INSTITUT DE FRANCE
Académie des sciences

Teaching computer science in France

Tomorrow can't wait

Report of the
Académie des Sciences
(French Academy of Sciences)

May 2013

Report of the Académie des Sciences

Teaching computer science in France

Tomorrow can't wait

MAY 2013

Foreword

The following report on the teaching of computer science was prepared by a working group of the *Académie des Sciences* (French Academy of Sciences) within the scope of its *Comité sur l'enseignement des sciences* (Committee on Science Education). It was presented on 6 March 2013 to the *Comité restreint de l'Académie* (a select committee made up of members of the Academy Bureau, delegates to sections of the Academy and members elected by the plenary assembly), and then, after a favourable opinion by the latter, on 9 April 2013 to its *Comité secret* (Secret Committee, a plenary assembly of Academy members).

This report deals with the place of computer science in primary and secondary education, as well as teacher training. It also briefly evokes higher education (prep schools and universities), a subject that would be worth developing further.

The working group is made up of Academy members, researchers and teachers: Serge Abiteboul (Académie member, Inria), Jean-Pierre Archambault (of the EPI association for public education & computer science), Christine Balagué (Institut Télécom), Georges-Louis Baron (Université Paris Descartes, Paris), Gérard Berry (Académie member, Collège de France), Gilles Dowek (Inria), Colin de la Higuera (Société Informatique de France and Université de Nantes), Maurice Nivat (Académy member), Françoise Tort (École Normale Supérieure de Cachan), Thierry Viéville (Inria). Gérard Berry serves as president and Gilles Dowek as secretary.

We would like to thank Academy members Marie-Lise Chanin (CNRS), Jean-Pierre Demailly (Université Joseph-Fourier, Grenoble), Pierre Encrenaz (Observatoire de Paris), Jean-Pierre Kahane (Université Paris-Sud), Pierre Léna (Université Paris-Diderot), Odile Macchi (CNRS) and Alain-Jacques Valleron (Université Pierre-et-Marie-Curie, Paris), as well as Michèle Artigue (Université Paris-Diderot) and Jean-Pierre Raoult (Université Paris-Est), for their comments and contributions regarding the preliminary versions of this report. It is important to note that an expert's contribution to this report does not necessarily mean he or she agrees with all of its conclusions.

Table of contents

Summary

The considerable impact of computer science in an increasing number of fields in industry, communications, leisure, culture, health, science, and society in general is universally recognised. We now speak of a "digital world" in the broadest sense, based on two powerful levers: computer hardware and computer science.

Computer science

- The development of digital applications is intimately linked to progress in computer science, which has become an autonomous field with its own ways of thinking and acting. While the applications and digital objects evolve at a rapid pace, computer science remains founded on a stable and homogeneous set of concepts and knowledge.
- Many of the most striking technological advances in recent years are the direct result of computer science: search engines and processing of massive amounts of data, large-scale networks, reliable computers built into objects, etc.
- Due to the universal nature of its object, computer science interacts closely with almost every other field of science. It is not only an auxiliary in calculations, but also contributes new ways of thinking.

The current situation

- Computer science is increasingly important in the creation of wealth and jobs throughout the world, whether directly in the computer industry or in fields that rely on it heavily such as aeronautics, the car industry, and telecommunications.
- In the computing field, Europe and France in particular are far behind, both conceptually and industrially, compared to more dynamic countries such as the United States and certain Asian nations. This situation is partially due to shortcomings in the teaching of computer science, which has been stagnating or limited to learning how to use basic products. Teaching limited in this manner cannot allow our country to shift from a nation of consumers of what is produced elsewhere to one that contributes to creating tomorrow's world.
- In France, awareness of the need for teaching computer science as a scientific discipline is growing. In 2012, a new Informatics and digital sciences optional teaching has been introduced in the final science-oriented year of secondary school; il will be extended to all other final years in 2014. The digital road map presented by the government in March 2013 insists on learning the uses and role of digital applications in pedagogy, but also explicitly cites the importance of teaching computer science. We can observe a similar awareness in other European countries.
- Circumstances are quite favourable for introducing real education in computer science: pressure from industry, which lacks personnel with computer science skills, natural attraction of students for all things digital, which are now part of their everyday environment, the possibility of creating applications in varied and attractive fields, excellent adaptation to online learning, which is developing everywhere, and development of a clearer understanding of what a curriculum should include in this area with the help of researchers.

Teaching computer science

- Teaching should address all citizens, so that they can understand the mechanisms and thought processes of the digital world that surrounds them and on which they depend.
- It should also address more thoroughly all those destined to create, adapt or simply use computer-related objects, in whatever field they work in.
- Particular attention should be paid to making computer science attractive to both genders as the field remains mostly masculine.
- Teaching should start at the primary level, through exposure to the notions of computer science and algorithms, made possible by highly varied examples like those used by the programme *La Main à la Pâte*. Teaching should be further developed in middle and secondary school.
- We can distinguish three main phases:
    - *Exposure*, mainly at the primary level, which can be done in a complementary manner by using computers or "unplugged" methods. Abundant and high-quality teaching materials are already available.
    - *Acquisition and autonomy* should start at the middle-school level and deepen understanding of data structures and algorithms. Initiation to programming is a mandatory phase that offers access to creative activities and therefore autonomy.
    - *Advanced training* should mainly take place at the secondary-school level, with in-depth lessons on basic notions and the widest possible range of experiments.

Teacher training

- Teacher training is a top priority. The government's road map proposes massive training of teachers in the uses of digital technologies, but it fails to specify anything in terms of training in computer science. This project needs to be defined and undertaken as soon as possible.
- For primary school teachers (*professeurs des écoles*), who are destined to expose pupils to computer science, training in concepts and basic examples should take on the form of a dedicated module in newly created teachers' colleges (*Écoles Supérieures du Professorat et de l'Éducation* or ESPE) .
- For middle-and secondary-school teachers, the Academy strongly recommends qualifications and recruitment methods aligned with those in other secondary-school disciplines, which are also likely to evolve as the Academy has already recommended.
- All teachers should be trained in the impact of computer science on the evolutions of their discipline: simulations in experimental sciences, use of databases in history and geography, analyses of texts in literature, automated translation, artistic creation, etc.

Recommendations

In all these recommendations the word computer science refers to the concepts, science, and techniques of computing, as explained in the full report, and is therefore not limited to the simple use of hardware and software.

Deciding to teach computer science

The essential decision consists in implementing a programme in computer science from the primary to the secondary-school level, oriented towards understanding and mastering computer science, and therefore going far beyond simple use of hardware and software. This implementation can no longer be delayed.

To achieve this it would be best to proceed, for each point, with an experiment on a sufficiently large scale and over a brief period, bolstered by solid support and whose extension would be the object of a multi-year strategy.

Programmes

Primary school:

- Include in primary school programmes an initiation to the concepts of computer science. Starting at this level, combine plugged in and unplugged activities.

Middle school:

- Introduce real teaching in computer science that is not diluted in other scientific and technical courses, but developed in cooperation and within an interdisciplinary approach.

Secondary school:

- Propose mandatory computer science courses for beginning secondary-school students.
- Make computer science mandatory in the second and final year of scientific programmes without ruling out the option of more in-depth specialisation in the final year.
- Propose optional computer science courses in the second and final years of literary and economic-social sciences programmes.
- Continue and develop courses in computer science within technical and professional programmes.
- Examine the balance of hours in various disciplines required by the introduction of computer science with, on the one hand, a specific time for computer science and, on the other hand, consideration of computer science content in other disciplines and their programmes.

An inventive coordination of disciplines cannot be based on this report alone. However, this must not serve as an alibi for a new delay in introducing computer science, which would constitute a major handicap for our country and our overall educational system.

Higher Education:

- For *Classes Préparatoires aux Grandes Ecoles* (post-Baccalauréat preparatory classes), or CPGE, increase the number of hours dedicated to computer science. The current volume of two hours in the first year and one hour in the second year is not sufficient to cover the cultural and professional needs of students in these classes.
- Develop specific courses in computer science culture for all students in Bachelor's and Master's degree programmes, particularly for those destined to become teachers.

Teacher training

- Include computer science in initial training of primary school teachers and train current teachers through aggressive professional development programmes so they can initiate their students in this discipline.
- In middle school aim for specific courses in computer science, led by teachers meeting skill and diploma requirements as stringent as those for other middle-school science programmes.
- In secondary schools, recruit teachers in computer science meeting skill and diploma requirements as stringent as those for other secondary-school science programmes.
- Envisage the inclusion of computer science in the teaching of traditional disciplines, in humanities as well as sciences.

Context and positioning of this report

This report extends the specific work conducted by researchers, teachers and *inspecteurs généraux* (education system general inspectors) on the teaching of computer science since 2008, when the reintroduction of computer science in secondary schools became an explicit goal of the French Ministry of Education. This collaborative work on the principles and programme started within the framework of the educational reform of 2008 and continued when the decision was made, shortly thereafter, to reintroduce secondary-school computer science courses in the autumn of 2012 in the form of a specialised curriculum, Informatics and Digital Sciences (ISN), for the final year of scientific programmes. This report also pursues the work of the French Academy of Sciences, which has already released partial opinions on the introduction of computer science in education, particularly reports published in 2007[1] and 2010[2]. In its pillar 3, the 2006 government text regarding core knowledge and skills also included paragraphs on understanding electronic and digital processing of information, following a proposal by the *Académie des sciences*.

