). Founding a new school course such as this requires much more effort than improving an existing one. For this, the Department for Education has collaborated with many other institutions committed to this transformation, such as academics and students of computer science in universities, private companies, and civil society organizations, including CAS as previously mentioned (CAS, 2015).

United States

In early 2016, President Obama launched the initiative *Computer Science For All (CS for All)*, with the aim of empowering all K-12 students to learn computing and to be equipped with *computational thinking* skills that would enable them to be creators in a digital economy and active citizens in a technology-driven world[19]. Based on the declaration of this initiative, computer science is a new basic skill required to take advantage of economic opportunities and to promote social mobility.

CS for All is the first federal initiative that provides a strategic enhancement to computer teaching in the American education system. Resources will be provided to states and school districts to train teachers, prepare instructional material, and forge strategic alliances with companies, media outlets, foundations, as well as other civil society organizations and professionals interested in collaboration, so as to expand learning of this discipline in schools. Similarly, the initiative will support institutions that have been developing computer teaching-support programs in secondary from long before, as is the case of the National Science Foundation (NSF)[20].

In fact, the United States has a long tradition of computer education in secondary schools, in the framework of information science courses or as part of efforts to strengthen the teaching of STEM disciplines. However, since the early 2000s, interest in these types of courses has declined. According to a study carried out in 2009-2010 by *The Association*

[17]https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study
[18]The objectives of the program of studies were highlighted in Chart No. 1 in section 2.
[19]https://www.whitehouse.gov/blog/2016/01/30/computer-science-all#Need
[20]See for example NSF's CS10K initiative at https://cs10kcommunity.org/

for Computing Machinery (ACM) and *The Computer Science Teachers Association* (CSTA), two-thirds of states included computer science concepts and skills in their secondary curricula, but in a very irregular and optional manner. In most cases, the use of ICTs in K-12 was focused on using technology to support multi-disciplinary learning in the curriculum and to provide students with basic digital literacy (CSTA, 2010).

While institutions such as NSF or CSTA have an extensive trajectory in supporting computer science teachers across the country, primarily providing curricular and professional development, the overall situation was that computing was not considered as part of the mandatory curriculum of school education. In this context, the president's initiative is expected to strengthen prior experiences and facilitate the incorporation of computer science in the teaching of all students, especially in secondary school.

Israel

This country has a long tradition in teaching coding and computers science. Since the 1970s, the secondary curriculum has had an elective course to teach the BASIC coding language, which incorporated LOGO coding and the use of general applications in the 1980s. In the early 1990s, the Ministry of Education formed a committee to rethink this course, which was reformulated to focus on the fundamental concepts of *computer science* and on the development of algorithmic thinking, where coding is seen as 'the means to get a computer to execute algorithms' (Gal-Ezer, 1995, p73). Throughout the following decade, this new curriculum was gradually implemented in secondary schools. In 2000, the Ministry created the *National Center for Computer Science Teachers (Machshava)*, an organization dedicated to fostering and supporting the professional development of teachers in this discipline.

New Zealand

In the mid-1970s, New Zealand introduced coding education as part of an applied math course at the end of specialized secondary, which lasted until the mid-1980s, when a technology course was created for practical study on a variety of technology-related topics -- from nutrition to digital -- which immersed coding in a much broader curriculum but without its own space. In 2009, the Ministry of Education convened a panel of experts with industry representatives, universities, and secondary schools to revise technology education. This panel's recommendations led to a new technology curriculum beginning in 2011. This new curriculum established special elective courses in the final years of secondary for digital technology education, which explicitly include coding and computer science (Bell et al, 2010).

Costa Rica

Costa Rica was the first Latin American country to launch a national policy to incorporate ICTs in schools in the second half of the 1980s -- the National Program for Educational Informatics (PRONIE) -- which is inspired by the perspective that technologies can contribute to students' cognitive development, and in particular, that coding can be a powerful

[20]Ver por ejemplo la iniciativa CS10K de la NSF en https://cs10kcommunity.org/

means for developing children's problem-solving and logical reason capacities. After nearly three decades of gradual growth, now most urban primary and secondary schools in Costa Rica have an educational computer science course with lab classes where students work on projects -- thematically linked to the school curriculum -- that result in coding products (they initially used the LOGO language, and currently Scratch is used). Although this educational computer science course is mandatory, its evaluation is not factored into student grading (Muñoz et al, 2013). Currently, the Omar Dengo Foundation, which is responsible for PRONIE, is proposing to modify the curriculum of this course to include contents unique to computer science, computational thinking, robotics, and maker courses.

Chile

As part of ENLACES, Chile's national policy to integrate ICTs into the school system since the early 1990s, the Ministry of Education has been essentially focused on using technology to support the teaching of different courses transversally in the primary and secondary curricula, and on developing digital information management skills on the Internet. Nevertheless, in recent years, Chile has promoted coding skills through robotics-coding workshops in secondary schools[21]. These workshops, however, have a limited scope, as they only cover about 25% of secondary schools and among them, only groups of interested students, and during the year when they receive explicit support from the Ministry, so they do not necessarily constitute a permanent offering for students.

Uruguay

Uruguay was the first country in the world to deliver a laptop to every primary and secondary school student, with the strategic goal of improving educational quality in a framework of equality. As part of this policy, launched in 2007 and called PLAN CEIBAL, Uruguay has developed a series of supplementary initiatives to use this infrastructure for student learning, one of them being the Laboratorios de Tecnologías Digitales (LabTeD). LabTeds are coding workshops that are held in nearly half of secondary schools and where students carry out projects, the final result of which is the design and construction of a technological artifact of hardware and/or software based on coding, robotics, and sensors. The result of these activities are expected to be that students develop superior competencies such as creativity, collaboration, and critical thinking, among others. Ceibal has also facilitated these labs forming part of the existing secondary-level computer science courses since the 1990s, which have been highly focused on office technology up until now (Jara, 2016).

