Microsoft Computer Science Curriculum Toolkit

Structure and Principles

BACKGROUND TO THE MICROSOFT COMPUTER SCIENCE FRAMEWORK

To develop the Microsoft Computer Science Framework (MCSF), Microsoft called on a group of internal and external experts with experience as computer science professionals, educators, and in academia. This group drew on Microsoft's expertise as a world-leader in technology and a decades-long employer of computer scientists to understand what content is important to include in a computer science curriculum framework. It consulted academic research in the learning sciences and curriculum development to design the structure and sequencing of the curriculum framework and to identify the right pedagogies to recommend for the content.

Further to this, the group conducted research into what other education systems around the world have been doing in terms of developing CS curricula at a school level. This research was conducted both as traditional desk research, in the form of a literature review, and as a discussion with Ministry of Education officials on computer science in the curriculum that was part of a live event hosted in June 2020 by Microsoft, European Schoolnet (a network of 34 Ministries of Education in Europe) and the STEM Alliance platform.[i]

These research initiatives served to teach the MCSF designers both what has been successful and what has been learned about past large-scale CS curriculum designs and implementations. It also gave the writers of the MSCF a clear idea of Ministry of Education priorities in terms of content, competences, and objectives. All of this research, summarized in the sections that follow, enabled the design of the innovative curriculum framework that is outlined in this document.

ADDRESSES MINISTRY OF EDUCATION FEEDBACK

Feedback from various Ministries of Education was collected from a live, online event co-hosted by Microsoft and European Schoolnet as described above. During this event, representatives from the European Commission provided a summary of computing, computer science, and computational thinking initiatives in several European countries. A panel and question and answer session were also held with representatives from the Ministries of Education in three European countries who had developed or were in the process of updating their national curricula to include computer science. Participants in the panel were asked about their countries' objectives in developing and implementing CS curricula and the outcomes they hoped to achieve. They discussed the barriers they had experienced, including those inherent in their own education systems, and support mechanisms they were putting in place at various stages in the process. Participants to the online event were able to ask questions to learn from the best practice and experiences of these countries.

One of the key takeaways from the event that was fundamental in the development of the MCSF was that computer science is not just about programming. One participant stated that her country's objective in developing a comprehensive CS curriculum was to create "digital wisdom" among its citizens. Government objectives in creating CS curricula were not necessarily about more students becoming computer scientists but giving more student access to the fundamentals of the subject so that they could use their skills in "known and unknown situations," as one participant said. Countries see the benefits of computer science as an outlet for students' creativity, problem-solving, and innovation. They want to show young people what is possible in terms of their own future careers and in addressing problems facing our world today and in the future.

The MCSF also takes account of feedback from Ministries that creating an outcome-based CS curriculum not only prevents it from becoming outdated or obsolete too quickly, but it provides more opportunities for students to be innovative. Other countries have found that overly prescriptive CS curricula that dictate the technologies to be used rather than the competencies to be achieved promote "teaching to the test" and do not allow students to be creative in their problem solving. Countries have had to provide support for educators to enable them to feel confident with this level of flexibility in teaching complex computer science concepts.

Some of the barriers to adopting CS curricula that these countries experienced will also need to be addressed by any users of the MCSF. Participants expressed that traditional exams are often an inappropriate method to assess competencies in computer science. Indeed, the MCSF organizes more advanced work into labs and projects that could be used to evaluate student learning rather than high-stakes exams that often measure student recall at a certain point in time. Learning projects and student portfolios, also mentioned during the panel discussion, allow a more in-depth examination of students' mastery of competencies and the progress they have made on their learning journey.

Finally, education systems face a challenge in finding enough educators qualified to teach computer science and in keeping their professional development up to date with the constantly changing field. Support structures are needed to provide initial training and continuous development for a computer science teaching workforce – for instance in England, the National Centre for Computing Education[1] was set up to "to inspire and support teachers in delivering the new computing curriculum and to help establish computer science as a foundational subject across all key stages."

MAKES USE OF INNOVATIVE PEDAGOGIES

The MCSF allows for flexibility in educators' choice of pedagogies as well as technologies, but has as its core an inquiry-led, problem-solving approach. The MCSF poses "Big Questions" to students, which not only allow the curriculum to be flexible and adaptable to accommodate the rapid pace of technology change, but also encourage students to apply what they have learned to solve real-world problems. There is abundant research on this kind of problem- or project-based learning (PBL) that shows a positive impact on students' development of skills and their motivation towards learning.[ii] For example:

- Students who have participated in PBL retain more knowledge, acquire better-developed critical thinking skills, and have increased motivation to learn than those who learned through traditional methods.[iii, iv]
- PBL is positively associated with skill development in students, regardless of the stage of education or level of expertise of the student.[v]
- PBL can benefit students considered "at risk" who may come from challenging home circumstances, for example. In mathematics education, use of PBL has decreased the achievement gap between these at-risk students and their peers.[vi]

Most critically, PBL provides the context needed to develop collaboration, problem-solving, systems thinking, communication, and the broader competencies that are essential to a modern education. A PBL approach to curriculum simulates the work learners will experience in most kinds of work later in life. A PBL approach help them learn to work in teams and directing their own acquisition of new knowledge and skills.

One of the challenges of PBL is the quality of the initial question or problem that needs to be answered or solved, which can vary substantially depending on the educator. A key feature of the MCSF is that it provides guiding and inspiring questions which students will learn competencies to address – a major difference to other existing curriculum frameworks and approaches.

FOSTERS EQUITY IN COMPUTER SCIENCE EDUCATION

Across the world, there are challenges of equity in CS education. Issues of equity manifest themselves in terms of the availability of CS courses in all schools in a country and the level of participation in CS courses from diverse groups of students.

Research has shown that there has been relatively low participation in higher level CS courses among girls and some minorities. This can be due to availability of courses, as mentioned, but some research indicates that it is connected to the way that CS is taught and reinforced at school. For example, in the United States, a review of research uncovered several reasons why girls are often deterred from taking computer science, including:

- Teaching style that uses lecturing rather than hands-on, real-world projects.
- Lack of collaboration with peers.
- Lower levels of self-confidence, and lack of clarity around careers in computer science.
- A feeling that mistakes or risk-taking are not permitted in computer science.[vii]

[1]https://www.stem.org.uk/secondary-computing

In those countries that have created new CS curricula, there is also a tremendous challenge in implementation: educators feel insufficiently prepared to teach CS, and there are low levels of recruitment of CS graduates to the teaching profession. When CS professionals do become educators, they often lack the pedagogical skills to understand how to teach the very topic they have used throughout their own careers.[viii] As a result, the need for in-service CS professional development is immense. Given these challenges, at global scale, there is a clear incentive for collective action to more effectively design CS curricula and prepare educators appropriately to engage students in impactful CS learning.

The MCSF takes advantage of several approaches supported by the research literature to help make the teaching of CS more equitable to students of all backgrounds and abilities. For example, equity is overtly woven into the learning goals, concepts, and challenges in the MCSF. Students study the impacts of computer science on issues of accessibility as well as the positive and negative impacts of AI on society. The deliverables associated with many CS tasks provide opportunities for non-native language speakers to demonstrate competency without having to produce as much written work as they might in other subject areas. The project-based and inquiry-based learning approach uses real-world, socially relevant problems to help engage females and underrepresented minorities in computer science.