Note: *This report was written before publication of the road map defined during the governmental seminar on digital technology in March 2013. This road map insists on the increasing importance of the digital industry and the role of digital technology in many businesses and the creation of new jobs. This report also insists on the importance of a true familiarity with computer science among the general population, as well as the need to train many more competent professionals in the field. Finally it indicates that discussions on the role of computer science at all levels of the educational system will soon be organised.*

A digital world created by computer science and techniques

The development of computer science and the systematic digitisation of all sorts of data profoundly affect society and its activities, leading to the creation of what is now broadly called the "digital world". The specific nature of digital activities and industries is clear: they consist in manipulating immaterial information that is very different from material objects. A computer program or a piece of data weighs nothing at all, is transported at little or no costs and can be infinitely duplicated. Yet in our country not everyone realizes three truths that the *Académie des sciences* (Academy of Sciences) insists on asserting and which this report develops in detail:

- The route towards a digital world is based on joint progress in computer science and techniques.
- Computer science has become an independent discipline with its own schools of thought and results.
- While it is indispensable and helps reduce the digital divide, education in digital practice exclusively through the use of software, computers and networks does not contribute much in terms of education in computer science.

We should note that the government now recognises explicitly that neglecting the importance of computer science in education would endanger the future of our country. This recognition translates into a proposed law for reorganising schools (*Loi pour la Refondation de l'École*) presented to the Council of Ministers on 23 January 2013, which entrusts schools with a new mission: "digital education". But it does not provide details and therefore does not specify the fundamental difference between use, science, and technique. The digital road map presented afterwards by the government in March 2013 explicitly mentions starting discussions on the role of computer science in education. This question indeed seems essential and will be the subject of this report.

[1] Report by the *Académie des sciences* (Academy of Sciences) on teacher training and the teaching of science, http://www.academie-sciences.fr/activite/rapport/avis131107.pdf.
[2] Report by the *Académie des sciences* (Academy of Sciences) on reforming secondary schools, 2010, http://www.academie-sciences.fr/activite/rapport/avis251110.pdf.

Vocabulary notes

First we need to specify the vocabulary we will use, as new acronyms are constantly cropping up to designate computer-related activities: ICT, NICT, ICST, etc[3]. The term "electronic" is still present in terms like "e-mail" and "e-commerce", etc. while "digital" is clearly gaining ground and used in a growing number of fields. In this report we will use the following definitions:

- "Computer science" will specifically designate the science and technique of processing information and, by extension, the industry dedicated to these subjects.
- The adjective "digital" can be associated with any activity based on digitized information: digital photography, digital sound, digital publishing, digital sciences, digital art, etc.

Thus we can speak of a "digital world" to express the use in an increasing number of activities of digitized information and a "digital economy" for all economic activities linked to the digital world and "digital" alone for all activities to which the adjective can be added. Since digitized information can only be processed via computing, computer science is the conceptual and technical driver behind the digital world.

About this translation

The French words *Informatique* does not translate well into English. Compared to English, *informatique* covers computer science but also other topics that deal with the processing of digital information, such as what is often called information science and technology, as well as the computing industry and commerce. We have nevertheless chosen in this report to translate *Informatique* by computer science when talking about education and research, and by computing when the context is that of industry. The situation could become simpler in the future, since the English word Informatics is gaining traction in international Academia. The French word *Numérique* has the same meaning as digital in English: *société numérique* is adequately translated by digital society.

Computer science at the heart of digital technology

Computer science is the object of considerable scientific research throughout the world, ranging from abstract research on algorithms and fundamental calculation structures to the design of innovative hardware and software used everywhere. This report is dedicated exclusively to the teaching of computer science and techniques within the educational system. Our aim is to demonstrate that teaching computer science at a much earlier stage has become a necessity for all, obviously at different levels depending on whether we are talking about training engineers or ordinary citizens.

General education in computer science should first provide all citizens with the keys to their future world, which will be even more digitized and therefore computerized than it currently is. This is necessary for them to understand its decisions and evolution and consciously take part in them rather than simply endure them and consume what is made and decided elsewhere. Computer science teaching must also prepare people for new sectors rich in employment opportunities engendered by computer science. For engineers and scientists it must provide knowledge, adequate scientific practice, and appropriate techniques so they can make the most of all computing tools at their disposal as well as adapt these tools to their own sectors and even create new ones. To various degrees it will therefore be necessary to educate everyone concerned by constant and rich interaction with computerised systems, whatever their profession or field: literary, scientific, artistic, or other. The corresponding curricula in computer science should be designed according to the diversity of these interactions and with the actors concerned, but without losing sight of the profound unity of its concepts: computer science does not differ according to disciplines and cannot be limited to certain

[3] Information and Communication Technologies, New Information and Communication Technologies and Information and Communication Science and Technologies.

disciplines, even if it is presented differently in different areas. Both its general concepts and specific forms need to be taught in a harmonious manner.

Teaching computer science, beyond the use of digital technologies

What this report proposes:

- A discussion of the nature and impact of computer science
- Proposals regarding the purpose of teaching computer science
- A skeleton curriculum for the primary and secondary levels
- Elements regarding the training and status of primary-, middle- and secondary-school teachers
- An overview of education in computer science around the world
- Recommendations for the 2020 horizon

What this report does not cover: It does not pretend to cover all the questions regarding the digital world, nor evolutions in teaching methods made possible by new digital tools in all subjects. It does not propose a detailed analysis of higher education.

What this training should not be: Some continue to believe that scientific training in computer science is not necessary, that familiarisation with the basics is enough and that, if need be, computer science can be learned "on the job". This opinion leads to a dangerous misunderstanding.

Of course, familiarisation with various uses is a must. This is simple and natural for young people in the 21st century for whom computing is nothing new since they were born with the technology. The B2I[4] qualification in France contributes to this. However, computer science has undeniably become much more than a provider of tools that can be used without too much thought. On the contrary, it has become a vast space for scientific, technical, industrial, and commercial creation, as well as one of the fields that creates the most direct or indirect jobs in the world, as observed in the report by LévyJouyet[5]; numerous reports by the CIGREF[6], OECD[7], and other bodies; and the government road map published on 28 February 2013[8]. The corresponding wealth is built up by those who create and contribute to progress in the field, not by those who only consume its products. While one can become a savvy consumer through immersion in a digital society, creation is necessarily based on real skills in computer science.

A major stake: software

In the 20th century many useful and innovative programmes were created by knowledgeable amateurs, some of them self-taught, which has contributed to the myth that computer science can be developed in a garage. However, applications that create value, and which we will discuss further on in this report, are much more ambitious and complex. They are created by solid teams of highly-trained and well-equipped people, whether in industry or in society for the development of free software.

We should note that in the 1980s France presented itself as a major industrial player in computing, particularly software. This is far from the case today, even though French research has remained on a high level internationally and even if France and Europe remain leaders in important niches like embedded systems for transportation, industry, and objects. France remains competitive in terms of SSII (consultancy service companies), but is weak in software publishing with only one French

[4] *Brevet Informatique et Internet* (basic training in computer tools and Internet)
[5] http://www.ladocumentationfrancaise.fr/var/storage/rapports-publics/064000880/0000.pdf
[6] *Club Informatique des Grandes Entreprises Françaises* (Computing Club of Large French Companies)
[7] Organisation for Economic Co-operation and Development
[8] http://www.gouvernement.fr/sites/default/files/fichiers_joints/feuille_de_route_du_gouvernement_sur_le_numerique.pdf

company in the top 100 international firms. At a time when industrial reconstruction is a preoccupation, and above all focused on floundering firms, it is essential to observe the unfortunate weakness of our industry compared to the extraordinary international expansion of the field, as well as the large number of French computing firms that fall into foreign hands due to their success, which is not a very good sign for the nation's future in terms of industrial health and independence. We believe that our current decline in an area that is so fundamental for the future is in large part due to late and limited recognition of the discipline in prestigious engineering schools, secondary education and in French society generally speaking. Not so long ago it was commonly believed that computing was not a science and therefore did not need to be taught; it was sometimes even considered a passing fad.

There is something wrong with the way we teach computer science and even more so in the fact we do not teach it at all.

The situation can be reversed by implementing real education in computing as a science rather than just a usage, in keeping with the traditional high standards of science education in France, as well as with the place of science starting at the primary level, which has been progressively reinstated since 1996 at the initiative of the Academy (see *La Main à la Pâte*).

The problem of recognising the need for real education in computer science is not limited to our country. It is increasingly the subject of debate, with similar arguments as those developed here, and many countries have already launched initiatives on the subject. Notably the United Kingdom has decided to introduce computer science in their equivalent of the Baccalaureate, on equal footing with other sciences[9] (see the section of this report on computer science around the world for other experiments).

This report contributes to raising international awareness. The originality of our committee is probably the participation of teachers and researchers who are highly involved in educational issues and the dissemination of science and who have already taken part in defining computer science curricula and writing associated textbooks.

The nature and impact of computer science

A ubiquitous science and technique

In the beginning of the 21st century computer science has radically changed many fields of human endeavour, from the transmission of knowledge to business management, from artistic creation to government administration. While computing in the 1980s essentially concerned scientific calculation, business management and telecommunications, current computing is involved in all forms of communication between people, leisure, most dimensions of industry, from the design of objects to their manufacture, commerce, transportation, a large part of the service sector and, just as well, science, the humanities, health, and care for dependent persons. For example, everyone can observe the impact of the Internet on communications between people, access to intellectual works and the transmission of knowledge. The fact that a huge amount of data is available on line and can be analysed by algorithms transforms our relationship with knowledge. We should note that the Internet has been, since its début, a resolutely computerized network based on the distributed IP protocol, which is very different from traditional centralized telecommunications technologies that could never have been expanded to such a scale.