With the exception of Costa Rica, the policies in our region lack tradition or institutionalized schemes for the teaching of computer science-related topics in schools. However, as illustrated in the cases reviewed, it is possible to find defined initiatives in the form of extra-curricular workshops that engage secondary students, and there is a growing interest to formally incorporate computing into informatics courses. Proof of this is Uruguay's case and others not addressed in this review, such as the City of Buenos Aires in

[21]See http://www.enlaces.cl/proyectos/mi-taller-digital/

Argentina, which recently modified the curriculum for the final years of secondary to incorporate aspects of this discipline[22].

For their part, the developed countries examined show a long trajectory with courses that form part of the curriculum, primarily secondary. Generally, these courses, often times electives, include coding, but are more focused on the use of technological applications. In recent years, these curricula have been subject to review to more decisively incorporate computer science and offer them to all students. These changes are expected to raise the status and quality of these courses and the teachers who impart them, and the incorporation of more relevant and challenging topics makes it possible to attract more students to study this discipline.

[22]http://www.lanacion.com.ar/1791189-la-informatica-ya-forma-parte-del-plan-en-la-caba

3.2. Difficult decisions

The experiences we have reviewed show that the decision-making process on the incorporation of computer teaching in education is long and complex, and responds to the unique histories and contexts of each country. Below are some of the principal questions to consider when it comes time to incorporate computer teaching into school education.

Should computing be taught in school education?

The main question to be asked is if computer science should be a school subject or if it should remain a post-secondary degree. As we have seen, the most developed countries are convinced that if it does not start in school, their technological leadership is at serious risk. It is worth asking if in other contexts one can justify incorporating the teaching of this new discipline during class hours, even at the cost of displacing other learning equally or more relevant for child development (Hubwieser et al, 2015).

In particular, in Latin America there are still extremely important debts in terms of basic coverage and learning which demand the attention of public policies (TERCE, 2015); it is also worth asking what priority should be given to incorporating computer science teaching into the education agendas of this continent. It is possible that different countries have different answers to this question, including an intermediary that proposes the gradual installation of computing in the academic system while at the same time developing the national capacities required for mass implementation. What seems inescapable is that, sooner or later, countries will have to respond to this major question and define a way to address it.

In any case, the challenges that would be faced in the process are no less complex than the decision to undertake it. Computer science is a relatively new discipline, and even more so is its recent emergence in the school context, where it is faced with a very different scenario than tertiary education. There are still many questions without a clear answer, and the countries embarked on this movement are taking a risk and paving new roads.

Is it for all children from an early age?

There is no single answer to this issue. Some incline to begin as early as possible, indicating that there are good experiences in that respect. Others indicate it is better when children have achieved a certain capacity for abstraction (for example, starting from 11 years), so that they are prepared to understand more complex concepts, such as algorithms, variables, recursion, and others. Perhaps the answer is that there is a continuum of possibilities, from the simplest (basic coding) to the most complex. The experience of most countries studied is that they begin computer teaching in secondary school, except for some cases such as England and Costa Rica, which do so from primary. Similarly, for many it is an optional extracurricular course, or elective, or specialization at the end of secondary, but the observed trend is to offer this educational experience to all students.

Therefore, there are various options for curricular design, depending on the chosen style, to introduce computer science teaching in schools. The first option to address is whether computer science education is offered as part of required courses for all students or in optional activities for only those who are still interested. If required class time is the case, one must decide if it will be covered in a single course, probably dedicated exclusively to computer science as is the trend, or transversally, incorporating different topics in various existing courses, such as mathematics and science (Carvalho et al, 2013).

The option to incorporate computer science in courses of other disciplines is a more complex and uncertain trajectory, as normally their curricula are already stressed from their own content and activities, which would entail training a wide range of teachers

on the fundamentals of computer science to be incorporated. Having its own course instead seems to be a safer alternative, although no less challenging, in particular in the countries where there does not exist a technology course that can be used with these purposes. Indeed, as class time is a limited resource, the incorporation of a new course would mean reducing the time of other courses that are no less important, the proponents of which must be convinced of the high value the new formative experience would have (Carvalho et al, 2013; Hubwieser et al, 2015).

What should be the content of the curriculum provided?

One question that is always hard to define is the minimum contents of school subjects, even in stable disciplines with a long tradition, such as math or science. In the case of young computer science, the challenge is even greater. However, various countries have carried out complete and detailed curricular designs, with years of experience testing them out in schools. This experience is certainly a very early point for defining the curriculum in question.

These definitions, however, are not made out of thin air, but rather in countries with a certain history of this; with teachers specialized in technology and coding; and with a network of organizations that can help in their implementation (such as universities and companies, among others). Therefore, one must be conscious of the need to modulate these models to the specific reality of the country, which often times does not have these same conditions.

The definition of the curriculum is also a space where interests and outlooks in each context are disputed, which must be articulated by the educational authority based on the country's strategic interests. England's case is also worth mentioning as an example of this type of processes. As various protagonists who participated directly in designing the new curriculum can attest, the first proposal prepared by an expert committee consulted by the Department for Education included a balanced mix of computer science content along with less specialized content related to culture and digital literacy. However, this proposal was replaced by another solely focused on computer science due to strong influence from the industry and other agents strongly connected to this discipline (Cobo, 2014).

How can teachers be prepared?