The MCSF takes into account that not all schools have adequate equipment to provide an “ideal” CS education. However, many topics, especially in primary schools, can be learned without computers. This curriculum makes many suggestions for “unplugged” learning: i.e., computing activities that don’t require physical hardware. Block-based programming helps to make some concepts of programming and CS more accessible to younger children. The MCSF also provides a variety of approaches to each topic, accomplished by the various learning goals, concepts, and challenges offered, that will help meet the varied learning needs of students of all ages.

Finally, to reach the widest possible range of learners throughout their education, the MCSF offers three possible graduation pathways: academia, vocational, or entrepreneurial. The MCSF believes that all learners should have access to a CS education, should they desire, regardless of their ultimate career trajectory.

PROVIDES FLEXIBILITY TO ENCOURAGES LOCAL ADAPTATION

It is important to note that Microsoft is not providing a prescriptive, one-size-fits-all computing curriculum. Rather, the MCSF is exactly as the name suggests: a framework. A fully developed curriculum requires substantial creation and adaptation of schemes of work, classroom activities, and a system of assessment, including the development of standards. This necessitates substantial input from local educators and education experts who understand the context of a country’s education system and the conditions inside its schools.

However, what Microsoft is supplying with this framework is more than just a matrix of topics to be covered and their progression. The MCSF gives guidance on the topics that should be covered in an innovative and enduring CS curriculum and the sequence in which they should be covered. It does not provide high-stakes assessments but suggests projects and lab work that can allow students to apply their learning to demonstrate their mastery of the necessary CS outcomes. The MCSF does provide links to Microsoft and other high-quality training, programs and learning resources that can be used by CS educators and students, but it leaves plenty of room for addition of local content and resources. In other words, the MCSF provides sufficient detail to be adopted by schools but is flexible enough for different countries to adapt to suit their context or preferences.

GIVES STUDENTS A FOUNDATION IN CS WHILE INTRODUCING THEM TO CUTTING-EDGE CONTENT

Before developing this framework, research was conducted to identify global best practices to learn from what has already been done in developing CS curricula at a school level.

Where possible, the authors examined countries that provided curriculum spanning all ages from K-12, but in some cases the computer science curricula were only offered in secondary school. In other countries, the computer science curriculum is part of a much larger Computing or Digital Technologies curriculum that also includes other technology-related topics.

Certain key topics were identified as being commonly used to organize the high-level content domains within the curricula. These include content domains such as:

- Computing systems: including system fundamentals, computer organization and hardware
- Data and data-related topics
- Networks
- Algorithms
- Programming
- Impacts: including cultural and social impacts of computing
- Computational thinking
- Problem solving: often a combination of computational thinking and algorithms.

The MCSF includes this core content so that even those countries who already have national curricula or standards in computer science might benefit from the Big Ideas, Big Questions or resources provided in the MCSF. However, the MCSF also includes content on such cutting-edge technologies as artificial intelligence (AI), cloud, blockchain, and quantum computing which, according to the research, are often omitted from other national curricula.

PROVIDES REAL-WORLD CONTEXT BY USING THE UNITED NATIONS SUSTAINABLE DEVELOPMENT GOALS

As part of the United Nations 2030 Agenda for Sustainable Development, member countries agreed on 17 Sustainable Development Goals (SDGs) which require countries to work together to solve the most urgent problems facing our world today. The 17 SDGs are related to each other – none can be achieved in isolation – and are present in some form in developed and developing nations alike.

The MCSF takes advantage of the UN SDGs as themes for the Big Questions across all Domains in Phases 3 and 4. Solving the SDGs will require innovative, global solutions, and the MCSF aims to prepare students to tackle challenging problems such as these with cutting-edge technology and superior problem-solving and analytical skills. Furthermore, by connecting CS learning objectives to real-world problems, the MCSF hopes to engage learners in the topics and inspire them to develop solutions to real-world problems rather than simply completing abstract tasks.[ix]

SUSTAINABLE DEVELOPMENT GOALS
1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

Specifically, SDGs have been mapped against each of the Big Question, Sprint, and Pitches in the project and business phases. Students are expected to research the given SDG and explore problems that can be solved with CS. They need to demonstrate an understanding of how CS can make a positive impact towards the SDG if their proposed solution was to be implemented and scaled in the real world. To this end they will need to identify and specify target segmentations and use cases. Ideally, students will build prototypes that demonstrate creative and realistic solutions.

TAKES ADVANTAGE OF MICROSOFT EXPERTISE, CONTENT, PROGRAMS AND SUPPORT RESOURCES

Through Microsoft's work in primary, secondary and higher education and with computer scientists and programmers, a vast array of content, training and other support resources are available for educators and students of computer science at all levels.

- List of programs and resources with brief description.

The MCSF provides a broader vision of CS – one that allows students to acquire an inspiring 360-degree experience of the world of computing and enables them to acquire the broader set of skills required to innovate with technology as an outlet for their creativity. The wealth of accompanying resources from Microsoft help make this possible.

ORGANIZATION OF THE CURRICULUM FRAMEWORK

The structure of the MSCF was developed to adhere to the tenets of what the academic literature says about quality instructional design and curriculum development, as well as what learning sciences research tells us about how students learn.

Guiding the organization of the MSCF is the principle from learning sciences research that all learning should build on learners' prior knowledge.[x] Thus, topics are arranged to provide a foundation for learners in early years and build upon that knowledge as the student's journey through the curriculum progresses. The real-world applicability of the questions the MCSF poses allows learners to relate to what is being taught by being able to compare topics to things they already know and understand from their own life experience.[xi]

The MSCF follows the "Understanding by Design" (UBD) curriculum design framework[xii] that is thought of by many as one of the standards for high-quality curriculum development.[xiii] While much curriculum development is driven by content, the UBD framework involves what the authors call "backward design." That is, instead of choosing content and designing learning around it, UBD dictates first defining the objectives around what students should be able to understand for each topic and designing the curriculum (including content and assessment) around those. These objectives should be based on social needs or expectations around the subject area, learning standards, and research into the kind of understanding need for application of learning.[xiv]

Finally, the structure of the MSCF has been developed in a way that aims to build expertise in computer science, which is necessary if students wish to pursue further study or enter the workforce as computer scientists. Many curriculum and exam systems focus on students' ability to absorb information and simply recall it on an exam. This does not develop expertise, which involves knowing the conditions and contexts in which it is appropriate to apply certain parts of the knowledge one has acquired.[xv] Experts have a lot of knowledge but are able to retrieve and employ what they need for the specific scenario, something which high-stakes exams and other tests can fail to assess.

The design and progression of the MSCF, including Domains, Big Ideas, and Big Questions with application in real-world scenarios, aim to take all of this research into account.