Up until recently the Internet was only accessible via desktop and laptop computers. But 21st-century

[9] http://www.bbc.co.uk/news/education-21261442

computers have taken on different forms, diversified and are integrated everywhere. Telephones and tablets connected to the Internet have rapidly replaced traditional computers, associating limitless mobility with touch screens, sound, video, novel functions, and a multitude of new applications using, for example, geolocalisation. Onboard computers in planes, trains, cars, and other everyday objects are even more numerous. These invisible computers are linked to increasingly varied forms of sensors and actuators. An automobile, for example, contains an impressive amount of computer equipment and software to assist in driving, braking, reducing fuel consumption, etc. Functions that were once mechanical, like control of the engine, steering, and suspension system are now computerised. The difference between one model and another often has to do with the onboard computer system. Soon cars will be constantly connected to the Web and collect data directly from the road, the city and a wide array of sensors in order to ensure greater safety and more fluid traffic conditions. As is the case for planes, automobile design is entirely assisted by computers, using geometric modelling and simulation tools. This is why automobile manufacturers are hiring just as many computer scientists today as mechanics.

Sciences that have gone digital and computer science

All natural sciences have now adopted computer science as a fundamental component in research, for example by designing new sensors and instruments for observation, together with experimentation that combine physics and computer science on a high scientific and technical level: telescopes, particle accelerators, medical imagery devices, DNA sequencers. Yet in science, computing is much more than a calculation tool. It leads to a different way of thinking, called "computational thinking", in which algorithms have the same importance as equations. Computational thinking also applies to life sciences where equations are sometimes lacking. Thus scientists in bioinformatics model life forms with algorithmic tools that allow them to more perfectly describe and even simulate life processes to understand them better. In medicine scientists are starting to model the dynamic functions of organs. Surgery is performed using augmented reality and intelligent prosthetics for hearing, sight, and movement are being developed. According to experts in experimental cognition, our brain has certain algorithmic aspects: once a purpose is defined, it searches for the means to reach it, gathering the necessary information and materials, and then executes a series of elementary physical or mental tasks to achieve it.

The links between mathematics and computer science are numerous and profound. Like physics, computer science relies heavily on mathematics and develops them for its own needs: logic, combinatorics, number theory, statistics, etc. Likewise, mathematicians use computer visualisation in geometry or dynamic systems. Hales' proof of the Kepler conjecture involves very large computations. Experts in applied mathematics and computer science work together to make digital and formal calculation tools use in a considerable number of scientific and industrial applications. These tools deal with difficult problems of physical data representation and approximation. Finally, new logics and algorithmic approaches to formal proofs have been applied in proving major mathematical theorems in a full formal way: recent examples are the four colour theorem, the Feit-Thompson “monster theorem” on odd-order group classification. The very same tools are used in computer science to formally verifying of circuits, compilers, and operating systems.

Social sciences are also concerned: philologists, historians and lawyers use computers to archive and index large corpora, but also to assemble fragments of statutes or manuscripts in order to reconstruct them. Computer science is gaining ground in the arts, with new ways of creating and disseminating the written word and images, generating computerised calculations in architecture, and new techniques for direct manufacturing of complex objects like 3D printing. Musicians use software to invent new digital instruments and new composition tools. Finally, athletes commonly use complex computerised instruments to improve training and increase performance. In these fields as well, we can observe the advent of computational thinking.

Automatic analysis of large volumes of data, which originated with databases and search engines, now has applications in a wide variety of fields such as genome biology, epidemiology, climate studies, journalism, customer analysis, etc.

Computer science is also a major provider of direct and indirect jobs around the world, with a constantly expanding economic network ranging from start-ups to service companies and large industrial empires. This expansion is particularly strong in North America and Asia. France and Europe have remained active, but their roles are somewhat limited, except in specialised sectors like embedded systems, telecommunications, computer-assisted design and video games. Entirely new ideas, such as search engines, social networks or smartphones, did not originate in Europe, where creativity is less important than in the USA and Asia.

A unified discipline: computer science

The wide range of effects of computer science we have described above does not mean there are several forms of computer science. On the contrary, it is the computer's absence of application domain specificity that explains its ubiquity. In fact computer scientists refer to them as "universal machines". While computing is a field where creation is fast and products are quickly obsolete, this creation and these products are based on fundamental computer science knowledge and stable know-how, relying on broad unifying concepts. Hence the recent specialised programme in French secondary schools, *Informatique et sciences du numérique* (Informatics and digital sciences) is organised around four concepts and their interactions: algorithm, language, information, and machine. These concepts existed[10] prior to computer science: the term algorithm, for example, comes from the name of the great Persian mathematician Al-Khawarizmi, who also invented algebra, and the notion of the algorithm has been used in mathematics throughout the ages. However, computer science has systematised and considerably extended these concepts.

To explain the generality of algorithms, we should bear in mind that many questions we face start with the word "How": How do we perform an addition? How do we recognise a face? How do we know a sentence is correct? How do we manage to walk? How do we know our geographical location? The answer to such questions is often an algorithm, whether it is performed by a human or by a machine: an algorithm for additions taught at school or performed by a circuit, a shape recognition algorithm, a grammatical analysis algorithm, a neuronal or robotic algorithm for walking, reading a map or a GPS algorithm that calls on general and special relativity for orientation. In computer science, algorithm design is based on a relatively limited number of major principles, even if the number of known and studied algorithms in now countless.

An algorithm is a conceptual object, like a number. In order to be materialised, a number must be written in a specific language, for example in positional notation, perhaps within a phrase written in natural language. In computer science, programs that incarnate algorithms must be uniformly formal and precise as they must be understood by both men and machines, the latter being infinitely stupid. The computer languages in which programs are written are therefore very different from natural languages as they are simpler, more specialised and less ambiguous. However they are also based on a limited number of concepts that can nevertheless be combined in various ways, hence the existence of multiple computer languages today.

Unlike human discourse, where the listener is supposed to understand the unsaid, expressing an algorithm in a program is a delicate operation as it can generate many bugs that can trigger erratic and potentially dangerous behaviours. The art of programming consists in writing programs that are scientifically organised so that they are correct and their correctness can be verified. Furthermore we can say that the starting point in computer science was the advent of the first high-level programming languages such as Fortran, Lisp and ALGOL, rather than the first computers. These languages allowed the accumulation of programs and their exchange, the reuse of a programe by others, its

[10] Programme available at the following address
http://www.education.gouv.fr/pid25535/bulletinofficiel.html?cidbo=57572

portability between different brands of computers, and the transformation of programs themselves into objects of study.

The concept of information, or data, is central in our way of thinking. In everyday life we are immersed in a sea of information. We need to learn to organise, sort and confirm this information, question its source, update and extract knowledge from it. We need to distinguish between the information we have and that which is lacking, but that we can search for or speculate about. Computer science simplifies these problems by providing new means of gathering, representing, organising and searching for information on a large scale. However it also exacerbates problems related to mastering information.

Finally there is a continuity in the history of machines that manipulate information, which is part of the general history of machines. It has led from mechanical to electric calculators and then to modern electronic computers. A uniformity of concepts also applies to these computers. Whether large or small, the circuits of these machines are all designed and manufactured according to the same principles, using computer assisted design software and simulation tools.

The fundamental character of computer science concepts has led some to propose a slogan for the children of this century, by adding a single word to the one already suggested by the *Académie des sciences* (Academy of Sciences):

Learn to read, write, count, reason, and program

The purpose of teaching computer science

Education in computer science should allow all students - including those who are not destined to become computer scientists - to understand and master the digital world around them and to access the new ways of thinking that come with the development of computer science. It should serve people in both their professional lives and their roles as citizens.

Training professionals in every sector

There are three professional sectors that are particularly concerned by computer science. First, there is of course the computer industry that is a major provider of qualified jobs. Then, computer scientists are massively employed by organisations (industries, administrations, etc.) to develop high-end applications. Finally, in a large group of other professions (scientific, technical, commercial, legal, etc.), people are in constant contact with sophisticated computerised objects.

In computer science, France and Europe have very high quality research and, as we have already pointed out, remain leaders in certain major fields. Nevertheless it is recognised that the French and European industrial fabric is lacking in vigour, with a small number of international leaders and frequent acquisitions of the most dynamic new French firms by foreign ones. This cannot be explained by industrial and financial factors alone. It also results of cultural phenomena, the first being a lack of understanding regarding the stakes of the computing industry among our elites and engineers, who often have not benefited from sufficient training in computer science.

Regarding computing in industry, and not only in the computer industry strictly speaking, industrialists in many western countries are regularly warning there are not enough well-trained computer scientists. It is surprising that there are also computer scientists who are unemployed. However, these are often "old-school" programmers, trained on the job, and who have difficulty adapting to changes in the sector due to weak understanding of its fundamental concepts. This offers yet another example of why the educational system can no longer turn a deaf ear regarding demand for real training in computer science that allows people to adapt to rapid changes in the sector. It is also an additional reason to start such training at an early age, to expose children early to the appropriate way of thinking about informatics and build their critical sense.