The primary challenge involved in incorporating computer science into school education is educating teachers who are able to convert established curricular guidelines into learning. Even in countries with the longest tradition in this field, there continue to be difficulties involved in the adequate preparation of instructors. To aid this process, Israel, for example, created a specialized center, and England is investing massive amounts to deploy a variety of strategies to transform these professionals (CAS, 2015). One must also consider that these countries have, in general, a high-quality teaching contingent, and in particular, instructors specialized in technology with at least a minimum amount of knowledge on coding.

Teaching of the new discipline requires teachers with very solid conceptual and practical knowledge, along with a pedagogic formation that enables them to convert this knowledge into effective learning experiences for students (CSTA, 2005). While the fundamentals of computer science are, at least in theory, available in pre-university degrees in the technology field, the teaching techniques or methodologies and their resources (texts, orientations, evaluations, languages, software) have recently begun to mature in the same classrooms. We still have a long way to go before we have a consolidated body of knowledge on the teaching of computer science in the school context, providing clarity of the different strategies and resources for each age (Hubwieser et al, 2015).

The countries that are making progress in this subject are defining standards for educating new computer science teachers, establishing requirements and trajectories for their initial education and certification systems. However, it is still unclear if enough teachers can be engaged in this new specialty, or how many decades it could take to have an acceptable level of teachers in each classroom in the country.

What infrastructure is required?

Although computer science has a conceptual component that can go without the use of computers to a certain extent, the learning of the discipline is materialized and enriched by solving coding-related problems on digital devices. Having easy access to computers in school is thus a minimum requirement for implementing these policies, whether in the form of labs with enough computers to perform coursework, or another type of access such as mobile carts or bags to carry laptops to class or simply students carrying their own mobile devices.

The reality in Latin America is therefore quite irregular: While there are countries where the majority of schools have labs or laptops, there is also a sizable group where the majority of schools lack this basic resource (Jara, 2015). The same occurs at students' homes. In this context, the teaching of computer science necessarily involves major investment in equipment.

Yet even with computer labs, as occurs in various countries in the region, the decision to use them in computer courses must be analyzed with caution. These labs are the result of policies that have invested decades in promoting the transversal use of ICTs in order to support learning in different courses. When assigning labs a new use, in the absence of other investment enabling ICT use in classrooms, teachers of other classes would be prevented from using technology with their students.

In short, implementation of an educational policy to introduce computer science learning into school education, as is being done in developed countries, requires that key matters be addressed -- such as gradualness, teacher education, and infrastructure -- which must be carefully designed, considering the context of aspirations and capacities of each country. It is recommended that policy decisions in these types of matters are supported by studies that consider all variables and investments required for the promises to be feasibly carried out. Otherwise, as often occurs in our region, it would entail another endeavor impossible to fulfill.

4. Independent initiatives

This section gathers part of the experience of numerous private and civil society organizations that promote computer science teaching, including participants in the meetings organized by Microsoft in Argentina, Chile, and Mexico in early-2016 (see appendix). These experiences involve private companies, foundations, nonprofits, international organizations, and groups of people such as unions or associations (software associations, for example). There are also initiatives powered by state entities which are not part of educational policies for the school system, but rather which are associated with policies aimed at youth or the digital development of the public in general, as is the case of projects promoted by the Ibero-American Youth Organization and ministries of youth directed at young people who have dropped out of the formal education system.

The experiences of these organizations, along with many others around the world, confirm the existence of a great wealth and variety of concrete initiatives regarding coding for youth and children. Evidence indicates that these initiatives no longer represent a few isolated and sporadic actions of some enthusiasts, rather that many of them and their supporting arguments comprise experiences -- in mass occasions -- of digital literacy and computer science education for children and youth, primarily in extra-curricular contexts, and in particular, among groups of youth who are at social risk. These initiatives complement -- and in many cases, advance -- government policies seeking to respond to social problems by using the opportunities provided by technology.

4.1. Initiatives and motivations

In this field, one can find a broad range of motivations, purposes, and focuses, as well as initiatives and organizations. For example, there are initiatives promoted by companies that offer Internet resources for young people interested in coding to learn on their own and to organize partnerships and events to promote their use in schools and clubs. There are initiatives undertaken by foundations and nonprofits that work directly with youth in vulnerable neighborhoods, with the purpose of expanding their work opportunities; there are some focused on working with women or students with special needs; and there are others that offer extra-curricular coding workshops at schools.

Below is a summary of some initiatives undertaken that illustrate the breadth of computer science-related efforts.

YouthSpark[23], is worth noting -- a global initiative by Microsoft in partnership with non-governmental organizations, companies, and families that seeks to increase youth access to computer science education, with the aim of empowering them to aspire to greater

[23]YouthSpark hub: https://www.microsoft.com/about/philanthropies/youthspark/youthsparkhub/

achievements for themselves, their families, and their communities. To date, more than 300 million youth from around the world have participated in Youthspark.

Participants can take computer science classes and use digital resources through which they learn about computational thinking and develop problem-solving skills. Some of Youthspark's numerous initiatives include: YoPuedoProgramar and the Microsoft Virtual Academy Course, "Learning to Code"; Kodu; and Imagine Academy, the aim of which is to design and create games and digital applications using IT tools and computer science knowledge; as well as DigiGirlz and Girls Who Code, directed at girls to foster their participation in computer science workshops to engage them in science, technology, engineering, and math by connecting them with female Microsoft employees and through participation in workshops with other girls.