STRUCTURE

At every stage, the MCSF aims to address the aforementioned issues with current CS curricula, and to meet the needs expressed by governments. The structure of the curriculum and the chosen terminology are designed to address the issue of CS curricula being unappealing to learners. The MCSF makes use of descriptors that are more engaging for learners and lend themselves to project- or problem-based approaches to learning computer science. Specifically:

- At the top organizational level of the MCSF are Domains. Domain replaces the term "topic" in a traditional curriculum.
- Underneath Domains are Big Ideas, which represent a learning pathway and continue throughout a student's journey through the MCSF, from age 5 to age 18. Big Ideas might be referred to as "themes" in a traditional curriculum. In other words, they are concepts that serve as the main point around which a learning pathway is structured.
- Each Big Idea contains Big Questions, which are akin to "modules" in a traditional curriculum. A Big Question specifies the main subject of a discreet package of learning and contains its own outcomes.
- Each Big Question contains Learning goals, Concepts and Challenges, a link to a UN Sustainable Development Goal (SDG), and specifications for supporting content.
- The challenges lead to specific Computational Thinking, Data Literacy and Design Thinking competencies.

Phases

The MCSF follows the International Standard Classification of Education (ISCED 2011) created by UNESCO, which is used by international organizations to provide common terminology for education phases.[xvi] It is described in Table 6.

Table 6. ISCED levels and their equivalent school level.

1	Primary education
2	Lower secondary education
3	Upper secondary education
4	Post-secondary non-tertiary education

These phases are mapped to ages, and each year is given a level. The age shown will be the maximum ages of students in a year group; for example, where Table 7 indicates Age 6, the age range for that level is ages 5 to 6. The levels map to the US "K" calibration.

Table 7. ISCED levels and their corresponding ages.

ISCED 2011 PHASES	AGE	LEVEL
1	6	1
	7	2
	8	3
	9	4
	10	5

Table 8 below sets out the key features of the MCSF as it applies to each phase. One of the MCSF's key competencies is "Design Thinking," and its inclusion is discussed in detail in the section entitled "Competencies, Design Thinking." Different levels of Design Thinking offer a way to structure the levels to reflect the broader goals of the MCSF. Students move through 4 levels of Design Thinking, lower to higher order thinking skills[xvii] - "Foundation, Product, Project and Business."

Table 8. Key features of the MCSF at each ISCED level.

ISCED 2011 PHASE		LEVELS OF THINKING	DESCRIPTION – MCSF
1	**PRIMARY EDUCATION**	**Foundation Level** Knowledge Comprehension • Interpreting • Explaining • Identifying *Factual - knowledge of terminology, specific details and elements*	Here, students should be provided with fundamental skills in CS and establish a solid foundation for learning. To enable students to follow clear learning pathways, the patterns of themes and topics established in Phase 1 are carried through to Phase 4. In most countries Phase 1 is less structured and rigidly timetabled as subsequent K-12 schooling phases, and the types of learning challenges presented to the students in this curriculum framework will reflect this.
2	**LOWER SECONDARY EDUCATION**	**Product Level** Apply • Implementing • Executing • Translating *Conceptual - Knowledge of classification and categories, principles, and generalizations*	Students in the first stage of secondary education will build on their primary CS education, following the same patterns and learning pathways established in Phase 1. The main difference between Phase 1 and Phase 2 is that CS will most likely be formally timetabled alongside other key subject areas.
3	**UPPER SECONDARY EDUCATION**	**Project Level** Analyze • De-constructing • Differentiating • Organizing *Conceptual - Knowledge of classification and categories, principles, and generalizations*	In the second stage of secondary education the MCSF specifies learning activities which prepare students for tertiary education and/or acquiring skills relevant to employment. Usually in this phase students will pick subjects from a range of options, so CS is likely to be competing with other subjects for student enrolment. The key difference between this phase and the previous phases is that students who select CS will now have more time to dedicate to the subject. This is reflected in the MCSF by an increase in the number of units that students would be expected to take in this phase.
4	**POST-SECONDARY NON-TERTIARY EDUCATION**	**Business Level** Synthesis • Coordinating • Critiquing • Testing *Procedural - Application of specific skills, techniques, and methods*	Here, the MCSF specifies learning experiences that build on Phase 3 and prepare students for the labor market; entry to tertiary education; or directly into entrepreneurship. Subject matter straddles upper secondary and tertiary education.

Compulsory CS education and the MCSF

There is very little international consistency in how countries apply their national curricula across year groups in primary and secondary education.[xviii] For example, in some countries CS is an option offered only at the secondary level, whereas in others, it forms part of the national, required curriculum throughout a child's primary and secondary school education. The MCSF curriculum is based on the expectation that Phases 1 and 2 are compulsory; in other words, all students in primary and lower secondary are entitled to take CS, while Phases 3 and 4 are optional. The MSCF is designed so that up to Level 10 the CS curriculum will be taken by all children.

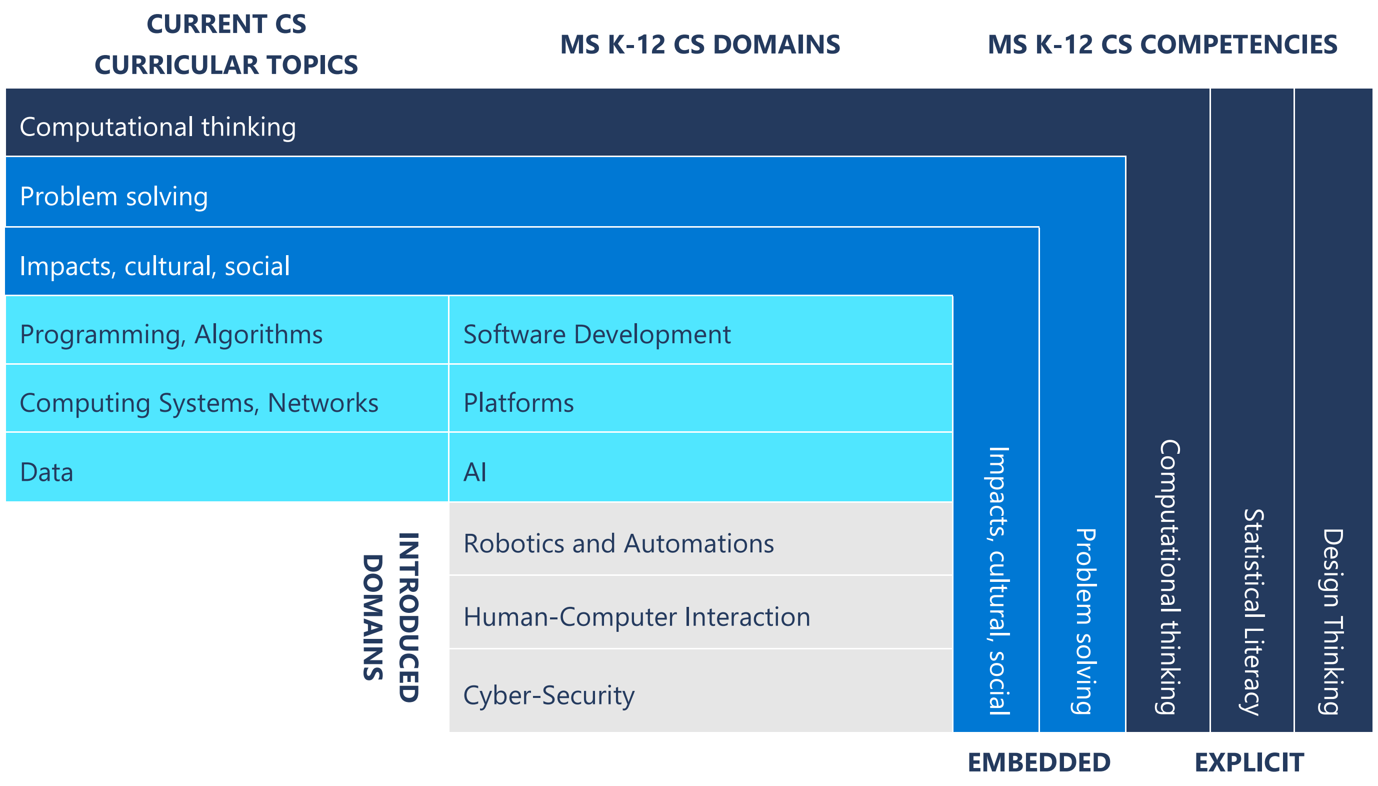
Between Levels 10 and 13, students elect to take the subject, so more time is spent in the curriculum compared to previous years.