For other businesses where people are constantly in contact with computer science - engineers, managers, salesmen, architects, etc. - the construction of real "computational common sense" is essential and can only be the result of real training as well. To quote the words of Louis Becq in his article *Contre l'Illettrisme Numérique en Entreprise* (Against digital illiteracy in the firm)[11]: "Undoubtedly industry needs more computer scientists, and increasingly well-trained and professional ones. [...] However, industry also needs, and just as much, personnel that has acquired and assimilated a general computer science culture allowing them not only to dialogue effectively with their computer scientist colleagues, but also to distance themselves sufficiently from their tools in order to be active vectors in their improvement and performance". For example, jurists are increasingly required to write and apply laws that concern questions linked to computer science. Recent experience has shown that their difficulty in understanding and following the subject leads to constant delays in terms of action and even contradictions as we have seen for questions involving identification, electronic voting or privacy issues: the widespread and erroneous belief that an IP address identifies a computer (within Internet Protocol version 4 or IPv4), blind confidence in electronic voting, which is certainly not shared with specialists in computer security, etc.

Whatever the profession, it is important to abandon the traditional idea of using applications whose elaboration is delegated to computer scientists. Let us take two final examples. A scientist designs an experiment and needs to process the data, but not only by adapting existing software. The way the data is processed is in fact part of the experimental design. A young man who uses a telephone does not want to wait until he has finished his studies in computer science to develop and put the applications he has invented on line. All these actors must gain their independence in the digital world.

Preparing citizens

Computer literacy for all

The "digital divide" is often restricted to the gap between those who possess a computer and Internet access and those who do not. This partial vision focused on equipment is highly limited and can lead to inadequate solutions in terms of education. For example, giving each first-year middle-school student a computer or equipping each classroom with an interactive whiteboard, as is the case in many French departments and regions, is only an ineffective way to deal with a real problem if this "gift" is not associated with educational support.

Once they are adults, these students will actually have to cope, more often than not, with many changes brought about by computers in many areas rather than a lack of machines and these changes will affect them directly. The real divide will be between those who possess the intellectual tools to understand these changes and those who do not. For example an unemployed person will have to register with the unemployment office via a computer. It will be easier to find one than to learn how to use it.

The transformations brought about by computer science carry a message of hope, one of greater transparency and proximity, of more possibilities for every person to express himself and to access information and education. However these changes can also affect our individual freedoms. Therefore they need to be accepted and controlled by citizens and society. This control will not be possible without a minimal understanding of the concepts involved. Currently this understanding is far from widespread; it can only result from a real educational programme. An in-depth study of the arguments put forward in recent debates reveals the necessity of such an educational programme: irrational confidence or wariness of electronic voting, whose strengths and weaknesses are poorly

[11] http://www.lemonde.fr/sciences/reactions/2012/12/06/contre-l-illettrisme-numerique-en-entreprise18012581650684.html

assessed, concerns regarding new ways of disseminating intellectual works, often seen exclusively from the angle of pirating, absence of control over the sharing of personal data, often collected without informing the persons concerned, difficulty for institutions in measuring the impact of allowing citizens to express their opinion on any and all subjects, the impact of screens and various computerised objects on the physical and mental health of young people[12], etc.

Reducing digital divides between genders and social classes

Gender equality in understanding computer science and the digital world is essential. While women were quite present in the early days of computer science and up to the generation trained in the 80s, today we can observe a marked inequality between the sexes in French Grand École and university degree programmes, which has resulted in the current proportion of around 15% in computer science professions[13]. This inequality does seem to be diminishing over time. Furthermore, we know that social differentiation between girls and boys increases with age, in particular their taste for scientific subjects. Reducing this type of inequality is not possible without constant and early efforts in education. For us this is an essential reason to teach computer science in schools.

Similarly, we must avoid broadening the digital divide between children of the affluent, who are exposed to the digital world and can, if they so wish, be trained in computer science, and others who are not so fortunate. This is another reason why it is essential that our educational system teach computer science starting at primary school level, so that access to this science is not reserved for an elite.

A favourable context

The conceptual unity of computer science and the wide variety of its applications provide a favourable context for creating an educational programme that is both conceptually well-designed and easy to implement via a wide variety of practical applications for student exercises and projects. Fields of application can be linked to other sciences, languages, the arts, history, geography, business management, government administration, etc. The wealth of available applications makes it possible to select fields adapted to the students' curricula and elective courses and that are close to their interests. From a computer science standpoint, the construction of conjugation tables or calendars represents a similar type of problem. One can teach the notion of nested loops with one or the other, but reception of these two examples can vary a great deal according to the students' centres of interest.

Another favourable factor for a quality educational programme that is interesting for students is that many of them are now highly familiar with the uses and logic of computer science. For 21st-century students, computer science is anything but a "new technology" since they have never known a world without it. A computer is no stranger for them than the sea, the mountains, a bicycle or a cat. On the contrary, some 20th-century objects are now strange. Soon history teachers will have to explain what a telephone box was used for or what a typewriter or vinyl disc is, even a CD. Moreover, computer science is often associated with the pleasant aspects of life: games, communicating with others, music, photos, videos, etc. This familiarity can be used as a means of gradually leading the youngest

[12] See the opinion published by the *Académie des sciences* (Academy of Sciences) JF Bach et al (2013), *L'Enfant et les Écrans* ("children and screens"), Ed. Le Pommier, Paris. Available online at http://www.academie-sciences.fr/activite/rapport/avis0113.htm and the *La Main à la Pâte* Foundation's *Les Écrans, le Cerveau et l'Enfant* (Screens, the brain and the child) pedagogical module.

[13] Marie-Paule Cani, *Filles et Garçons Face à l'Informatique*, (Girls and boys and computer science) in the proceedings of the symposium *Filles et Garçons en Sciences et Techniques, Un Enjeu Européen et Planétaire* (Girls and boys in science and techniques, a European and planetary challenge) co-organised by the association *Femmes et Sciences* (Women and science), the *Association pour la Parité dans les Métiers Scientifiques et Techniques* (Association for gender equality in scientific and technical professions), or APMST, and the *Mission pour la Place des Femmes* au CNRS (Mission for the place of women within the French national centre for scientific research) Grenoble, Nov. 15 2008.

students from uses to concepts. A good indication of students' appetite for computer science is the success of the *Castor informatique* competition[14] for middle- and secondary-school students.

In computer science, scientific and technical activities are often closely linked. Even in the context of a highly practical application, there is often the need for a sorting program, which will have to be imagined and written. The teacher could therefore lead the students to analyse the algorithm they have programmed, to question its complexity and to study a greatest lower bound to the complexity of sorting algorithms. The path that leads progressively from the abstract to the concrete is the essence of teaching computer science. Such teaching can play a truly positive role in developing the students' taste for sciences and techniques, whether in those who do not enjoy studying science or do not see what it is for, or, conversely, in those who are easily bored by classes in technical subjects that disregard the abstract dimension.

A final favourable factor is the emergence of MOOC (Massive Open Online Courses). Even though they involve significant investments, they offer a new lever, with strong economies of scale, and allow the emergence of cooperative communities and the development of an open pedagogy. This type of teaching seems particularly well-adapted to computer science. This report will come back to the subject of MOOC when dealing with continuing and teacher training. However, it is important to note this trend is developing rapidly and must be reckoned with over time in the teaching of computer science in general.

Some general principles

Before outlining a skeleton curriculum, we will specify some general principles that should guide the teaching of computer science, at any level.

Striking a balance between theory and experimentation: Regarding the objects studied, computer science is close to mathematics as both concern abstract objects. Regarding the method, computer science is close to natural sciences in that it relies on material objects, i.e. computers, to implement abstract algorithms. Thus it allows concrete realisations that are directly linked to reality: analysing a text or image, designing the wing of a plane in order to optimise air flow, composing, and playing a piece of music, etc. It is important to impart this dual anchoring to students by initiating them to the most abstract aspects of the discipline, while showing them how these naturally produce concrete applications, close to their personal centres of interest and their professional plans, thus linking theory and experimentation as, for example, in the *La Main à la Pâte* programme.

Linking computer science to the real world and other disciplines: One practical way of teaching some computer science is to remain focused on simple mathematical examples. However, mathematical calculations represent only a part of computer applications and are only likely to interest those students already interested in maths. The flexible nature of computing allows for a wide and varied choice of examples: textual data, images, sound and video, robots, and everyday digital systems, physical simulations, etc. All these fields are equally formative and computational thinking is essentially the same within them.

Ensuring the relevance of course content over time: One characteristic of computer science is the contrast between the stability of its basic concepts and the rapid evolutions of the objects themselves. For example, many students used to learn Lisp, a language that has now been basically abandoned. However, the notions that form the basis of Lisp - the function call, recursion, tree data structures, etc. - are fundamental notions that are obviously still in use. Thus it is important to formulate the objectives of a programming course in terms of concepts rather than languages. Even if we teach programming using a specific language, which seems inevitable, this teaching must be sufficiently general for students to learn other languages by themselves later on. Therefore we must systematically seek to teach fundamentals and concepts, instead of training students in the details of tools that will soon be obsolete, otherwise the students risk to be as useless as the objects studied.

[14] http://castor-informatique.fr

Reducing digital divides: To reduce inequalities we suggest very early training and a diversified approach to the discipline that combines abstraction and applications in the real world, in particular far from the masculine image of an uncommunicative person glued to a machine.

