Conocimiento sobre programas: un abordaje filosófico, técnico y educativo.

Resumen

En la actualidad disciplinas tan distintas como la medicina, la economía y la ingeniería, por mencionar algunos ejemplos, son impensables sin el uso de programas. La programación ocupa un lugar central en la actualidad, por sus aplicaciones prácticas y por introducir una serie de conceptos que operan de un modo transversal en distintas áreas del conocimiento y la cultura.

Sin embargo, las siguientes preguntas, no sólo no tienen respuestas fáciles, sino, que ni siquiera contamos con un marco teórico adecuado para plantearlas: ¿qué es un programa?, ¿por qué confiamos, o no, en que su funcionamiento/resultado es correcto?, ¿qué significa que un programa es correcto?, ¿cuál es el vínculo entre las pruebas formales de corrección y la ejecución efectiva?, ¿cómo introducir en la educación en programación estos problemas ontológicos y epistemológicos?

El planteamiento de este tipo de preguntas produce una serie de confusiones e incomprensión que exigen aclaración. Determinar un marco teórico en el cual se haga precisa la relación que existe entre la dimensión lógica y la dimensión física del programa, entre la corrección formal y la ejecución, entre la noción de error y mal funcionamiento, es necesario para aclarar aquel tipo de pregunta y para elaborar una didáctica sensible a este tipo de problemas.

El problema del conocimiento sobre programas tiene consecuencias, tanto éticas como legales. Educar en programación y aprender a programar, desde una perspectiva integral, no sólo introduce una ganancia cognitiva, sino, también, sensibiliza a las personas sobre la dimensión epistemológica y ética de la programación.

Estos problemas tienen un espesor conceptual que por su propia naturaleza requieren de un abordaje interdisciplinario, donde se puedan estructurar aspectos conceptuales, técnicos y educativos. Avanzar en el desarrollo de un marco teórico en el cual se plantee el problema del conocimiento sobre programas y sus consecuencias prácticas es el objetivo del proyecto.

Objetivo/s generales y específicos en relación a las tres funciones universitarias

El problema del *conocimiento sobre programas*, desde la perspectiva en que se va a desarrollar en este proyecto interdisciplinario, supone la convergencia de tres áreas del conocimiento dentro de una misma unidad de análisis. Las tres áreas son la filosofía de la computación, los métodos formales en computación y la didáctica de la computación.

La unidad está dada por el problema del *conocimiento sobre programas*, es decir, por los problemas que surgen al responder: *¿qué significa que el agente S sabe que el programa P es correcto?* Lo cual supone elaborar la relación que existe entre el *programador* (el agente cognitivo), las *reglas formales* que constituyen el programa-texto y el *programa-físico* que las ejecuta.

Esta pregunta tiene una dimensión filosófica, técnica y educativa.

1. *¿Qué es un programa?*, ¿qué tipo de conocimiento podemos tener sobre los programas según las distintas epistemologías de la computación?, ¿cuál es el vínculo entre la dimensión lógica y física del programa?

2. *¿Qué significa desde un punto de vista científico tener una prueba formal de corrección?*, sobre todo, considerando la discusión del punto anterior.

3. *¿De qué modo se pueden incorporar los problemas epistemológicos de la programación en su didáctica?* La respuesta debería armonizar las discusiones de los puntos anteriores y una perspectiva didáctica consistente que refleje las dificultades propias que existen en la comprensión de los problemas anteriores.

A.1) Objetivos generales en Investigación

El objetivo general de la investigación consiste en desarrollar un marco teórico en el cual:

1. Se puedan plantear de un modo sistemático los problemas ontológicos y epistemológicos que surgen en relación a la pregunta por el conocimiento acerca de los programas, así como el estatus científico de la computación.

2. Evaluar qué tipo de conocimiento adquirimos sobre los programas cuando tenemos una prueba formal de corrección, y la importancia del problema de la corrección de programas en relación a los sistemas críticos.

