Investigación sobre el desarrollo del pensamiento computacional en la educación

Jesús Moreno León - KGBL3

Conferencias de investigación para posgrado
Programa de doctorado en ingeniería informática

17/06/2021

UNIVERSIDAD COMPLUTENSE MADRID

Motivation

Coding in schools

- One of the main trends in the educational landscape worldwide
- Code to learn vs learn to code
- Huge disparity of criteria in terms of target ages, approaches, methodologies...
- Lack of standarization due to the lack of research in this field
- **Urgent attention from academia is required: assessment, transference and affecting factors**

Five broad categories of technologies

- Unplugged
- Arrow-based visual environments
- Block-based visual environments
- Textual programming languages
- Connected with the physical world

Which technologies are using educators at schools?

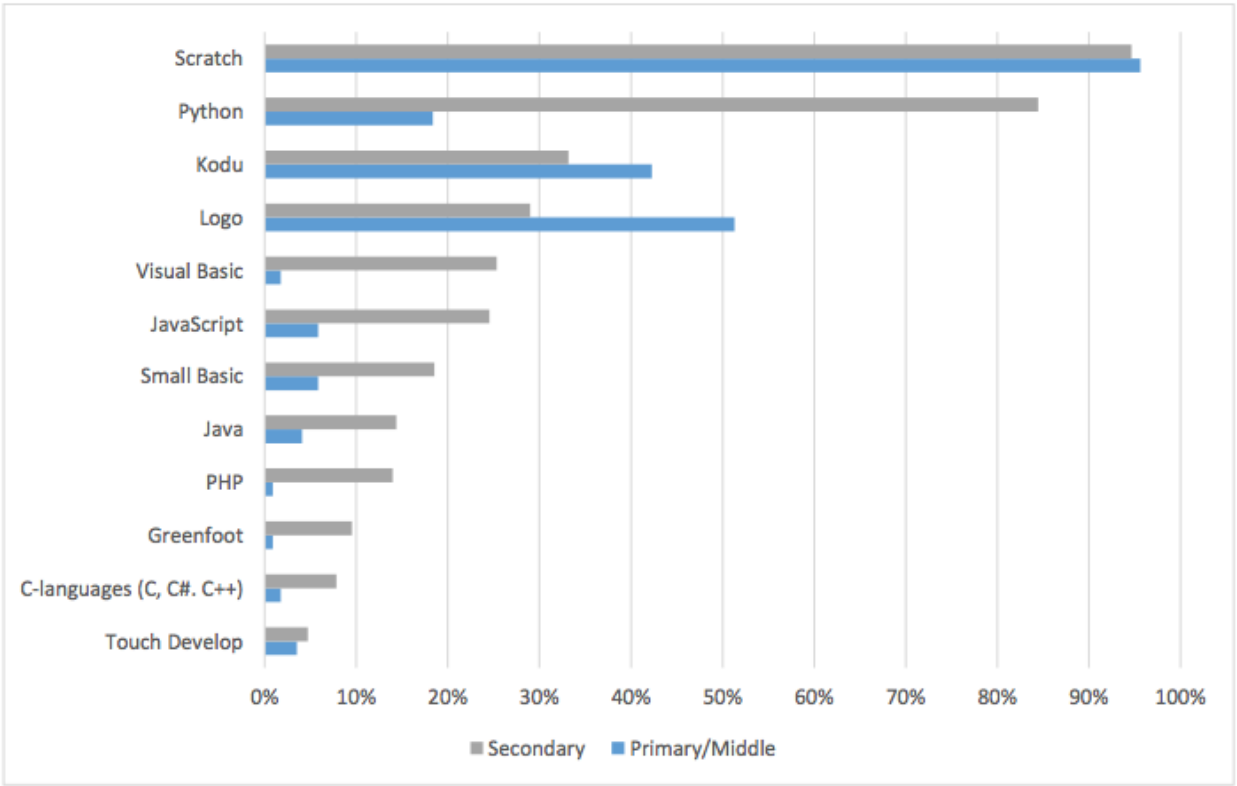
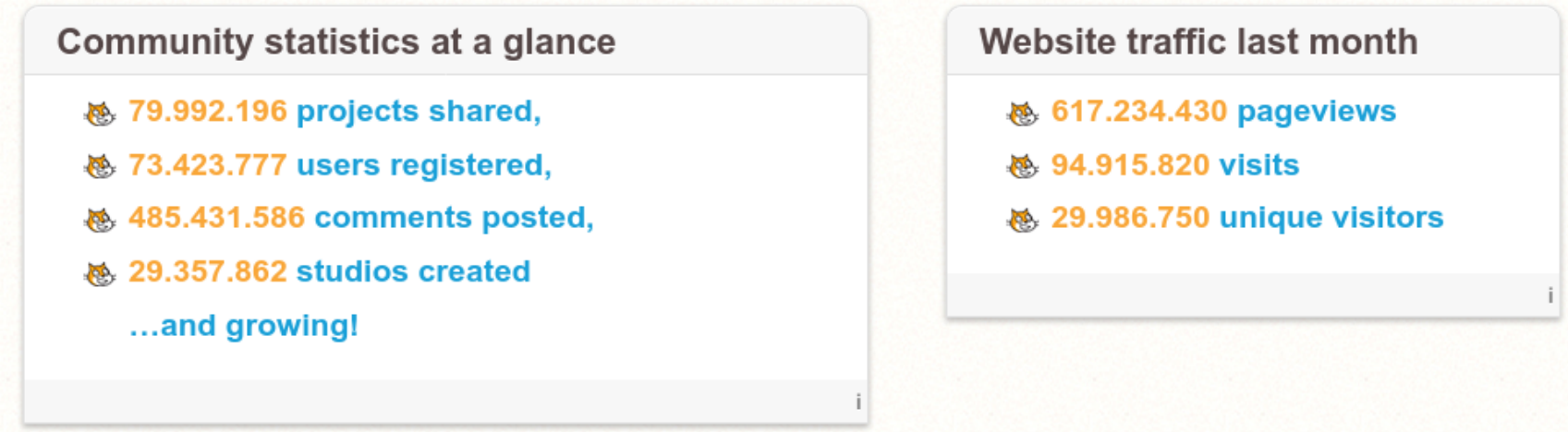
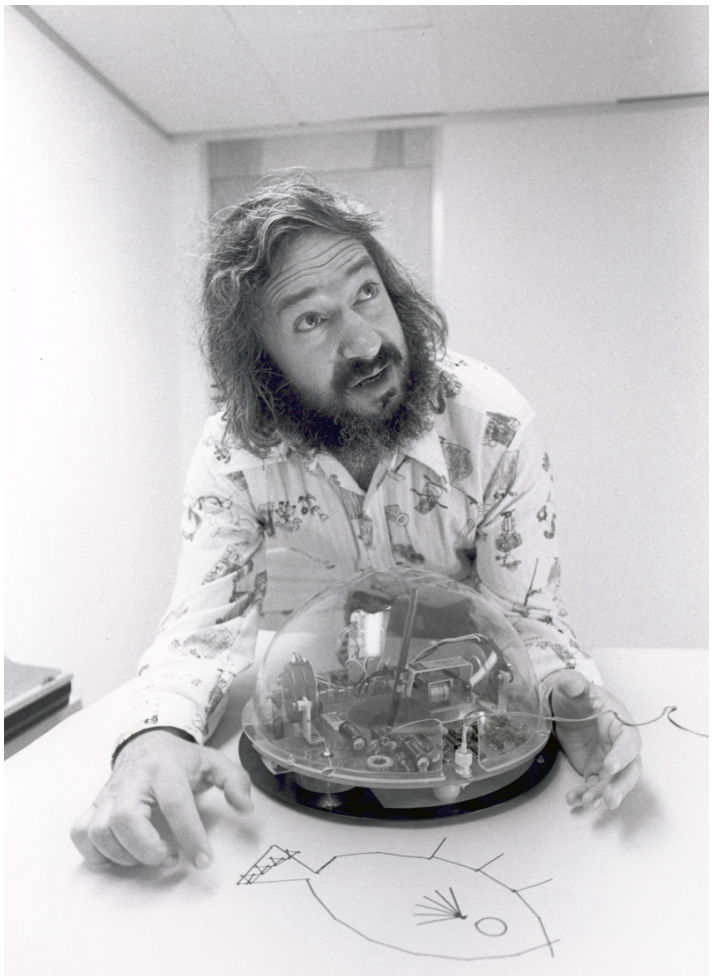


Figure: Programming languages taught at primary and middle schools (blue) and secondary/high schools (gray) [Sentance, 2015].

Scratch statistics



The pioneers



Logo programming language

- Developed in the 1960s
- Millions of students learnt to program at school during the 1970s and 80s
- "Disappeared" from the educational landscape since mid-90s

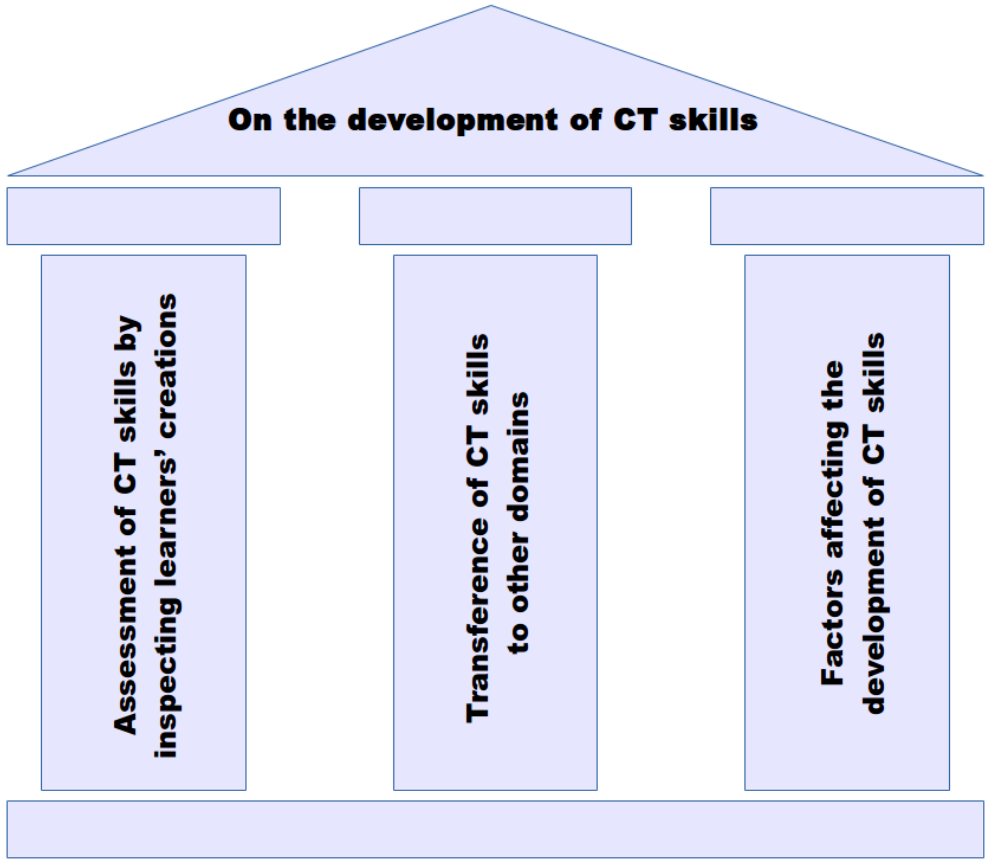
Seymour Papert's picture: jgora.net

Worrying signs from England

- "Only a small fraction of pupils choose, or have the option to choose, to take qualifications at GCSE (5.5%) or A level (1.7%)"
- "teachers just do not have the knowledge to teach this subject"
- "computer science could become a niche subject, taught in only a few schools"

Our first goals

The three pillars of our research

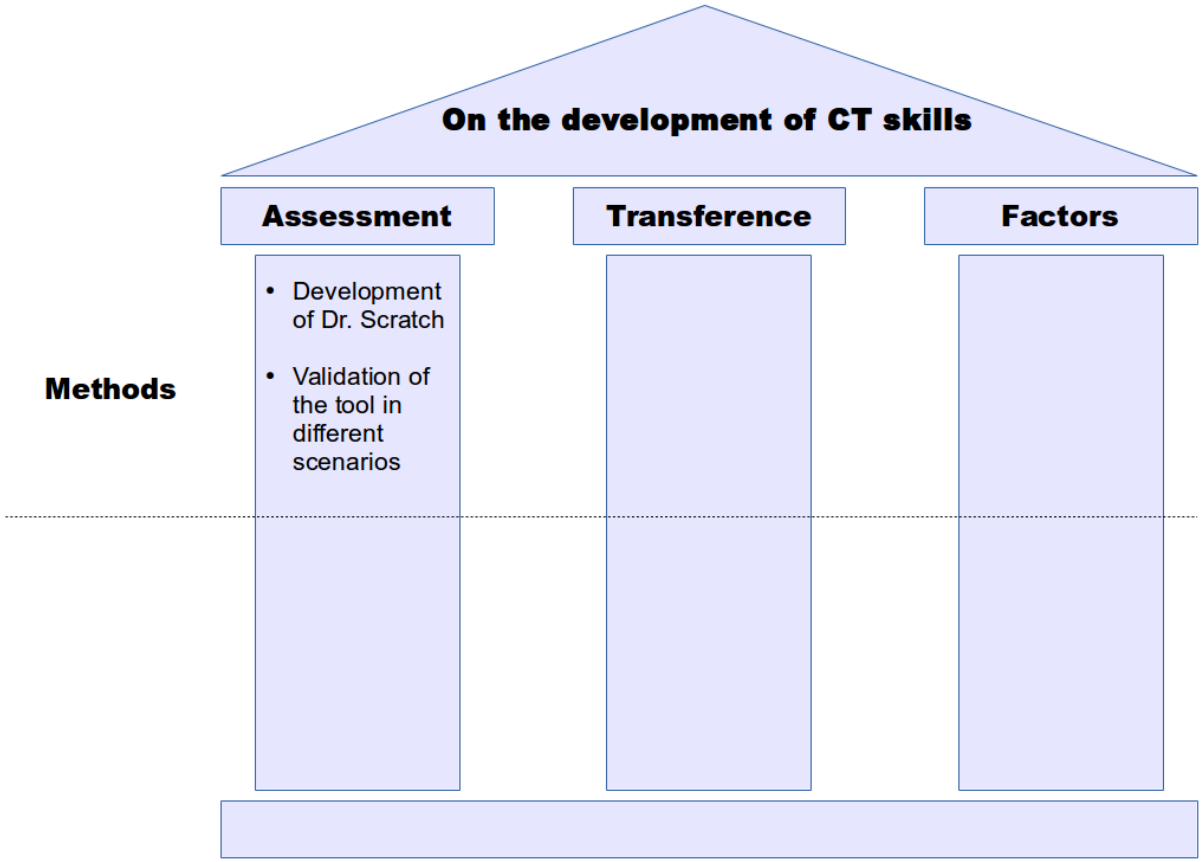


Research questions

1. **Is it possible to automatically assess the development of CT skills by inspecting learners' `Scratch` creations?**
2. **Does the development of CT skills through programming activities with `Scratch` enhance the learning of other subjects?**
3. **Are there social and non-cognitive factors affecting the development of CT skills when programming with `Scratch`?**

Main investigations

Methodology



A formative assessment tool for Scratch projects

- Could we create a `lint`-like tool to support learners and educators?
- `Dr. Scratch`, inspired by `Scrape` and is based on `Hairball`

Free software - standing on the shoulders of giants

Total Physical Source Lines of Code	1,809
Dominant language	Python
Schedule Estimate, Months	4.42
Estimated Average Number of Developers	1.01
Total Estimated Cost to Develop	$50,345

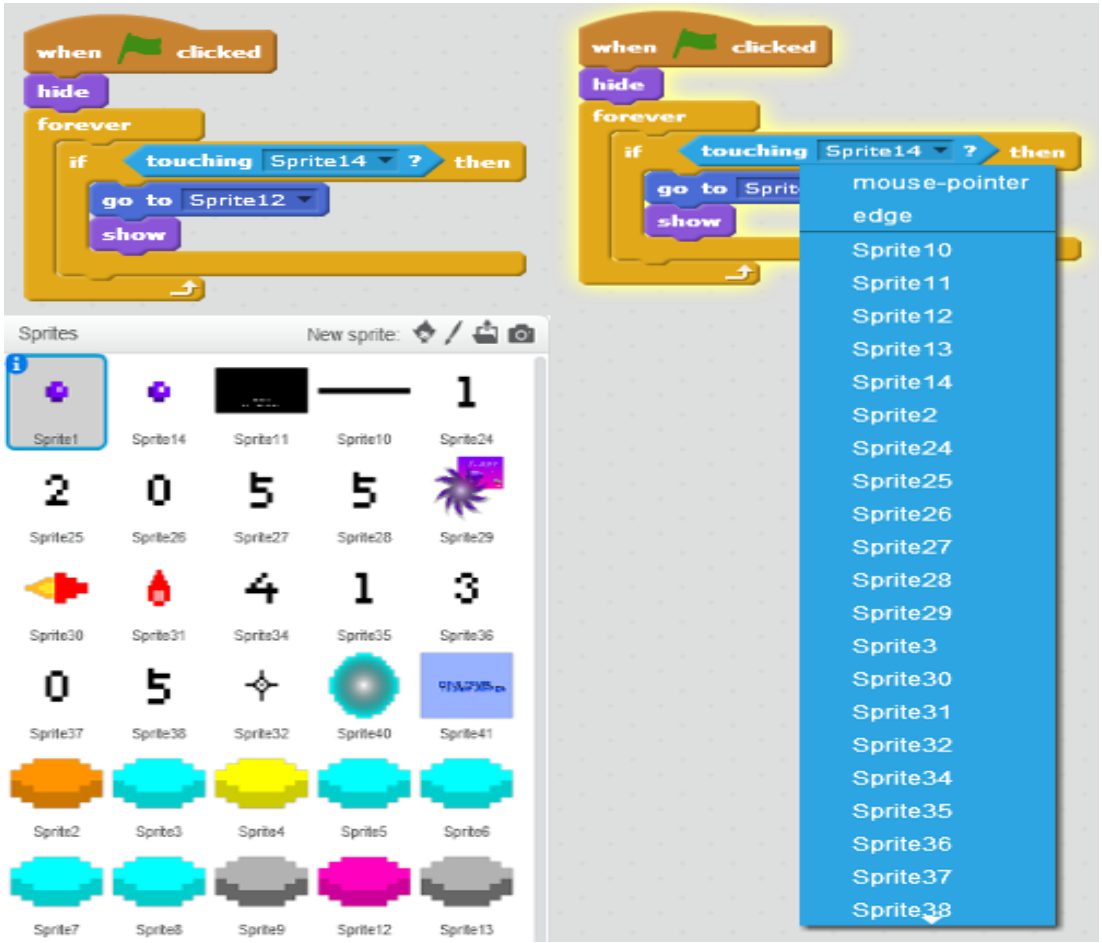
Table: Estimated effort for the development of the `Hairball` fork (using the *basic* COCOMO estimation model [Boehm et al., 2000]).