Going beyond usage and training "on the job": Contrary to an unfortunately widely accepted idea, training in computer science cannot be limited to its applications - word processing, spreadsheets, Web browser, etc. - for the same reason that training in thermodynamics cannot be limited to learning how to read a thermometer or a barometer, or mechanics to obtaining a driver's license. This type of training would only produce users at the mercy of techniques and not actors in tomorrow's world. Similarly, we must repeat that learning "on the job", without a gradual approach to the concepts, can only work for the use of tools designed by others. Real training is required to imagine the objects, services, and sciences of the future, as all professions that create value will lead to real innovations in computing and not only the consumption of standard tools. An architect's training, for example, should not be limited to learning how to use design software that is available today, but to acquiring skills that will allow him to invent tomorrow's applications, like Franck Gehry, whose new buildings could not have been designed without computers. Computational thinking has become a central factor in their conception.

Skeleton curriculum

At all ages and especially in primary school

Training in computer science is organised differently according to whether the courses are for primary-school, middle-school, secondary-school, or university students, but the question of the curriculum should be approached globally, as the content of courses at secondary-school level, for instance, depends on the level of the students when they begin secondary school, i.e. what they have learned in middle school. This report offers a unique opportunity to address this question of globality.

Three ways of learning

We can distinguish three ways of learning computer science that differ in their purpose: discovery, acquisition and autonomy, and mastering concepts.

Discovery: One way to have students discover computer science is to teach them how to use computerised objects and to lead them, through this learning process, to raising questions and searching for answers. Many students, for example, know how to send an email, but they do not necessarily wonder how such a message ends up in the recipient's inbox. Learning how to use an email program offers an opportunity for them to raise the question and search for the answer. This question can be introduced in an attractive way, for example in the form of a riddle. Students can look for the answer collectively. We can place the question in its historical context: the problem of sending messages is very old. This question has been raised before, for example, during the reign of Genghis Khan, whose empire stretched from Asia Minor to the China Sea. It can be used to introduce the notion of networks - computers are linked together by electric cables or radio - and routing - a message must find its way through a labyrinth made up of billions of interconnected computers.

This initiation to computer science also involves discovering the fundamental concepts of algorithms, language, data, etc. without necessarily using a computer.

Acquisition and autonomy: Understanding computer science is not restricted to using objects designed by others, but also consists in knowing how to design them oneself.

The acquisition and autonomy phase requires learning the basics of programming languages and methods. Obviously the aim is not to train programmers, but to offer students an opportunity for hands on experience, so they can understand what goes into making a computer program, an indispensable key to understanding the world around them. The objective is to teach them to move beyond the role of simple spectators and become full-fledged participants in our digital world.

In-depth study of concepts: Once a certain level of autonomy has been acquired, students can really enter the universe of computer science. They can understand how programming languages are designed, how public key cryptography works, how a database system is organised, how a computer or a network works, why certain problems cannot be solved by an algorithm or why others require more or less time to calculate. For example, consulting an index of several terabytes takes only a few tenths of a second, but decrypting a short message without knowing the key can take centuries.

These divisions in the teaching of computer science are not exclusively chronological. For example, students can learn at a very early stage how to write little programs, while perfecting their skills later by learning and understanding different computing languages. However, it is still reasonable to say that the first way of learning should dominate in kindergarten and at the primary level, the second in middle school and the third in secondary school and higher education.

Kindergarten and primary school: discovery

With programming languages appropriate for their age and in unplugged mode

In primary schools training in computer science is sometimes limited to use of a computer. However, this is a misrepresentation of a scientific discipline in which abstraction plays an essential role. "Doing computer science" does not consist in spending hours in front of a screen. Initiation in computer science in kindergarten and primary school should therefore be balanced between activities using a computer and "unplugged"[15] tasks. The children's use of computer science, as well as the fascination it exerts on them, can be mobilised to initiate them in the mysterious workings of their own brains, as proposed in the module created by *La Main à la Pâte* for the primary level[16].

With a computer Activities involving the use of a computer can start with learning about the most common software: email, search engines and Web browsers, word processing, spreadsheets, etc. As we have pointed out before, this teaching should serve as an excuse to wonder about how objects work, a line of questioning that should lead in turn to the discovery of certain computer science concepts. We have already cited the case of email. The Web is another example. While sending a page from a Web server to the school computer obeys the same principles as sending an email, a new question is raised: Where does this information come from? Whose words are these? Where is this information saved? How is it identified? It is also possible to evoke a keyword search and the notion of an index, thus lifting the veil on the mysterious workings of search engines. Pupils can even design their own Web page and become active participants on the Internet, which is the best way of understanding that anyone can say anything in the virtual world and questioning the relevance of the information one can find there. The Web can also be placed in its historical context, through a question regarding the difference between a Web page, a parchment and a printed page. For example, just like a traditional encyclopaedia, an online encyclopaedia can contain articles about dinosaurs or highly specialised information on the typology of electrical outlets throughout the world that would have been eliminated from a traditional encyclopaedia due to the number of pages required. While middle school students learn the difference between an encyclopaedia, a technical manual, a tourism guide, etc. today they also learn that an online encyclopaedia is all of that at once.

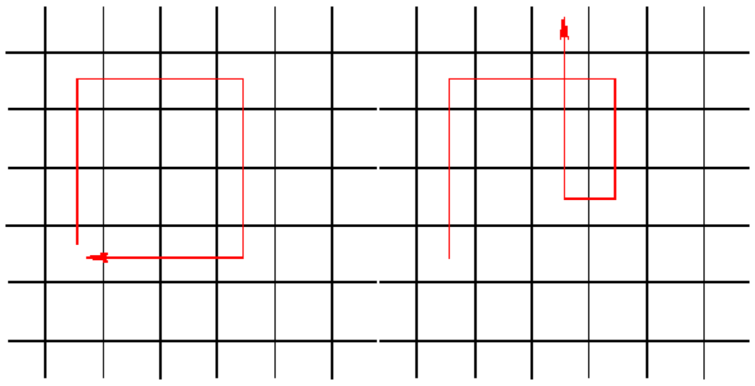
In unplugged mode: Alongside these activities, which start with the use of a computer to raise various questions, "unplugged" activities, i.e. which do not require the use of a computer, allow an approach to computer science on a more conceptual level. They aim to introduce students to three fundamental notions in computer science: language, information and algorithms, notions that are also part of "connected" activities. These highlight, moreover, the fact that concepts in computer science apply to the real world just as much as the virtual world inside computers.

A formal language is different from a natural one due to its specialisation, artificial character, limited lexicon and the simplicity of its grammatical rules. One simple example is a language formed with just four words: "North", "South", "East" and "West" and of a construction, a sequence, that is used to form series of these words. This language can be used to give directions on a square grid, for

[15] See also the opinion of the *Académie des science* (Academy of Sciences), *L'Enfant et les Écrans* ("children and screens").

[16] B Descamps-Latscha et al (2013) *Les Écrans, le Cerveau... et l'Enfant* (Screens, the brain...and the child), Ed. Le Pommier, Paris, a pedagogical module designed by *La Main à la Pâte* in support of the opinion of the *Académie des sciences* (*op.cit. supra*)

example, like the tiles of the school yard. The expression "North, North, North, East, East, East, South, South, South, West, West, West" indicates, for example, that a child should move three squares to the North, then three squares to the East, then three squares to the South and finally three squares to the West, thus drawing a square on the ground.



This same movement can be expressed in another language: "forward, forward, forward, right turn, forward, forward, forward, right turn, forward, forward, forward, right turn, forward, forward, forward, right turn". This time there are only three words: "forward", "right turn" and "left turn", composed by a sequence operation.

Activities around the notion of language consist, for example, in interpreting instructions given by another student or finding a phrase to instruct him to go from one point of the school yard to another. Afterwards it is possible to try more elaborate exercises, like translating an expression from one language to another -for example an expression formed in the first of the two languages presented above into the second one - demonstrating the redundancy of a language - for example, "left turn" can be replaced by a series of three "right turn". During this type of activity it is also possible to evoke the notion of "bugs". A tiny error in an instruction expressed in the second language, for example one "right turn" too many, completely changes the drawing and sends the student anywhere, as the figure on the upper right shows.

This type of activity helps students understand, in a very simple context, a few of the essential traits of written language: its conventional character, the need for rules and the correspondence between words and actions. It also allows them to understand that it is possible to calculate with words instead of numbers.

Speaking to little ones about information, language and algorithms with their words and based on examples in daily life

The notion of information is also a formidable key for entering the world of computer science. The first notion that needs to be transmitted is that any form of information can be represented numerically, that is to say by a series of symbols, like 0 and 1. Images, sounds, texts and numbers all have a digital reflection, a code that allows this information to be memorised, transmitted and reproduced infinitely. It is possible to introduce the atom of information, the bit, as early as the primary level and to ask how many bits are required to express a piece of information. To know whether a light is switched on or off, a single bit is enough, while to express a person's hair colour - brown, chestnut, blond or red - two bits are needed. Here the notion of the quantity of information contained in a message appears quite naturally, which is, at first reckoning, its size. However, redundancy complicates this notion: if a message expresses that an animal is an insect, adding that it has six legs, the size of the message increases, but not the quantity of information transmitted.

A third notion that "unplugged" teaching can tackle is the algorithm. An algorithm is essentially a way of solving a problem by performing elementary operations "without thinking". Algorithms that transform symbols - addition, subtraction, declensions, conjugations, etc. - are as old as these symbols themselves, that is to say as old as writing. However, mankind obviously used algorithms well before the advent of writing, to prepare food, weave cloth, shape silex, etc. Initiation to the notion of algorithms can start by identifying simple ones that children use every day: when you dress you need to put on your T-shirt before your sweater; to make an apple pie you need to add the apples before you bake the crust, but for a strawberry pie you need to add the fruit afterwards. The second stage consists in examining the constructions used to express an algorithm such as:

- a sequence: do this then that;
- a test: if a given condition is true, then do this, or else do that;
- a loop: do this three times, or until a given condition is true.

