How to define a coherent curriculum for K-12: the example of France

Gilles Dowek

http://www.societe-informatique-de-france.fr/
1024-no9-enseigner-linformatique-de-la-maternelle
-a-luniversite-a-paraitre/

Why defining a curriculum?

I. The situation of informatics education in France

Prehistory (before 2012)

In the 80's and 90's: an optional course in high school

Removed twice

For a couple of decades: no informatics in French "general" schools

2012-2016

2012: Elective course in 12th grade for “science” students
2013: Course in preparatory classes
2015: Elective discovery course in 10th grade
2016:

- Notions in kindergarten, elementary school, and 6th grade
- Course in middle school (7th, 8th, and 9th grade)
- Optional course in 11th and 12th grade

More diverse situation in technical and professional high school

Why was it removed in the 90's?

Lack of math teachers (who also taught informatics)

No need for computer scientists: "At the economic level, are we in such a need of computer scientists that we should incentive a large number of young people to go, very early, on this track?", *Quel lycée pour demain ?*, 1991, Conseil National des Programmes, p. 140

Informatics had changed: no need to program computers anymore, nothing to understand, just use (use the disruption narrative carefully)

Why was it put back in 2012?

- Pervasiveness of informatics: professional life, personal life, democratic life (e.g. recent state surveillance)
- University professors: teaching tests and loops to 20 y.o. students
- *La main à la pâte* (Hands on), *Castor informatique* (Beaver), IOI (Olympiads)...
- Many off school training programs (for children, students...)
- Academy of sciences: 2005, a one day seminar (Maurice Nivat)
- *Collège de France*: Gérard Berry's course 2007
- Action of high-school informatics teacher professional organization (EPI), of the *Société informatique de France*, from 2007
- A report of the Academy of sciences, of the *Conseil National du Numérique* 2013
- An open letter to the President (signed by two former Prime ministers) 2014
- ...

Names

No way to call it "Informatics"

It would disgust non "science / technology oriented" students ("such as women")

But pleonasm does not disgust our education department "Informatics and digital sciences", "Informatics and digital creation"

Implicit: teach science and technology to "science / technology oriented" students and not to the others

Explicit: "The main problem of school is not informatics, it is reading", Bayrou, "From 11th grade the teaching is organized around three mandatory courses : French / Philosophy, History and Geography, and English, plus optional courses" Le Maire

To have and have not

Students $\checkmark$
Curricula $\checkmark$

Teachers $\times$
Time $\times$

Teachers

Kindergarten, elementary school: one teacher teaches everything
Solution: a little bit of informatics in teachers' initial training × +
a little bit of informatics as continued learning ✓(Class'Code)

Middle, high school: specific teachers
Solution: recruit informatics teachers ×
Next year: an informatics option for the math teachers' recruiting exam (informatics teachers?)

Teachers of another topic (maths, technology...)
Taken (passed?) a continued learning program: 3 to 300 hours
(other subjects: 5 years in university)

Time

Kindergarten, elementary school: one teacher organizes the schedule

Middle school: split between the math course and the technology course

High school: optional or elective

Why not a subject

Three groups of people want to limit the development of informatics in schools

Teachers of other sciences (e.g. math)
Those who fight against sciences and technology in general
Those who believe they know what informatics is (and easy)

II. Building a coherent curriculum

A new problem

Build a curriculum for a 15 level structure, without repetitions

A possible disaster: `for` loop the 1st year, `for` loop the 2nd year, ..., `for` loop the 15th year

This already happened:

- in middle and high school continental drift six times (geography and earth sciences)
- at university: termination of simply typed lambda-calculus three times

A problem for the *Conseil Supérieur des Programmes*
Also a problem for the *Société informatique de France*

The principle: three equilibria

Between the four concepts of informatics: algorithm, machine, language, and information

Between science and technology

Between the core of the subject and its interactions

Equilibrium I: the four concepts of informatics

What is informatics?

A science defined by the objects it studies (e.g. life) and the method it uses (e.g. experimental method)

Informatics studies several kind of objects: languages, information, machines, algorithms

$$\text{algorithm} + \text{machine} \longrightarrow \text{language} + \text{information}$$

A stable definition of informatics
(Method-based definition possible but more difficult)

The four concepts

Languages: programming language constructions, other languages, grammars

Information: representing numbers, texts (ascii, unicode, html), images, sounds, Boolean functions, structuring information (file systems, databases, Web), compression, cryptography, error-correction

Machines: Boolean gates, time and memory, structure of a computer, distributed computing, networks (internet), robots

Algorithms: addition, multiplication, binary decomposition, sorting and searching, graphic algorithms, graph algorithms

More unity needed

Robots: machines, process information (through sensors...), implement original algorithms (feed-back, analog to digital transformations...), led to (reactive) programming languages
Thus a part of informatics
Yet, some think robotics is not part of informatics
Interaction with mechanics: an autonomous science

Same for numerical analysis

Broader "numerical sciences" (while "informatics" broad enough)

In curricula

What do math teachers want to teach?

What do technology teachers want to teach?

An example: algorithm

Primary school: the systematic character of known methods: cooking recepies, weighing an object with a Roberval balance...

Middle school: basic algorithms, basic principles (binary decomposition, generate and test...)...

High school: classical algorithms (text procesing, image processing...), complexity of an algorithm...

Equilibrium II: science and technology

Informatics is both
answering questions: e.g. what is the complexity of binary search?
building objects with a purpose: e.g. binary search programs

Students should both
learn that binary search is logarithmic
build their own binary search program

Viewpoints to be fought

A science, nothing to do with technology: I cannot fix your computer (I cannot change a wheel), astronomy and telescopes

Only a technology: nothing to learn, just do

Equilibrium III: the core of the subject and its interactions

Informatics is everywhere, but everything is not informatics

Use a digital pH meter, Photoshop...

Informatics is a branch of no other science or technology

Informatics is not part of maths, not part of technology, not part of physics, not part of linguistics, not part of media studies...

But it has interactions with other sciences: mathematics, physics, linguistic, archaeology...
Dilution vs. interdisciplinary projects

Viewpoints to be fought

A part of mathematics (constructive, discrete...)

It is not autonomous: always coupled with a mechanical system (e.g. in a car)... complex systems

Society questions

E.g. privacy and social media

Are important

Provided

- they are in interaction with scientific and technological content
  Information: travels at velocity c, is duplicated at cost 0, is there for ∞ time
  vs. “Do not do this”, “be very careful when doing that”...
- they do not eat the full cake

Organizing teaching in time

Imagine you teach RSA: public / private key, Fermat's little theorem, bla
Two possible conclusions
(1) let us program it, and play with it
(2) computer scientists use this to encrypt messages

Learning how to write a program ("to code") is a key step to autonomy: from spectator to actor

Three steps:

1. before learning how to program (Kindergarten, elementary school)
2. learning how to program (middle school)
3. after learning how to program (high-school and after)

Before:

- Computer science unplugged (north-south-east-west...)
- Discovering mail, Web, text-processing... (excellent source of questions)
- Interaction with other subjects: algorithms in math, cooking recipes orientation in space...

After: informatics as we know it (compilation, cryptology, complexity, machine architecture...)

A structured and sustainable curriculum

Keep a balance between the four concepts

Keep a balance between science and technology

Keep a balance between the core of the subject and its interfaces

Learn informatics in three steps