First step: extending the features of Hairball

Plug-ins to detect bad programming habits

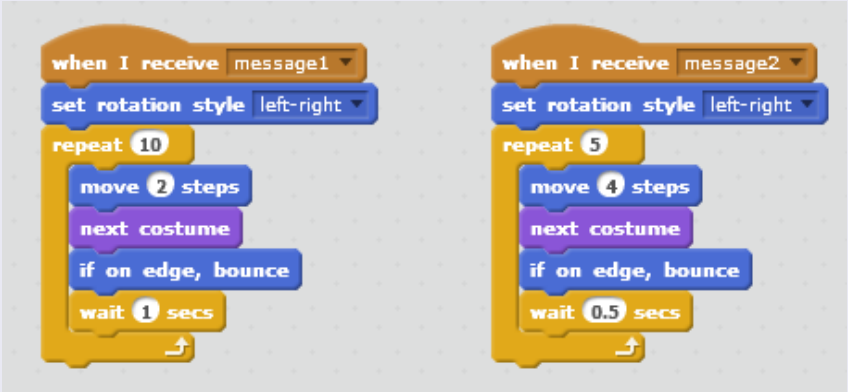
1. Characters that use the default, non-meaningful name that `Scratch` assigns to new objects
2. Repetition of code

Bad/default naming of sprites

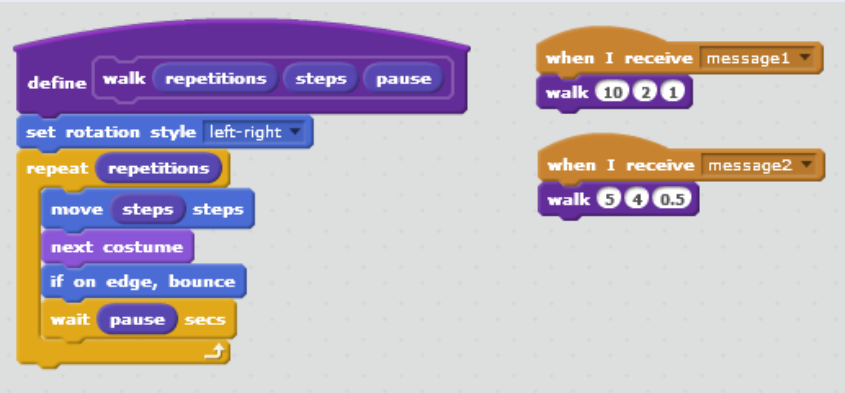


Repetition of code

Example of repeated code



Solution to avoid repeated code



Blocks should be created to avoid repetition of code

Scratch projects repository analysis

	Default names	Duplicated scripts	Defined blocks
Projects	79	62	17
Mean	5.94	7.23	1.11
Median	3	2	0
Maximum	67	71	25

Table: Analysis of 100 ramdonly downloaded Scratch projects

Second step: development of the CT assessment feature

Remixing other researchers' ideas

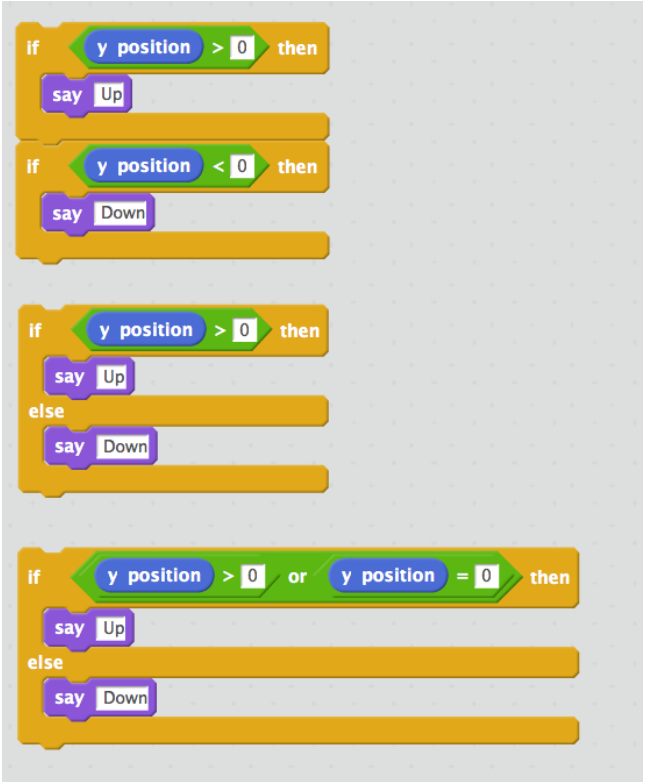
- New frameworks for studying and assessing the development of CT [Brennan and Resnick, 2012].
- Progression of Early CT Model [Seiter and Foreman, 2013].
- Evaluation of games to gauge understanding of programming concepts [Wilson et al., 2012].
- Towards the Automatic Recognition of CT for Adaptive Visual Language Learning [Koh et al., 2010].

Assessment of CT

CT dimension	Basic	Developing	Proficient
Data representation	modifiers of sprites properties	operations on vars	operations on lists
Logical Thinking	if	if else	logic operations
User interactivity	green flag	key pressed, sprite clicked, ask and wait, mouse blocks	when %s is >%s, video, audio
Algotithmic notions of flow control	sequence of blocks	repeat, forever	repeat until
Abstraction and problem decomposition	more than one script and more than one sprite	def block	when I start as clone
Parallelism	Two scripts on green flag	Two scripts on key pressed, two scripts on sprite clicked on the same sprite	Two scripts on when I receive message, two scripts when %s is >%s, two scripts on when backdrop change to
Synchronization	wait	Broadcast, when I receive message, stop all, stop program, stop programs sprite	wait until, when backdrop change to, broadcast and wait

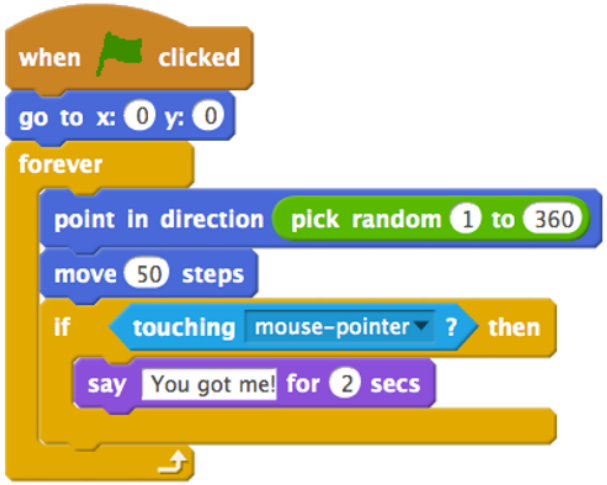
Table: Level of development for each CT dimension

Assessment of CT: Logical Thinking



Different levels of development of logical thinking: basic (top), developing (center) and proficient (bottom).

Assessment of CT: an example



CT dimension	Basic	Developing	Proficient
Data representation	modifiers of sprites properties	operations on vars	operations on lists
Logical Thinking	if	if else	logic operations
User interactivity	green flag	key pressed, sprite clicked, ask and wait, mouse blocks	when %s is >%s, video, audio
Algotithmic notions of flow control	sequence of blocks	repeat, forever	repeat until
Abstraction and problem decomposition	more than one script and more than one sprite	def block	when I start as clone
Parallelism	Two scripts on green flag	Two scripts on key pressed, two scripts on sprite clicked on the same sprite	Two scripts on when I receive message, two scripts when %s is >%s, two scripts on when backdrop change to
Synchronization	wait	Broadcast, when I receive message, stop all, stop program, stop programs sprite	wait until, when backdrop change to, broadcast and wait

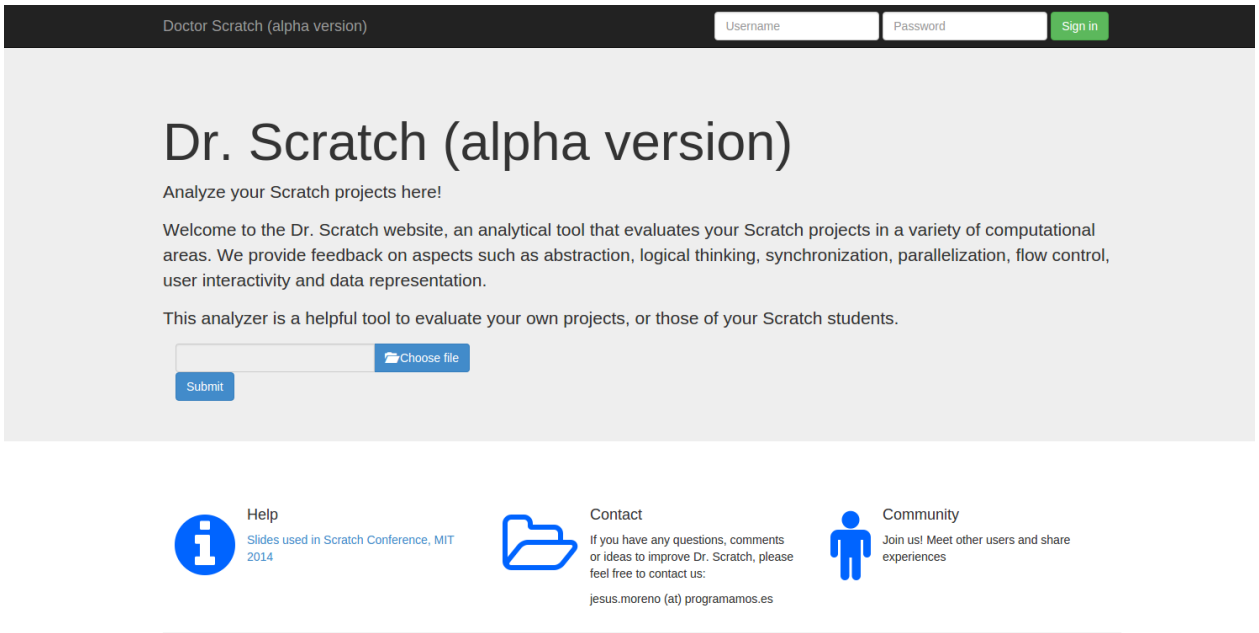
Evaluation of *Catch me if you can* (available at https://scratch.mit.edu/projects/138397021/)

Assessment of CT: an example

```
chen@th:~$ hairball -p mastery.Mastery Catch\ me\ if\ you\ can.sb2
Catch me if you can.sb2
{'Abstraction': 0, 'Parallelization': 0, 'Logic': 1, 'Synchronization': 0, 'FlowControl
': 2, 'UserInteractivity': 2, 'DataRepresentation': 1}
Total mastery points: 6/21
Average mastery points: 0.86/3
Overall programming competence: Basic
```

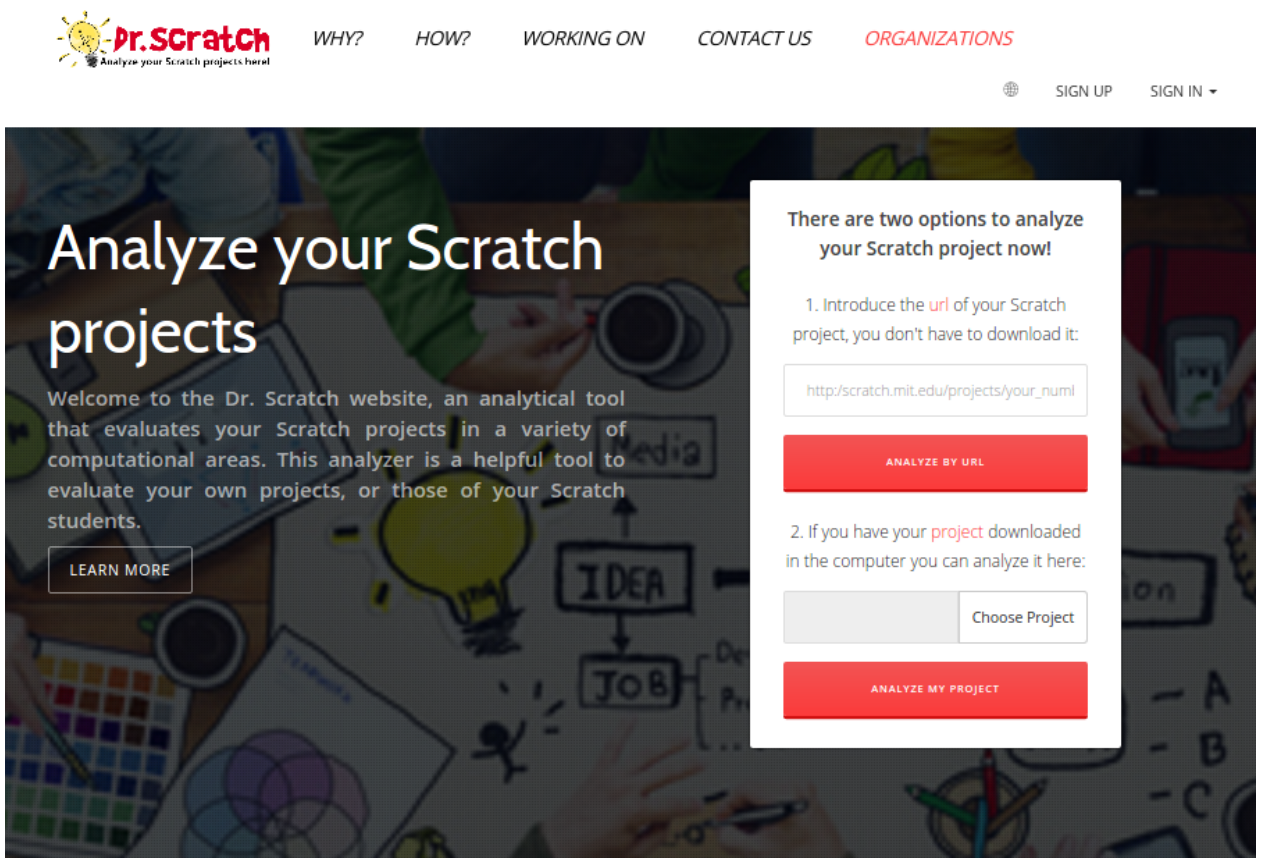
Evaluation of *Catch me if you can* (available at https://scratch.mit.edu/projects/138397021/)

Web-based service: Dr. Scratch



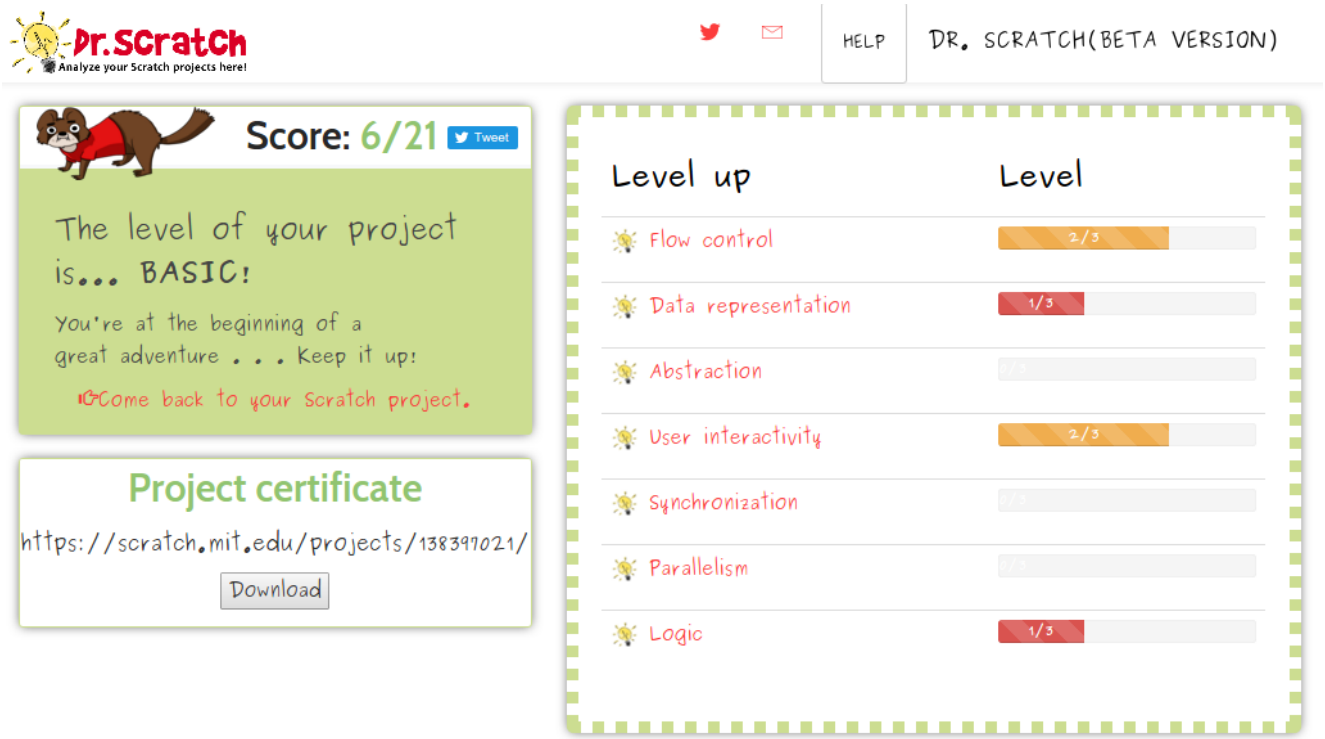
Alpha version of the tool, presented during the Scratch Conference 2014 at the MIT (Cambridge, MA, USA)