Once these bases have been established, it is possible to introduce the notion of a parallel algorithm: to make an apple pie, a pastry chef cuts the apples while another prepares the crust; on the other hand, when tying a bowline it is difficult for two sailors to divide the task. This line of thinking can be pursued, even at the elementary level, by the difference between an algorithmic and a non-algorithmic definition: defining the difference $n - p$ of two numbers as the remaining number of pebbles when we place n in a bag and we remove p provides us with an algorithm for calculating the difference of two numbers. On the other hand, defining the difference of two numbers n and p as the number we need to add to p to obtain n offers no direct way of computing the difference.

The objectives behind discovering the notions of language, information and algorithms are twofold. The first is obviously to prepare for more advanced education in computer science in middle school and secondary school, which makes this discovery phase the first step in training throughout the student's education. However, it is important to understand its overall utility for the education of young children. Many of the first things young children learn concern language and algorithms. Children learn rules that allow them to solve problems, including communicating with people around them. Just as the rules of a game are not constraints imposed by an authority, but a necessary condition for the game to exist at all - you cannot play hopscotch without following the rules as to where you are allowed, or not, to place your feet - grammatical rules, rules for finding a sum or one's way in a city are not constraints imposed by an authority, but vital components in methods used to express ideas, calculate something or figure out one's position. Here, grammatical rules define a language, coding rules can express information and rules expressed by algorithms can be used to solve a problem. This initiation to computer science is therefore an opportunity for children to start discovering at an elementary level this dimension of our culture, notably our written culture, made of signs and rules for manipulating them.

It is also possible to propose "unplugged" activities around the notion of the machine, for example, by showing students how difficult it is for them to act like robots and where this difficulty comes from. It is also advisable to call the student's attention to a few key elements in the history of machines. Mankind first made tools, i.e. objects that require a certain level of skill to use. Then it made machines equipped with motors to execute operations independently, which sometimes meant adjusting machines to perform one task or another. For example, fabric looms can be adjusted to weave one pattern or another. Then machines capable of manipulating symbols were made to process information. For example, a special machine was built in the United States to process the results of the 1890 census. However, what sets computers apart from other machines is their universal character, i.e. the possibility for a computer to execute an arbitrary algorithm: one computer is all computers.

Middle school: acquisition and autonomy

In middle school, combining science and techniques

One particularity of computer science is that it is both a science and a technique: its purpose is to construct both knowledge and objects. These objects are either concrete, like computers, or abstract, like, for example, programs. In this sense, computer science may be a sign of new relationships between sciences and techniques in the 21st century. While the construction of knowledge dominates at the kindergarten and primary-school level, the construction of objects, particularly programs, should, in our opinion, dominate teaching at the middle-school level. Writing a program oneself is indeed an essential step in becoming an actor, instead of using objects constructed by others. This is also an essential step in understanding how programs are made and mastering the principle concepts of computer science, many aspects of which can be discovered through programming.

Because certain objects are abstract in computer science, it is possible to construct them with very few means compared to other technical fields like aeronautics or even electronics. This means practical activities can play a much more important role than in other disciplines.

Discovering programming in order to master the digital world

Students start to learn about programming with the rudiments of programming language, limited, for example, to the notions of assignment, sequence, test, loop and array, without necessarily introducing more advanced notions like functions or dynamic allocation. However, learning about programming also means learning know-how that allows students to move from a description of what a program should do, like, for example, printing a calendar, to an idea of the way to do it, for example, by using two nested loops. This also means learning how to develop programs that often contain bugs and fix these bugs.

Whereas more traditional courses rely on a series of short exercises, performed individually and at the same time by all students, teaching computer science involves exercises that are performed over longer periods of time, for example several weeks, and in small groups, usually in pairs, working on different exercises. These exercises are often called "projects". A large share of the task required of students involves, moreover, defining, in part by themselves, the subject of their project.

This programming activity is therefore an opportunity to develop a form of teaching at the middle-school level devoted in part to projects, an idea that dates back to John Dewey. This is currently practiced by *the Enseignement Intégré de Sciences et Technologies au Collège* (Integrated teaching of sciences and technologies in middle schools) program, or EIST, and the *La Main à la Pâte* Foundation, and it can be seen as an introduction to research methods.

When teaching in project mode, teachers must not be silent as the students are highly unlikely to discover knowledge that mankind has taken centuries to construct all by themselves. The teacher needs to guide the students, answer their questions and accompany them in their research, without stifling their creativity. It is also possible to ask the students to read a chapter in a book before starting their project or to search for resources on line.

Despite all its qualities, teaching in project mode is not enough as the knowledge transmitted can often be too fragmented. The link between different forms of knowledge may be missing. It is essential to establish a balance between learning through projects and traditional classes that contribute to structured thinking.

Learning about programming is a way to discover the rudiments of computational thinking. As soon as one learns programming one understands that certain problems, like detecting the presence of a face in an image, which seem simple enough to solve but are very difficult to program, contrary to others, like checking if the letter 'A' is present in a word. This awareness is a first phase that leads to wondering how our brain is capable of recognising a face in an image. A new meaning of the interrogative adverb "How" emerges: the answer to the question "How does our brain recognise a face?" is an algorithm, undoubtedly programmed by evolution, that we are only partially familiar with.

Middle school is also an opportunity to learn about other aspects of computer science, or delve deeper into them. First, while certain computer systems are autonomous, like desktop computers for example, others are used within more complex systems such as planes, trains, cars, etc., that combine mechanical and computer components. The interface between computers and the mechanical systems they control consists in sensors and actuators that are also machines. This is a very rich subject because it mixes hardware and algorithmic dimensions as algorithms are required to translate an analog value into a digital one inside a sensor or a digital value into an analog one inside an actuator. Other algorithms are essential to control the system by acting on the actuators according to the values picked up. As early as middle school it is possible to have the students program a small robot, like a car, so that it follows a white line: when the car drifts too far to the right, the program must turn the steering wheel to the left, and when it drifts too far to the left, it must turn the wheel to the right. This type of work raises awareness of problems like analog-digital conversion and control with closed-loop feedback, which can only be examined in depth at secondary-school level.

Middle school is also a level where networks can be studied, and particularly the notions of addressing and routing in an "unplugged" way, i.e. without the use of a computer. While in primary school it is only possible to examine the way a package finds its way through a network and provide relatively simplified answers, it is possible at middle-school level to introduce the notions of router, routing tables and their updating, and routing algorithms. This introduction to the notion of networks is obviously an ideal moment to introduce the notions of security, privacy or property that their development implies.

We have mentioned the profound changes computer science has brought to our society. It is in middle school that students can start to think about these changes. For example, teenagers learn at a very early stage how to download music. Beyond the legal implications of such practices, it is possible to ask them why the question of illegality concerns an mp3 file, but not a vinyl record or even a CD. The answer to this question can lead them to understand that non-rivalry is an essential dimension of material goods: since a digital copy is instantaneous and perfect, possessing information does not prohibit others from possessing it as well. Changes in practices, but also in standards and even the concept of private property are determined by changes in the properties of the objects exchanged.

Secondary school: consolidating knowledge and know-how

Computer science is for all students in all programmes

From the primary level students will have discovered the notions at the heart of computer science: the notions of the machine - computer, network, robot, etc. - of the algorithm, language, in particular programming languages, and information. Training at the secondary-school level offers an opportunity to stand back and take stock of these notions and, above all, to understand the way they work together.

The notion of language, for example, is seen in middle school, essentially through programming activities. It is possible at the secondary-school level to introduce more advanced constructions such as the notion of function, recursion, dynamic allocation, types of data and objects, and parallelism. And it is also possible to start inventing little languages of one's own - like the language "North-South-East-West" introduced in kindergarten - and to write interpreters for such simple languages. This is also a time to understand the plurality of languages used within the same computer system, for example, the fact that one program written in a high-level language is translated into machine language in order to be executed by a machine. Finally, secondary school is ideal for introducing the notions of source code and compiled code, which are very important in understanding technical aspects ranging from optimising a program's response time to legal issues regarding software licenses.

The notion of information can be tackled from two angles: its coding and its structure. Coding is a set of methods that allow computers to manipulate numbers, texts, images, sounds, etc. when they only know, in a primitive sense, 0s and 1s. This idea of coding also includes methods that are used to compress information, correct transmission errors and encrypt information to protect it. Structuring information involves the entire set of methods associated with complex coding, from file systems to databases and the Web, which allow a computer to manage astronomical amounts of data.

The notion of the machine can be dealt with in a more analytical way at secondary-school level by insisting on the multiple descriptions of a single object depending on its scale, ranging from the most concrete to the most abstract: the scale of the transistor, logic gate, processor, computer, local network, Internet, etc. It is at this level that distributed programming can be evoked, for example in client–Web server architecture and mobile phone applications.

Finally, the notion of the algorithm is dealt with in and of itself, while insisting on the subtle difference between algorithm and program. It is in secondary school that students can understand that an algorithm can be the object of a resolutely abstract study, other than its incarnation in a particular language or its execution on a specific machine, but at the same time of very concrete research regarding, for example, the resources it requires in terms of time and space.

