

Millionex17

Castellón de la Plana

IV Jornada sobre
Innovación Educativa utilizando TICs
10 de mayo de 2017

El pensamiento computacional en el currículum

Jordi Adell

Universitat Jaume I

 @jordi_a

FOOLA

temas:

- ¿Qué es el pensamiento computacional (PC).
¿Alfabetización universal? Elementos o componentes del PC.
- La integración en el currículum. ¿Por qué? ¿Cómo?
Actividades didácticas: *unplugged*, juegos, programación, robótica.
- Las críticas al PC.
- La formación del profesorado en PC.
- Conclusiones/Debate.

¿Qué es el “pensamiento computacional” (PC)?



toteltitular: "computational thinking"



Acadèmic

Aproximadament 1.210 resultats (0,10 s)

Les meves cites

i1.210!

Articles

[PDF] **Computational thinking.**

JM Wing - Communications of the ACM, 2006 - microsoft.com

• Abstract interpretation in systems biology• Model checking applied to arrhythmia, diabetes, pancreatic cancer• DNA sequences are strings in a language• Boolean networks approximate dynamics of biological networks• Cells as a self-regulatory system are like

Citat per 2697 Articles relacionats Totes les 114 versions Importa a BibTeX Desa Més

Computational thinking and thinking about computing

JM Wing - ... transactions of the royal society of ..., 2008 - rsta.royalsocietypublishing.org

La meva biblioteca

En qualsevol moment

Des de 2017

Des de 2016

Des de 2013

Interval específic...

Ordena per rellevància

Ordena per data

Qualsevol idioma

Cerca pàgines en anglès i castellà

patents incloses

inclou cites

Crea una alerta

Abstract Jeannette Wing's influential article on **computational thinking** 6 years ago argued for adding this new competency to every child's analytical ability as a vital ingredient of science, technology, engineering, and mathematics (STEM) learning. What is **computational**

Citat per 284 Articles relacionats Totes les 11 versions Importa a BibTeX Desa Més

[PDF] **New frameworks for studying and assessing the development of computational thinking**

K. Branan, M. Resnick ... of the 2012 annual meeting of ... 2012 ... scratched.goe.harvard.edu



"computational thinking"

Acadèmic

Aproximadament 11.500 resultats (0,08 s)

i11.500!

Articles

[PDF] **Computational thinking.**

JM Wing - Communications of the ACM, 2006 - microsoft.com

La meva biblioteca

• Abstract interpretation in systems biology• Model checking applied to arrhythmia, diabetes, pancreatic cancer• DNA sequences are strings in a language• Boolean networks approximate dynamics of biological networks• Cells as a self-regulatory system are like

En qualsevol moment

Citat per 2697 Articles relacionats Totes les 114 versions Importa a BibTeX Desa Més

Des de 2017

Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.



Computational thinking builds on the power and limits of computing.

cisely. Stating the difficulty of a problem accounts for the underlying power of the machine—the computing device that will run the solution. We must

Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, what is appropriately incestuous about this vision is that computing and computers facilitate the spread of computational thinking.

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.

Having to solve a particular problem, we might ask: How difficult is it to solve? and What's the best way to solve it? Computer science rests on solid theoretical underpinnings to answer such questions pre-

ization of dimensional analysis. It is recognizing both the virtues and the dangers of aliasing, or giving someone or something more than one name. It is recognizing both the cost and power of indirect addressing and procedure call. It is judging a program not just for correctness and efficiency but for aesthetics, and a system's design for simplicity and elegance.

Computational thinking is using abstraction and decomposition when attacking a large complex task or designing a large complex system. It is separation of concerns. It is choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable. It is using invariants to describe a system's behavior succinctly and declaratively. It is having the confidence we can safely use, modify, and influence a large complex system without understanding its every detail. It is

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.

Pero hay más...

“El Pensamiento Computacional son los procesos de pensamiento implicados en la formulación de problemas y sus soluciones para que estas últimas estén representadas de forma que puedan llevarse a cabo de manera efectiva por un procesador de información" (Wing, 2011, p. 1)

Es decir...

1. El Pensamiento Computacional es un proceso de pensamiento, por lo tanto **independiente de la tecnología.**

2. El Pensamiento Computacional es **un tipo específico de resolución de problemas** que implica capacidades distintas, por ejemplo, ser capaz de diseñar soluciones para ser ejecutadas por un ordenador, un humano, o una combinación de ambos.

Pero sigue sin haber
acuerdo...

Submitted: SIGCSE 2014, 5-8 March, Atlanta GA

Computational Thinking: The Developing Definition

Cynthia C. Selby
University of Southampton
Highfield
Southampton UK
44 (0) 2380 593475
C.Selby@soton.ac.uk

John Woollard
University of Southampton
Highfield
Southampton UK
44 (0) 2380 592998
J.Woollard@soton.ac.uk

Table 1 summarizes the justification for each prospective term's inclusion in or exclusion from a proposed definition of computational thinking.

Term	Status	Justification
A thought process	Include	Consensus found in the literature
Abstraction	Include	Consensus found in the literature
Decomposition	Include	Consensus found in the literature
Logical thinking	Exclude	Broad term, not-well defined
Algorithmic thinking	Include	Well-defined across multiple disciplines
Problem solving	Exclude	Broad term, evidences the use of skills; develops acquisition of skills
Evaluation	Include	Well-defined across multiple disciplines
Generalization	Include	Well-defined concept, although the term may not be familiar
Systems design	Exclude	Evidences the use of skills
Automation	Exclude	Evidences the use of skills
Computer science content	Exclude	Evidences the use of skills
Modeling, simulation, and	Exclude	Evidences the use of skills in their creation; manipulation develops acquisition of skills

Table 1. Computational Thinking Definition Terminology

6. CONCLUSION

There is a genuine need for a robust and agreed definition of computational thinking. The definition can facilitate the development of computer science curriculums in line with Wing's original vision to encourage computational thinking for all. The definition may also ensure that the K-12 curriculums will not become just a collection of interesting resources presented at teachers' discretions.



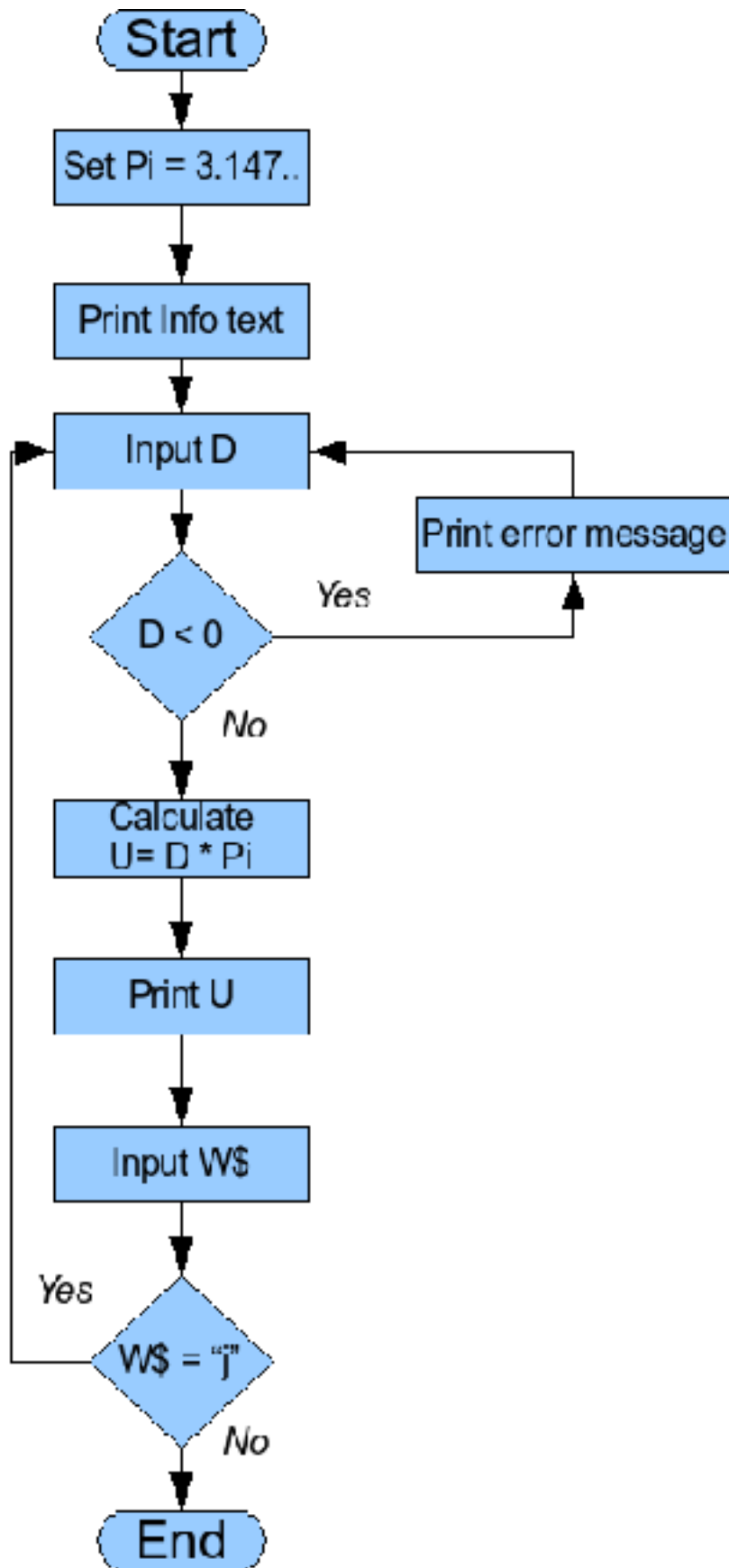
 **COMPUTATIONAL THINKING** teacher resources second edition