Web-based service: Dr. Scratch



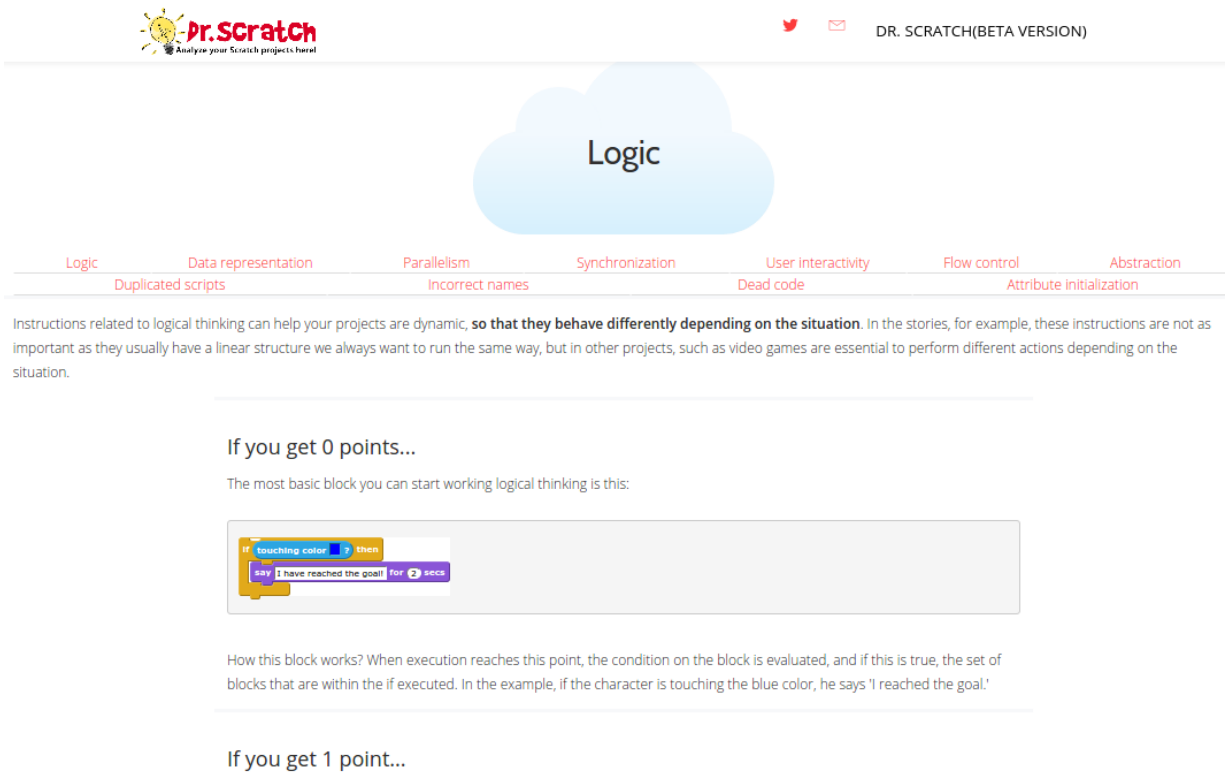
Present version of the tool, developed after the Google RISE Award 2015

Web-based service: Dr. Scratch



Feedback report

Web-based service: Dr. Scratch

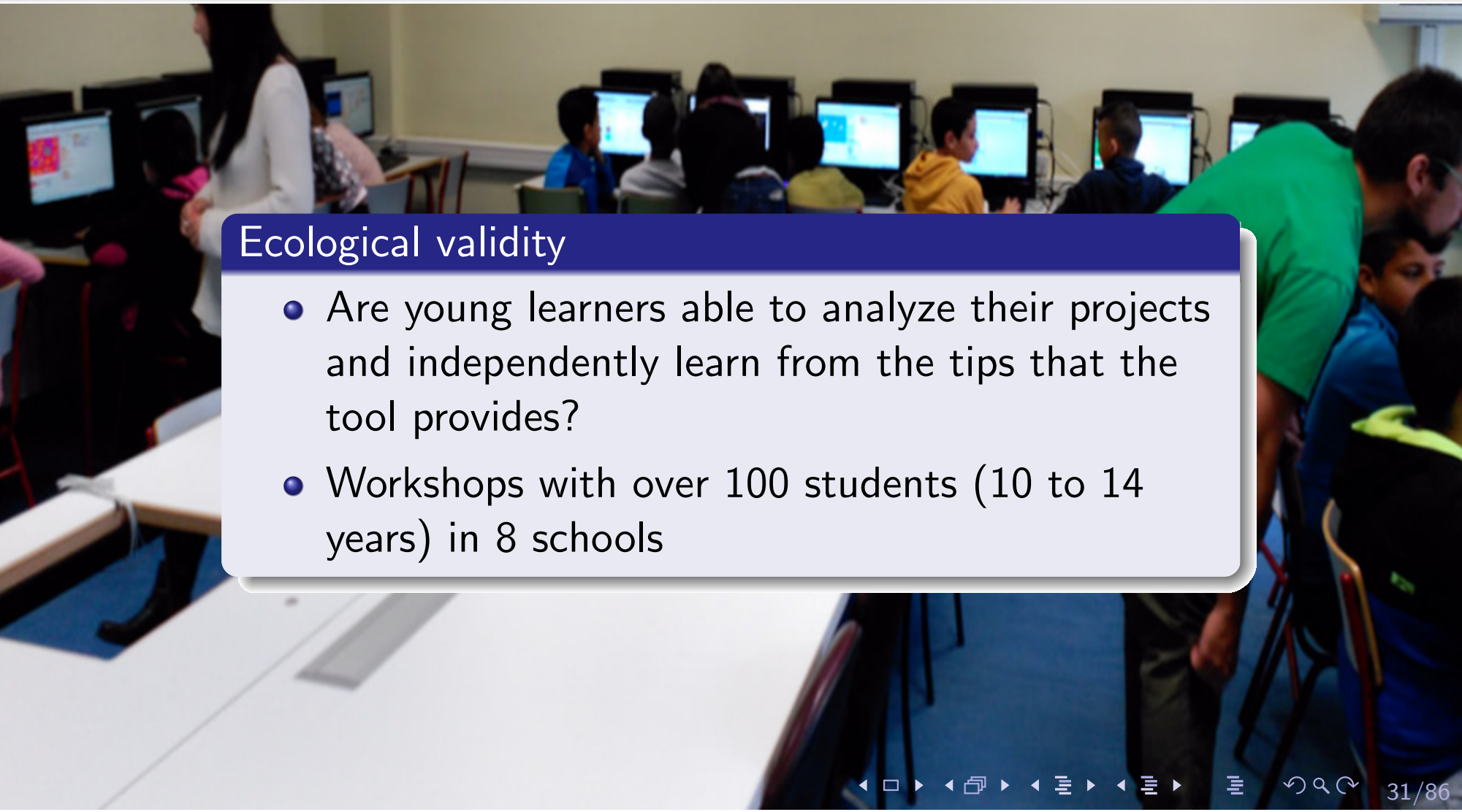


Ideas to improve the score

Validation Process

- Ecological validity
- Convergent validity
- Discriminant validity

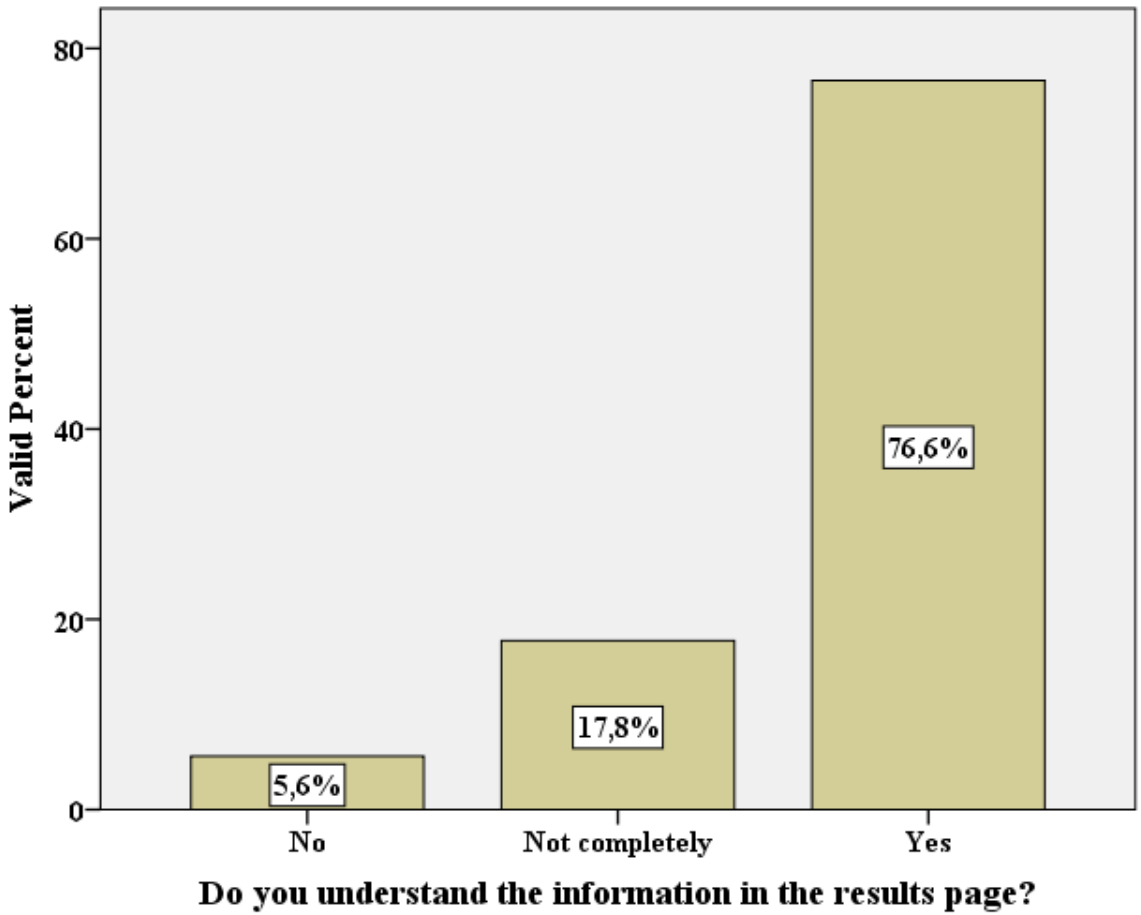
Validation Process: Ecological Validity (I)



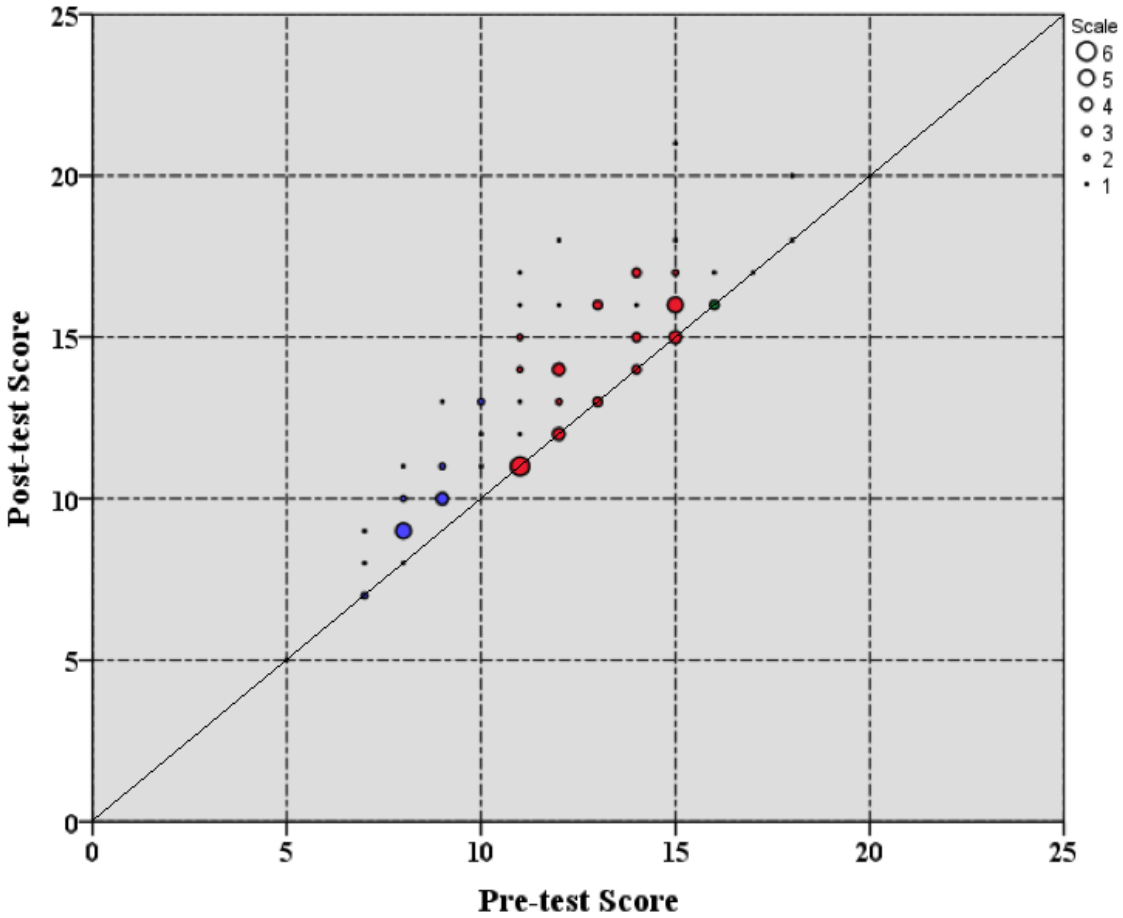
Ecological validity

- Are young learners able to analyze their projects and independently learn from the tips that the tool provides?
- Workshops with over 100 students (10 to 14 years) in 8 schools

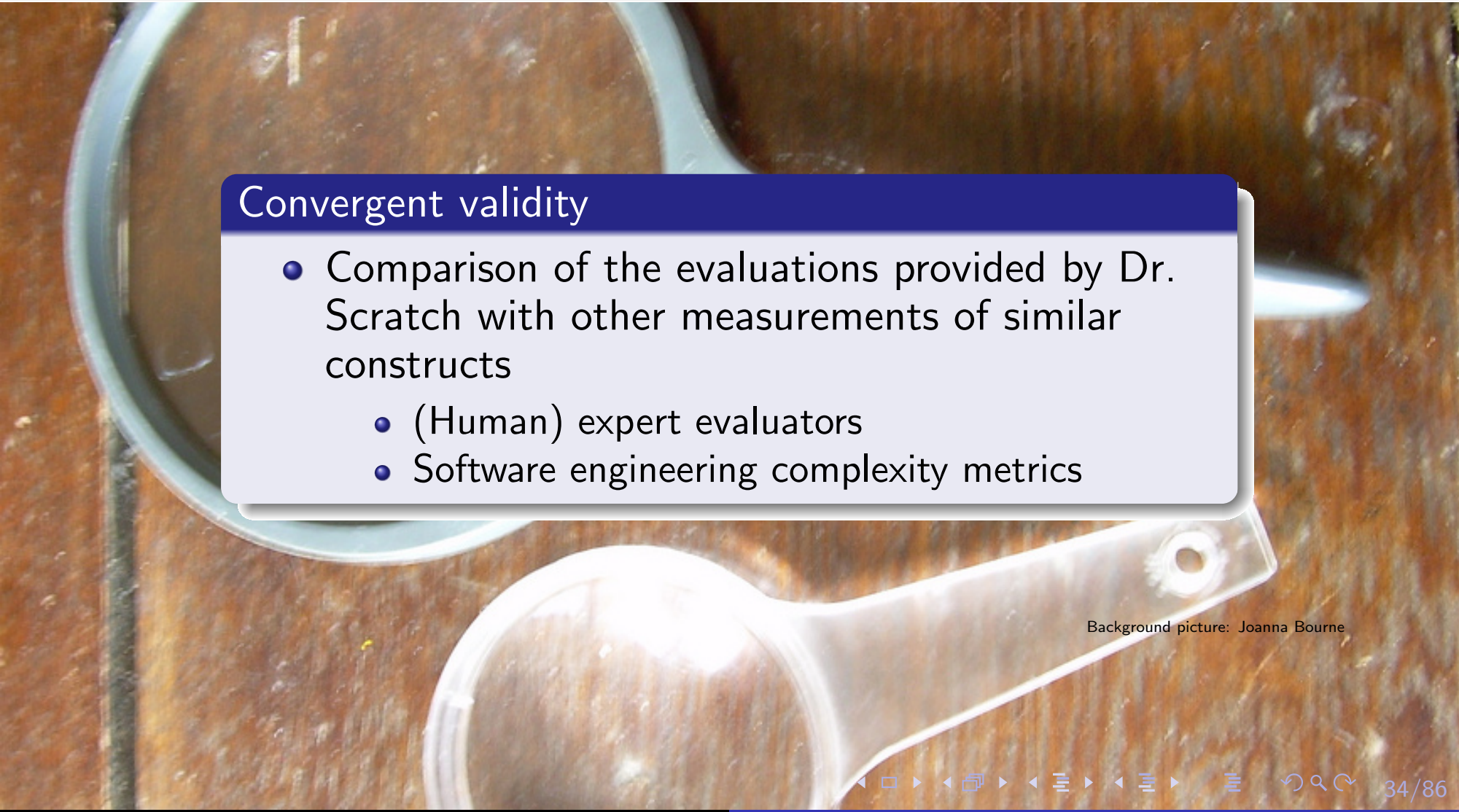
Validation Process: Ecological Validity (II)



Validation Process: Ecological Validity (and III)



Validation Process: Convergent Validity (I)

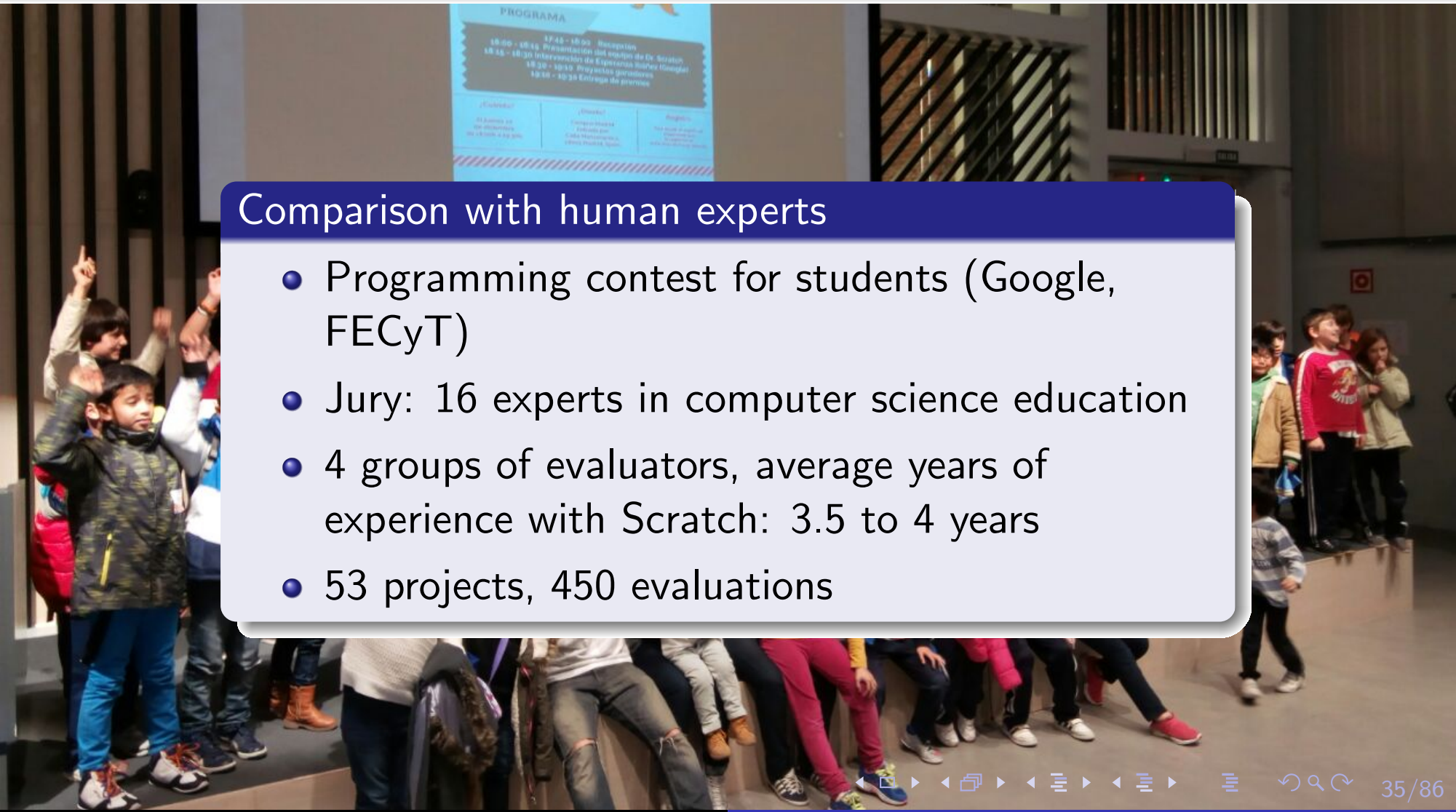


Convergent validity

- Comparison of the evaluations provided by Dr. Scratch with other measurements of similar constructs
  - (Human) expert evaluators
  - Software engineering complexity metrics