Domains

This curriculum specifies six Domains. Three of these Domains align with established computer science curriculum topics. Three new Domains have been established to address currency and appeal problems with existing curricula. Each of these Domains and what they contain are explained in detail in section X.

Figure 1 shows how existing CS curricular topics (see Table 3) have been integrated into the curriculum, where new Domains have been created, and competencies added.

Figure 1. How CS curricular topics are integrated in the MCSF

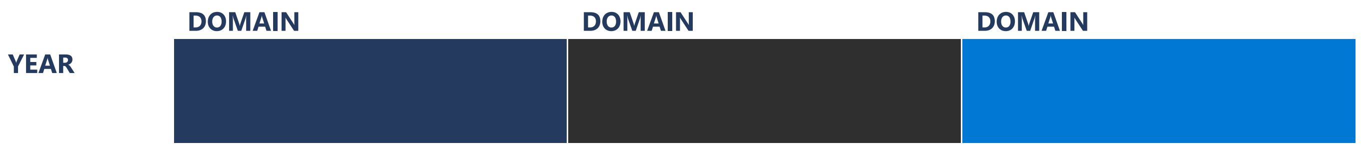


Domains in primary education

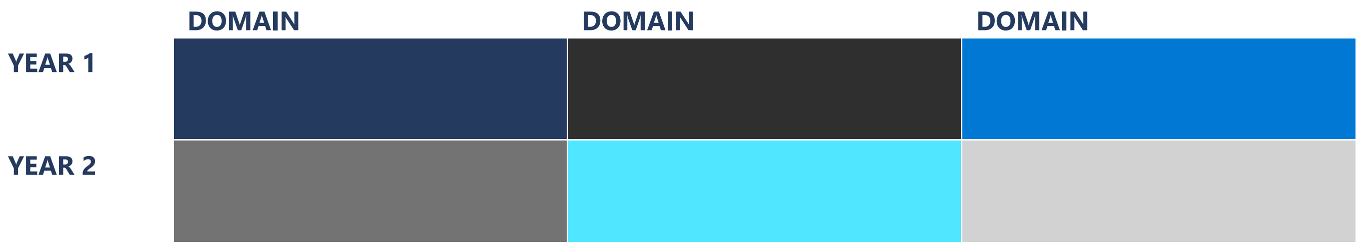
While each of the six domains listed in Table 2 are also present in Phase 1 of the MSCF, the terminology in the domain titles might seem daunting or overly complex for learners in primary school. For this reason, the six domains have been organized under three over-arching categories that are more suitable to primary school educators, learners, and parents. Those three categories are working with code, working with data, and working with computers. The domain organization underneath those categories is represented as follows.

WORKING WITH CODE		**WORKING WITH DATA**		**WORKING WITH COMPUTERS**	
SOFTWARE DEVELOPMENT	**ROBOTICS & AUTOMATION**	**DATA AND AI**	**PLATFORMS AND CLOUD**	**HUMAN-COMPUTER INTERACTION**	**CYBERSECURITY**
Solving Complexity	Sensing Your World	Solving Intelligence	Making Machines Compute	Making Computing Interactive	The Challenge of Digital Safety
Writing Programs	Controlling Your World	Learning from Data	Connecting Computers	Designing User Experiences	Securing Computing
Developing the Web	Making Environments Smart	Making AI Fair	Delivering Web Services	Making Computing Accessible	Infotagion

Most countries operate between 2 and 5 academic terms.[xix] Therefore three "Domains" are specified per year. Each year, students will complete a set of learning objectives for three Domains.



As there are six Domains in each Phase, three Domains are covered in one year, and the other three Domains are covered the following year.



This process repeats until Level 10.

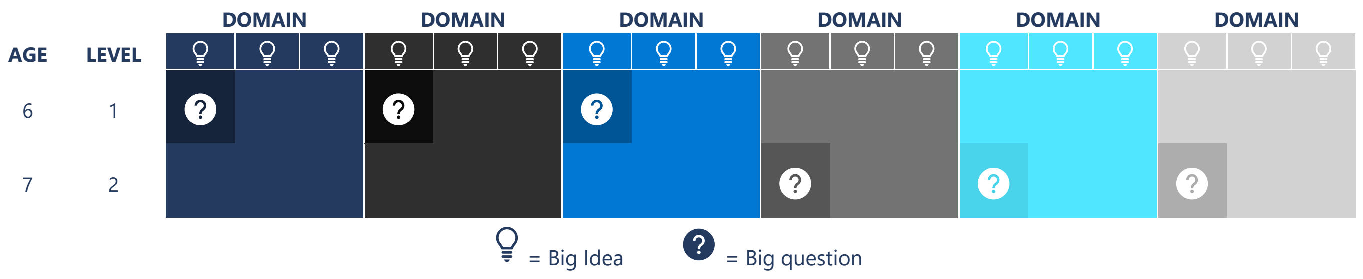
Big ideas

Each Domain contains three Big Ideas which contain learning pathways. An important feature of this curriculum framework is that learning pathways – Big Ideas - extend from Phase 1 to Phase 4.

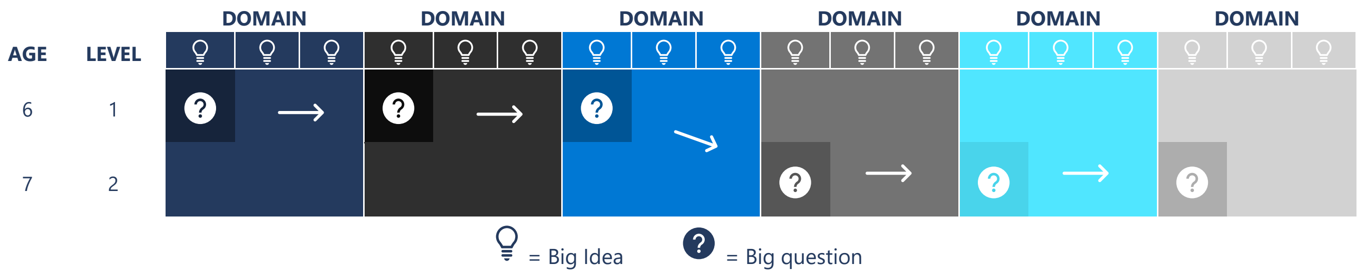
			DOMAIN		
	AGE	LEVEL	BIG IDEA 1	BIG IDEA 2	BIG IDEA 3
	6	1			
	7	2			
1	8	3	↓	↓	↓
	9	4			
	10	5			
	11	6			
	12	7			
2	13	8	↓	↓	↓
	14	9			
3	15	10	↓	↓	↓
	16	11			
4	17	12	↓	↓	↓
	18	13			

Big questions

Each Big Idea contains Big Questions. In Phases 1 and 2, students will work through two or three Big Questions per Domain. The graphic below shows the placement of the Big Questions in the darker boxes.