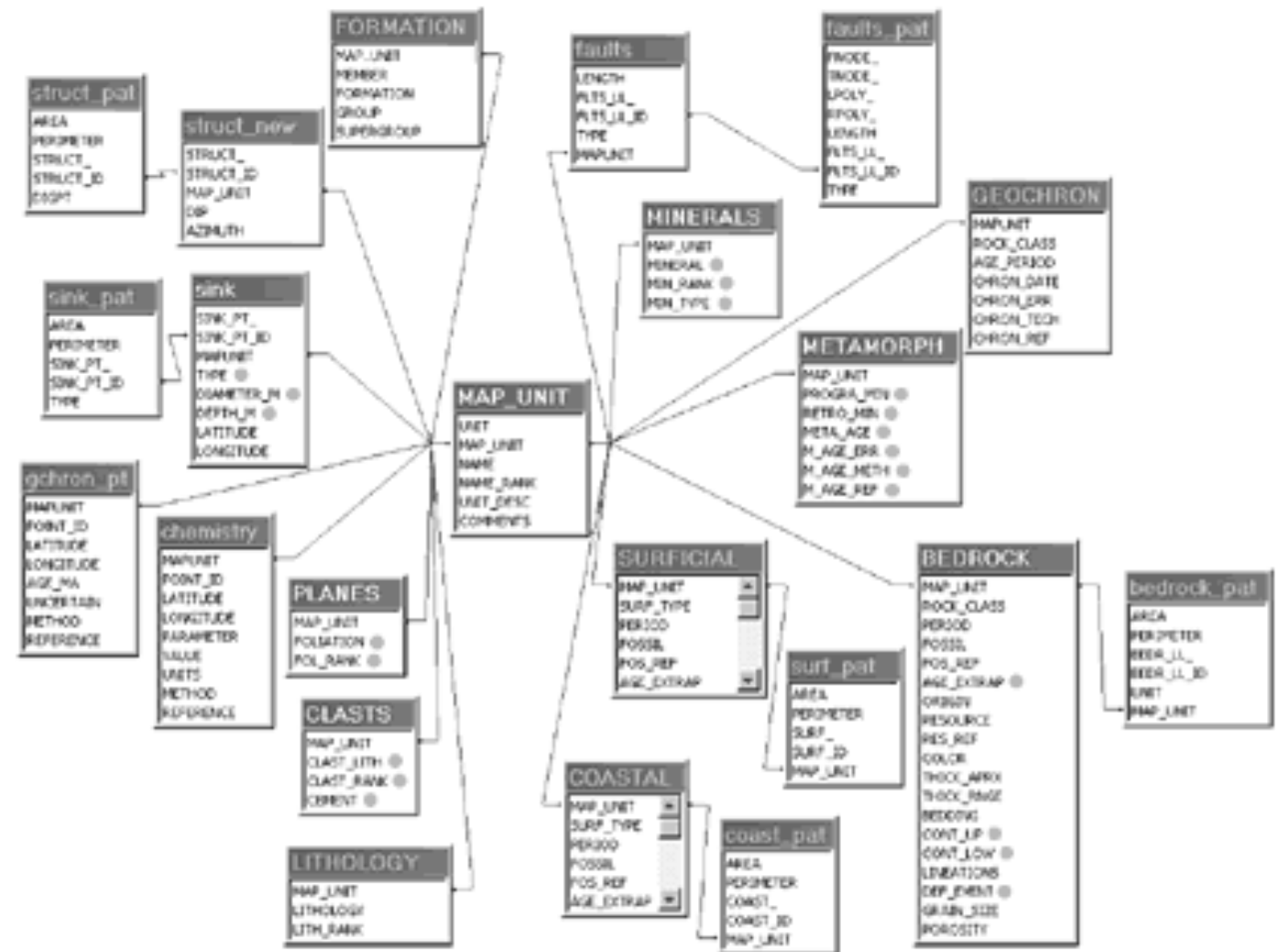
Computer Science Teachers Association & International Society for Technology in Education (CSTA & ISTE, 2009, p.1)

Definición “operacional”

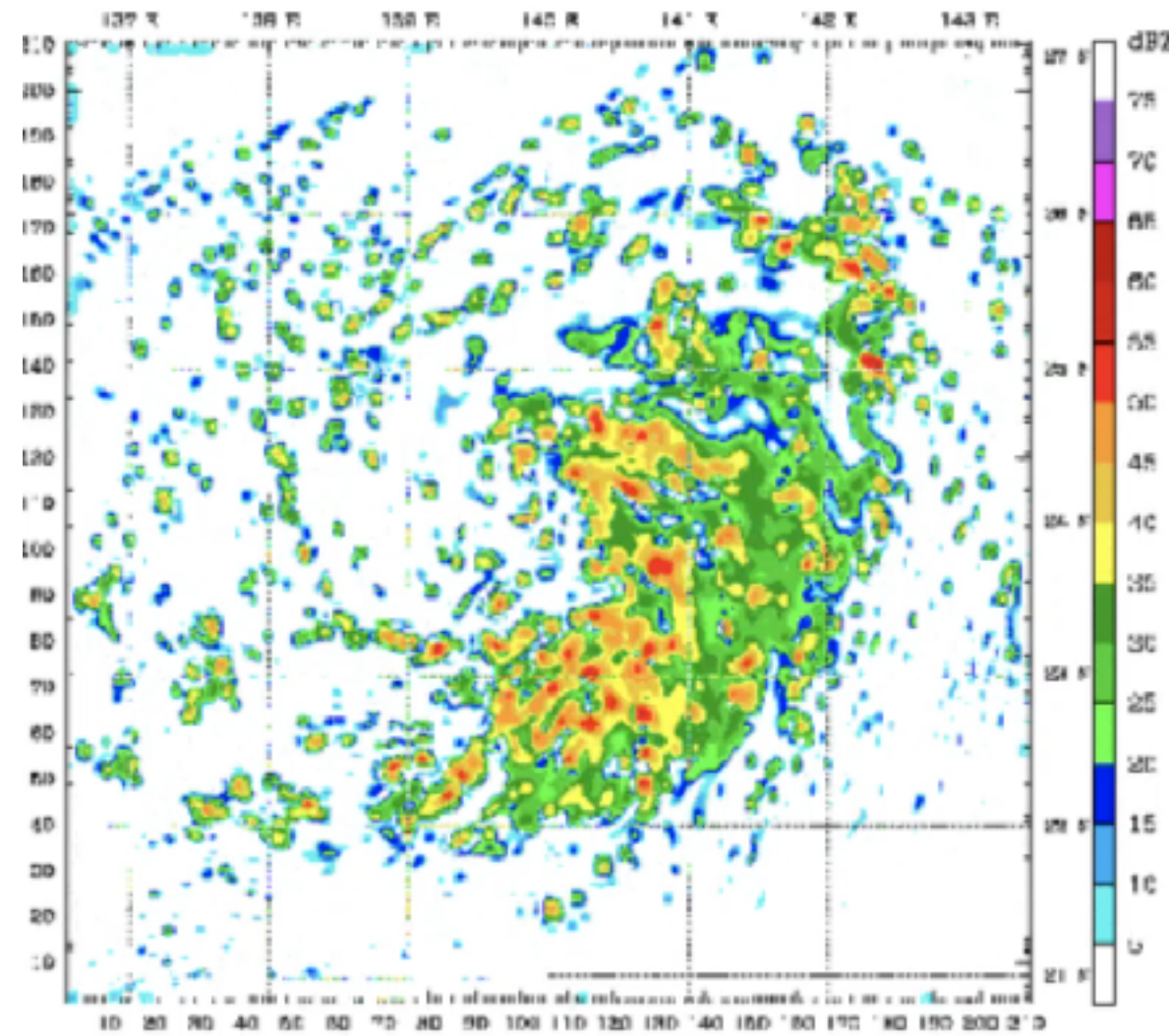
El *pensamiento computacional* es un proceso de resolución de problemas que incluye (pero no se limita a) las siguientes características:



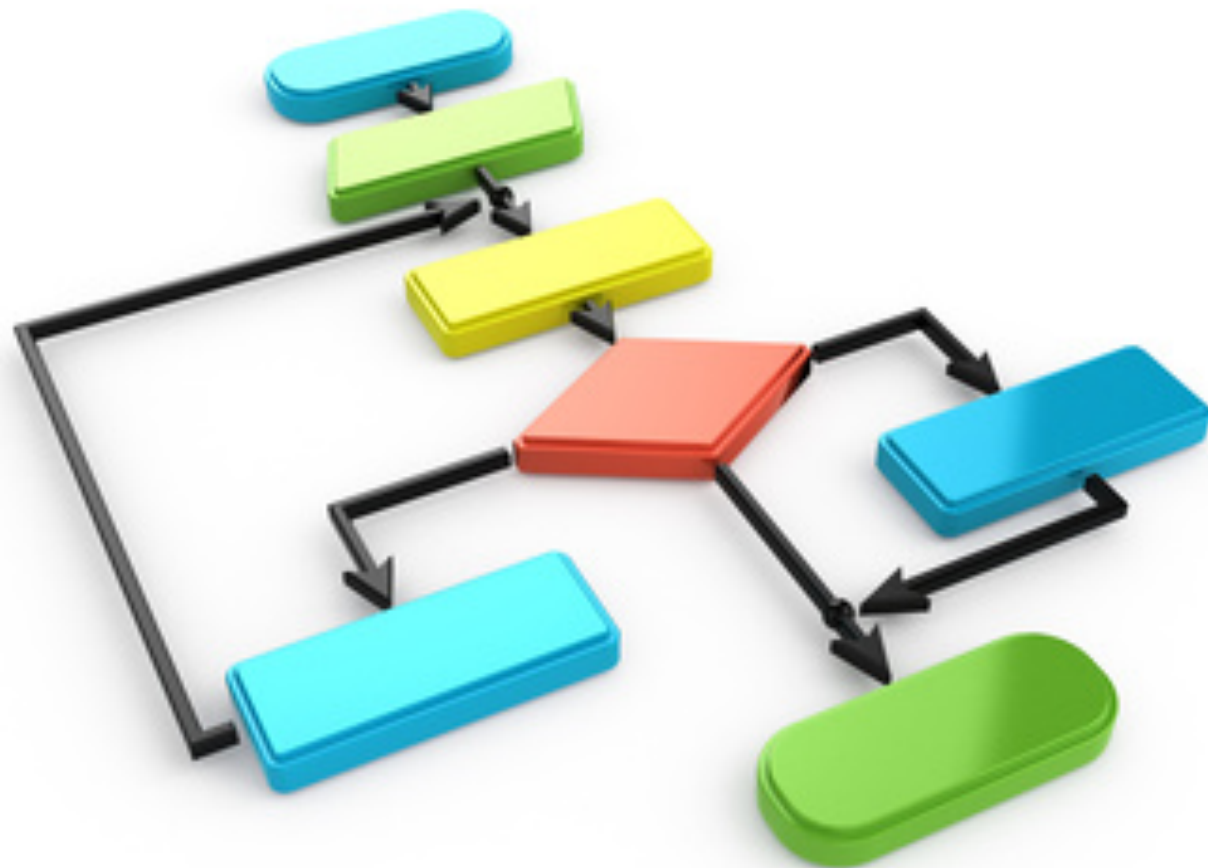
1. Formular problemas de manera que se pueda utilizar un ordenador y otras herramientas para ayudar a resolverlos.



2. Organizar y analizar datos de una manera lógica.



3. Representar datos mediante abstracciones tales como modelos y simulaciones.



4. Automatizar soluciones mediante el pensamiento algorítmico (una serie de pasos ordenados).



5. Identificar, analizar y aplicar posibles soluciones con el objetivo de conseguir la combinación más eficaz de pasos y recursos.



6. Generalizar y transferir este proceso de solución de problemas a una amplia variedad de tareas y problemas.

Y además, las siguientes actitudes:

- La confianza para tratar con la complejidad.
- La persistencia en el trabajo con problemas difíciles.
- Tolerancia a la ambigüedad.
- La capacidad para hacer frente problemas abiertos.
- La capacidad de comunicarse y trabajar con otros para alcanzar una meta o solución común.

The Computational Thinker: Concepts & Approaches

Concepts

Logic
predicting & analysing

Algorithms
making steps & rules

Decomposition
breaking down into parts

Patterns
spotting & using similarities

Abstraction
removing unnecessary
detail

Evaluation
making judgement



Tinkering
experimenting & playing

Creating
designing & making

Debugging
finding & fixing
errors

Persevering
keeping going

Collaborating
working together

Approaches

La integración en el currículum



JRC SCIENCE FOR POLICY REPORT

Developing Computational Thinking in Compulsory Education

*Implications for policy and
practice*

Authors: Stefania Bocconi, Augusto Chioccariello,
Giuliana Dettori, Anusca Ferrari, Katja Engelhardt

Editors: Panagiotis Kampylis, Yves Punie

2016

[http://publications.jrc.ec.europa.eu/
repository/bitstream/JRC104188/
jrc104188_computhinkreport.pdf](http://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf)

El Pensamiento Computacional en la Enseñanza Obligatoria (Computhink)

Implicaciones para la política y la práctica

Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado (INTEF)
Departamento de Proyectos Europeos

Febrero 2017

<http://educalab.es/intef> @educalNTEF <http://educalab.es/blogs/intef/>



Kindergarteners Learning to Code, de [Kevin Jarret](#), en [Flickr](#), con licencia [CC BY 2.0](#)

Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K. (2016). [Developing computational thinking in compulsory education – Implications for policy and practice](#); EUR 28295 EN; doi: 10.2791/792158



Esta obra está bajo una licencia [Creative Commons Atribución-CompartirIgual 3.0 España](#)