Background picture: Joanna Bourne

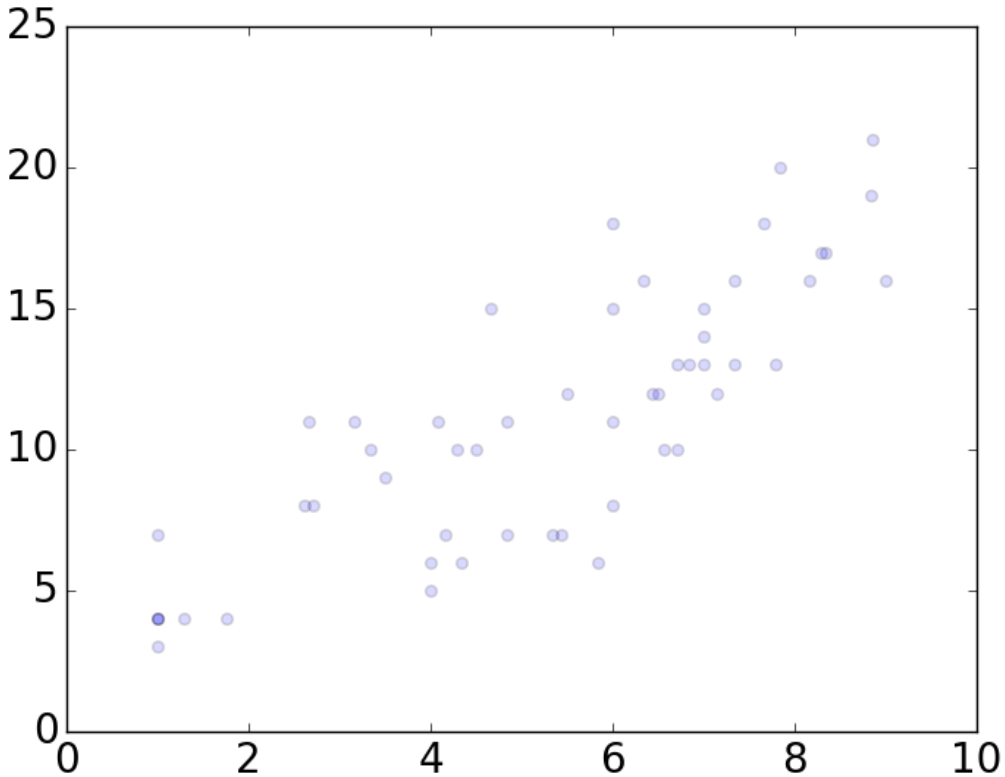
Validation Process: Convergent Validity (II)



Comparison with human experts

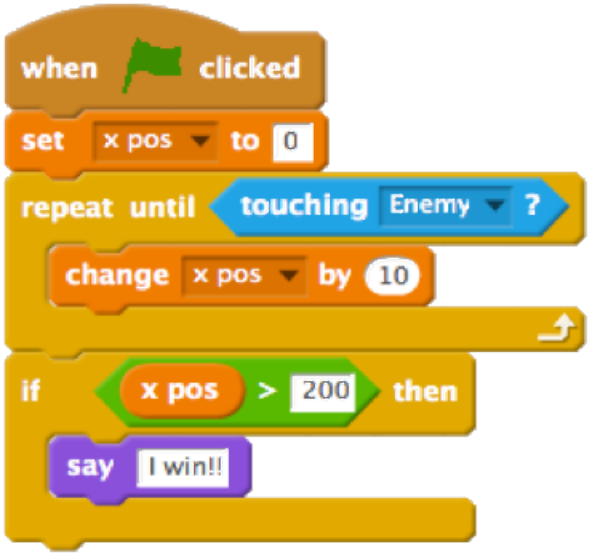
- Programming contest for students (Google, FECyT)
- Jury: 16 experts in computer science education
- 4 groups of evaluators, average years of experience with Scratch: 3.5 to 4 years
- 53 projects, 450 evaluations

Validation Process: Convergent Validity (III)



Scatter plot for experts evaluation (x-axis) and Dr. Scratch assessment (y-axis).

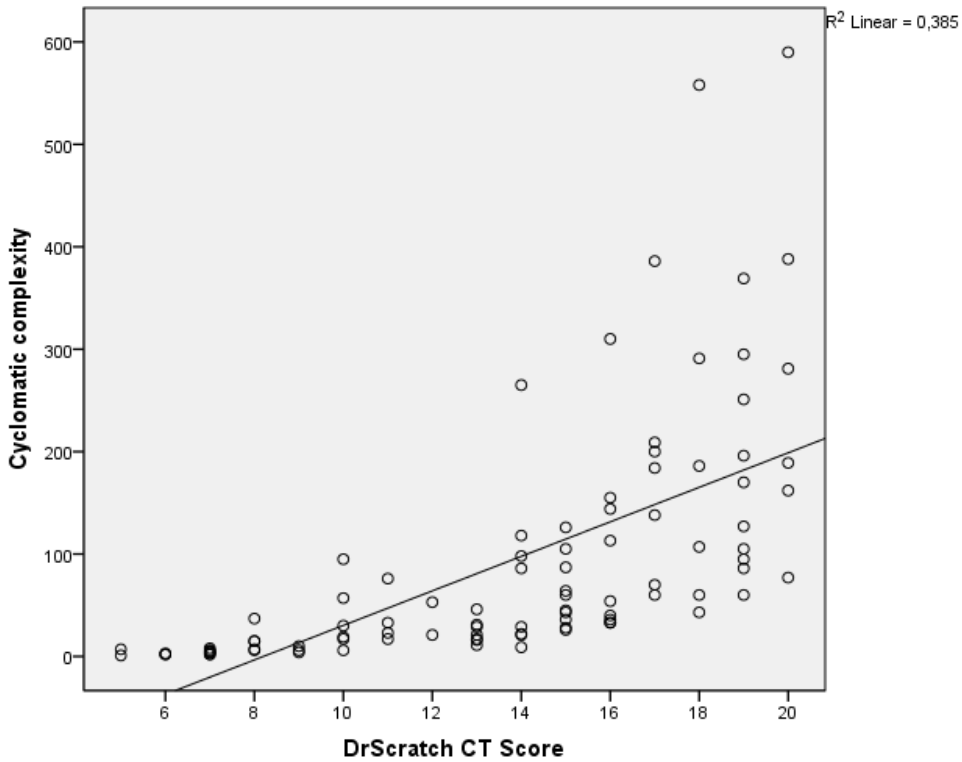
Validation Process: Convergent Validity (IV)



Metric	Value
Cyclomatic complexity	3
Vocabulary	14
Length	14
Volume	66.42
Difficulty	53.30
Effort	293.86

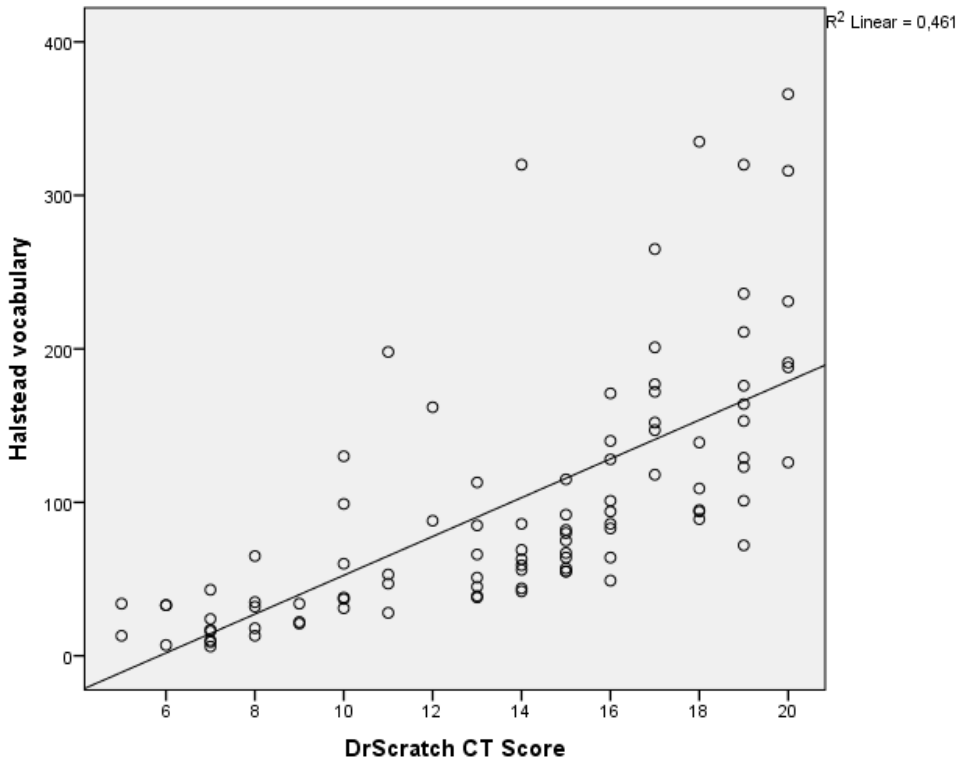
Implementation of CC and Halstead's analyzer for `Scratch` projects

Validation Process: Convergent Validity (V)



Scatter plot for Dr. Scratch assessment (x-axis) and Cyclomatic Complexity (y-axis).

Validation Process: Convergent Validity (and VI)



Scatter plot for Dr. Scratch assessment (x-axis) and Halstead's vocabulary (y-axis).

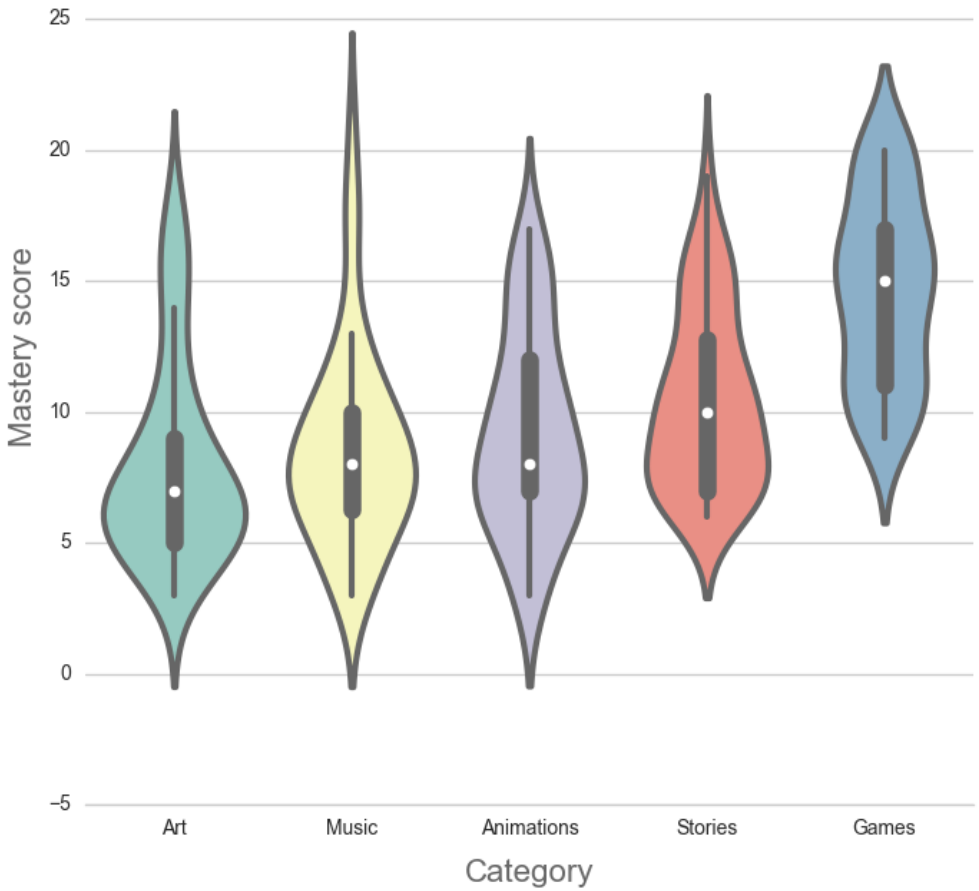
Validation Process: Discriminant Validity (I)

Discriminant validity

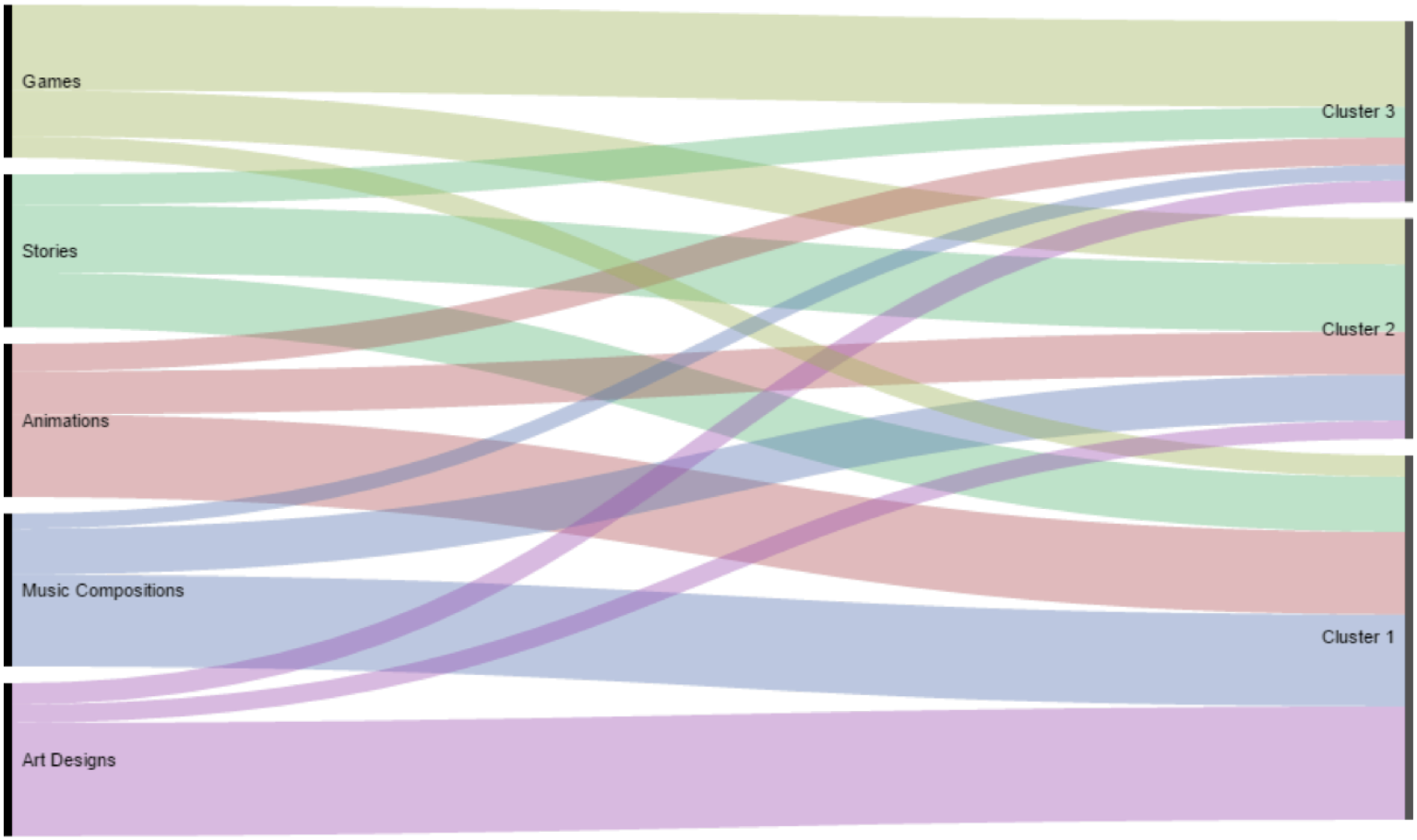
- Projects shared in the `Scratch` repository are categorized under one or more project types: games, animations, music, art and stories.
- Is `Dr. Scratch` able to detect differences in the CT dimensions developed when programming different types of projects?