The *YouthSpark* program for Latin America *YoPuedoProgramar*, has had a considerable impact, with more than 3 million participants. This initiative allows participants to begin their first stages of coding and opens the doors to those interested in continued learning and to obtain a special diploma in Microsoft's Virtual Academy[24]. This initiative emerged in 2013 based on the recommendations for the Post2015 agenda of the Ibero-American Youth Organization (OIJ), which included teaching coding for formal and non-formal education. The scope of *YoPuedoProgramar* has been enriched by the collaboration of ministries of youth that facilitated the use of youth centers, as well as the outlook of the OIJ, which has recommended strengthening these joint efforts to increase opportunities for youth[25].

Also of note is the initiative *Hour of Code*[26], a global movement with the participation of more than 100 million students in 180 countries. It involves an hour-long introduction to computer science, designed to show that anyone can learn to code, regardless of prior experience, and starting at four years of age, the ability to understand the basic fundamentals of the discipline. The tutorials, all of which last one hour, are available in more than 40 languages.

CodeClub[27], for its part, is a world network of volunteers who teach students ages 9 to 11 to program after-school for one hour per week, whether at school or in a community center. To date, there are more than 6,000 clubs in the world. Coding projects include digital games, animations, and webpages.

Finally, ScratchEd[28] emerged from the large popularity of the Scratch coding language, whose user page now has more than 15 million projects that can be shared, used, and modified. Launched in 2009, ScratchEd is an online community where educators and those interested in coding with Scratch can share curricular experiences of this language, in the classroom or at home, exchange resources, submit questions, and contact other teachers[29].

These globe-spanning initiatives stand out in a sea of actions of various scopes, focuses, and duration. Despite this diversity, in this field of rich activity it

[24]https://mva.microsoft.com/
[25]Preparation of this agenda, which is part of the Sustainable Development Goals of the OIJ, included contributions and experience of Microsoft and its philanthropy and education programs. Similar recommendations have been indicated in Ibero-American youth summits and in the Ibero-American Presidential Summit. See recommendations in OIJ's Post-2015 Agenda in "AGENDA FOR DE-VELOPMENT AND SOCIAL INVESTMENT IN YOUTH: A POST-2015 STRATEGY FOR IBERO-AMERICA" (OIJ, 2013).
[26]Hour of Code https://hourofcode.com/es
[27]Code Club https://www.codeclubworld.org
[28]ScratchEd http://scratched.gse.harvard.edu
[29]Scratch https://scratch.mit.edu

is possible to identify shared outlooks, diagnostics, and goals. In particular, many share the interest of using the positive attitude and basic technological knowledge of children and youth, especially in vulnerable sectors, to empower their opportunities for employment and entrepreneurship. This outlook is supported by the following elements:

Firstly, surveys, research, and experiences indicate that youth nowadays are hyper-connected (transcending socio-economic levels) through social networks. Although this motivation offers opportunities for students to use technologies for their formation and future careers, it is also known that the use of social networks does not imply the development of sufficient skills for efficient, broad handling of opportunities offered by technology. The dynamic of jobs in all productive and services sectors requires new skills, many of which are associated with the ability to effectively use and create digital technologies, so it is necessary to prepare youth to better approach the new employment scenario. In some countries with high drop-out rates, this is related to the need to apply student retention strategies in secondary schools and offer them tools for employability.

Secondly, technology is positioned as an important, and sometimes central, component in start-ups. Youth wishing to create new companies or nonprofits must be informed, construct proposals, and establish networks -- tasks for which technology is necessary. Along these lines, various nonprofits consider the transformative potential of ICTs, by observing their effects in youth and adults seeking new ways to approach the job market or to enrich their endeavors. Similarly, developed countries -- and gradually, developing countries -- are blaming the deficit of a qualified workforce on the lack of digital literacy. This is both a challenge and an opportunity for our countries -- preparing youth to undertake the professional challenges of tomorrow.

4.2. Work strategies with youth

The experience of various initiatives conveys the importance of creating strategies to engage and attract youth to coding, especially those who are less motivated by it or who have not had the opportunity to access initiatives they can undertake. Some of these strategies reported by different organizations working the field include:

One common factor seems to be implementing spaces for learning and engagement in the same places where youth are, in particular in their neighborhoods or places of recreation. The slogan is, "go where the youth are." For example, in vulnerable neighborhoods, there are numerous initiatives such as "Casas del Futuro" of the Under Ministry of Youth in Argentina, "Casas Poder Jóven" of IMJUVE in Mexico, "DesarrollArte" and "Acá estamos" by INJUV-Chile.

Another aspect that is required in organizations active in this field is communication. For youth and teachers, it is important to communicate good experiences with coding, to tell positive life stories – personal testimonies conveying personal or group development, success stories, etc. It is also necessary

to eliminate the stigma that coding is boring or only for technology addicts, through testimonies of girls and boys who develop their projects in a collaborative manner, working in groups, in learning environments that are both appealing and playful.

Some organizations are making a special effort to integrate girls and youth into coding, based on projects that are personally appealing to them. Other organizations' spotlight is on people with special educational needs, and their testimonies reflect that it is an area of great potential for development, with approaches toward areas of entrepreneurship and employability, and in some cases even related to specific disabilities. Nonprofit organizations, such as CDI, Trust for the Americas at the regional level, ProAcceso in Mexico, Laboratoria in Peru, Chile, and Mexico, and Comunidad IT in Argentina, among many others, have served as a source for education in technology and Computer Science skills. These efforts seek to create concrete economic opportunities for employment and entrepreneurship for youth in adverse socio-economic contexts, including job placement in company technology departments.

Additionally, the Maker movement is worth noting, which is the focus of various initiatives in the region.