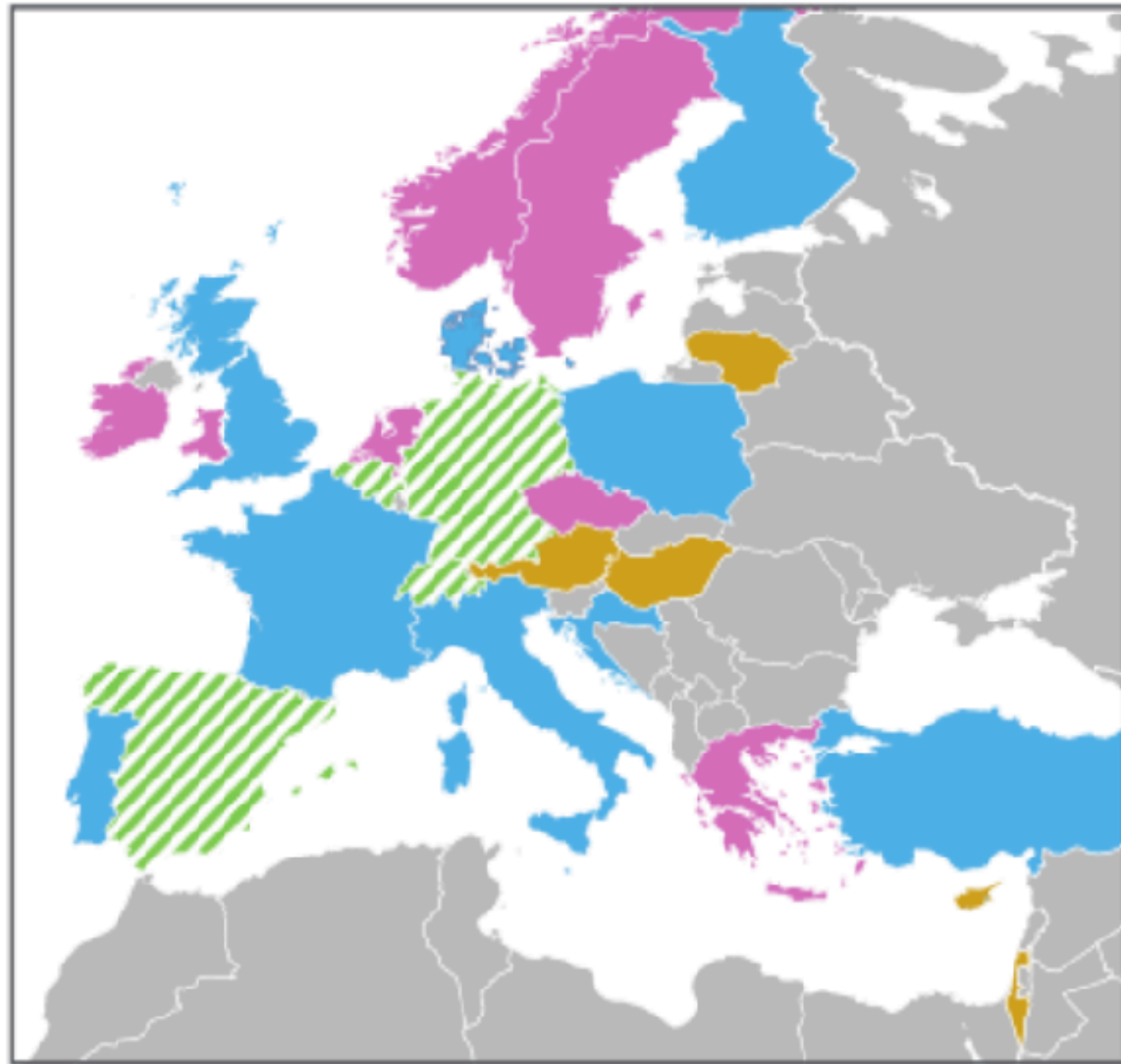
http://blog.educalab.es/intef/wp-content/uploads/sites/4/2017/02/2017_0206_CompuThink_JRC_UE-INTEF.pdf




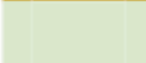
¿Por qué?

“En resumen, surgen dos tendencias principales respecto a la justificación para incluir el Pensamiento Computacional en la enseñanza obligatoria:

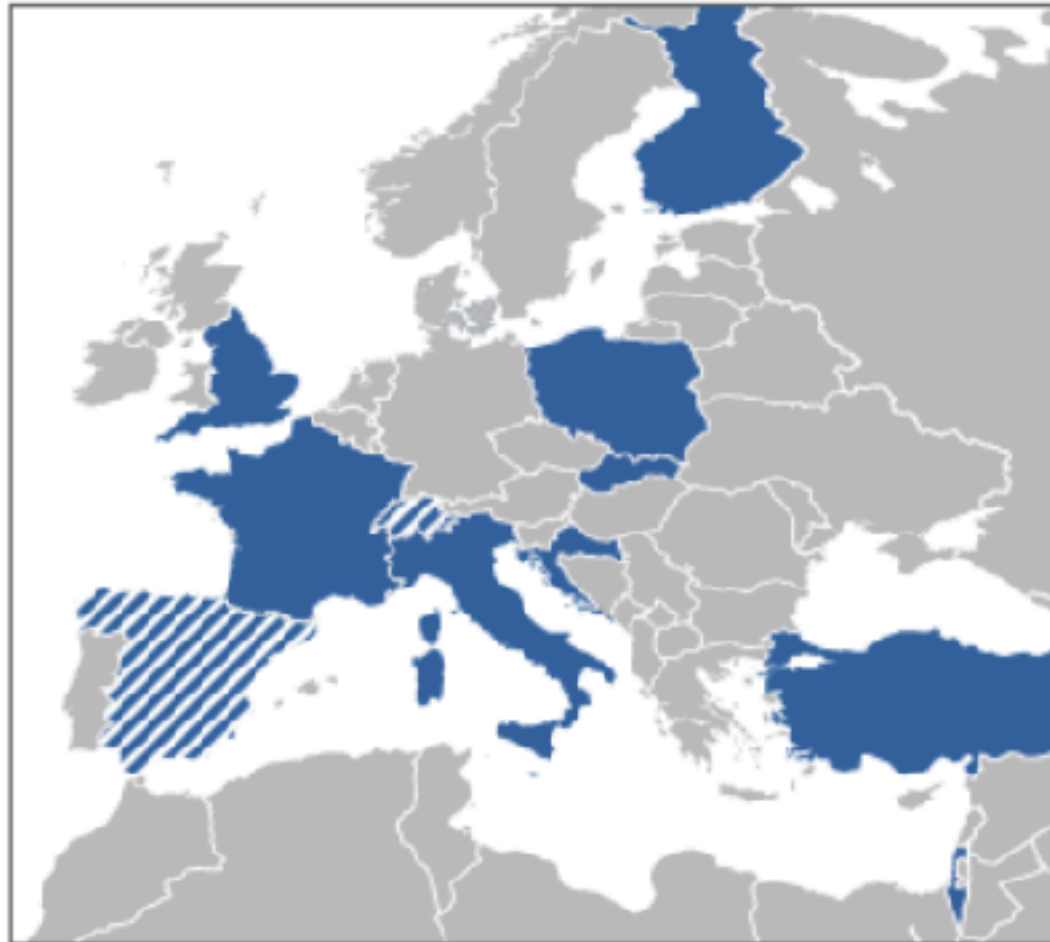
1. El desarrollo de **habilidades** de Pensamiento Computacional en niños y jóvenes para que puedan pensar de manera diferente, expresarse a través de una variedad de medios, resolver problemas del mundo real y analizar temas cotidianos desde una perspectiva diferente.
2. El fomento del Pensamiento Computacional para impulsar el **crecimiento económico**, cubrir puestos de trabajo TIC y prepararse para futuros empleos.”

(Bocconi et al., 2016, pág. 25).



-  Renovación del currículo para integrar el Pensamiento Computacional
-  Con planes para introducir el Pensamiento Computacional
-  Larga tradición en Computación
-  Políticas a nivel regional

Iniciativas de Pensamiento Computacional en Primaria



Iniciativas políticas sobre Pensamiento Computacional ya implementadas en centros de enseñanza primaria



Iniciativas políticas sobre Pensamiento Computacional a nivel regional

Iniciativas de Pensamiento Computacional en Secundaria



Iniciativas políticas sobre Pensamiento Computacional ya implementadas en centros de enseñanza secundaria



Iniciativas políticas sobre Pensamiento Computacional a nivel regional

España

- No hay documento curricular oficial estatal que mencione el "Pensamiento Computacional".
- Conceptos relacionados en determinadas asignaturas: Tecnología (ESO), "Tecnología Industrial" y "Tecnologías de la Información y la Comunicación" (Bachillerato)
- Conceptos relacionados en asignaturas ofrecidas en comunidades autónomas: Andalucía, Canarias, Cantabria, Castilla-La Mancha, Castilla y León, Murcia, Madrid, La Rioja y Comunidad Valenciana.
- En Cataluña: hay "programación" en la descripción de la "Competencia Digital" en el currículum de primaria. En secundaria: aspectos de programación y robótica están presentes en la enseñanza de Tecnología.



Department
for Education



Computing programmes of study: key stages 1 and 2

National curriculum in England

Purpose of study

A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world. Computing has deep links with mathematics, science, and design and technology, and provides insights into both natural and artificial systems. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming. Building on this knowledge and understanding, pupils are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world.

Aims

The national curriculum for computing aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology.

Attainment targets

By the end of each key stage, pupils are expected to know, apply and understand the matters, skills and processes specified in the relevant programme of study.

Schools are not required by law to teach the example content in [square brackets].