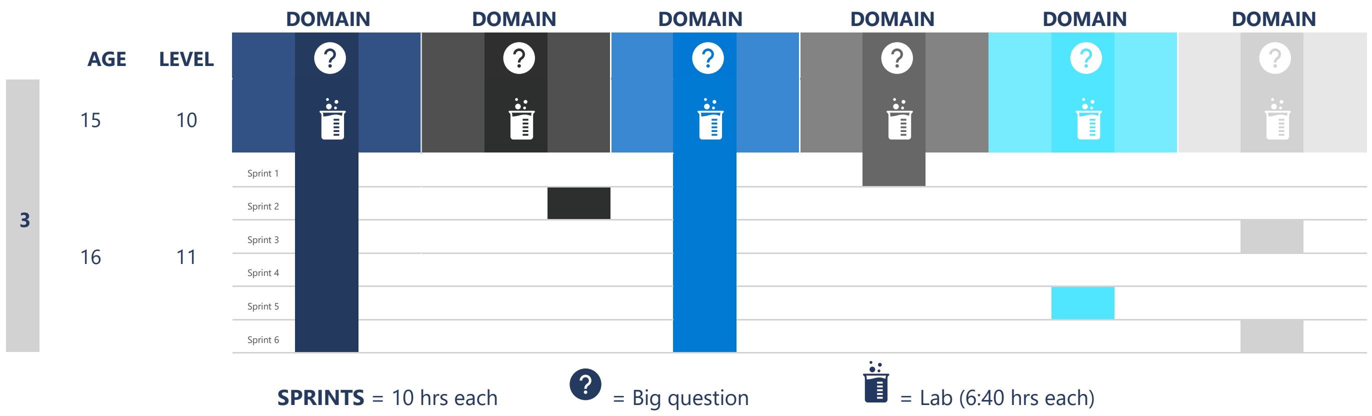


Chronological progression through the curriculum involves students working on one Big Question in one Domain at a time.

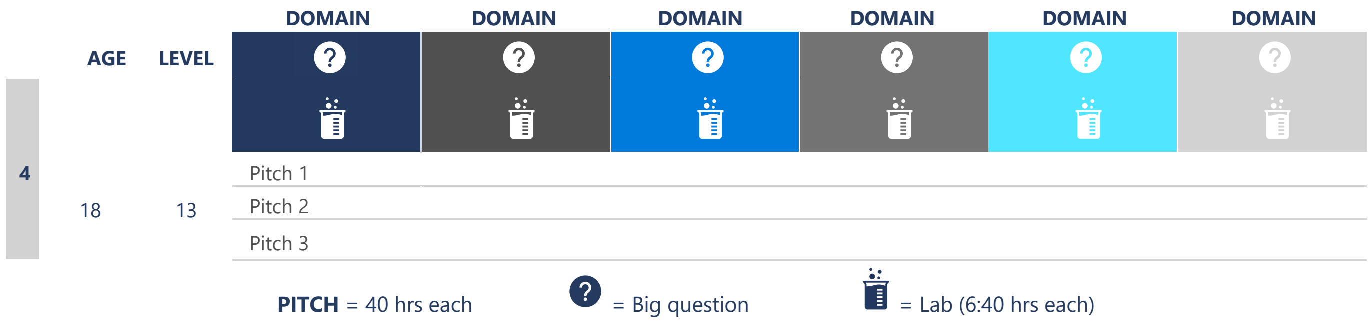


Three Big Questions are specified per year for Phases 1 and 2.

In Phase 3 (based on UK GCSE) year 1, the MCSF recommends six "labs" followed by six short open-ended projects in year two (one per half-term using the UK as a template). The Big Question of the lab is similar to what students would do in science and engineering subjects at University – i.e., experiments or practical activities with a specific learning goal.



In Phase 4 (based on UK A-Level) the MCSF specifies 18 "labs" for year 1, followed by three longer-term, open-ended projects in the final year.



Time allocation

To develop the following time allocation recommendations for each phase of the curriculum, the authors consulted guidelines provided by the two United Kingdom exam boards (OCR and AQA), the International Baccalaureate computer science curriculum, and multiple schemes of work provided to primary and secondary school educators on the Computing at School online community.[xx] These guidelines are to aid educators in their planning for implementing this curriculum; it is possible that the actual time required may differ slightly or may need to be adjusted to suit the requirements of a local education system.

PHASE	1-YEAR PROVISION	TIME EACH	TOTAL TIME
1	3 Big Questions	10 hrs	30 hrs
2	3 Big Questions	10 hrs	30 hrs
3-1	6 "Labs"	10 hrs	60 hrs
3-2	6 Projects (Sprints)	10 hrs	60 hrs
4-1	18 "Labs"	6:40 hrs	120 hrs
4-2	3 Projects (Pitches)	40 hrs	120 hrs

Learning Goals

Each Big Question sets out learning goals. Learning goals will begin with active verbs that refer to cognitive processes set out in k12cs.org's K-12 Computer Science Framework,[xxi] including, for example -

- Understand
- Appreciate
- Know

Concepts

Each Big Question specifies what key concepts should be learned.

Challenges

Each Big Question contains practical challenges. Each Challenge is designed to produce clear, assessible outcomes and enable students to acquire competencies in computational thinking, data literacy, and design thinking, as discussed in the following sections.

Computational Thinking

Andreas Schleicher, Director of the OECD Directorate for Education and Skills recommends focusing on the computational thinking skills that students can use to shape the technologies of tomorrow.[xxii] Key abilities include:

- Decomposition – breaking down a complex problem or system into smaller, more manageable parts.
- Pattern recognition – looking for similarities among and within problems
- Abstraction – focusing on the important information only, ignoring irrelevant detail
- Algorithms – developing a step-by-step solution to the problem, or the rules to follow to solve the problem

Data Literacy

Data Literacy is the ability to read, work with, analyze, communicate about, and question data and statistics. It is an essential skill for understanding and being able to explain the workings of AI, for example.

Design Thinking

Developed at the Stanford d.school,[xxiii] Design Thinking is a methodology that teaches individuals new strategies to solve problems. The design process challenges students to combine empathy, ingenuity, and rationality to meet user needs and create successful solutions with an innovator's mindset. Students are taught to defer judgment early in the process, which reduces fear of failure and encourages thinking outside the box.

A core innovation skillset, design thinking goes beyond problem solving. Design thinking is an iterative approach that uses prototyping for continuous user feedback and engagement and is an essential skill for developing products – a key goal of the MCSF.