The Maker movement

Martin (2015) summarizes the philosophy of this movement and the promises it offers to transform learning. In summary, three converging factors are recognized which have given substance and energy to the Maker movement.

First of all, new low-cost processors (such as Arduino, RaspBerry, Intel Galileo) have given life to the so-called Internet of Things, popularizing coding of all types of domestic and industrial artifacts, without requiring advanced knowledge of electronics or coding. In addition to these technological advances are 3D printers and CNC machines, the costs of which have dropped in "Makers Spaces." These can now be found in schools as well. In second place, computer coding in schools -- which has been addressed in this document -- has re-emerged in recent years with easy-to-understand languages, even for beginning-level, basic education students (Scratch, Snap), freely available and in Spanish. Third is the renewed drive for education related to project-based learning (PBL) that is meaningful for students and which offers ways to approach teaching-learning processes in an interdisciplinary manner to solve real, interesting, and complex problems. In this, teachers gradually assume a role of learners and students take on an active role, with collaborative work and collective and gradual construction of knowledge during planning and development of their projects.

These three associated trends breathe life into the Maker movement in schools, homes, and workshops. Examples include the Makers[30] faire in many countries, with thousands of youth sharing their creations of simple electronic components, programmed in simple languages. The educational potential of Makers is linked to curricular initiatives (labs in particular) in subjects such as physics and

[30]http://makerfaire.com

mechanics (electricity, movement, sound, heat, light, principles of engines, generators, renewable energies, etc., biology), biology and botany (plant treatment, nutrients, irrigation, photosynthesis, gardens, etc.) and even music (sound generation in analog-digital devices) and art (connecting the value of sensors with colors and shapes).

The additional interest that draws teachers working with students in the Makers line is the ability to develop their creativity, curiosity, entrepreneurial spirit, and "learn by doing" for projects that are appealing to youth. Given the low costs and simplicity to take the first steps, a strategy to test and prototype is being used in a progressive manner. This helps students "dare to dream" and take risks with new ideas.

One interesting aspect of the Maker movement is the attitude it sparks among youth, as they feel empowered to create inventions they usually see to be beyond the reach of their capacities (for example, making a mini-submarine, control the safety of a space or code an automatic irrigation system). This is important, 'especially among girls who tend to give up on the realms of science and math in secondary education[31].' Students need challenges that genuinely motivate them to be inquisitive (and not just because school requires it). The projects one can develop as part of the Makers line precisely inspire curiosity and exploration of ideas, using modern, low-cost technologies.

4.3. Challenges

From the experience of various initiatives also emerges a set of challenges for their future development that merit attention. Particularly:

It is worthwhile to encourage other organizations to join these initiatives, which have often failed to see this opportunity or have believed it is beyond their focus of attention. Yet experience shows there is a great diversity of nonprofits incorporating coding for youth into their focus of action. For this challenge, there must be effective communication of the work performed in different contexts and for different actors (particularly in neighborhoods of vulnerable contexts), in a language that challenges them. Specifically, the challenge is: What and how to report on these initiatives to attract more support? This segment also includes parents and guardians who should know the benefits their students learn from computing, as they can arouse engagement in school initiatives. For public policies, it is necessary to articulate a solid set of arguments, experiences, evidence, and testimonies, which sensitize decision-makers in ministries and departments of education and of social issues.

The experience of those who work in the area indicate that in excluded sectors and vulnerable contexts there is a gap of information regarding opportunities that technology can offer or can be used through technology. Those who live and work in these sectors must be able to answer the question, what can I learn? What can I do with what I'm learning? And at the

[31]http://www.weareteachers.com/blogs/post/2015/04/03/how-the-maker-movement-is-transforming-education

same time, be able to connect with other people in similar contexts to establish informal networks, solve problems, and undertake employment or educational opportunities.

Although there is already certain experience in gender-inclusion initiatives in some initiatives, additional efforts must be made to collect more arguments and testimonies encouraging girls and young women to engage. One option is to disseminate the testimonies of girls who have stood out: What have they done and how have they done so? What do their experiences say and what do they indicate for other girls?

5. Conclusions

This document aims to contribute with decisions and efforts regarding the teaching of computer science to new generations of Latin American children and youth. It includes the principal background, experiences, and lessons on this matter, obtained by gathering specialized literature and testimonies of those in charge of initiatives occurring in three meetings organized by Microsoft in Argentina, Chile, and Mexico, in early 2016.

This review has helped shed light on the growing interest of promoting computer science education -- computer science, computational thinking, and/or coding -- for children in youth both in and outside of the region. Proof of this is that educational policies around the world are revising their school curricula to strengthen teaching for all students, and that we are witnesses of various initiatives promoted by companies and civil society organizations that complement these public efforts to expand opportunities for youth.

While the origins of this movement date back to the 1970s, in the past decade there has been growing concern that technology education for new generations is not enough to take advantage of the benefits of the digital revolution. In response to this scenario, many countries are changing the focus of their technology curricula in formal school education from digital literacy to the systematic study of the discipline of computer science, computational thinking, and coding. This change also implies a massive effort to train the teachers of these courses, and to provide all the support required for student learning (equipment, texts, companionship, etc.). It involves a long-term transformation, the results of which must be reflected both at the individual level, in greater problem-solving skills, increased understanding of the digital world, and more interest for technology degrees, as well as at the social level, in the form of increased development of technology-based industries and greater economic competitiveness of countries.

Additionally, a variety of organizations are deploying initiatives to teach digital and coding skills to children and youth, especially beyond the school context and to those whose social situations are at risk, with the confidence that these new skills will help them to enter labor markets with more value and strength. There are initiatives of various types -- some offer online, self-learning resources available anywhere in the world, others work in marginalized urban neighborhoods with vulnerable groups in a small city. But they all share the vision that greater mastery of technology leads to new opportunities for youth and their communities.