Published: September 2013



Department
for Education

Computing programmes of study: key stages 3 and 4

National curriculum in England

Purpose of study

A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world. Computing has deep links with mathematics, science, and design and technology, and provides insights into both natural and artificial systems. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming. Building on this knowledge and understanding, pupils are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world.

Aims

The national curriculum for computing aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology.

Attainment targets

By the end of each key stage, pupils are expected to know, apply and understand the matters, skills and processes specified in the relevant programme of study.

Schools are not required by law to teach the example content in [square brackets].

Published: September 2013

Digital Technologies: Sequence of content F-10 *Strand: Processes and production skills*

	F-2	3-4	5-6	7-8	9-10 (Elective subject)
Generating and designing			Design a user interface for a digital system (ACTDIP018) Design, modify and follow simple algorithms involving sequences of steps, branching, and iteration (repetition) (ACTDIP019)	Design the user experience of a digital system, generating, evaluating and communicating alternative designs (ACTDIP028) Design algorithms represented diagrammatically and in English, and trace algorithms to predict output for a given input and to identify errors (ACTDIP029)	Design the user experience of a digital system by evaluating alternative designs against criteria including functionality, accessibility, usability, and aesthetics (ACTDIP039) Design algorithms represented diagrammatically and in structured English and validate algorithms and programs through tracing and test cases (ACTDIP040)
Producing and implementing		Implement simple digital solutions as visual programs with algorithms involving branching (decisions) and user input (ACTDIP011)	Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020)	Implement and modify programs with user interfaces involving branching, iteration and functions in a general-purpose programming language (ACTDIP030)	Implement modular programs, applying selected algorithms and data structures including using an object-oriented programming language (ACTDIP041)
Evaluating	Explore how people safely use common information systems to meet information, communication and recreation needs (ACTDIP005)	Explain how student solutions and existing information systems meet common personal, school or community needs (ACTDIP012)	Explain how student solutions and existing information systems are sustainable and meet current and future local community needs (ACTDIP021)	Evaluate how student solutions and existing information systems meet needs, are innovative, and take account of future risks and sustainability (ACTDIP031)	Evaluate critically how student solutions and existing information systems and policies, take account of future risks and sustainability and provide opportunities for innovation and enterprise (ACTDIP042)
Collaborating and managing	Create and organise ideas and information using information systems independently and with others, and share these with known people in safe online environments (ACTDIP006)	Plan, create and communicate ideas and information independently and with others, applying agreed ethical and social protocols (ACTDIP013)	Plan, create and communicate ideas and information, including collaboratively online, applying agreed ethical, social and technical protocols (ACTDIP022)	Plan and manage projects that create and communicate ideas and information collaboratively online, taking safety and social contexts into account (ACTDIP032)	Create interactive solutions for sharing ideas and information online, taking into account safety, social contexts and legal responsibilities (ACTDIP043) Plan and manage projects using an iterative and collaborative approach, identifying risks and considering safety and sustainability (ACTDIP044)

Digital Technologies: Sequence of content F-10 *Strand: Knowledge and understanding*

	F-2	3-4	5-6	7-8	9-10 (Elective subject)
Digital systems	Recognise and explore digital systems (hardware and software components) for a purpose (ACTDIK001)	Identify and explore a range of digital systems with peripheral devices for different purposes, and transmit different types of data (ACTDIK007)	Examine the main components of common digital systems and how they may connect together to form networks to transmit data (ACTDIK014)	Investigate how data is transmitted and secured in wired, wireless and mobile networks, and how the specifications affect performance (ACTDIK023)	Investigate the role of hardware and software in managing, controlling and securing the movement of and access to data in networked digital systems (ACTDIK034)
Representation of data	Recognise and explore patterns in data and represent data as pictures, symbols and diagrams (ACTDIK002)	Recognise different types of data and explore how the same data can be represented in different ways (ACTDIK008)	Examine how whole numbers are used to represent all data in digital systems (ACTDIK015)	Investigate how digital systems represent text, image and audio data in binary (ACTDIK024)	Analyse simple compression of data and how content data are separated from presentation (ACTDIK035)

Digital Technologies: Sequence of content F-10 *Strand: Processes and production skills*


	F-2	3-4	5-6	7-8	9-10 (Elective subject)
Collecting, managing and analysing data	Collect, explore and sort data, and use digital systems to present the data creatively (ACTDIP003)	Collect, access and present different types of data using simple software to create information and solve problems (ACTDIP009)	Acquire, store and validate different types of data, and use a range of software to interpret and visualise data to create information (ACTDIP016)	Acquire data from a range of sources and evaluate authenticity, accuracy and timeliness (ACTDIP025) Analyse and visualise data using a range of software to create information, and use structured data to model objects or events (ACTDIP026)	Develop techniques for acquiring, storing and validating quantitative and qualitative data from a range of sources, considering privacy and security requirements (ACTDIP036) Analyse and visualise data to create information and address complex problems, and model processes, entities and their relationships using structured data (ACTDIP037)
Creating digital solutions by:					
Investigating and defining	Follow, describe and represent a sequence of steps and decisions (algorithms) needed to solve simple problems (ACTDIP004)	Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them (ACTDIP010)	Define problems in terms of data and functional requirements drawing on previously solved problems (ACTDIP017)	Define and decompose real-world problems taking into account functional requirements and economic, environmental, social, technical and usability constraints (ACTDIP027)	Define and decompose real-world problems precisely, taking into account functional and non-functional requirements and including interviewing stakeholders to identify needs (ACTDIP038)



Guidance and information

News

Dysg Newsletters

Visit **Hwb**  for classroom resources

GUIDANCE

Alternative versions ▼

How-to 

Digital Competence Framework

Last updated: 1 Sep 2016

Use the filters below to view the parts of the Framework you want to see.

SELECT A STAGE/YEAR

RfL routemap

A steps

B steps

C steps

Nursery

Reception

etc.

Críticas

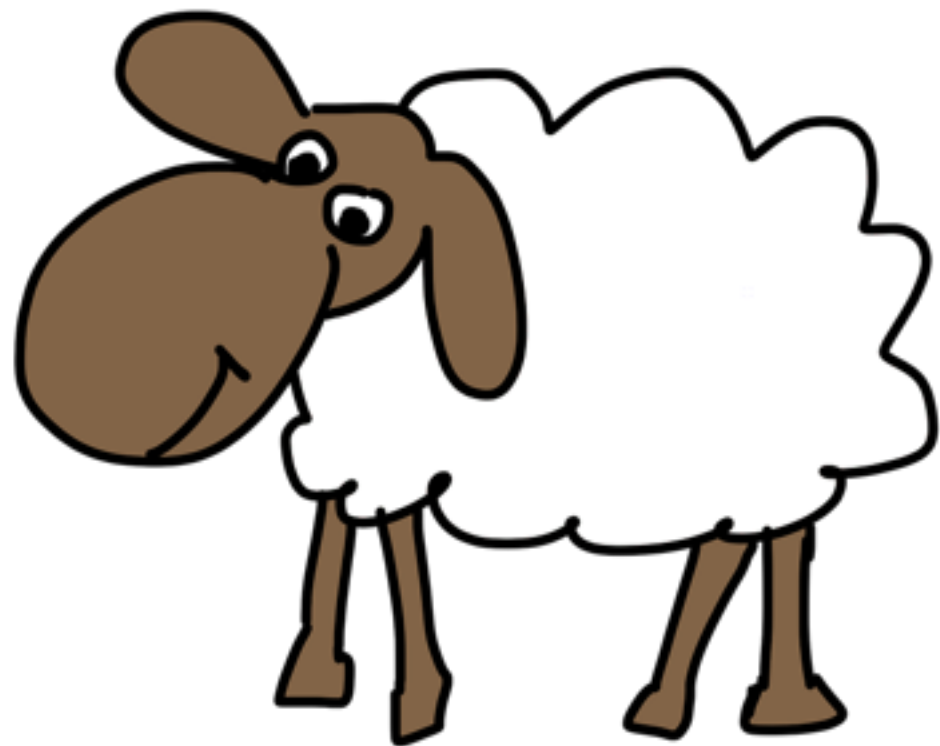
... los profesionales de la computación pueden intentar resolver todos los problemas a través de medios algorítmicos, mientras no perciben aquellos que no pueden ser expresados usando las abstracciones del pensamiento computacional.