3. Evaluar el tipo de dificultades cognitivas y didácticas que surgen en relación al problema de la educación en programación cuando se intenta dar cuenta de los problemas anteriormente mencionados.

A.2) Objetivos específicos en Investigación

Definiremos los objetivos específicos en investigación separados en tres grupos, pero mostrando las conexiones entre ellos.

A. Filosofía de la computación:

1. Problema ontológico:

    1. Explicitar y plantear de un modo crítico las respuestas al problema ontológico, es decir, a la pregunta, ¿qué es un programa? sobre todo, en relación a las tres corrientes principales que determina Eden (2007).

    2. Evaluar críticamente la posición dualista Colburn (2000), Moor (1978).

    3. Evaluar la distinción entre procesos lógicos y físicos, Eden (2007), Copeland (1996).

2. El problema de la implementación:

    1. Explicitar y plantear de un modo crítico el problema del dualismo y la relación entre el programa-texto y el programa-físico, Colburn (2000).

2. Explicitar y evaluar la respuesta de Rapaport (1999, 2005), en términos de implementación del programa-texto en un dispositivo físico.

3. Explicitar y evaluar los argumentos en relación al pancomputacionalismo, que surgen en los trabajos Putnam y Searle, Chalmers (1996), como respuesta al tipo de relación que existe entre el programa-texto y el programa-máquina. Explicitar las respuestas que aparecen en el trabajo de Piccinini (2008).

3. Especifación, función, error y mal funcionamiento:

    1. Explicitar el problema de la especificación, Summerville (2012), en particular, el problema de la especificación funcional, Kroes (2010).

    2. Explicitar la teoría de la función de Kroes (2012), el problema de la especificación, como criterio externo al proceso-físico y como condición de posibilidad para un criterio de corrección.

    3. Explicitar un marco teórico para aclarar la diferencia entre error y mal funcionamiento. Este problema tiene consecuencias legales importantes, en la medida en que el programador no es responsable del mal funcionamiento, aunque sí es responsable del error. Sin embargo, la distinción depende de una serie de distinciones conceptuales que en general no son explícitas y no siempre son claras.

4. Errores categoriales en la discusión sobre corrección de programas:

    1. Explicitar y evaluar el problema del conocimiento sobre programas. ¿Qué tipo de conocimiento tenemos sobre el programa cuando sabemos que es formalmente correcto?

    2. Explicitar y evaluar el debate que se ha llevado adelante acerca del valor que tienen las pruebas de corrección formal de los programas. Se suele confundir en esta discusión el nivel lógico del programa con el nivel físico. Blanco (2008) plantea el problema en términos de un error categorial que suele operar en la discusión. Aclarar la siguiente pregunta: tener una prueba de corrección formal ¿significa que el programa no puede fallar? La respuesta intuitiva es: no. Sin embargo, esto no significa que no podamos conocer a priori el comportamiento del programa, lo cual genera una serie de confusiones conceptuales. En la investigación se buscará dar un marco teórico para responder a esta pregunta.

5. La discusión sobre pruebas matemáticas:

    1. Explicitar la discusión sobre corrección de programas tiene consecuencias en discusiones matemáticas. El problema de los cuatro colores ha sido un ejemplo paradigmático del tipo de dificultades que surgen en la matemática cuando algún resultado depende de un cálculo que realiza una computadora. En ese caso, ¿la matemática sigue siendo a priori? Las discusiones de los puntos anteriores pueden iluminar este tipo de problema.

2. Evaluar el vínculo entre matemática y computación, en particular, la relación que tiene el problema del conocimiento sobre programas con el problema del conocimiento sobre los algoritmos matemáticos.

B. Métodos formales:

1. El problema de los sistemas críticos:

    1. Los sistemas críticos son aquellos en los cuales un error podría tener consecuencias graves. En el marco de este tipo de sistemas, la discusión sobre corrección formal adquiere otra relevancia. Se buscan explicitar las condiciones en las cuales un sistema se considera crítico. Se busca explicitar en qué sentido en este tipo de sistemas las pruebas de corrección son importantes.