The international experience of policies and initiatives such as those reviewed in this document are still not enough to confirm with solid evidence the benefits promised by computer science education, particularly its impact on the cognitive capacities of students -- logical reasoning, problem-solving. Yet the need to provide youth with more effective tools to have more work opportunities and contribute with innovation and growth in an increasingly digital world, seem to now be sufficient arguments to sustain these investments.

Nonetheless, Latin American countries must face the question of whether they will address in some way computer science education of children and youth, as the most developed countries are doing. There are many paths to take for this -- some of them are more radical, such as England's case, in which it established a required computer science course starting in primary school, to more gradual and optional strategies inside and outside of the formal educational system, such as extra-coding workshops, Olympic-style games or fairs that support and mobilize those who are engaged. Accordingly, the work of organizations that develop independent initiatives can be a very valuable resource to promote the development of these new capacities in many countries of the region. While policies and initiatives have been presented separately in this document, giving the false impression they are on separate tracks, the synergy between both efforts is a very important route to explore.

In any case, computer science teaching requires that various challenges be addressed, which have been reviewed in the previous sections of this document. Of note among these is the need to design relevant curricula, ensure proper training of teachers, provide the required infrastructure, and coordinate efforts of different policies and initiatives in networks allowing their impact and sustainability to expand.

Bibliographical references

Anderiesz, M. (2014). Erase/Rewind- The Story of Computer Science Education in the UK.

Bell, T., Andreae, P., & Lambert, L. (2010). Computer science in New Zealand high schools. Procedures 12th Australasian Computing Education Conference (ACE 2010), Brisbane, Australia, 15–22. Retrieved from http://dl.acm.org/citation.cfm?id=1862223

Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. Computers & Education, 58(3), 978–988. http://doi.org/10.1016/j.compedu.2011.10.006

Brown, N. C. C., Kölling, M., Jones, S. P., Ave, J. J. T., Humphreys, S., & Sentance, S. (2013). Bringing Computer Science Back into Schools : Lessons from the UK, 269–274.

BRS (2012). Shutdown or Restart: The way forward for computing in UK schools. The Royal Society: London.
Carvalho, T., Andrade, D., Silveira, J., Auler, V., Cavalheiro, S., Aguiar, M., Foss, L., Pernas, A., Reiser, R. (2013). Discussing the Challenges Related to Deployment of Computational Thinking in Brazilian Basic Education. In 2013 2nd Workshop-School on Theoretical Computer Science (pp. 111–115). IEEE. http://doi.org/10.1109/WEIT.2013.27

CAS (2009). Computing at Schools: the state of the nation, Computing at Schools Working Group Report for the UK Computing Research Comitee: UK.

CAS (2011). Computing at Schools: International Comparison, Version 5: UK.

CAS (2015). Closing the gapto achieve a world class computing teaching workforce, Computing at Schools, British Computer Society, the chartered institute for IT: UK.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. Journal of Educational Psychology, 76(6), 1051–1058. http://doi.org/10.1037/0022-0663.76.6.1051

Cobo, C. (2014). Experiencia del caso inglés en la integración de TIC y la definición de estándares de habilidades TIC para docentes (1997-2013), Enlaces: Santiago

Computational Thinking: https://medium.com/@lorenaabarba/computational-thinking-i-do-not-think-it-means-what-you-think-it-means-6d39e854fa90#.n4h5bmeoh

CSTA (2005). The New Educational Imperative: Improving High School Computer Science Education. Using worldwide reasearch and professional experience to improve U.S. Schools. Ccomputer Science Teacher Association Curriculum Improvement Task Force: New York.

CSTA (2010). Running on Empty: The failure to teach K-12 Computer Science in the Digital Age, ACM-Association for Computing Machinery/ Computer Science Teachers Association: USA

CSTA (2011). CSTA K-12 Computer Science Standars, Computer Science Teachers Association: New York
EUN (2014). Computing our future Computer programming and coding - Priorities, school curricula and initiatives across Europe. European Schoolnet.
Retrieved from http://www.eun.org/c/document_library/get_file?uuid=521cb928-6ec4-4a86-b522-9d8fd5cf60ce&groupId=43887

Gal-Ezer, J., Beeri, C., Harel, D., & Yehudai, a. (1995). A high school program in computer science. IEEE Computer, 0198, 1–21. Retrieved from http://teacher.tchcvs.tc.edu.tw/mhtsai/essay/high-school-program.pdf

Garneli, V., Giannakos, M.N., Chorianopoulus, K. (2015). Computing Education in K-12 Schools: A review of Literature, Conference Paper, IEE Global Engineering Education Conference, March 2015, Tallinn University of Technology: Tallinn, Estonia.