Easterbrook, S. (2014). From computational thinking to systems thinking: A conceptual toolkit for sustainability computing. In *Proceedings of the 2nd international conference on information and communication technologies for sustainability (ICT4S'2014), stockholm, sweden, 24-27 august, 2014.*

El martillo de Maslow

*Si tu única herramienta es un martillo,
tiendes a tratar cada problema como
si fuera un clavo.*

Abraham Maslow, 1966. "The Psychology of Science".



**Problemas *mansos* vs.
problemas retorcidos**



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Featured content](#)
[Current events](#)
[Random article](#)
[Donate to Wikipedia](#)

Not logged in [Talk](#) [Contributions](#) [Create account](#) [Log in](#)

Article [Talk](#)

[Read](#) [Edit](#) [View history](#)

Wicked problem

From Wikipedia, the free encyclopedia

A **wicked problem** is a problem that is difficult or impossible to solve because of incomplete, contradictory, and changing requirements that are often difficult to recognize. The use of term "wicked" here has come to denote resistance to resolution, rather than evil.^[1] Moreover, because of complex [interdependencies](#), the effort to solve one aspect of a wicked problem may reveal or create other problems.



WIKIPEDIA
La enciclopedia libre

[Portada](#)
[Portal de la comunidad](#)
[Actualidad](#)
[Cambios recientes](#)
[Páginas nuevas](#)
[Página aleatoria](#)

No has iniciado sesión [Discusión](#) [Contribuciones](#) [Crear una cuenta](#) [Acceder](#)

Artículo [Discusión](#)

[Leer](#) [Editar](#) [Ver historial](#)

Problema retorcido

Un "**problema retorcido**" (en inglés, "**wicked problem**") es un concepto utilizado en [planificación social](#) para describir un problema que es difícil o imposible de resolver dado que presenta requisitos incompletos, contradictorios y cambiantes que generalmente son difíciles de reconocer. El término "retorcido" no se utiliza en un sentido de malvado, sino antes bien como resistencia a la solución.¹ Además, dada la existencia de complejas [interdependencias](#) en este tipo de problemas, los esfuerzos para resolver un aspecto de un problema retorcido podría revelar o crear nuevos problemas.

La crítica “política” al “learning to code”

Political computational thinking: policy networks, digital governance and ‘learning to code’

Ben Williamson*

School of Education, University of Stirling, Stirling, UK

Reflecting political shifts toward both ‘network governance’ and ‘digital governance’, the idea of ‘learning to code’ has become part of a major reform agenda in education policy in England. This article provides a ‘policy network analysis’ tracing the governmental, business and civil society actors now operating in policy networks to project learning to code into the reformed programs of study for computing in the National Curriculum in England. The insertion of learning to code into the curriculum provides evidence of how the education policy process is being displaced to cross-sector ‘boundary organizations’ such as ‘policy labs’ that act as connecting nodes to broker networks across public and private sector borderlines. It also examines how the pedagogies of learning to code are intended to inculcate young people into the material practices and systems of thought associated with computer coding, and to contribute to new forms of ‘digital governance’. These developments are evidence of a ‘reluctant state’ deconcentrating its responsibilities, and also of a computational style of political thinking that assumes policy problems can be addressed using the right code. Learning to code is seen as a way of shaping governable citizens that can participate in the dynamics of digital governance.

Keywords: computing curriculum; computational thinking; learning to code; governance; policy networks; policy labs

Actividades didácticas para el desarrollo del PC

<http://csunplugged.org>

CS UNPLUGGED

Computer Science without a computer

[Home](#)

[The Book](#)

[Activities](#)

[Videos](#)

[Community](#)

[Promotional](#)

[About](#)



Update: New version of CS Unplugged in development

We are currently updating the CS Unplugged content and website, and expect to release this content in late 2017. [Click here for more information.](#)



Search





ROBOT Turtles
The Game for Little Programmers!

Subscribe for Updates!



Game Info

Coding Literacy

Adventure Quests

Resources

Buy Now

ROBOT Turtles™

It's the first board game for little programmers!



It's really simple.
Just get your Turtle
to the matching
colored jewel.

Ages
4 & up





C O
D E

Puzzle

16

20

I've finished



▶ Run Program

Blocks

move forward

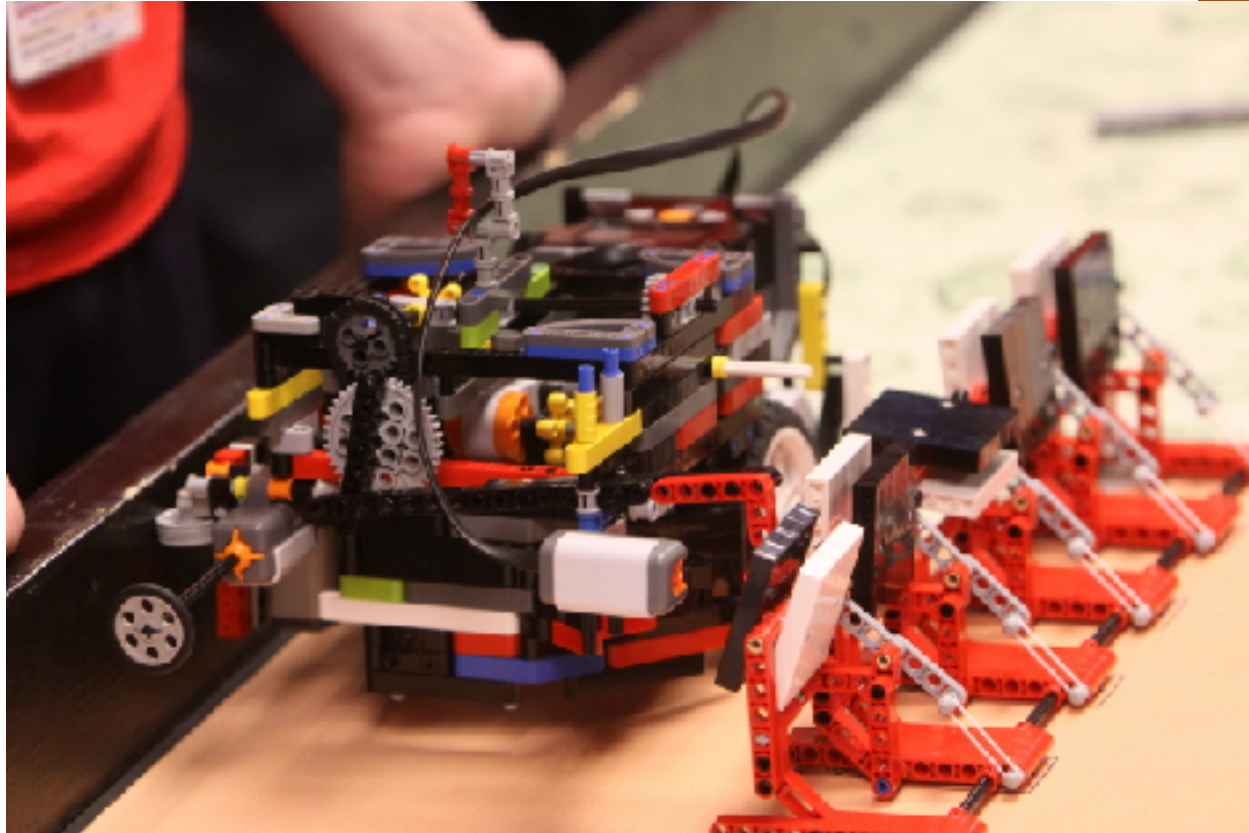
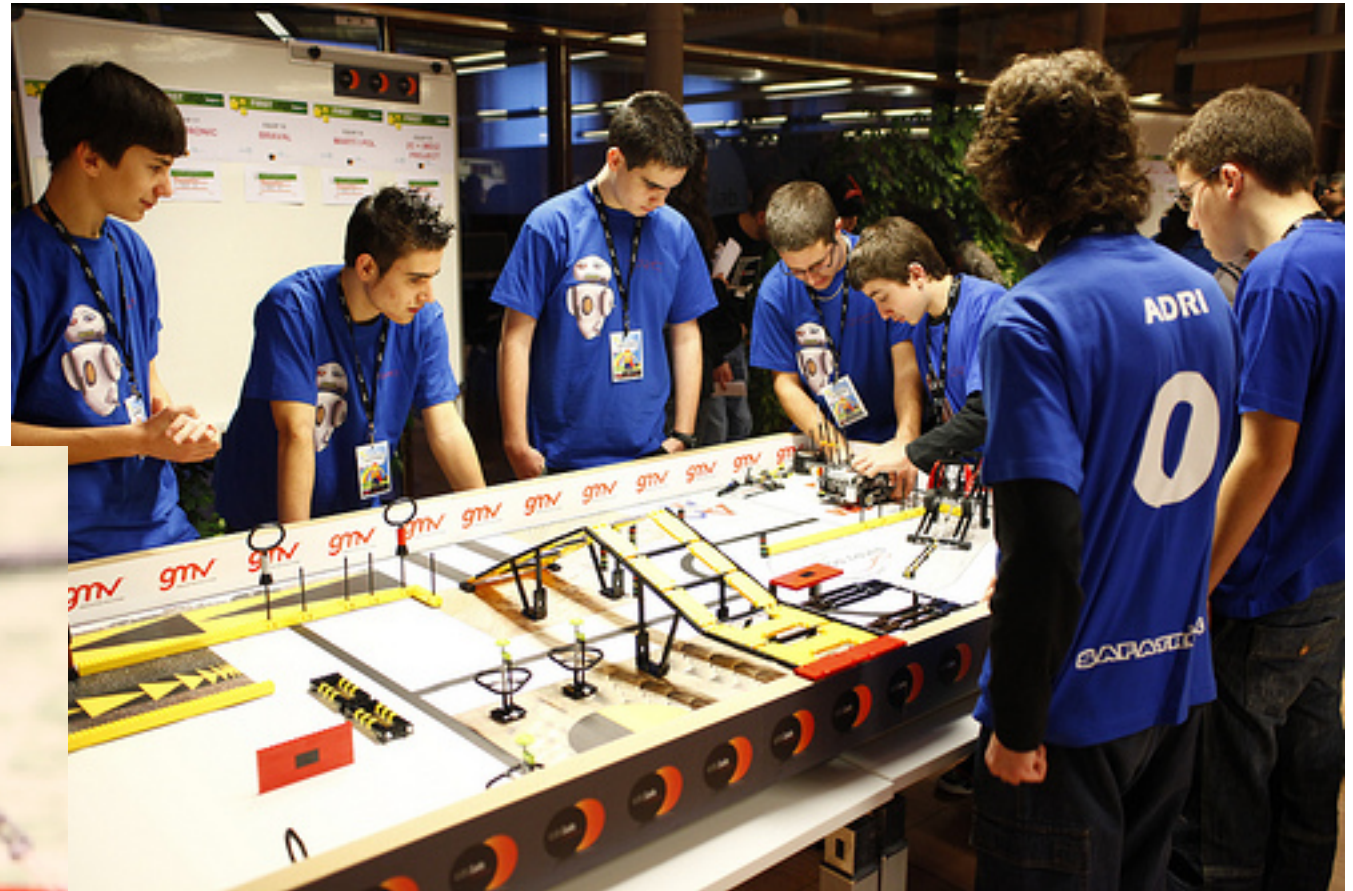
turn left ↺