2. Se busca desarrollar un caso de estudio sobre sistemas críticos: descripción del problema, modelo del sistema, programación, pruebas de corrección.

3. Asistentes de prueba: evaluar pruebas de corrección de programas en por lo menos dos asistentes de prueba.

C. Didáctica de la informática:

El proyecto busca contribuir a la construcción de un área de investigación que estudie los problemas cognitivos y didácticos que son propios de la educación en programación. La didáctica de la informática se ha desarrollado en los últimos años, con la finalidad de responder de un modo sistemático a las preguntas: *qué, cómo, por qué y para quién* educar en la disciplina [ítem A]. El marco teórico que fundamenta nuestra investigación, es por un lado la epistemología genética de Jean Piaget [ítem B], que nos lleva desde el *cómo educar* hacia el *cómo aprenden* los estudiantes y por otro la *teoría de las situaciones didácticas* de por Guy Brousseau [B.13] que propone pautas de aplicación de la teoría piagetina para la práctica docente en matemática. Nuestra metodología de investigación se basa en el diseño, la implementación y el análisis de resultados de estudios empíricos llevados a cabo con estudiantes, fundamentados en principios del marco teórico. Señalamos tres etapas en la investigación y las relaciones entre ellas:

1. *El diseño del algoritmo*: aquí aparecen una serie de dificultades cognitivas, lógico-formales, vinculadas al pensamiento algorítmico, que han sido investigadas en una serie de estudios empíricos [ítem C], y que seguirán siendo analizadas en futuros trabajos. Aquellas investigaciones desembocaron en la construcción de un modelo de aplicación de la teoría piagetiana para la didáctica de algoritmos y estructuras de datos, que fue presentado en [C.2]. La pregunta que orienta este punto es: ¿cuáles son las dificultades cognitivas del pasaje del conocimiento instrumental al conceptual y formal? [C. 2, 3, 4, 5, 6,7]

2. *El problema de la implementación*: la programación pone en un juego una serie de dificultades que están vinculadas a la conexión que existe entre el algoritmo que es pensado y diseñado en algún lenguaje, y su implementación y ejecución física, es decir, que relacionan el pensamiento algorítmico con el pensamiento computacional. Esto introduce problemas cognitivos que no son estrictamente lógico-formales, que tienen que ver con las ideas preconcebidas del estudiante con

respecto a la máquina física. En el estudio empírico [C.1] se presenta una aproximación al problema y se espera profundizar en este línea de investigación de modo de avanzar en nuestro modelo hacia una didáctica de los programas. Las preguntas que orientan este punto son: ¿qué tipo de dificultades aparecen como consecuencia de preconceptos e ideas pre concebidas de los estudiantes sobre el vínculo entre el algoritmo y su implementación y ejecución física?, ¿cómo ayudar a los estudiantes en la construcción de conocimiento sobre la correspondencia entre las acciones que ellos describen como un algoritmo y las acciones que un computador ejecuta (un programa)?

3. *El problema de la corrección*: la discusión acerca de la ontología y epistemología de los programas, nos permitirá elaborar estudios empíricos en los cuales sea posible preguntarle a un estudiante: ¿el programa que diseñaste e implementaste es correcto?, ¿en qué sentido es correcto?, ¿cómo podrías diferenciar un error de un mal funcionamiento?, ¿cuáles son las consecuencias éticas y legales involucradas en la corrección de los programas? Estudiar la introducción de este aspecto permitirá completar el modelo teórico para la didáctica de la programación.