Secondary school is also a time for greater specialisation. Depending on the students' choices, it is necessary to adapt training in computer science to their centres of interest. The fundamentals of computer science are the same for all specialisations. However, it is possible to vary the examples, computer applications and projects to suit the students' preoccupations. In the final two years of *terminale L* (Literature), for example, questions regarding processing language and text can occupy a central role. From simple algorithms used to pluralise a word - including irregular plurals like "child", "man", or "mouse", etc. - to algorithms used in statistical analyses of texts to identify the author, for example. In the final two years of *terminale ES* (Economics and Social studies) and *terminale STG* (Management Sciences and Technologies) it is the notion of information that is central to training, via studying the place of data in social sciences and above all the way information systems have changed the way firms operate. In the final two years of *terminale STI2D* (Sciences and Technologies of Industry and Sustainable Development) it is the notion of the machine that is central to training, via an in-depth study of computer architectures and their insertion in machines dedicated to specific tasks. In *terminale S* (Science) the notions of the algorithm and modelling will dominate. Finally, it is in secondary school that students can be introduced to the use of computer science in more global contexts. First, it is possible to evoke how information plays a pivotal role in organising a set of activities in a laboratory, by processing the results of an experiment or conducting a simulation, or in a firm, via its information systems. It is also possible to show how computers are integrated in extremely complex technical ensembles, such as aircrafts. These subjects lead to examining societal questions, whose study, as we have said before, is part of the phase aimed at mastering computer science. This type of work, which is pan-disciplinary in nature, will involve cooperation between professors in philosophy, French, music, economics, etc.

After secondary school: preparing for all professions in the digital world

Evolving with training in computer science at the secondary level

Training in computer science after the Baccalaureate will certainly need to be redesigned when students that have acquired real basic knowledge in computer science reach secondary-school level. The programme will naturally be different when addressing future computer scientists, engineers or scientists in other specialities, or students destined for other careers. We shall only discuss the main principles here.

Computer scientists: Training in computer science has existed long enough for the curriculum of a computer scientist to be well understood, even if significant nuances remain. Training of computer scientists must constantly be adapted according to the most recent developments in research. However, one should not be mistaken. While new products are launched at a rapid pace for commercial systems, computer science is evolving at a much more reasonable speed.

The importance of mathematics in learning computer science implies solid training in this area. This is essential so that students can really learn the basics of computer science. Most computer scientists we train do not work in the computer industry, but across the spectrum of other industries, even actively taking part in research in other fields like medicine, biology or nuclear power, for example. It is therefore important to give students in computer science solid training in scientific basics and avoid the temptation of overspecialisation in computer science. General training in mathematics and experimental sciences is vital for them to find their place over the long term in a rapidly evolving world.

Engineers and scientists in general: We have already insisted on the importance of training engineers in professions directly related to computer science in preserving the competitive edge of our industries. Training in computer science for engineers and scientists in other fields requires the same special attention. One French specificity is the path to excellence in the training of engineers and scientists through preparatory schools and *Grandes Écoles* (leading higher education institutions). This path leads a large share of students to an engineering degree at the same level as the third or fourth year of university studies in mathematics and physics. It is essential to define similar objectives in computer science. This means reinforcing training in computer science in preparatory schools - and this has started. Many engineering schools grant a great deal of importance to training in computer science. It is important to ensure course content is appropriate and, as stated before, that the basics are not overlooked in favour of tools and techniques that could soon become obsolete. At university level, computer science is already present in most scientific programmes. This movement should be encouraged.

Other students: It makes sense that training in computer science for non-scientific students includes a general part intended to broaden their general culture in the field and another part oriented more specifically according to their training and future profession. We should note that a growing number of American universities, such as Princeton, offer an introductory course in computer science that is taken by the vast majority of students. A student in cinema can take more specialised courses in image processing or synthesis and special effects *afterwards*; a student in business administration can study information systems in greater depth; a student in architecture can study computational geometry, etc. For these students as well, we are touching on the necessity of a global approach to the problem of teaching. To the extent that in the future computer science will be the object of real training in secondary school, they will be able to take only specialised courses adapted to their discipline if they so desire. However, this is currently not the case.

The main risk seems to be that in certain degree programmes, out of ignorance regarding the real nature of the subject and due to a lack of qualified teachers in computer science, the courses focus on the use of tools without devoting serious thought to the underlying concepts.

Professional development in all sectors[17]

Professional development plays an essential role in computer science from a dual point of view. First, it will bring generations that have learned the basics of computer science up to level. For these individuals, this will in fact be initial training in computer science, but in differed mode. Secondly, it will allow everyone to keep abreast of changes in the field, in a world where we are constantly confronted with the advent of new tools and concepts. In this area, training content is usually highly contextualised according to the professional skills targeted. Generally speaking, it is important to remain aware that training in concepts, as opposed to specific tools, often ensures the long-term success of these initiatives.

The use of MOOC allows the implementation of this type of training on a large scale and a shared basis, in cooperation with large continuing training organisations. Popular education in computer sciences also involves the sharing of online content, in cooperation with actors in scientific and technical culture. Such content can serve as a resource for local initiatives promoting scientific dissemination, as is already the case to a certain degree.

[17] Extension of the former term "continuing education".

Teacher training and status

Teacher training is a top priority

Teacher training and status in general is the object of many questions that this report alone is not intended to address as they concern all disciplines. However, it is possible to suggest some general principles to ensure computer science is taught at all levels by teachers who are well-trained, as is the case for other disciplines. The first of these principles is that, apart from historical considerations, there is no valid reason to treat computer science differently from other disciplines.

At each level it is important to treat initial training at the ESPE (*École Supérieure du Professorat et de l'Éducation* / teacher's college) differently from professional development of teachers who are already active in the field.

Primary school

At the kindergarten and primary levels it is not customary to call on specific teachers for each discipline and it does not seem advisable to do so for computer science. It is therefore necessary to include computer science in the curriculum of primary school teachers, just like other disciplines, so that they can teach it in the same manner.

It is also necessary to propose training to primary school teachers who are already working in the field. Regarding this point, it is important to note that existing solutions set up to train a few dozen teachers a year are grossly inadequate. We need a real national training plan if we want to tackle computer illiteracy in our country today.

Middle school

In our opinion, the question of training for middle-school teachers deserves particular attention. It is currently based on two divisions between disciplines. Sciences are traditionally taught separately, with classes in mathematics, physics, chemistry, life sciences and earth sciences. However, the same teacher can be competent in several areas, teaching both physics and chemistry or life and earth sciences. Conversely, in technology, techniques are traditionally taught as an ensemble: technology is not broken down into mechanical engineering, electrical engineering, biotechnologies, etc. This division is not written in stone, as the *Académie des sciences* and the *Académie des technologies* have promoted, since 2006, experiments in EIST *(Enseignement Intégré de Science et Technologie/* integrated teaching of science and technology).

We need to study the position of computer science within this framework. First, computer science is both a science and a technique and it is not possible to restrict one of these two aspects without severely mutilating it. Furthermore, computer and software engineering are too specific to blend into a vast ensemble that would also include, for example, mechanical and electrical engineering. Finally, while computer science is partly a technique whose purpose is to construct objects, these objects are often abstract in nature. It therefore seems impossible to integrate computer science within training in technology at middle-school level, which is oriented around material objects. Training in computer science should be sufficiently specific and ensured by teachers specifically trained at Masters level, even if it must obviously interact with other subjects. Otherwise, if it is too diluted it there is a considerable risk of it losing its relevance.

Secondary school

As is the case for all other disciplines, a teacher at the secondary-school level should have a minimal level corresponding to four to five years after the Baccalaureate. This principle can be put into practice by setting up competitive exams in computer science, similar to the ones that already exist for traditional subjects, and recruiting teachers with a Master in computer science, etc. It is possible to envisage more open solutions such as by using teachers in other disciplines having partial training in computer science, as is the case in the final year of secondary school for ISN (*Informatique et Sciences du Numérique* / specialisation in computer and digital sciences) or by following the proposal put forward by the *Académie des sciences* in 2007 of a CAPES competitive exam with a major and minor, either of which could be in computer science. However, we must be very wary of diluting supposedly minor subjects, which is a valid complaint regarding chemistry compared to physics or earth sciences compared to life sciences. For example, having only mathematics teachers ensure courses in computer science could lead to training in only part of the subject, the part most closely related to mathematics, which would not correspond to the objective cited earlier in this report.

One idea worth exploring consists in sharing means between universities and secondary schools and even middle schools. Strong ties existed between French universities and secondary schools up until the 20th century, but unfortunately have since been dissolved. The introduction of a new scientific discipline could be a way to revive them. However, we should note that this only displaces the problem since university computer science departments are often understaffed.

Learning on line: The development of MOOC (massive open online courses) offers a new lever for improving continuous teacher training with potentially significant economies of scale. Experimental forms have been set up to support teacher training in the specialised ISN programme (*Informatique et Sciences du Numérique /* computer and digital sciences) in the final year of secondary school, and we already have feedback regarding their advantages and limitations. Online courses can serve to support solid traditional courses, providing they are rooted in concrete communities. Then they can contribute to the creation of privileged ties between research-lecturers and middle- or secondary-school teachers, which form the basis of such communities. The collaborative aspect of online courses can reinforce this synergy as well as the sharing of course content and practices, on both the national and local level.