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

Grover, S., Cooper, S., & Pea, R. (2014). Assessing computational learning in K-12. Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education - ITiCSE '14, (April 2016), 57–62. http://doi.org/10.1145/2591708.2591713

Havenga, M., Breed, B., Mentz, E., Govender, D., Govender, I., Dignum, F., & Dignum, V. (2013). Metacognitive and Problem-Solving Skills to Promote Self-Directed Learning in Computer Programming : Teachers ' Experiences. SA-eDUC JOURNAL, 10(2), 1–14. Retrieved from http://www.nwu.ac.za/sites/www.nwu.ac.za/files/files/p-saeduc/sdl issue/Havenga et al. Metacognitive and problem-solving skills to .pdf

Hubwieser, P., Armoni, M., & Giannakos, M. N. (2015). How to Implement Rigorous Computer Science Education in K-12 Schools? Some Answers and Many Questions. ACM Transactions on Computing Education, 15(2), 1–12. http://doi.org/10.1145/2729983

Jara, I. (2016). Revisión comparativa de iniciativas nacionales de aprendizaje móvil en América Latina. El caso del Plan Ceibal de Uruguay. UNESCO: Paris

Jara, I. (2015). Infraestructura digital para educación: Avances y desafíos para Latinoamérica. IIPE-UNESCO: Buenos Aires.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through

programming: What is next for K-12? Computers in Human Behavior, 41, 51–61. http://doi.org/10.1016/j.chb.2014.09.012

Martin, L. (2015). The Promise of the Maker Movement for Education. Journal of Pre-College Engineering Education Research Journal of Pre-College Engineering Education ResearchJ-PEER Journal of Pre-College Engineering Education Research, 5(5), 1–30. http://doi.org/10.7771/2157-9288.1099

Microsoft YoutSpark: http://www.microsoft.com/about/philanthropies/youthspark/

Muñoz, L., Brenes, M., Bujanda, M.E., Mora, M., Nuñez, O., Zúñiga, M. (2014). Las políticas TIC en los sistemas educativos de Américalatina: Caso Costa Rica,: Buenos Aires Unicef.

Papert, S. (1988). A critique of technocentrism in thinking about the school of the future. Children in the Information Age: Opportunities for Creativity, Innovation and New Activities, (2), 3–18.

Papert, S. (1993). Mindstorms: Children, Computers and Powerful Ideas. Perseus Books: New York.

Selby, C., Woollard, J. (2013). Computational thinking: the developing definition. Retrieved from http://eprints.soton.ac.uk/356481/7/Selby_Woollard_bg_soton_eprints.pdf

Selwyn, N. (2002). Learning to Love the Micro: the discursive construction of 'educational' computing in the UK, 1979-89, British Journal of Sociology of Education, vol 23, no 3, pp 427-443.

Sunkel, G., Trucco, D. (2012). Las tecnologías digitales frente a los desafíos de una educación inclusiva en América Latina: Algunos casos de buenas prácticas, CEPAL: Santiago de Chile

Stevenson Committee (1997). Information and Communications Technology in UK Schools: an independent inquiry, London: The independent ICT in School Commission.

Valverde-Berrocoso, J., Fernández-Sánchez, M. R., & Garrido-Arroyo, M. del Ć. (2015). El pensamiento computacional y las nuevas ecologías del aprendizaje. RED - Revista de Educación a Distancia, 46(46), 1–18. http://doi.org/10.6018/red/46/3

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33. http://doi.org/10.1145/1118178.1118215

Wing, J. M. (2010). Computational Thinking: What and Why? Thelink - The Magaizne of the Varnegie Mellon University School of Computer Science, (March 2006), 1–6. Retrieved from http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

TERCE (2015). Tercer Estudio Regional Comparativo y Explicativo (TERCE). OREALC/UNESCO: Santiago. Disponible en < http://www.unesco.org/new/es/santiago/terce/>

Appendix

In March and April 2016, Microsoft Latinoamérica organized three meetings with experts and organizations involved in the promotion of computer science among youth in the region. The meetings were held in Buenos Aires (Argentina), Santiago de Chile, and Mexico City. The conversations held during these meetings has been an important resource for preparing this document. Below are the participants of each meeting.

Argentina

Alicia Bañuelos – Ministra de Ciencia y Tecnología de la Provincia de San Luis *(Minister of Science and Technology of the Province of San Luis)*
Pedro Robledo – Subsecretario de Juventud de la Nación – Ministerio de Desarrollo Social de la Nación *(National Under-Minister of Youth - National Ministry of Social Development)*
Camila Crescimbeni – Directora Nacional de Juventud – Subsecretaría de Juventud de la Nación – Ministerio de Desarrollo Social de la Nación *(National Director of Youth – National Under-Ministry of Youth – National Social Development Ministry)*
Silvia Carranza – Presidente - CILSA *(President – CILSA)*
Laura Wyerzlo – Directora Ejecutiva - CILSA *(Executive Director – CILSA)*
Karina Cuzzani – Coordinadora Académica - CILSA *(Academic Coordinator – CILSA)*
Pablo Godfried – Director - ComunidadIT *(Director – ComunidadIT)*
Mariel Sabra – Especialista FOMIN - IDB/MIF *(FOMIN Specialist – IDB/MIF)*
Alberto Croce – Director Ejecutivo - Fundación SES *(Executive Director – SES Foundation)*
Alejandra Solla – Directora Adjunta - Fundación SES *(Adjunct Director – SES Foundation)*
Natalia Jasin – Responsabilidad Social Empresaria y Sustentabilidad – Intel Argentina *(Corporate Social Responsibility and Sustainability)*
Mariela Relman – Directora - Chicos.net *(Director – Chicos.net)*
Marcela Czarny – Directora - Chicos.net *(Director – Chicos.net)*
Aníbal Carmona – Presidente - CESSI *(President – CESSI)*
Norberto Capellán – Presidente - CICOMRA *(President – CICOMRA)*
Agustín Dellagiovana – Director Nacional – Subsecretaría de Responsabilidad Social - Ministerio de Desarrollo de la Nación *(National Director – Under-Ministry of Social Responsibility – National Social Development Ministry)*
Fifi Palou – Dirección Nacional de Responsabilidad Social - Ministerio de Desarrollo Social *(National Directorate of Social Responsibility –Ministry of Social Responsibility)*
Ludovico Grillo – Secretario de Educación – Municipalidad de Vicente López *(Secretary of Education – Municipality of Vicente López)*
Gustavo Cucuzza – Adicra