Design Thinking is also a practical tool for integrating 21st century skills into the classroom. It makes direct connections between content students learn in class and the world beyond their school. Key steps in the process include:

- Empathize
- Define
- Ideate
- Prototype
- Test

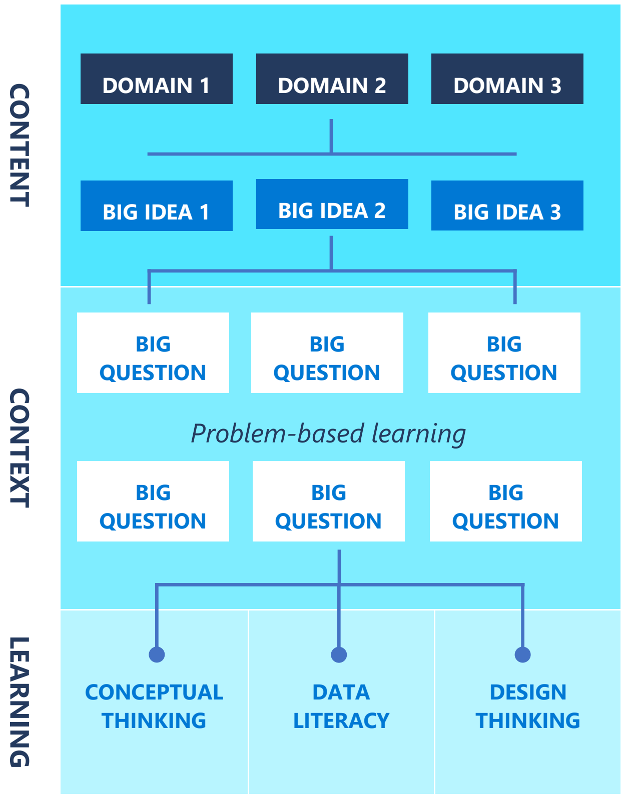
SUPPORTING CONTENT

Each Big Question includes recommendations for content to support the challenges, which originate from a wide range of sources. Local content can easily be used to replace or augment the examples provided.

Structure Summary

At the highest level, the curriculum framework structure is depicted in the following figures:

Context is provided through project-based learning approaches at the Big Question level. Content is organized at Domain levels, where each Domain contains three Big Ideas under which are the Big Questions. The end goal is the learning, which is defined by the acquisition of three competencies – Computational Thinking, Data Literacy, and Design Thinking.



ATTAINMENT TARGETS AND PROGRESSION

The MCSF attainment targets and progressions have two elements:

- Competencies
- Transversal skills and knowledge

Competencies

The MCSF is aligned to "Computing Progression Pathways, CAS[xxiv]" and "Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017.[xxv]"

However, MCSF goes beyond these standards with additional innovative thinking that reflects the need to emphasize areas such as AI, The Cloud, IoT, and emerging human-computer interfaces; and to encourage creativity and entrepreneurialism.

Therefore, the MCSF explicitly prescribes the following sets of competencies:

- Computational Thinking
- Data Literacy
- Design Thinking

Computational Thinking

The computational thinking descriptors are based on two sets of standards:

- Computer Science Teachers Association (CSTA) K-12[xxvi] – labeled "CTSA" in the following tables.
- Computing At School (CAS)[xxvii] – marked in blue, labeled "CAS" in the following tables.

Other descriptors are bespoke to the MCSF, and are marked in green and labeled "MCSF" in the following tables.

BY THE END OF PHASE 1 (FOUNDATION LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
1	Understand the process of computing – input, memory, process, output	MCSF	
1	Explain that computers collect data from various input devices, including sensors and application software	CAS	H&P 3
1	Understand that computers have no intelligence and that computers can do nothing unless a program is executed	CAS	H&P 1
2	Explain the role of electricity in computing systems and processes	MCSF	
3	Write sequences, events, loops, and conditionals	MCSF	
3	Create programs that include sequences, events, loops, and conditionals	CSTA	1A-AP-10
3	Use post-tested loops e.g., “until,” and a sequence of selection statements in programs, including an “if,” “then,” and “else” statement	CAS	P&D 3
3	Program robots	MCSF	
4	Make a cipher work	MCSF	
5	Construct static web pages using HTML and CSS	CAS	C&N 5

BY THE END OF PHASE 2 (PRODUCT LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
7	Manage complexity with diagrams, procedures, and tools to develop, organize, version, share and reuse code	MCSF	
7	Use C code functions, variables, and structures in a control solution	MCSF	
7	Apply practical experience of a high-level textual language	CAS	P&D 5
8	Describe how internal and external parts of computing devices function to form a system	CSTA	1B-CS-01

8	Understand the main functions of the operating system	CAS	H&P 4
8	Explain and use strong passwords to protect devices and information from unauthorized access	CSTA	1A-NI-04
8	Discuss real-world cybersecurity problems and how personal information can be protected	CSTA	1B-NI-05
8	Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts	CSTA	3A-NI-06
9	Develop programs that combine control structures, including nested loops and compound conditionals	CSTA	2-AP-12

BY THE END OF PHASE 3 (PROJECT LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
10	Use data analysis tools and techniques to identify patterns in data representing complex systems	CSTA	3B-DA-05
10	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects	CSTA	3A-AP-17
10	Access pre-existing functionality from standard libraries	MCSF	
10	Program an RPA process	MCSF	
10	Know the relative merits of different network methods, the role of Internet protocols and how packets, IP addresses, and memory work	MCSF	
10	Understand data transmission between digital computers over networks, including the Internet i.e., IP addresses and packet switching	CAS	C&N 5
10	Explain the names of hardware e.g., hubs, routers, switches, and the names of protocols e.g., SMTP, iMAP, POP, FTP, TCP/ IP, associated with networking computer systems	CAS	C&N 5
10	Explain security issues that may lead to compromised systems	CSTA	3B-AP-18

10	Apply multiple methods of encryption to model the secure transmission of information.	CSTA	2-NI-06
11	Apply prior descriptors to each of the six Sprints		

BY THE END OF PHASE 4 (BUSINESS LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
12	Understand the power of OOP for modelling real- world phenomena, create a DRY OO program with high modularity and extensibility	MCSF	
12	Apply JavaScript to create responsive, interactive webpages	MCSF	
12	Describe how artificial intelligence drives many software and physical systems	CSTA	3B-AP-08
12	Integrate different types of sensors in an autonomous system	MCSF	
12	Apply a range of mechatronic skills to solve robotics problems	MCSF	
12	Apply a range of IoT skills to solve smart city/smart environment problems	MCSF	
12	Write a Classical Machine Learning algorithm to classify Earth images	MCSF	
12	Demonstrate an understanding of number systems, computing electronics, memory, ICs, processor types, fetch-execute cycle	MCSF	
12	Create parameters for a Neural Network to solve a predictive modeling problem	MCSF	
12	Write code to ingest data from a public API	MCSF	
12	Explain the fundamentals of Quantum computing	MCSF	
13	Application of prior descriptors to each of the three Pitches		

DATA LITERACY

Data Literacy is the ability to understand and reason with statistics and data. It is an essential skill for understanding and being able to explain the workings of AI. These competencies also incorporate aspects of the Australian ACARA Digital Technologies Curriculum[xxviii] and Numeracy Progressions[xxix] as well as the CAS and CSTA standards described above.