Background picture: Bodie Pyndus

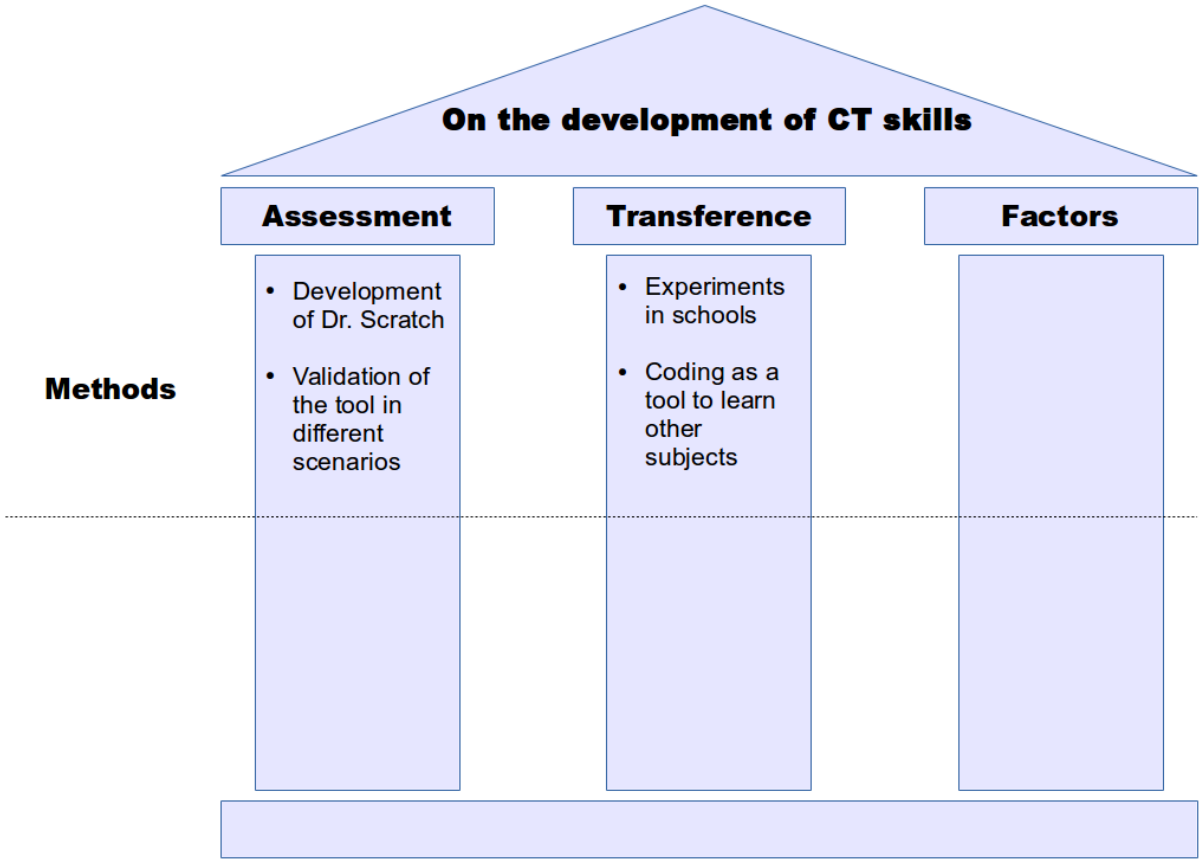
Validation Process: Discriminant Validity (II)



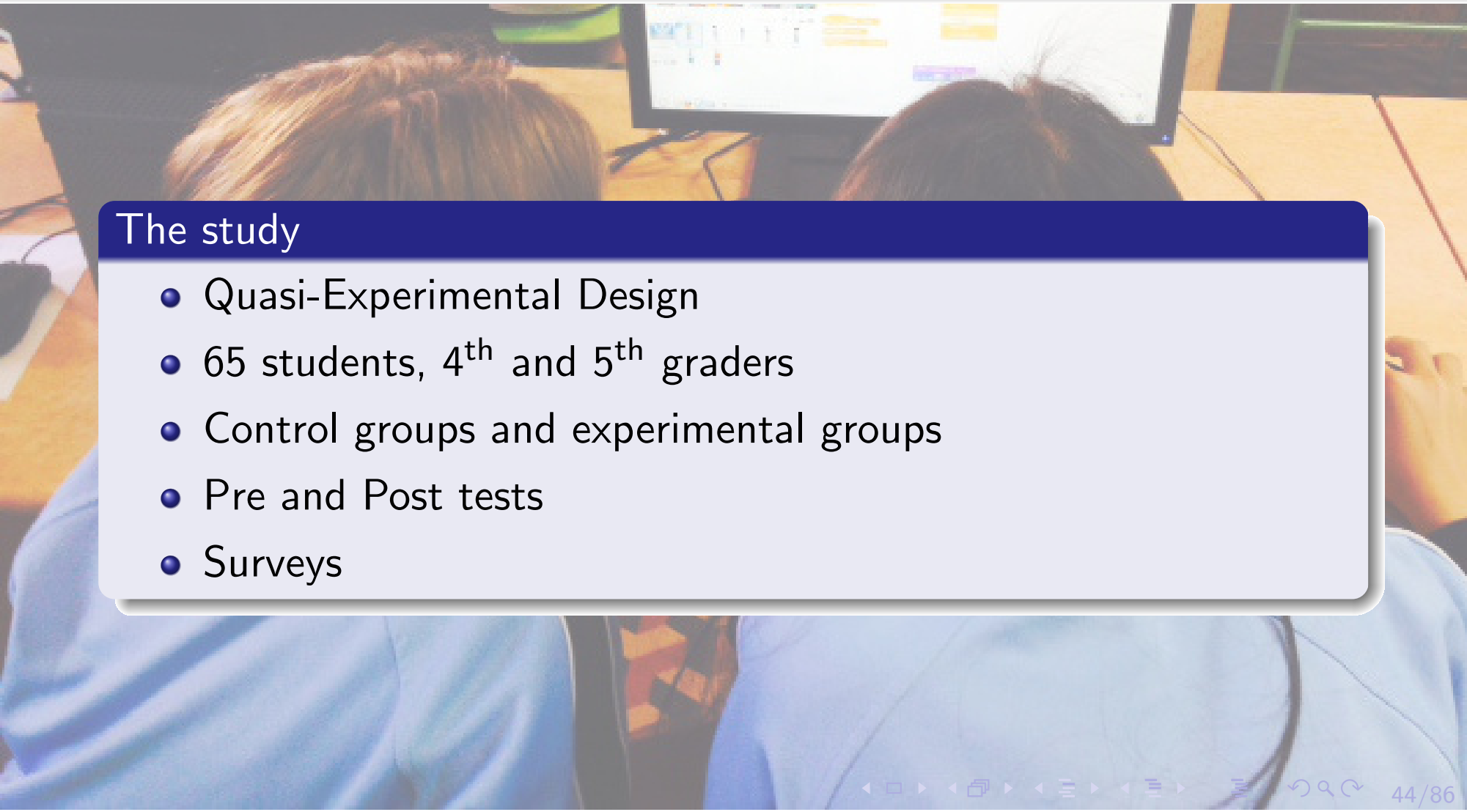
Validation Process: Discriminant Validity (and III)



Methodology



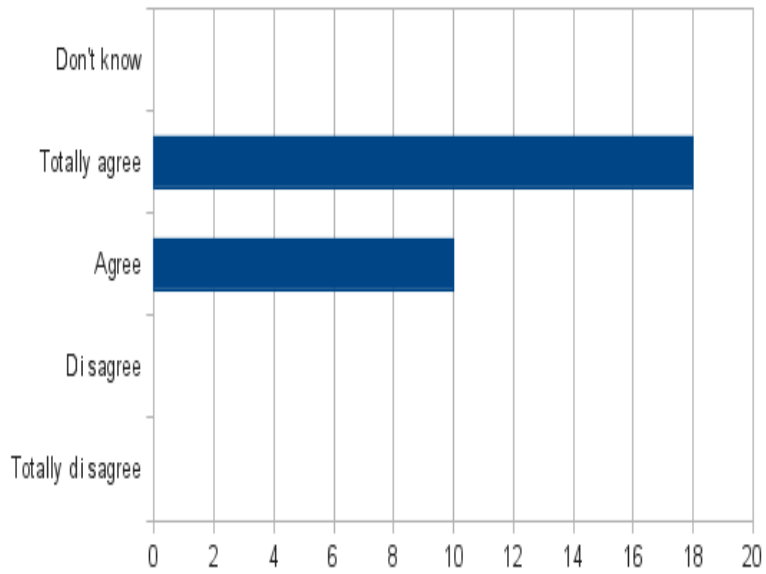
Coding in the English classroom (I)



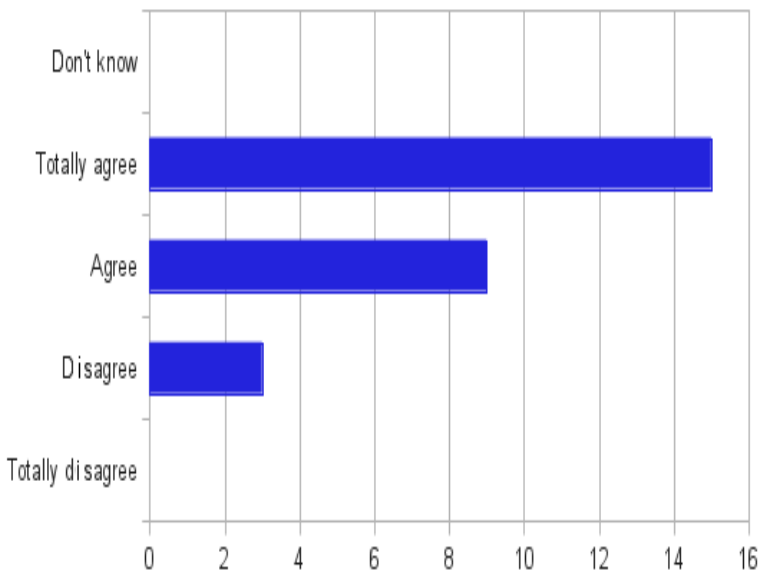
The study

- Quasi-Experimental Design
- 65 students, 4^{th} and 5^{th} graders
- Control groups and experimental groups
- Pre and Post tests
- Surveys

Coding in the English classroom (II)



Scratch helped me to learn English



Scratch made me want to learn more English

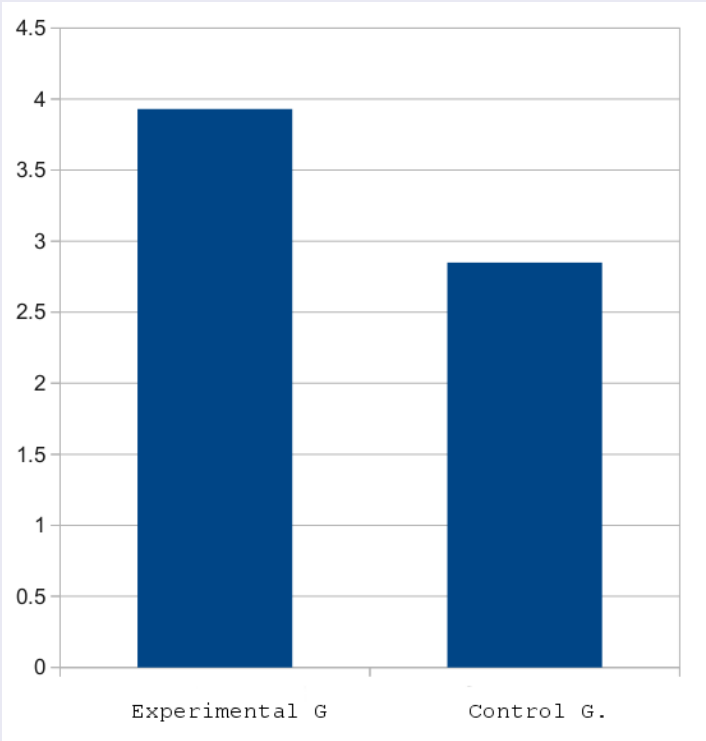
Coding in the English classroom (III)

	Experimental group	Control group
Initial test	5.05	5.13
Final test	7.7	7.55
Improvement	2.65	2.42

Table: Pre-test and Post-test results

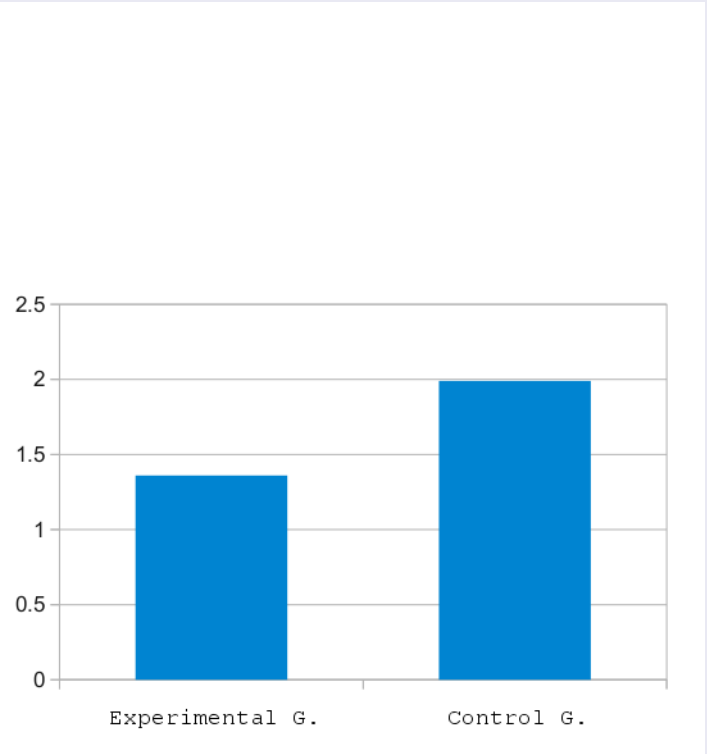
Coding in the English classroom (and IV)

4th grade



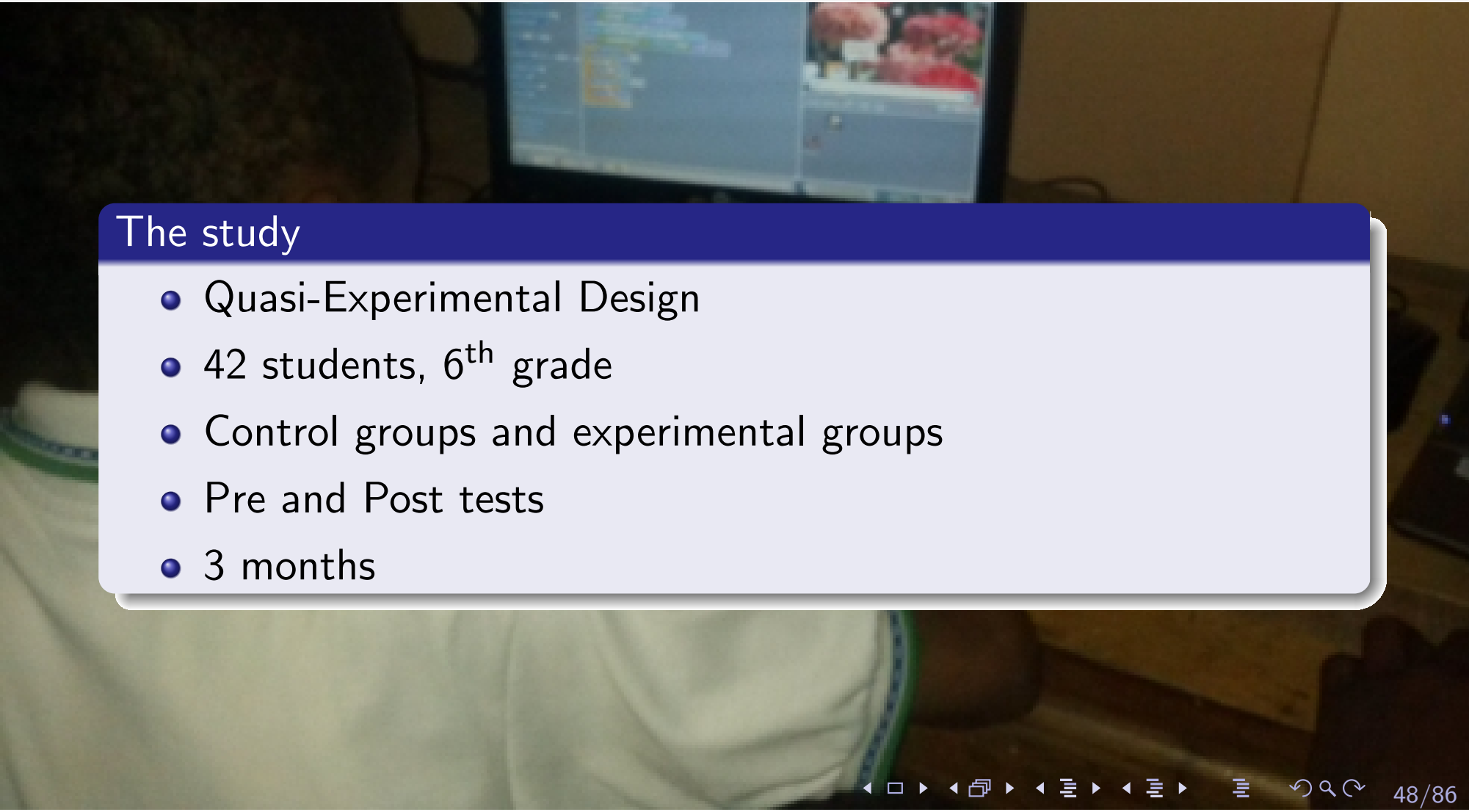
Teacher with coding experience

5th grade



First-time programmer teacher

Coding in the Maths classroom (I)



The study

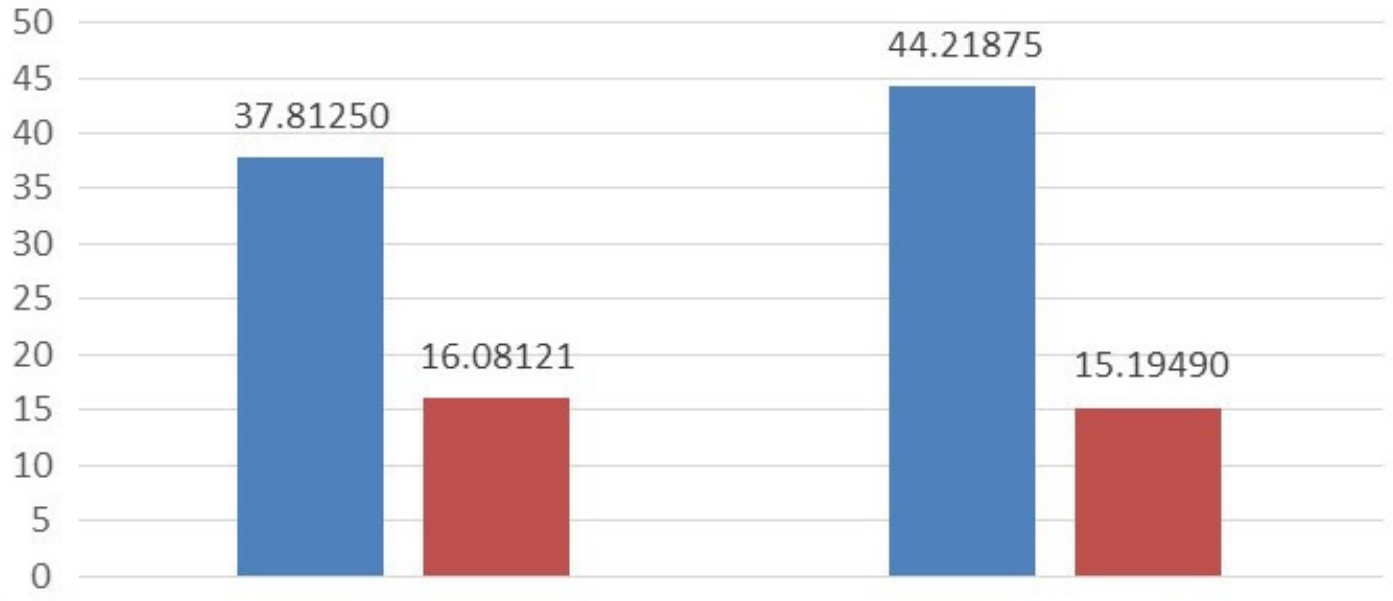
- Quasi-Experimental Design
- 42 students, 6^{th} grade
- Control groups and experimental groups
- Pre and Post tests
- 3 months