Algunas observaciones generales:

i. La investigación en didáctica de la informática permite generar pautas para la práctica docente, de modo que el tratamiento de los problemas didácticos deje de estar reducido a observaciones de experiencias aisladas o de opiniones personales.

ii. Conocer los problemas inherentes a la educación en programación en la Enseñanza Media es una de las preocupaciones centrales de la comunidad en informática debido a la importancia que tiene la disciplina en la formación de los estudiantes, desde temprana edad [A.2,3,5,9]

iii. Ser conscientes de los problemas ontológicos y epistemológicos de la programación, genera una actitud crítica, basada en los fundamentos de la disciplina, que nos distancia de una visión naturalizada e instrumental centrada en solamente en las aplicaciones tecnológicas (Rushkoff, 2010)

B.1) Objetivos generales en Enseñanza

Desde el punto de vista de la Enseñanza se busca introducir el problema del conocimiento sobre programas, desde un punto de vista interdisciplinario, tanto a nivel universitario, como a nivel de la Enseñanza Media.

B.2) Objetivos específicos en Enseñanza

Se busca desarrollar un curso interdisciplinario: *Introducción al problema del conocimiento de los programas: un abordaje filosófico, técnico y educativo.* El curso estará organizado en tres módulos:

Módulo 1: Filosofía de la computación: introducción al problema del conocimiento sobre programas.

Temario:

1. ¿Qué es un programa? El problema del dualismo.
2. La relación entre programa-texto y programa-físico.
3. Argumentos pancomputacionlistas. Problemas y posibles respuestas.
4. El problema de la implementación.
5. El problema de la especificación, la función, y los criterios de corrección.
6. La distinción entre error y mal funcionamiento.
7. ¿Qué conocimiento nos brinda la corrección formal sobre programas?

Módulo 2: Corrección formal.

Temario:

1. El problema de los sistemas críticos.
2. Pruebas formales de corrección de programas.
3. Asistentes de prueba.

Módulo 3: El problema de la corrección en la didáctica de la programación.

Temario:

1. Aprender a programar, diseño e implementación de algoritmos. Ejecución de programas.
2. Criterios de corrección.
3. Dificultades cognitivas vinculadas al problema de la corrección de algoritmos y de programas. ¿Cuáles son? ¿Cómo determinarlas?

El curso será ofrecido tanto para estudiantes de posgrado en filosofía y computación, como para profesores de Enseñanza Media de matemática, de informática y de filosofía , y para estudiantes del profesorado de informática y matemática. Cabe destacar que hemos dictado un curso de Filosofía de la ciencia de la computación en dos ediciones, un curso de matemática y programación para profesores de matemática de todo el país desde el año 1999 y un curso sobre epistemología y didáctica desde 2012. (Ver https://www.fing.edu.uy/grupos/nifcc/). Asimismo destacamos que se integran al proyecto docentes con experiencia en investigación y enseñanza de métodos formales [ítem E].

C.1) Objetivos generales en Extensión

El *Documento del Pro Rectorado de Extensión y Relaciones con el Medio, abril 2015* expresa una especial preocupación por *" ... difundir la cultura, contribuir al estudio de los problemas de interés general y propender a su compresión pública ...",* así como *"... a tarea extensionista y de actividades en el medio con acento en la generación de experiencias interdisciplinarias ..." http://www.extension.edu.uy/node/6730*

El abordaje interdisciplinario de nuestro Núcleo ha tenido desde sus inicios el objetivo de contribuir en forma significativa en la construcción social y cultural de la perspectiva de la informática como disciplina científica, con sus métodos y problemas, diferenciada del uso de productos y servicios tecnológicos basados en la informática. (Ver https://www.fing.edu.uy/grupos/nifcc/material/2012/formulariounico.pdf)

Consideramos de interés general que los ciudadanos accedan a una cultura que les permita distinguir la ciencia de la computación de los servicios y herramientas

tecnológicas derivados de aplicaciones de dicha ciencia. Por lo tanto es objetivo de nuestro Núcleo, contribuir a la comprensión pública de la relación entre tecnología e informática, de modo de fomentar el estudio de elementos básicos y fundamentales de la disciplina.