BY THE END OF PHASE 1 (FOUNDATION LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
1	Collect, visualize, and explain patterns in data	MCSF	
1	Recognize that data can be structured in tables to make it useful	CAS	D&DR 1
1	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data	CSTA	1A-DA-05
1	Identify and describe patterns in data visualizations, such as charts or graphs, to make a prediction	CSTA	1A-DA-07
1	Demonstrate an understanding of the basic concepts of human intelligence	MCSF	
2	Know that digital computers use binary to represent all data, how it can represent numbers and images, how computers transfer data in binary relationship between binary and file size	CAS	D&DR 5
3	Identify your position and calculate the position of a landmark or object relative to your position in the real-world	MCSF	
3	Express probabilities numerically	MCSF	
3	Understand that data can be learned from	MCSF	
5	Explain how IoT can be combined with Cloud Computing to deliver aggregated data from around the world	MCSF	
5	Explain the concepts of bias and fairness in the context of AI and automation	MCSF	
5	Understand that personal data is collected, and it is important to keep this information secure	MCSF	

5	Explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access	CSTA	1A-NI-04
5	Keep login information private, and log off of devices appropriately	CSTA	1A-1C-18

BY THE END OF PHASE 2 (PRODUCT LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
6	Use units and scales of computing	MCSF	
7	Process Serial Data in a control solution	MCSF	
7	Understand how data can be used in the creative world	MCSF	
9	Create interactive data visualizations using software tools to help others better understand real-world phenomena	CTSA	3A-DA-11
9	Develop, apply, interpret, and communicate statistical models, judgements, and arguments	MCSF	
9	Use algorithms to make predictions by turning probability concepts into code	MCSF	
9	Apply combined control technologies in a Cloud-based IoT solution	MCSF	
9	Develop, apply, interpret, and communicate statistical models, judgements and arguments	MCSF	
9	Explain how data is encoded from a range of media, how it is used to make predictions, and the effects of bias	MCSF	
9	Discuss issues of bias and accessibility in the design of existing technologies	CSTA	2-IC-21
9	Query data on one table using a typical query language	CAS	D&DR 5

BY THE END OF PHASE 3 (PROJECT LEVEL), STUDENTS CAN:

LEVEL (K)	COMPETENCY DESCRIPTOR	STANDARD	REF
10	Create a computational model that represents the relationships among different elements of data collected from a phenomenon or process.	CSTA	3A-DA-12
10	Evaluate quality, authenticity and accuracy of data and extrapolate from a trend or pattern. Use a range of models and charting methods to analyze, predict and communicate data stories	MCSF	
10	Clean and prepare textual data for analysis and Machine Learning	MCSF	
10	Understand the role of mathematics in encryption	MCSF	
11	Apply prior descriptors to each of the six Sprints		

BY THE END OF PHASE 4 (BUSINESS LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
12	Explain what a relational database is, and understand the benefits of storing data in multiple tables	CAS	D&DR 8
12	Query, join, transform, and summarize data into useful information using a typical query language such as SQL	MCSF	
12	Demonstrate understanding of the principles of Deep Learning and its power, potential and limitations	MCSF	
12	Apply supervised learning methods to classify image data	MCSF	
13	Apply prior descriptors to each of the three Pitches		

DESIGN THINKING

Design Thinking is the ability to combine empathy, ingenuity, and rationality to meet user needs to solve problems. It is an essential creative skill for being able to develop solutions to problems and products. These competencies incorporate aspects of the CAS and CSTA standards described above.

BY THE END OF PHASE 1 (FOUNDATION LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
2	Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware)	CSTA	1A-CS-02
2	Model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination	CSTA	1B-NI-04
2	Explain the relative merits of different types of interface	MCSF	
2	Seek diverse perspectives for the purpose of improving computational artefacts	CSTA	1B-IC-20
2	Demonstrate safe and responsible computing	MCSF	
2	Compare how people live and work before and after the implementation or adoption of new computing technology	CSTA	1A-1C-16
4	Think of ways to improve the accessibility and usability of computing	MCSF	
4	Evaluate the trustworthiness of digital content and consider the usability of visual design features when designing and creating digital artefacts for a known audience	CAS	IT 6

BY THE END OF PHASE 2 (PRODUCT LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
6	Arrange elements of a Cloud-based stack, including a database, in a diagram	MCSF	
6	Recognize and understands the function of the main internal parts of basic computer architecture	CAS	H&P 5
6	Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users	CSTA	3A-AP-19
6	Recommend improvements to design of computing experiences	MCSF	

6	Establish ethical protocols for the online world	MCSF	
6	Compare trade-offs associated with computing technologies that affect people's everyday activities and career options	CSTA	2-IC-20
6	Describe trade-offs between allowing information to be public and keeping information private and secure	CSTA	2-IC-23
7	Manage complexity with diagrams, procedures and tools and know how to organize, version, share and reuse code	MCSF	
8	Use reverse engineering to understand a solution	MCSF	

BY THE END OF PHASE 3 (PROJECT LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
10	Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices	CSTA	2-CS-01
10	Apply principles of UX design	MCSF	
10	Understand the importance of fail-safe and zero errors in safety-critical systems	MCSF	
11	Produce use case scenarios, plan sequences, goals, and outcomes	MCSF	
11	Combine hardware and software network technologies	MCSF	
11	Modify, remix, or incorporate parts of an existing program	MCSF	
11	Apply Design Thinking – Empathize, Define, Ideate, Prototype, Test	MCSF	

BY THE END OF PHASE 4 (BUSINESS LEVEL), STUDENTS CAN:

Level (K)	Competency Descriptor	Standard	Ref
12	Collaboratively deploy and manage software artefacts on a code management platform (Git)	MCSF	
12	Explain the difference between Waterfall, Agile, and Scrum	MCSF	
12	Apply Business Model Canvas	MCSF	
12	Discuss the "five tribes of AI"	MCSF	

12	Demonstrate a working understanding of XAI and AI Ethics	MCSF	
12	Demonstrate a working understanding of the criticality of factoring energy use into a solution	MCSF	
12	Demonstrate a working understanding of haptics, tracking and other advanced HCI technology	MCSF	
12	Use User-centric Design methods	MCSF	
12	Factor-in accessibility when designing solutions	MCSF	
12	Design smart contract processes	MCSF	
12	Demonstrate an appreciation of technology law	MCSF	
13	Apply sound fundamentals of system organization, architecture, and solution design	MCSF	
13	Collaborate effectively in a software team using version control and continuous deployment for commerce-ready software	MCSF	
13	Effectively use prototyping	MCSF	
13	Pitch convincingly to investors	MCSF	

TRANSVERSAL SKILLS AND KNOWLEDGE

The following transversal skills and knowledge cut across all Domains, Big Ideas, and Big Questions:

Digital Literacy

By the time students are 7 years of age (level 2), they should be able to use technology purposefully and safely to create, organize, store, manipulate, and retrieve digital content.[xxx]

By age 10 (level 5) students should be able to use basic cloud services independently.

The Microsoft Digital Literacy Course (MDL) offers structured pathway to enable this. The table below shows how it maps to the MCSF Big Questions.

LEVEL (K)	BIG QUESTION	MDL UNIT	DESCRIPTION
2	What does “digital” mean?	1 & 2	Work with Computers Access information online
2	How can computers and people interact?	3 & 5	Communicate online Create digital content
2	How Can You Stay Safe Online?	4	Participate Safely and Responsibly Online
5	How can we connect sensors across schools?	6	Collaborate and Manage Content Digitally

Completing the MDL course by the end of the Foundation Phase (Phase 1), means that students will have the key basic digital literacy skills that they need to be successful in the MSCF and across the rest of their schooling.