Coding in the Maths classroom (II)

Mathematical processes investigated

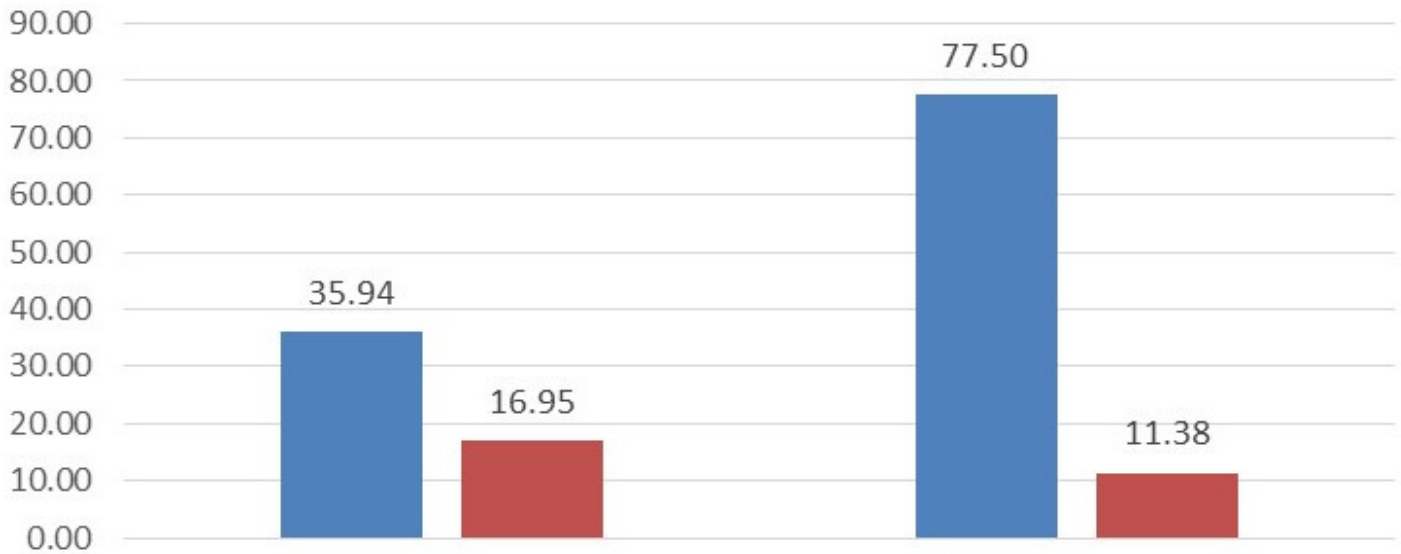
- Modeling of processes and reality phenomena
- Reasoning (making predictions and justifying arguments)
- Problem formulation and problem solving
- Exercising (comparison and execution of procedures and algorithms)
- Following guidelines set by the Ministry of National Education of Colombia

Coding in the Maths classroom (III)



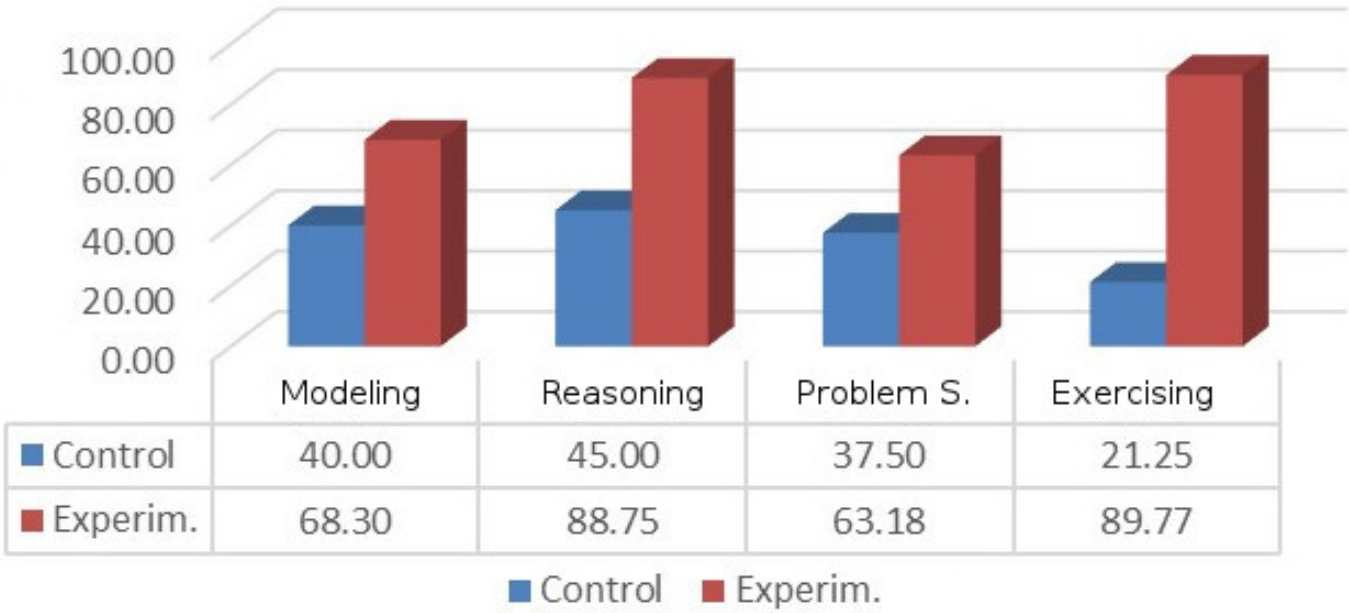
Pre-test. Mean and deviation for control (left) and experimental (right) groups.

Coding in the Maths classroom (IV)



Post-test. Mean and deviation for control (left) and experimental (right) groups.

Coding in the Maths classroom (and V)



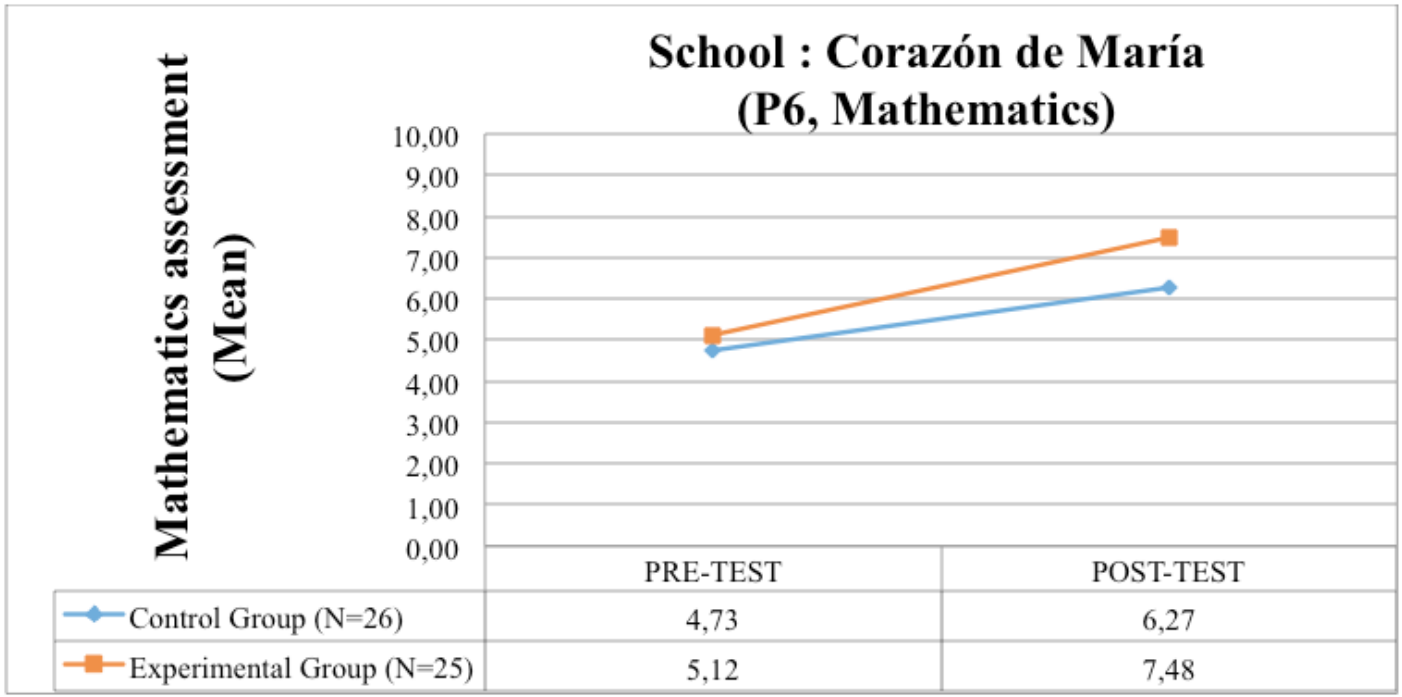
Post-test comparison. Means obtained for the control and experimental groups by mathematical process.

Comparing three quasi-experimental research designs (I)

Comparing three quasi-experimental research designs (II)

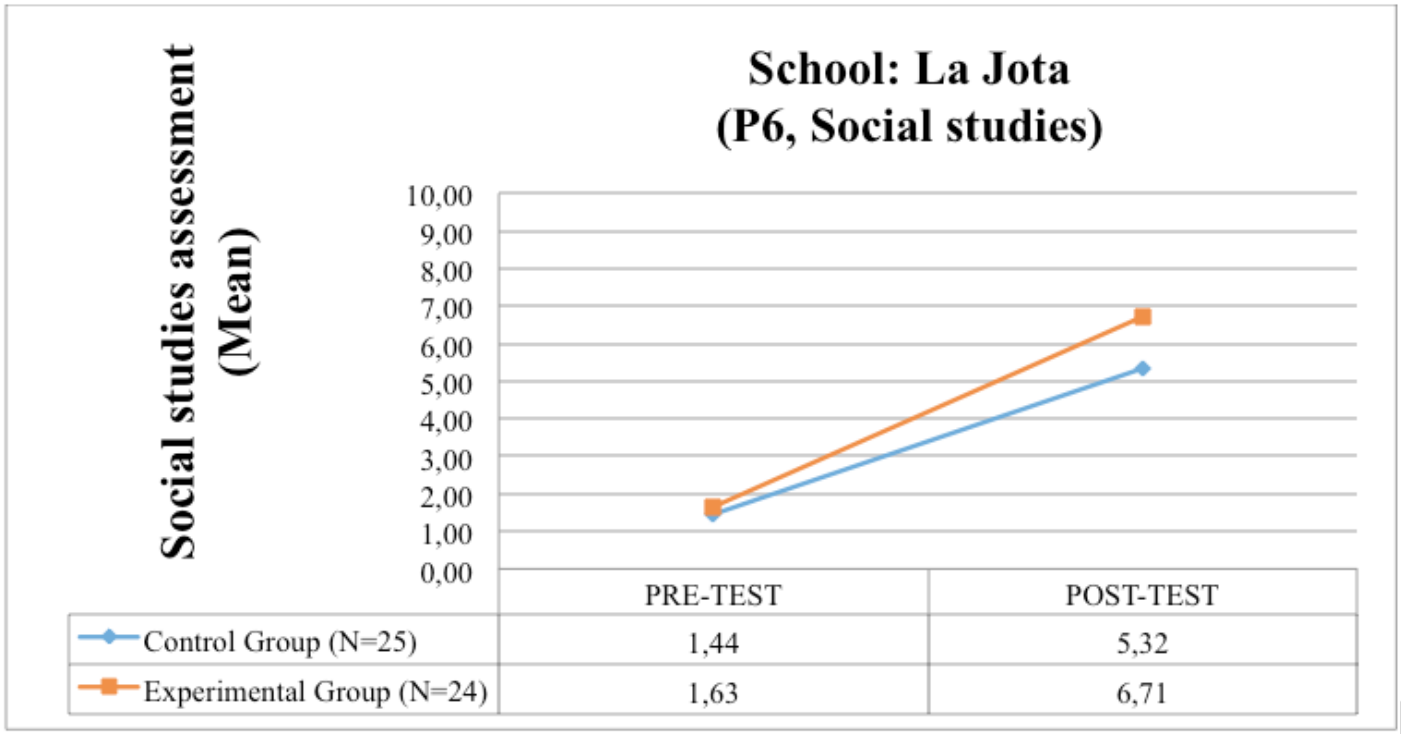
School	Grade	n (control)	n (exp.)	Subject
Corazón de María	6th	26	25	Math
La Jota	6th	25	24	Social studies
La Inmaculada	2nd	15	14	Language arts

Comparing three quasi-experimental research designs (III)



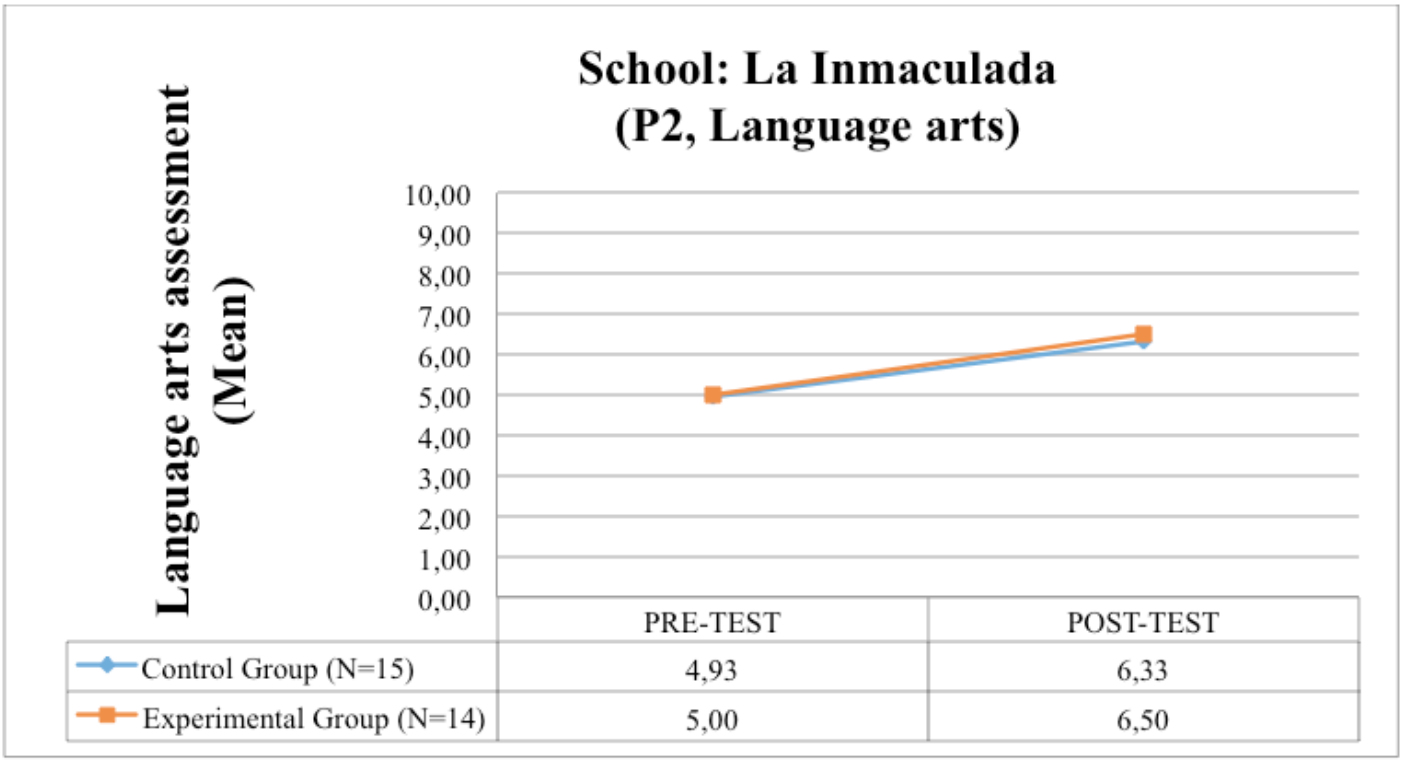
	PRE-TEST	POST-TEST
Control Group (N=26)	4,73	6,27
Experimental Group (N=25)	5,12	7,48

Comparing three quasi-experimental research designs (IV)



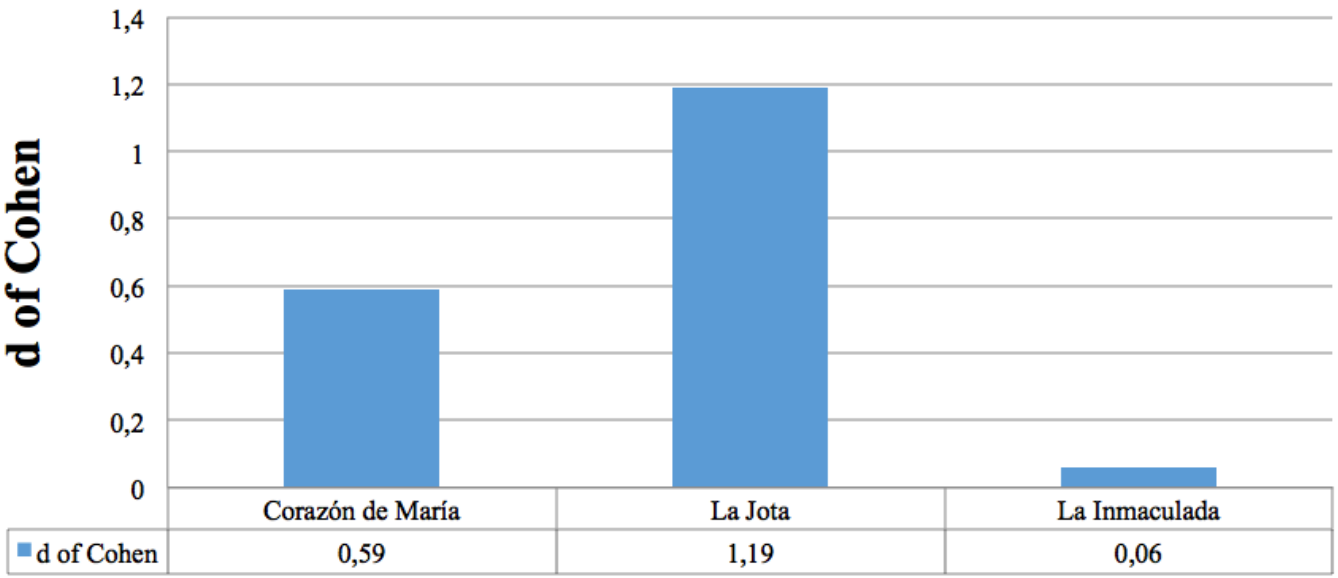
	PRE-TEST	POST-TEST
Control Group (N=25)	1,44	5,32
Experimental Group (N=24)	1,63	6,71