Consideramos que el sistema educativo es uno de los principales ámbitos para superar prejuicios e ideas preconcebidas sobre informática y tecnología, por medio de introducir el enfoque interdisciplinario de la disciplina y sus problemas didácticos. Por lo tanto, es objetivo de nuestro Núcleo, mantener y afianzar las relaciones que mantenemos desde hace casi 20 años con docentes de la Enseñanza Media de todo el país, tanto de matemática como de informática. Como resultado de ese proceso contamos hoy con cursos, proyectos de investigación y realización de eventos en conjunto con la Inspección de Matemática de Enseñanza Secundaria, el Instituto Normal de Enseñanza Técnica (INET) (donde se dicta el profesorado de Informática), el PEDECIBA y nuestras facultades, alcanzando a decenas de estudiantes y profesores en todo el país.

C.2) Objetivos específicos en Extensión

Organizar y realizar un taller dirigido especialmente a profesores de Enseñanza Media de matemática y de informática, basado en el siguiente temario:

- Beneficios del estudio de la informática en la formación general del ciudadano
- Problemas éticos y legales, vinculados a la corrección de programas
- El problema de la informática en la educación formal
- Cómo afecta la informática la educación en otras disciplinas
- Distinción entre informática como ciencia básica y sus aplicaciones
- Límites teóricos de la informática, interacción hombre-máquina
- Uso de algoritmos en situaciones cotidianas

Organizar y realizar dos eventos con participación de invitados de la región, similares a los realizados anteriormente.
(Ver Difusión en https://www.fing.edu.uy/grupos/nifcc/material.html)

Justificación y carácter interdisciplinario

La conformación del Núcleo Interdisciplinario Filosofía de la Ciencia de la Computación (NI FCC) (http://www.fing.edu.uy/grupos/nifcc/), en base a una propuesta presentada al llamado del año 2011 del Espacio Interdisciplinario de la UDELAR, nos permitió comenzar a abordar algunos de los problemas descritos en este documento. Desde entonces el grupo se ha esforzado por mantener el área tanto internamente como en sus vínculos regionales [ítem D][1], y también, por refinar el carácter interdisciplinario de la propuesta.

Actualmente hemos elaborado una unidad teórica, esencialmente interdisciplinaria, en relación al *problema del conocimiento sobre programas.* Determinar con precisión qué significa que *el agente S sabe que el programa P es correcto*, y determinar las consecuencias de este problema, tanto para la disciplina como para su didáctica trasciende los límites de cada una de las disciplinas por sí mismas. Los tres tipos de

1 Algunas referencias bibliográficas están agrupadas por ítems temáticos.

problemas involucrados en la aclaración de la pregunta anterior son:

i) Discusiones metateóricas en computación, acerca del problema del conocimiento sobre programas

El problema de la corrección de los programas tiene una larga tradición en ciencia de la computación. Desde los trabajos de Turing, Von Neumann, Hoare, hasta nuestros días, el área se ha desarrollado sostenidamente en varias direcciones. Por otra parte, el desarrollo de software tuvo un enorme crecimiento desde aquellos años. Los programas se aplicaron en infinidad de problemas, desde problemas simples hasta problemas de gran escala, lo cual contribuyó a que la complejidad de los programas aumente exponencialmente con respecto a los inicios de la computación. Esta situación hizo que sea casi imposible prenteder una prueba de corrección de todos los programas que se desarrollan, abriendo un debate sobre el problema de la corrección en la industria del software. Las pruebas de corrección sobre programas se han reservado para sistemas críticos; por ejemplo, para un programa que controla un marcapasos, algo que ha sido desarrollado en el país. Desde hace unos 30 años se han producido una serie de discusiones que dieron lugar a una reflexión metateórica acerca del estatus ontológico y espistemológico de los programas, en las que se integran algunos de los problemas mencionados anteriormente. Aclarar conceptualmente el significado de una prueba de corrección formal, en qué casos se requiere una prueba de corrección formal, en qué sentido podemos diferenciar un error de programación y un mal funcionamiento, etc., constituyen problemas metacientíficos que son relevantes para la disciplina y su didáctica.