turn right ↻

repeat until 
do

if path to the left ↺
do

repeat until 
do
if path to the left ↺
do turn left ↺
move forward



PC y formación del profesorado de Primaria

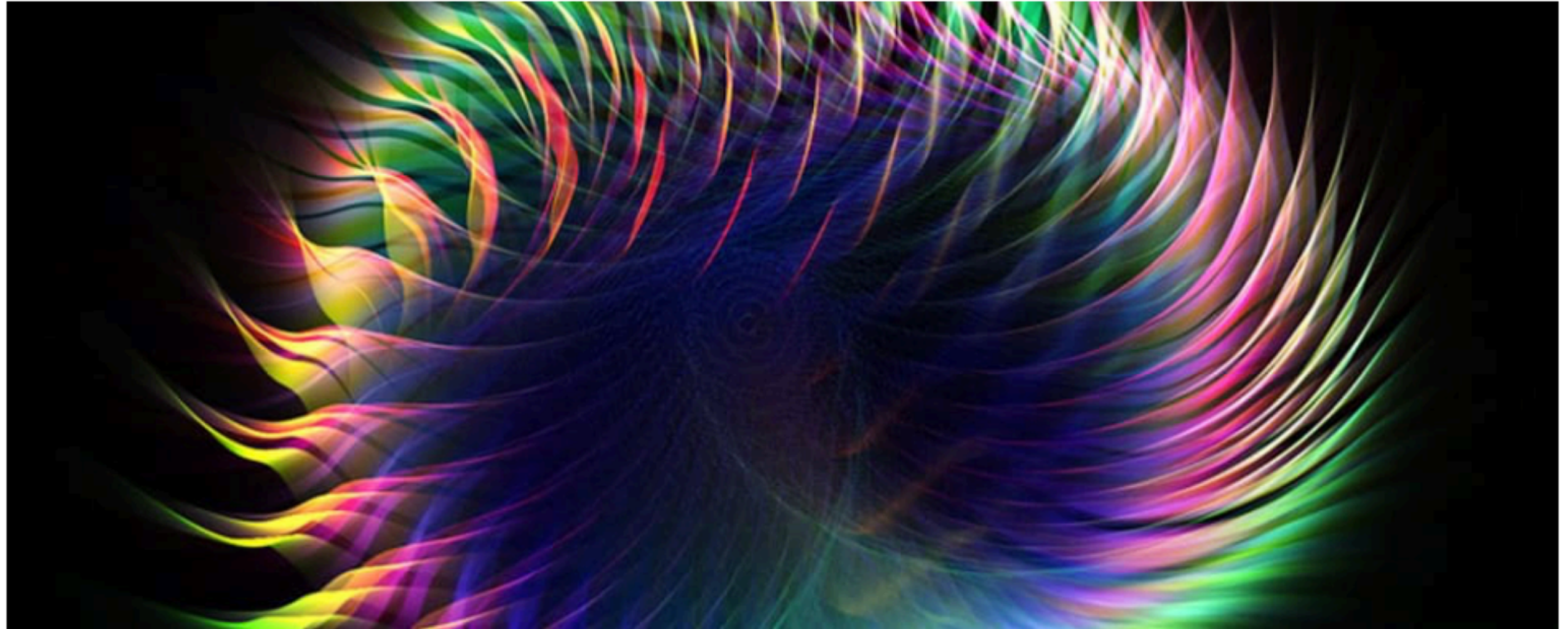
Ideas clave:





News > [Technology in School](#) > [Coding](#)

Learn to Code, Code to Learn



CC BY 2.0 flickr



By [Mitchel Resnick](#) May 8, 2013

Is it important for all children to learn how to write? After all, very few children grow up to become journalists, novelists, or professional writers. So why should everyone learn to write?

This article is in our guide:

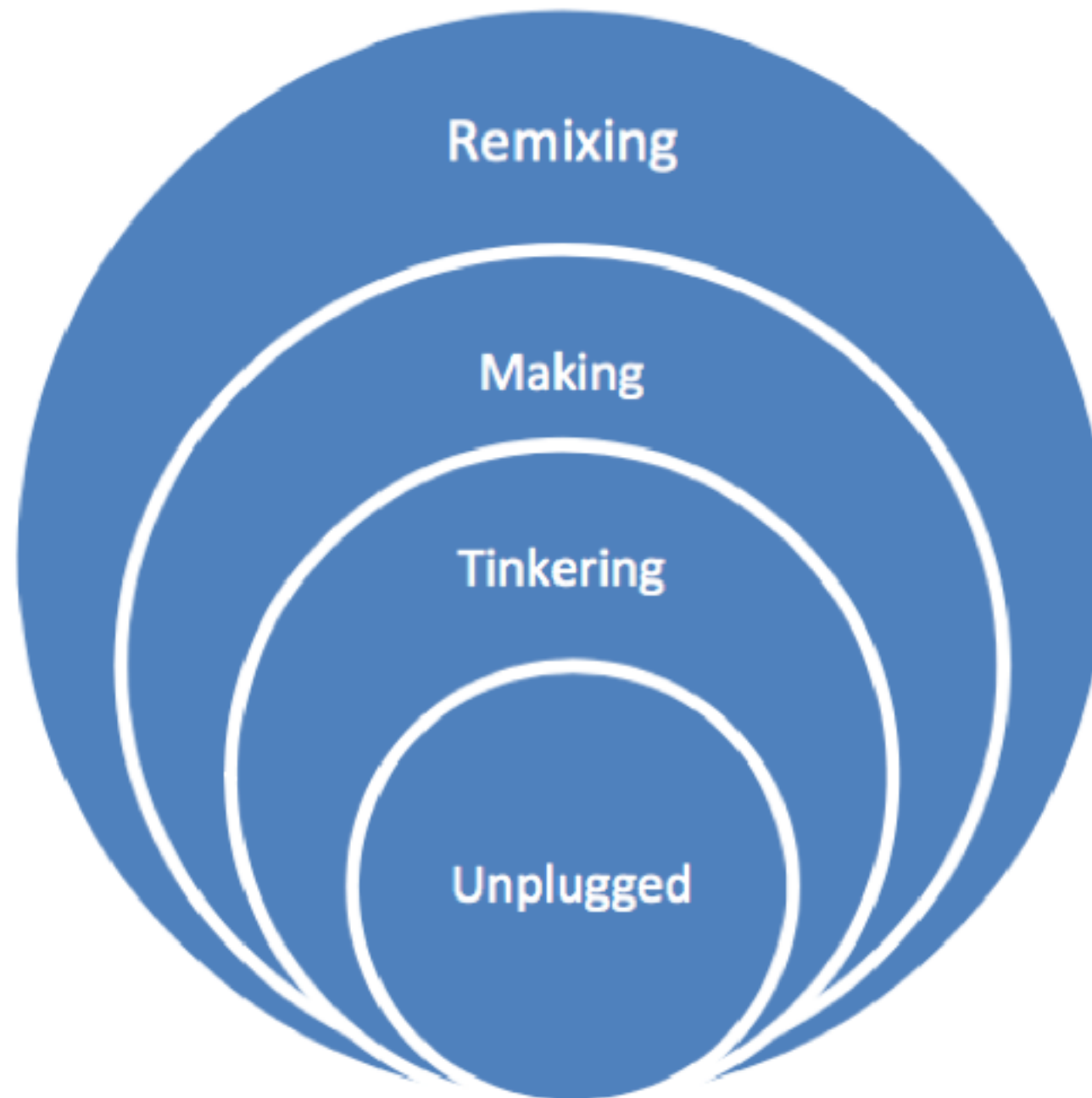
[Give Your Kids a Most Excellent Summer Coding Adventure](#)

Sponsored by [Dig-It Games](#)



Figure 1: The kindergarten approach to learning

Resnick, M. (2007). All I Really Need to Know (About Creative Thinking) I Learned (By Studying How Children Learn) in Kindergarten. Presented at Creativity & Cognition conference, June 2007.



(1) desenchufado, (2) jugar, (3) hacer, y (4) remezclar,

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A Pedagogical Framework for Computational Thinking. *Digital Experiences in Mathematics Education*, 1-18. DOI:10.1007/s40751-017-0031-2.

TPACK

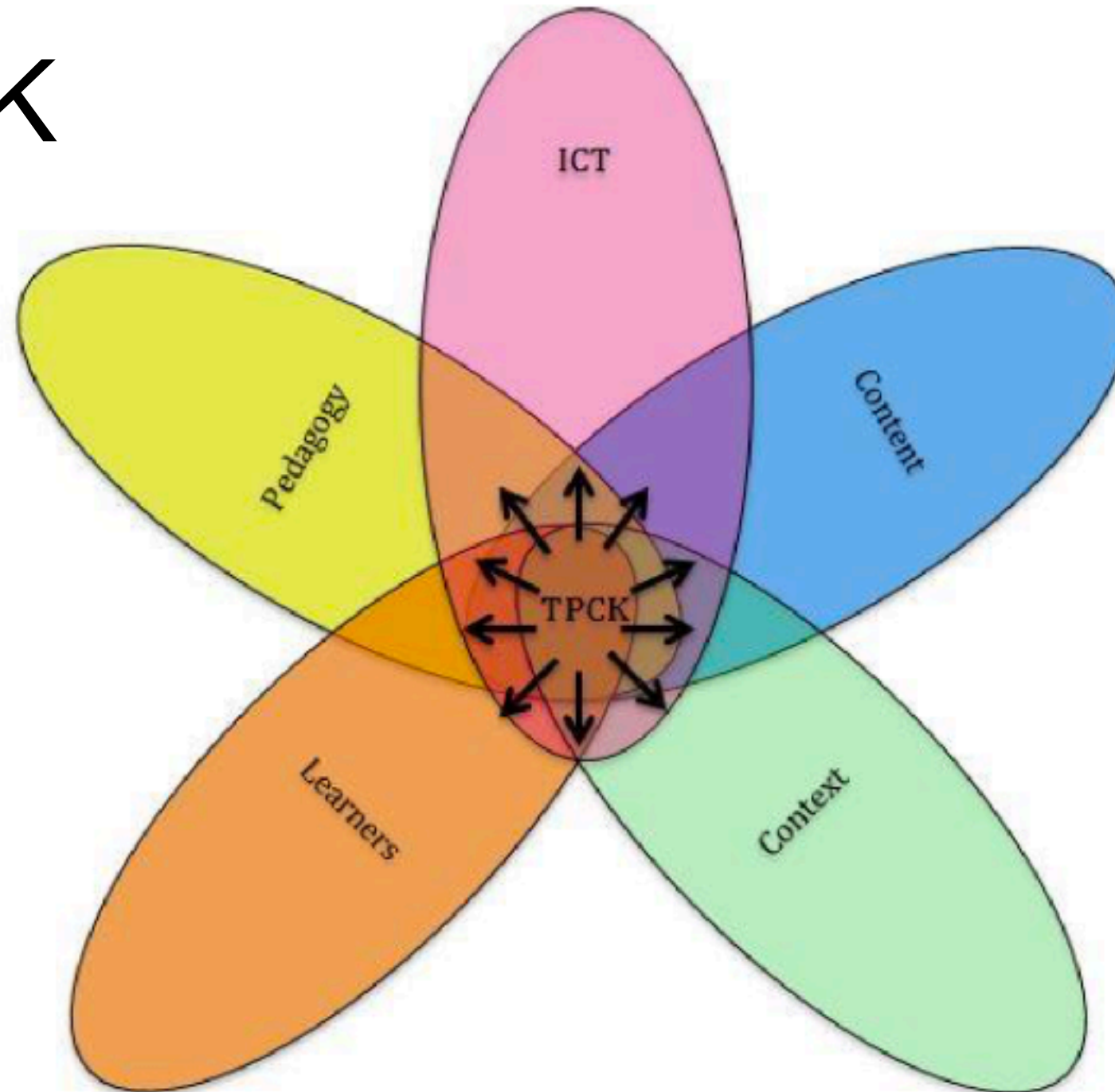



Figure 1. Technological Pedagogical Content Knowledge (adopted from [Angeli & Valanides, 2005](#))

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 Computational Thinking Curriculum Framework- Implications for Teacher Knowledge. *Educational Technology & Society*, 19(3), 47–57.

CÓDIGO 21
TECNOLOGÍAS CREATIVAS



Programar para aprender: orientaciones para el profesorado de Primaria

 Gobierno de Navarra
Departamento de Educación

<http://codigo21.educacion.navarra.es/2015/01/12/disponible-para-descargar-la-guia-programar-para-aprender-en-primaria/>

Conclusiones

1. El PC ya está aquí,
y está para quedarse.

2. Los desacuerdos académicos en su definición y elementos no parecen ser un obstáculo para su integración en el currículum de la educación obligatoria y secundaria no obligatoria en muchos países.

3. Predicción: en la próxima reforma del currículum a nivel estatal aparecerá el PC.

4. Es posible “orientar” la enseñanza y el aprendizaje del PC de formas diversas y no excluyentes: PC, programación-robótica, alfabetización computacional (*code literacy*), parte de la competencia digital, etc.

5. Deberíamos prepararnos:

- formación del profesorado en ejercicio;
- formación inicial del profesorado;
- desarrollo del currículum;
- dotación de medios y recursos a los centros, etc.

5. Hace falta más investigación:

- Formación inicial del profesorado.
- Diseño y desarrollo de actividades didácticas PC.
- Evaluación del PC.

<http://www.flickr.com/photos/alwaysbecool/2871346522>

**Moltes
gràcies!**



¿Preguntas?
¿Comentarios?