Information Technology

Information Technology (IT) focusses on using computers whilst the focus of CS is on the science of computing and being able to be create computing solutions. CS requires extensive use of IT, so the use of a wide range of IT tools and services is embedded within the MCSF, including those listed in the following table.

PHASE	IT Tools and services
FOUNDATION	Windows, Office, PDF, MakeCode, M:EE, OneDrive, sensors, Azure
PRODUCT	Excel, Visio, Project, Arduino IDE, MR interfaces, Raspberry Pi, Azure Notebooks, Visual Studio
PROJECT	IDLE, RPA, bots, network management technologies, encryption
BUSINESS	GitHub, UML, SQL, Azure Machine Learning, Blockchain, Haptics

FUTURE-READY SKILLS

Skills relating to cloud computing, artificial intelligence, machine learning, productivity and more are already in demand in organizations around the world, yet jobs remain unfilled, and the talent gap persists and is set to widen. More than half of today's jobs require technology skills, but in less than a decade that number will grow to more than 77%.[xxxi]

The Microsoft Future-Ready Skills program provides a framework to support students and educators from K-Career to be ready to innovate and create in an increasingly digital world. The following table shows how the MCSF maps directly to the Microsoft Future-Ready Skills program.

PHASE	FOUNDATION	PRODUCT	PROJECT	BUSINESS
FUTURE-READY SKILL	Block-based visual programming	Text-based programming		
	Digital Literacy	Productivity		
		Cloud		
	Modern skills – communication, collaboration, creativity, critical and computational thinking			
	Computer Science Fundamentals			
	Programming and design thinking			
	Technology ethics and data concepts			

TROUBLESHOOTING

Core attributes required by students learning how to create technology as future practitioners of CS are resilience and the ability to troubleshoot.

The MSCF also maps to the development of troubleshooting skills as described by the Computer Science Teachers Association (CSTA) standards from the United States as indicated in the following table.

PHASE	TROUBLESHOOTING SKILL	CSTA STANDARD
FOUNDATION	Describe basic hardware and software problems using accurate terminology	1A-CS-03
PRODUCT	Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies	1B-CS-03
PROJECT	Systematically identify and fix problems with computing devices and their components	2-CS-03
BUSINESS	Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors	3A-CS-03

IMPACTS OF COMPUTING

In addition to the technical and career-focused skills that the MCSF aims to develop, it's also essential that students understand the impacts of computing on both themselves and other people and develop awareness of what counts as acceptable or otherwise.

The development of the awareness of the impacts of computing, as described by CSTA, is depicted in the following table.

PHASE	IMPACT OF COMPUTING KNOWLEDGE AND SKILLS	CSTA
FOUNDATION	• Compare how people live and work before and after the implementation or adoption of new computing technology • Work respectfully and responsibly with others online • Keep login information private, and log-off devices appropriately	1A-IC-16 1A-IC-17 1A-IC-18
PRODUCT	• Discuss computing technologies that have changed the world and express how those technologies influence, and are influenced by, cultural practices • Brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users • Seek diverse perspectives for the purpose of improving computational artefacts • Observe intellectual property rights and give appropriate attribution when creating or remixing programs	1B-IC-18 1B-IC-19 1B-IC-20 1B-AP-14
PROJECT	• Compare trade-offs associated with computing technologies that affect people's everyday activities and career options • Discuss issues of bias and accessibility in the design of existing technologies • Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artefact • Describe trade-offs between allowing information to be public and keeping information private and secure	2-IC-20 2-IC-21 2-IC-22 2-IC-23
BUSINESS	• Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices • Test and refine computational artifacts to reduce bias and equity deficits • Demonstrate ways a given algorithm applies to problems across disciplines • Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields • Explain the beneficial and harmful effects that intellectual property laws can have on innovation • Explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users • Evaluate the social and economic implications of privacy in the context of safety, law, or ethics	3A-IC-24 3A-IC-25 3A-IC-26 3A-IC-27 3A-IC-28 3A-IC-29 3A-IC-30

---

[i]The event, *Hacking future skills: Computer Science Education*, was held online on 25 June 2020.

[ii]See Dochy et al, 2003, Merritt et al, 2017, and Holmes, V. & Hwang, Y. (2016). Exploring the effects of project-based learning in secondary mathematics education. *The Journal of Education Research,* 109(5), 449-463.

[iii]Dochy et al, 2003.

[iv]Holmes & Hwang, 2016.

[v]Merritt et al, 2017.

[vi]Ibid.

[vii]Ashcraft, C., Eger, E., Friend, M. (2012). *Girls in IT: The facts*. National Center for Women and Information Technology. Accessed from: https://www.ncwit.org/sites/default/files/resources/girlsinit_report2012_final.pdf

[viii]Weatherby, K. (2017), *Teacher participation in online communities of practice: a mixed*-methods *study of community, context and practice*, University College London, London. Accessed from: https://discovery.ucl.ac.uk/id/eprint/1566655/1/Weatherby_2017%20K%20Weatherby%20PhD%20_-%20FINAL.pdf

[ix]https://www.un.org/development/desa/disabilities/envision2030.html

[x]Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn* (Vol. 11). Washington, DC: National Academy Press.

[xi]Ibid.

[xii]Wiggins, G., & McTighe, J. (2005). *Understanding by design*. Alexandria, VA: Association for Supervision and Curriculum Development.

[xiii]Darling-Hammond, L., & Bransford, J. (Eds.). (2007). *Preparing teachers for a changing world: What teachers should learn and be able to do*. San Francisco, CA: John Wiley & Sons.

[xiv]Ibid.

[xv]Bransford et al, 2000

[xvi]http://uis.unesco.org/sites/default/files/documents/international-standard-classification-of-education-isced-2011-en.pdf

[xvii]https://www.researchgate.net/publication/283550497_Design_Thinking_pedagogy_the_Educational_Design_Ladde
r

[xviii]There is no http://www.mempowered.com/children/international-curricula

[xix]https://en.wikipedia.org/wiki/Academic_term

[xx] https://www.computingatschool.org.uk/

[xxi] https://k12cs.org

[xxii] https://oecdedutoday.com/should-schools-teach-coding/

[xxiii] https://static1.squarespace.com/static/57c6b79629687fde090a0fdd/t/5b19b2f2aa4a99e99b26b6bb/1528410876119/dschool_bootleg_deck_2018_final_sm+%282%29.pdf

[xxiv] https://www.tes.com/teaching-resource/computational-thinking-guide-and-progression-pathways-6436712

[xxv] https://www.esd105.org/cms/lib/WA01920102/Centricity/Domain/175/2017%20CSTA%20K-12%20Standards%20Progression%20Chart%20CT.pdf

[xxvi] https://csteachers.org/Page/standards

[xxvii] https://csteachers.org/Page/standards

[xxviii] https://www.australiancurriculum.edu.au/f-10-curriculum/technologies/digital-technologies/

[xxix] https://www.australiancurriculum.edu.au/resources/national-literacy-and-numeracy-learning-progressions/

[xxx] https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/239033/PRIMARY_national_curriculum_-_Computing.pdf

[xxxi] https://educationblog.microsoft.com/en-us/2019/12/empowering-students-to-be-future-ready/