ii) El problema de la corrección de programas en un sentido técnico

Seguramente uno de los problemas centrales en la discusión filosófica sobre los programas, sea la visión dualista acerca de los mismos, en la cual se reconoce una dimensión lógica y una dimensión física del programa, lo cual impacta directamente sobre el modo en el que entendemos el problema de la corrección. La distinción entre un proceso lógico que está integrado en el diseño del algoritmo y un proceso físico vinculado a la ejecución efectiva en un dispositivo físico, coloca a la computación en una situación epistemológica a medio camino entre la matemática y la ingeniería. ¿Cómo es posible conocer a priori un programa, es decir, mediante métodos formales, cuando el proceso físico de ejecución no puede ser conocido a priori? Este tipo de problemas requieren de un abordaje en el que se integre la discusión técnica sobre métodos formales, con la discusión filosófica acerca del dualismo sobre programas y la fundamentación del conocimiento. La incapacidad para dar cuenta de estas discusiones tiene consecuencias prácticas, por ejemplo, en una mala comprensión acerca del problema de la corrección formal, incluso dentro de la propia disciplina.

Iii) La dimensión de las dificultades cogntivas y didácticas

Estos problemas se traducen en la didáctica, como dificultades cognitivas de un cierto tipo que aún se están por entender. Las investigaciones en didáctica, se han concentrado, sobre todo, en los problemas que surgen al razonar con algoritmos, pero aún queda mucho trabajo por hacer en relación a las dificultades involucradas en la comprensión del vínculo entre el algoritmo y su implementación y ejecución física (Blanco, J et al. 2011).

Actividades previstas y cronograma

2017	Marzo a Julio	participan	Agosto a Diciembre	participan
	Reunión semanal de discusión de material teórico y tareas prácticas	Todos	Reunión semanal de discusión de material teórico y tareas prácticas	Todos
	Planificación curso de módulos del curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna	Implementación y organización de módulos del curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna
	Preparación de estudio didáctico	S.daRosa, F.Gómez, P.Añón	Realización de estudio didáctico	S.daRosa, F.Gómez, P.Añón
2018	Febrero a Julio	participan	Agosto a Diciembre	participan
	Mantenimiento de página y elaboración de materiales	S.daRosa, F.Gómez, L.Sierra, A.Chmiel	Mantenimiento de página y elaboración de materiales	S.daRosa, F.Gómez, L.Sierra, A.Chmiel
	Dictado curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna	Evaluación curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna
	Análisis de resultados del estudio didáctico y escritura artículo	S.daRosa, F.Gómez, A.Chmiel	Escritura artículo de didáctica, submisión a congreso.	S.daRosa, F.Gómez, A.Chmiel
	Escritura artículo sobre filosofía de la computación	A.Chmiel, I.Cervieri, G.Nigro, G.Calderón,	Escritura artículo sobre filosofía, submisión a congreso.	A.Chmiel, I.Cervieri, G.Nigro, G.Calderón,
	Preparación de taller de extensión	S.daRosa, A.Chmiel, F.Gómez, C.Luna, P.Añón	Realización taller y evaluación del mismo	S.daRosa, A.Chmiel, F.Gómez, C.Luna,
	Preparación evento	S.daRosa, A.Chmiel, F.Gómez, P.Añón	Realización de evento con profesor invitado	S.daRosa, A.Chmiel, F.Gómez, J.Blanco
	Mantenimiento de página y elaboración de materiales	S.daRosa, F.Gómez, L.Sierra, A.Chmiel	Mantenimiento de página y elaboración de materiales	S.daRosa, F.Gómez, L.Sierra,A.Chmiel

2019	Marzo a Junio	participan	Julio a Octubre	participan
	Dictado curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna	Evaluación curso	S.daRosa, A.Chmiel, I.Cervieri, G.Nigro, G.Calderón, C.Luna
	Evaluación de actividades y preparación de informes	S.daRosa, A. Chmiel, I.Cervieri, G.Nigro, G.Calderón	Realización de evento con profesor invitado	S.daRosa, A.Chmiel, F.Gómez, J.Blanco

Integrantes nacionales e internacionales

Responsables del proyecto: Alejandro Chmiel (FHCE) Sylvia da Rosa (FING).