Comparing three quasi-experimental research designs (V)

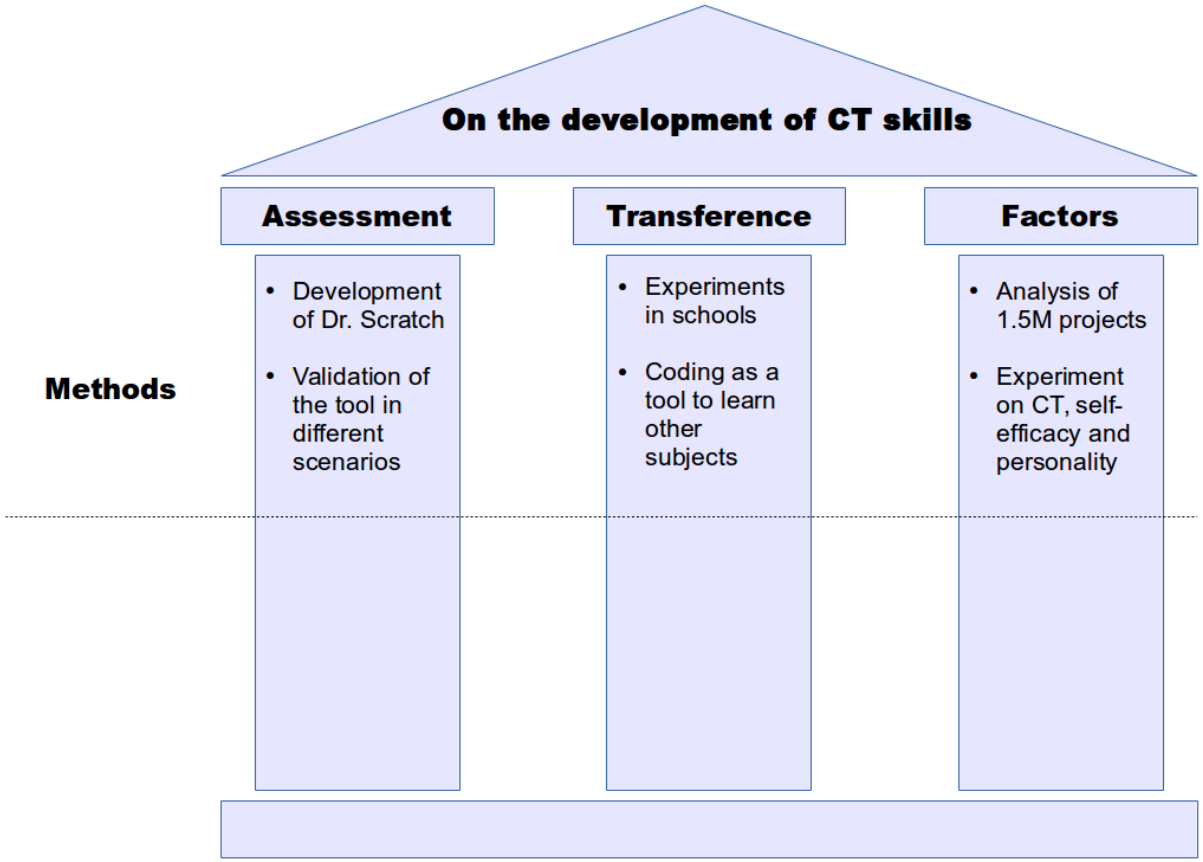


	PRE-TEST	POST-TEST
Control Group (N=15)	4,93	6,33
Experimental Group (N=14)	5,00	6,50

Comparing three quasi-experimental research designs (and VI)



Methodology



How socialization affects the learning of coding skills (I)

Data set of public activity in the Scratch online community during its first five years of activity (2007–2012)

- **1,056,951 users**
- **1,928,699 projects**
- **120,097 galleries**
- **1,313,200 friends**
- **7,788,414 comments in projects**

Background picture: xkl

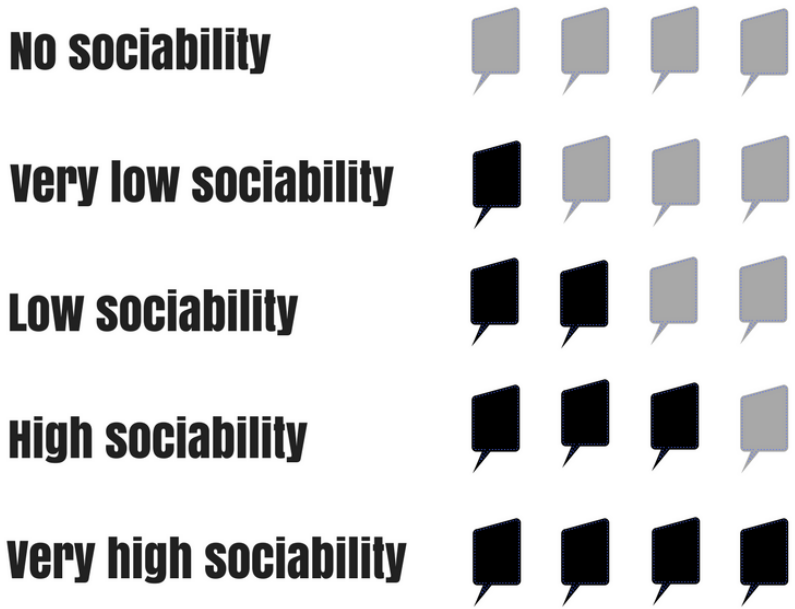
How socialization affects the learning of coding skills (II)

Measuring the sophistication of the projects from three points of view:

- **Breadth**: range of different features that programmers use
- **Depth**: amount with which programmers use those features
- **Finesse**: user's ability to solve programming problems effectively and creatively

How socialization affects the learning of coding skills (III)

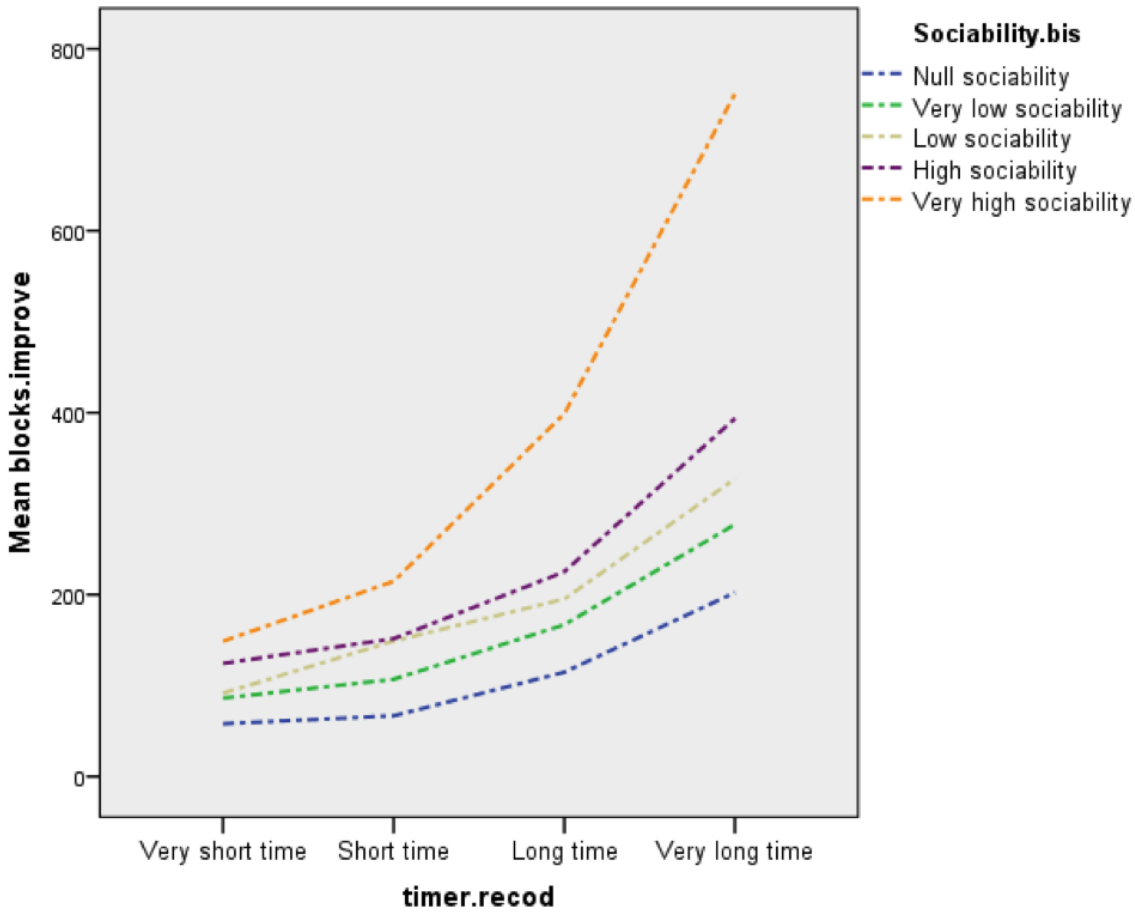
Sociability is measured as the sum of the social actions: number of favorites, galleries, friends and comments



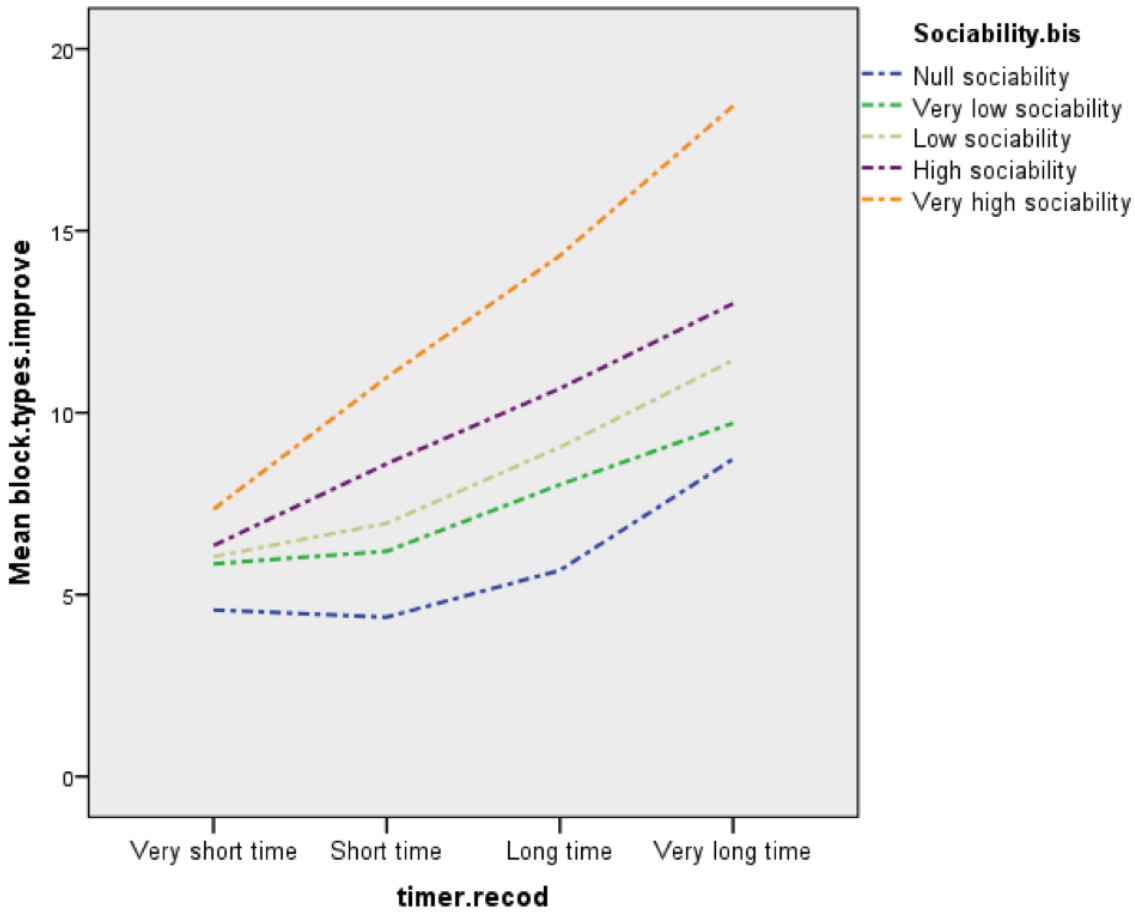
How socialization affects the learning of coding skills (IV)

4 groups of users using the quartiles of the the number of days that they spend in the community

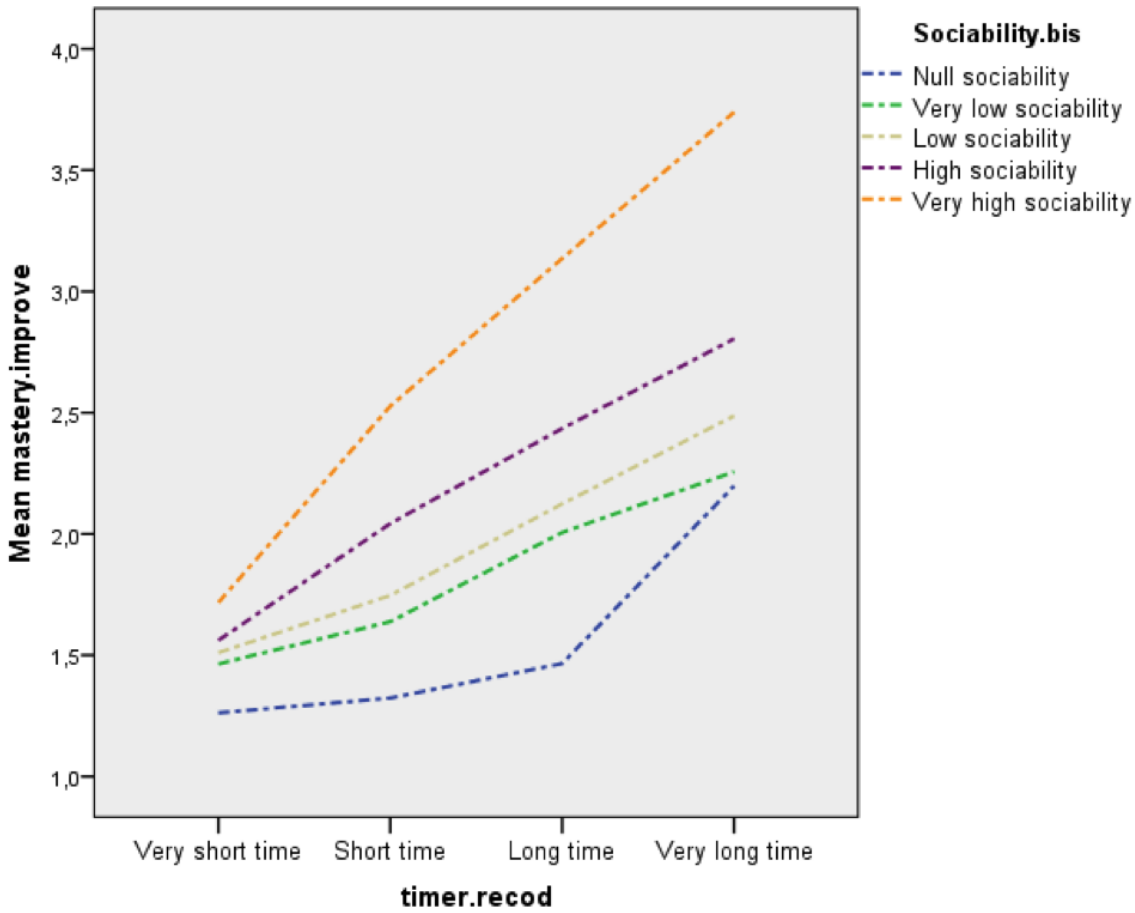
How socialization affects the learning of coding skills (V)



How socialization affects the learning of coding skills (VI)



How socialization affects the learning of coding skills (VII)



Non-cognitive factors of CT (I)