Training in computer science around the world

The idea of teaching computer science in schools dates back to the 1960s. This general training was first implemented in the 1970s[18]. During the 1980s new general courses in computer science were set up, mostly as electives. In the second part of the decade there was a shift in interest among public authorities, all over the world, towards the implementation of new computing tools. In the years 2000 there was an awareness of the need to confirm computer skills among students. Then, in the second half of that decade, specific courses in computer science were set up in industrialised as well as developing countries, with great diversity existing between nations and sometimes even within in them.

The following lines present a summary of research and reports published before 10 December 2012. The purpose of this section is to offer an image of a situation that is rapidly changing.

Contrasting situations throughout the world, with a growing interest in the teaching of computer science

The terminology used around the world to describe training in computer science varies and in the same way there is a wide variety of ages and levels of training at which different concepts are introduced. We should make the distinction between two types of training: courses focused on familiarising students with the use of information technologies (IT, ICT) according to a so-called integrated approach, and specialised training in computer science. Familiarisation with uses is particularly present in the lower levels of education systems (IT literacy). This is much more widespread than training in computer science.

In countries where computers science is taught, there are different approaches as to the importance of practical realisations and learning theoretical concepts. However, the orientations adopted generally concern computational thinking, with a significant share of the programme devoted to algorithms and programming. Programming languages are not always specified as the focus is on the acquisition of concepts. Some countries introduce networks and information systems management as early as the end of secondary school.

A comparison of five cases (Finland, Japan, Massachusetts, Ontario and Singapore) conducted by the *Royal Society*[19] shows that the age at which specialised courses in computer science are introduced varies from 12 to 16.

An overview of the current situation in Europe

In Europe, at primary and general secondary levels, most countries have considered computing, up until recently, as a set of tools that can be used to develop skills in other disciplines rather than a discipline in its own right.

Regarding the secondary level, and especially secondary school, several countries teach computer science as an independent discipline, with an official programme, specifically trained teachers and dedicated course hours. We can note a wide variety of terms for this discipline: *ICT*, *IT literacy*, *Informatics*, *Computer Science*. Courses entitled Computer Science are based more on computer

[18] CERI-OECD. (1971). *L'Enseignement de l'Informatique à l'École Secondaire* (Education in computer science at secondary school level) Paris: OECD

[19] Source : International Comparison of Computing in Schools, Report for the Royal Society, L. Sturman & J. Sizmur, September 2011

science as it is understood in this report. For programmes referred to as ICT (information and communications technology) or IT literacy situations can vary. Generally they involve the use of a range of applications and knowledge of how computers work.

However, some countries explicitly include developing programming aptitude in their course objectives (notably in Germany, Greece, Spain, Italy, Poland and now the UK)[20]. Furthermore, there is a great deal of diversity regarding the status of training in computer science, depending on the type of courses offered: some countries have mandatory courses, others have electives, others are currently trying to integrate skills and knowledge in computer science within existing disciplines.

It is important to note a recent trend towards implementing specific curricula in computer science. We will discuss two examples here: the United Kingdom and Germany.

The example of the UK

The UK has recently experienced changes that are undoubtedly representative of a more general trend. Information and communications technologies have been part of the national programme for a long time. However, important changes have been recently announced in response to severe criticism of this training.

In 2010 the Royal Society of London set up a task force whose assignment was to study education in computer science. The title of the report submitted in 2011[21] speaks for itself: "*Shut down or restart?"* The report concluded that it is vital to change the way computer science is taught, essentially based on ICT, and to focus on computer science instead.

At the same time, *Computer At School* (CAS), a group of teachers and professionals in computing, has proposed creating courses in computer science[22]. In June 2012, Michael Grove, secretary of state for education, announced the end of courses in ICT in order to allow schools to propose a real programme in computer science. He encouraged them to adopt the programme proposed by CAS.

In February 2013 it was announced that computer science would now be part of education in science and A-level exams with the same status as physics, chemistry or biology.

Shut down or restart : **A few recommendations from the Royal Society**

The question of terminology seems essential: the report recommends not using the vague term ICT, but rather *Digital Literacy*, *Computer Science* and *Information Technology*. Each of these terms is defined in detail. The use of fuzzy terminology is seen as having a highly negative effect on the creation of consistent policies.
Teacher training is a vital stake: in the UK only 35% of ICT teachers are trained in the subject.
The existing A-level exam in ICT has fallen out of favour. The failure of this A-level is due to a lack of coordination with higher education, which does not admit students in the university degree programmes that normally should accept them. In order to revive this A-level more concerted efforts are required.
Training in computing (including computer science as a discipline) is recommended. It is proposed that at age 14 students should have the basic knowledge they need to make a well-informed choice as to whether or not to pursue training in computer science. This question of a "well-informed choice" is also deemed important in the analysis of gender issues. A complementary study[23] conducted in different countries (Japan, Finland, Serbia...) confirms this analysis.

[20] Eurydice, 2001, op cite p 39.
[21] http://royalsociety.org/uploadedFiles/Royal Society Content/education/policy/computing-in-schools/2012-01-12-Computin-in-Schools.pdf
[22] Computer science as a school subject. Seizing the opportunity. CAS. March 2012
http://www.computingatschool.org.uk/index.php?id=documents

[23] http://www.nfer.ac.uk/nfer/publications/cis101/cis101.pdf

The example of Germany

In Germany, a federal republic, the situation varies from state to state. Overall, an integrated approach to information technologies, seen as tools in other disciplines, has been adopted for the most part at the primary level. As for the middle- and secondary-school levels, in most states, except for three, the general programme includes courses in computer science, which are either mandatory or optional[24]. Bavaria in particular has had mandatory classes in computer science since 1998. Researchers at the University of Munich took part in writing the curricula and have continued to work at a sustained pace on the didactics of computer science [25].

In 2008, the *Gesellschaft* für *Informatik*[26], a special interest group of scientists and professionals in computing, has published a proposed curriculum for middle schools (up to age 16). The *Länder* that have decided to review their computer science programmes are working in the direction of this proposal.

Brief overview of the situation in the rest of the world

Asia is a vast region with many contrasting situations. International studies indicate there is a great interest for training in computer science starting at the secondary-school level (particularly in South Korea, Vietnam, India, etc.). In India an optional course in computer science is offered for students age 14. Before this age, schools are free to offer training in computer science according to their means. There are several institutions that establish programmes and certifications for schools. The two main organisations propose complete curricula for training at the secondary-school level[27]. Since 2008 a group of university professors has been working on a new curriculum[28].

The situation in Africa is also highly contrasted. However, it is interesting to note that a certain number of countries, especially in Northern Africa, have set up courses in computer science starting at the middle-school level. Countries in Sub-Saharan Africa have also strived to offer specific training, as a means of ensuring future development[29].

In Israel[30], in 1990, the ministry of education conducted a project in order to elaborate a curriculum in computer science for the secondary level. This programme was implemented in 1998 with the creation of two programmes at the secondary-school level: one consisting in 270 hours divided into 3 modules devoted to basics in computer science and a 450 hour programme with 5 modules on advanced notions. Two modules on fundamentals are mandatory and students can choose between another four modules. Student course books and teacher's manuals have been published. The teachers recruited must have at least a Bachelor in computer science[31]. The students are assessed by a final exam. Currently a curriculum for middle schools is being created. It includes a mandatory and an optional part. The programme is focused on programming, robotics and the use of spreadsheets in scientific fields. The stated objective is to interest students in scientific fields.

[24] Synopse zum Informatikunterricht in Deutschland, I. Starru§ 2010. Online: http://dil.inf.tu/dresden.de/uploads/media/Bakkalaureatsarbeit_Isabelle_Starruss_01.pdf

[25] See the work of Peter Hubwieser

[26] GrundsŠtze une Standards für die Informatik in der Schule Online: http://www.informatikstandards.de

[27] See the Council for the Indian Certificate Examinations and the curriculum proposed by the Central Board of Secondary Education: http://www.cbse.nic.in/welcome.htm

[28] Model Computer Science Curriculum for School, Ier et als. 2010

[29] We should note the very recent implementation of courses in "Information and Communication Technologies" announced in Ivory Coast for kindergarten to the final year of middle school: http://www.education-ci.org/portail/node/250

[30] Source: " A Model for Secondary-School Computer Science Education: The Four Key Elements that Make it!" , O. Hazzan, J. Gar-Ezer and L. Blum, SIGCSE'08, March 2008, Portland, Oregon, USA

[31] Ragonis, N., Hazzan, O., & Gal-Ezer, J. (2010). A survey of computer science teacher preparation programs in Israel tells us: computer science deserves a designated secondary-school teacher preparation! In Proceedings of the 41st ACM technical symposium on Computer science education (pp. 401–405)

Summary

The foreign experiments presented here show that implementing a programme in computer science takes time and implies satisfying a certain number of conditions:

- Clarification of the terminology used: Countries such as Germany and Israel, which have designed and implemented curricula in computer science, do not seem to have had this debate on terminology. They have adopted the terms Computer Science and Informatics.
- Design of a complete curriculum: In several countries associations or special interest groups gathering researchers, scientists and specialists in education have proposed complete curricula in computer science for middle- and secondary-school levels. This is notably the case in Germany, Israel, Ontario and the United States. These curricula have greatly contributed to the creation of courses and their implementation in schools. One advantage is they provide a clear description of what computer science is.
- Implementation of robust teacher training or recruitment of teachers with a Bachelor level in computer science. In several countries, the relative failure of ICT-oriented programmes is due to a lack of teacher training (see in particular the box devoted to the UK). Countries that have created a training programme in computer science, like Israel, have also made an important effort in training and/or recruiting qualified teachers.