Melina Masnatta – Coordinadora de Investigación en Educación – Centro de Implementación de Políticas Públicas para la Equidad y el Crecimiento (CIPPEC) *(Center for Implementation of Public Policies for Equality and Growth (CIPPEC))*
Adrian Escandarani – Docente – Escuela ORT *(Teacher – ORT School)*
Diego Fernandez Slezak – Docente – Facultad de Ciencas de Exactas (Universidad de Buenos Aires) *(Teacher – College of Sciences (University of Buenos Aires))*

Chile

Nicolás Farfán - Director Nacional – Instituto Nacional de la Juventud (INJUV) (National Director – National Youth Institute (INJUV))
Eugenio Severin – Consultor en Educación y Tecnologías. Director Ejecutivo - Tu Clase mi País. (Education and Technology Consultant. Executive Director – Tu Clase mi País)
Pedro Hepp – Académico de la Pontificia Universidad Católica de Valparaíso. (Academic of Pontificia Universidad Católica de Valparaíso)
Marisol Alarcón – Directora Ejecutiva – Laboratoria Chile. (Executive Director – Laboratoria Chile)
Mónica Retamal – Directora Ejecutiva - Fundación Kodea (Executive Director – Kodea Foundation)
Carolina Rivera – Directora Ejecutiva – Innovacien (Executive Director – Innovacien)
Camila Batista – Global Network Manager – CDI
María Cristina Benavente – Investigadora - Comisión Económica para América Latina y el Caribe CEPAL (Researcher – Economic Commission for Latin America and the Caribbean ECLAC)
Eugenio Vergara – Director Ejecutivo – CDI (Executive Director – CDI)
Hugo Martínez –Director Pedagógico de Colegium/Eduinnova (Teaching Director of Colegium/Eduinnova)
Ariel Gringaus – Gerente General – Colegium (General Manager – Colegium)
Samuel Delgado - Director de Programa – Panal Chile (Program Director – Panal Chile)
Daniela Huanca – Directora de Desarrollo - Panal Chile (Development Director – Panal Chile)
Juan Luis Ramírez – Director Ejecutivo - Fundación de Vidal Rural (Executive Director – Fundación de Vidal Rural)
Claudia Peirano – Fundadora - Grupo Educativo (Founder – Grupo Educativo)
Komal Dadlani – CEO y Cofundadora - Lab4you (CEO and Co-Founder – Lab4you)
Javier Ignacio Errázuriz Araneda – Asesor – Mineduc (Advisor – Mineduc)
Marcelo Vera. - Director Ejecutivo del Centro de Educación y Tecnologías del Mineduc - Enlaces, Mineduc (Executive Director of the Center for Education and Technology of Mineduc – Liasion, Mineduc)
Rodrigo Ferrada – Socio Director – Magenta (Managing Partner – Magenta)
Paulina Rojas –Magenta Consultora (Magenta Consultant)
Macarena Badilla – Analista Responsabilidad Empresarial - Fundación AES Gener (Corporate Responsibility Analyst – AES Gener Foundation)
Daniela Trucco – Oficial de Asuntos Sociales, Division de Desarrollo Social, - CEPAL (Officer of Social Affairs, Division of Social Development, CEPAL)
Cristóbal Letelier – Periodista – Instituto Nacional de la Juventud – INJUV. (Journalist, National Institute of Youth, INJUV)

México

René Asomoza – Director General, ILCE (General Director, ILCE)
Miguel Angel Cardona – SEP @prende
César Santos – SEP @prende
Pilar Muñoz – SNTE
Alfredo Martínez de la Torre – Director, ANUIES
Héctor Bernal – Director, Eural
Carlos Astengo – Director, Tec de Monterrey
Lorenzo Valle – Director de Vinculación, Tec de Monterrey (Outreach Director, Tec de Monterrey)
Juan Miguel Pérez Rangel – Fundación Proacceso (Proacceso Foundation)
Rosario Jessica Diaz – Fundación Proacceso (Proacceso Foundation)
Kristian Salazar – Gerente de Fomento y Promoción, SCT – AEM (Promotion Manager, SCT – AEM)
Ignacio Meza – Director de Instituciones y Cultura, Lazos (Director of Institutions and Culture, Lazos)
Josue García Dávila – Líder de proyecto, Grupo Carso (Project Leader, Grupo Carso)
Angélica Mora – Directora General, Executive Global System (General Director)
Juan Carlos Martínez – Coordinador de Tecnologías Aplicadas, CONALEP (Coordinator of Applied Technologies, CONALEP)
Dr. Carlos León Hinojosa – Director General, CONOCER (General Director, CONOCER)
Humberto Merritt Tapia – Director, IPN
Ana Tamez – UT Santa Catarina
Ulises Beltrán, IMJUVE
Juan Carlos Rico Campos, Director de Bienestar y Estímulos a la Juventud, IMJUVE (Director of Wellbeing and Stimulus for Youth, IMJUVE)
Milagros Fernández – Director Ejecutivo, CLASE (Executive Director, CLASE)
Trust for the Amerias – Sergio Pérez
Daniela Rivera – Directora de los Puntos México Conectado, SCT (Director of Access Points, México Conectado, SCT)
Catalina Demidchuk – Directora de Vinculación, Codeando Mexico (Outreach Director, Codeando Mexico)
Miguel Salazar – Director General, Codeando Mexico (General Director, Codeando Mexico)
Rafael Linares – Supera Mexico

Esta versión se terminó de editar en agosto de 2016.