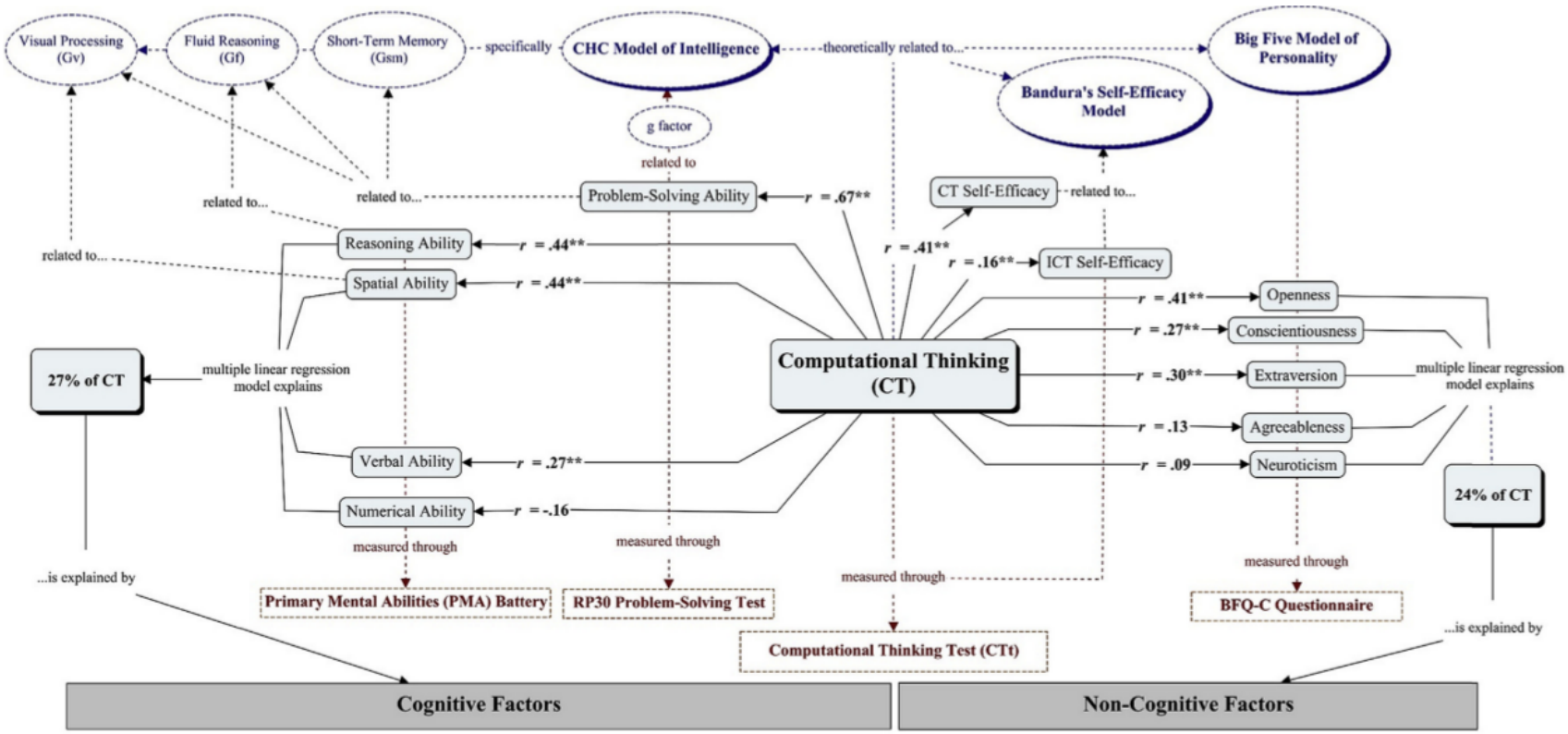
The study

- Correlations between CT, self-efficacy and the dimensions of the Big Five model of personality: Openness to Experience, Conscientiousness, Extroversion, Agreeableness, and Neuroticism.
- 1,251 students from 24 schools, enrolled in CS from 5^{th} to 10^{th} grade
- CTt [Román-González, 2015] and some additional self-efficacy items
- 99 students took the Big Five Questionnaire-Children version [Barbaranelli et al., 2003]

Non-cognitive factors of CT (II)

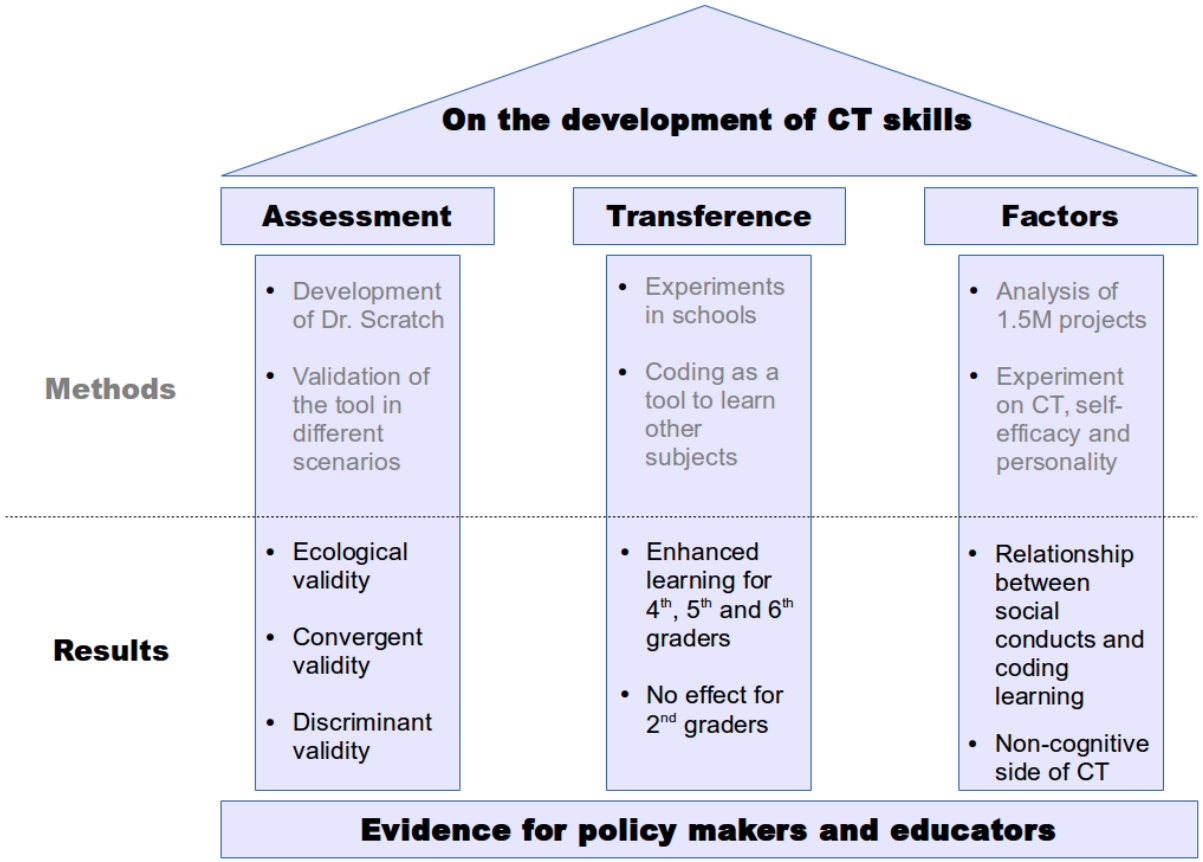
- As expected, positive correlations with Openness and Conscientiousness
- Unexpectedly, positive correlation with Extraversion
- Positive correlation with CT-Self efficacy (medium size difference in favor of males)

Non-cognitive factors of CT (and III)



Nomological network of CT including cognitive and non-cognitive factors

Summary of results



Conclusions

Who is using Dr. Scratch?

- Learners and educators from all over the world: over 500,000 analyzed projects since August 2015
- Scholars:
  - To analyze students' performance
  - To assess teachers' CT and programming skills
  - To find *code smells* and quality issues
  - To validate other tools
  - As part of an evaluation framework

Transference of CT across the K-12 curriculum

- School level: differences based on the age of the students
- School subject: simple vs complex subjects
- Teacher training: great impact

Social and non-cognitive factors affecting CT

- Educators could choose between platforms for learning to program based on their social and software evolution features
- The one-size-fits-all approach is leaving some students, especially adolescent girls, behind
- A diversity of computing contexts must be offered

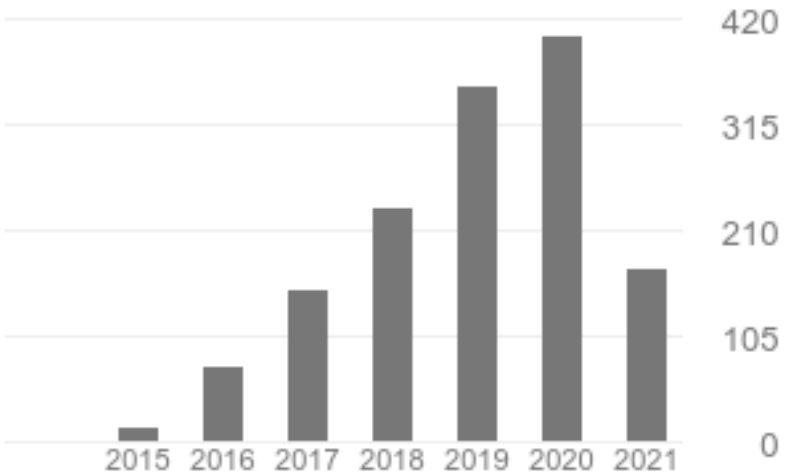
Main publications

- Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch**. IEEE Transactions on Emerging Topics in Computing, 2020.
  DOI: 10.1109/TETC.2017.2734818. JCR Q1
- Marcos Román-González, Juan-Carlos Pérez-González, Jesús Moreno-León and Gregorio Robles. **Extending the nomological network of computational thinking with non-cognitive factors**. Computers in Human Behavior, 2018.
  DOI:10.1016/j.chb.2017.09.030. JCR Q1
- Jesús Moreno-León, Marcos Román-González, Casper Harteveld, and Gregorio Robles. **On the automatic assessment of computational thinking skills: A comparison with human experts**. In Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems, CHI EA '17, pages 2788–2795, New York, NY, USA, 2017. ACM.
  DOI: 10.1145/3027063.3053216. GGS Conference Rating: CORE A++
- Jesús Moreno-León, Gregorio Robles, and Marcos Román-González. **Examining the relationship between socialization and improved software development skills in the Scratch code learning environment**. Jounal of Universal Computer Science, 22(12):1533–1557, 2016.
  DOI: 10.3217/jucs-022-12-1533. JCR Q4

Google Scholar

Citado por

	Total	Desde 2016
Citas	1420	1393
Índice h	20	20
Índice i10	26	26



Lots of things to be done...

These days...

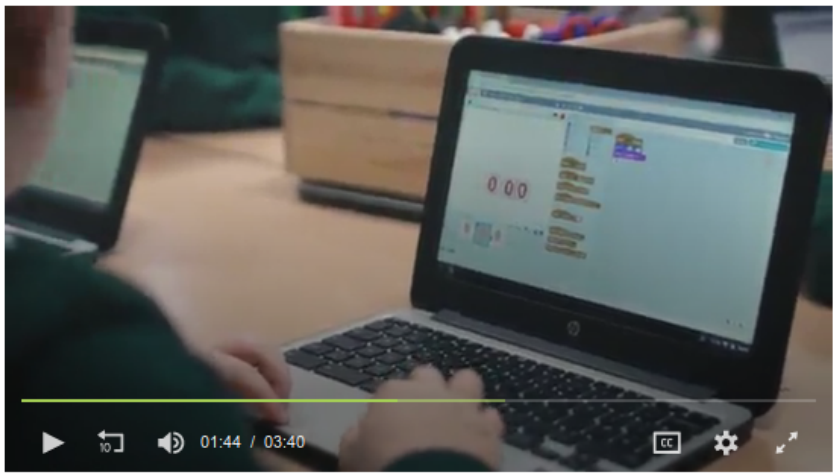
Replication of the Scratch Maths project

UCL

News and events | Departments and Centres | Clients and Partners | People | About the IOE | Contact the IOE

UCL Home » Institute of Education » Research » Projects » ScratchMaths

ScratchMaths



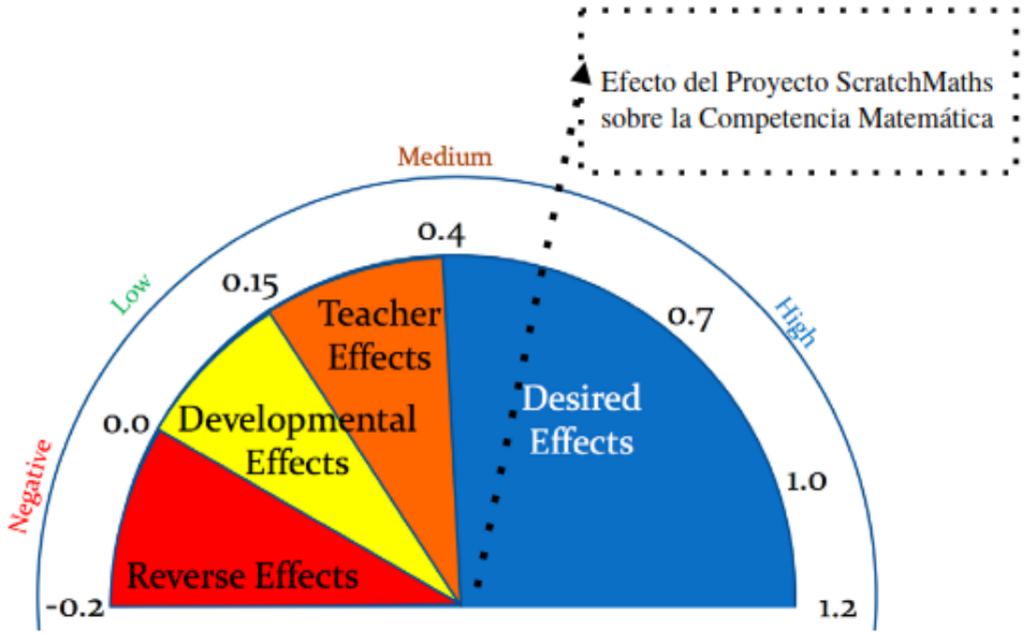
ScratchMaths

Contact us

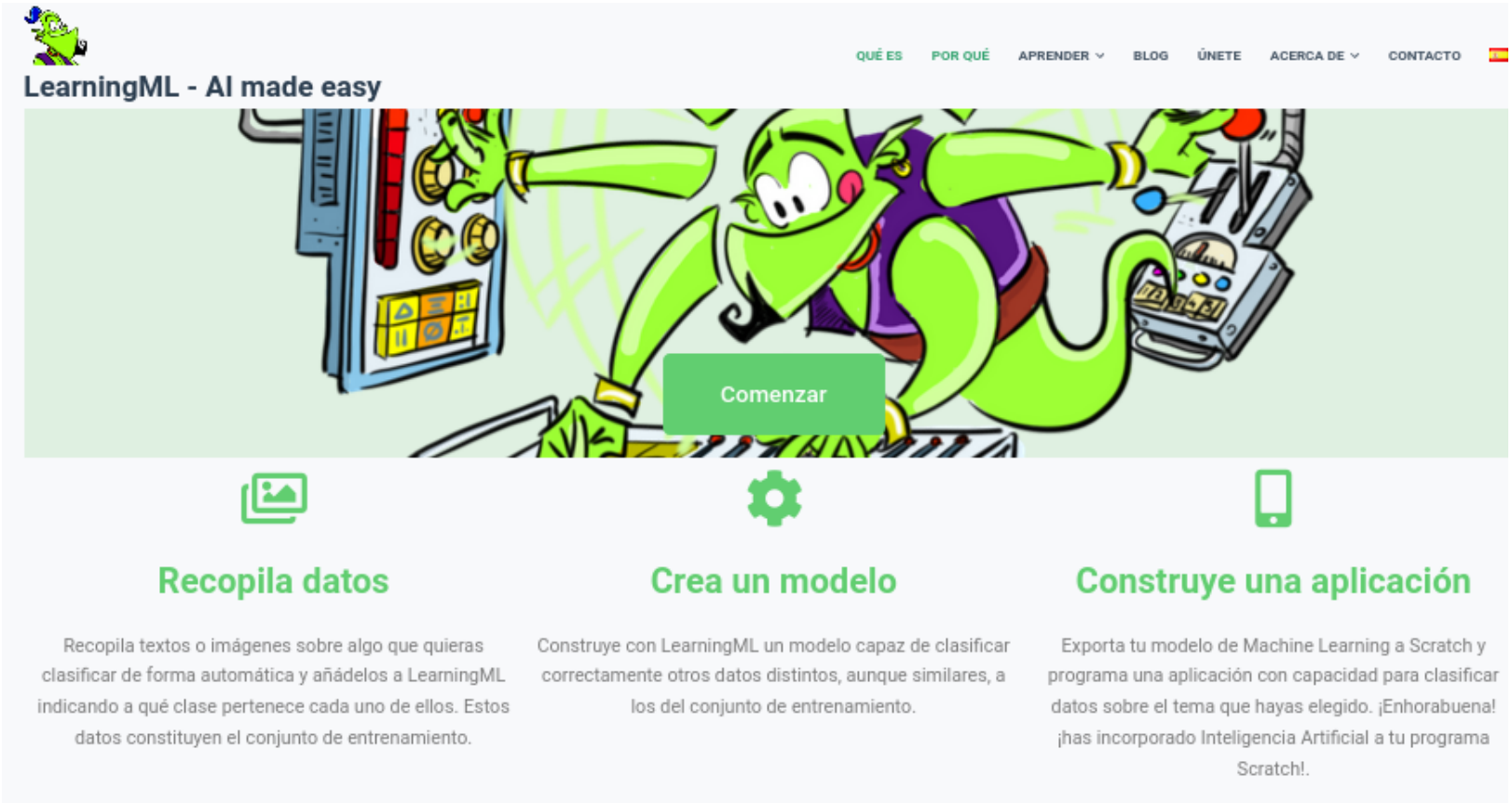
UCL Knowledge Lab
Department of Culture, Communication and

Replication of the Scratch Maths project

Gráfico 12. "Barómetro de la influencia" (Hattie, 2009).



Learning Machine Learning



At SIGCSE 2021

RESEARCH-ARTICLE

Evaluation of an Online Intervention to Teach Artificial Intelligence with LearningML to 10-16-Year-Old Students

Authors: Juan David Rodríguez-García, Jesús Moreno-León, Marcos Román-González, Gregorio Robles

Authors Info & Affiliations

SIGCSE '21: Proceedings of the 52nd ACM Technical Symposium on Computer Science Education • March 2021 • Pages 177–183 • https://doi.org/10.1145/3408877.3432393

Published: 03 March 2021

0 163

Get Access

SIGCSE '21: Proceedings of the...
Evaluation of an Online Intervention t...
Pages 177–183

← Previous Next →

AI investigation with over 6K students

Investigación sobre el desarrollo del pensamiento computacional en la educación

Jesús Moreno León - KGBL3

Conferencias de investigación para posgrado
Programa de doctorado en ingeniería informática

17/06/2021

UNIVERSIDAD COMPLUTENSE MADRID

References I

Barbaranelli, C., Caprara, G. V., Rabasca, A., and Pastorelli, C. (2003).
A questionnaire for measuring the big five in late childhood.
Personality and Individual Differences, 34(4):645–664.

Boehm, B. W., Madachy, R., Steece, B., et al. (2000).
Software cost estimation with Cocomo II with Cdrom.
Prentice Hall PTR.

Brennan, K. and Resnick, M. (2012).
New frameworks for studying and assessing the development of computational thinking.
In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada*, pages 1–25.

Koh, K. H., Basawapatna, A., Bennett, V., and Repenning, A. (2010).
Towards the automatic recognition of computational thinking for adaptive visual language learning.
In *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*, pages 59–66.

Román-González, M. (2015).
Computational thinking test: Design guidelines and content validation.
In *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies (EDULEARN 2015)*, pages 2436–2444.

References II

Seiter, L. and Foreman, B. (2013).

Modeling the learning progressions of computational thinking of primary grade students.
In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ICER '13, pages 59–66, New York, NY, USA. ACM.

Sentance, S. (2015).

Annual national survey 2015: Results.
Technical report, Computing At School.

Wilson, A., Hainey, T., and Connolly, T. (2012).

Evaluation of computer games developed by primary school children to gauge understanding of programming concepts.
In *European Conference on Games Based Learning*, page 549. Academic Conferences International Limited.