Integrantes del proyecto con compensada: Carlos Luna (FING), Guillermo Calderón (FING).

Integrantes del proyecto para quienes se creará un cargo: Guillermo Nigro (FHCE), Ignacio Cervieri (FHCE).

Asesores nacionales: Luis Sierra (FING), Federico Gómez (FING), Patricia Añón (ANEP).

Asesor extranjero: Javier Blanco (Universidad Nacional de Córdoba - Argentina).

Resultados esperados y estrategias de difusión

Se espera escribir tres artículos y submitirlos a congresos internacionales:

1. Un artículo en filosofía de la computación, en relación a la distinción entre error y mal funcionamiento.
2. Un artículo en filosofía de la computación con un perfil técnico, en relación al problema de los asistentes de prueba y las pruebas de corrección sobre sistemas críticos.
3. Un artículo en didáctica de la informática, sobre los problemas cognitivos que surgen en relación a la distinción programa texto-máquina y sus incidencias en la educación.

Se espera preparar el curso en 2017 y dictarlo en 2018 y 2019: *Introducción al problema del conocimiento de los programas: un abordaje filosófico, técnico y educativo*.

Se espera llevar a cabo las actividades de extensión, a saber, un taller y dos eventos especificados arriba. Se hará uso de las salas de video conferencia de la Facultad de Ingeniería con el objetivo de llegar a la mayor cantidad de profesores e interesados en general de todo el país. A través de la convocatoria a periodistas interesados en estos temas, se espera tener artículos de divulagación en prensa.
(Ver Difusión en https://www.fing.edu.uy/grupos/nifcc/material.html)

Anexo: Referencias bibliográficas

Eden, Amnon, 2007, "Three Paradigms in Computer Science", Minds and Machines 17(2): 135–167.

Moor, J. H., 1978, "Three Myths of Computer Science," *The British Journal for the Philosophy of Science*, 29(3): 213–222

Colburn, T. R., 2000, *Philosophy and Computer Science*, Armonk, N.Y.: M.E. Sharp.

Copeland, B. J., 1996, "What is Computation?" *Synthese*, 108(3): 335–359.

Rapaport, W. J., 1999, "Implementation Is Semantic Interpretation," *The Monist*, 82(1): 109–30. [Rapaport 1999 available online (pdf)]

——, 2005, "Implementation as Semantic Interpretation: Further Thoughts," *Journal of Experimental & Theoretical Artificial Intelligence*, 17(4): 385–417. [Rapaport 2005 available online (pdf)]

Piccinini, G., 2008, "Computation without Representation," *Philosophical Studies*, 137(2): 206–241. [Piccinini 2008 available online (this version lists 2006 as publication date but that seems to be the online publication date)]

Summerville, I., 2012, *Software Engineering*, Reading, MA: Addison-Wesley. (First Edition, 1982.)

Kroes, P, 2010, "Engineering and the Dual Nature of Technical Artefacts," *Cambridge Journal of Economics*, 34(1): 51–62.

Kroes, P, 2012, *Technical Artefacts: Creations of Mind and Matter: A Philosophy of Engineering Design*, Dordrecht: Springer.

Blanco, J. (2008), A categorial mistake in the formal verification debate (con P. García), E-CAP, Montpellier, Francia.

Blanco, J., García, P. y Cherini, R. (2011) *Convergencias y divergencias en la noción de computación,* revista CTS nr. 19, vol 7.

Rushkoff, 2010: *"program or be programmed"* [available online: http://www.rushkoff.com/wp-content/uploads/2015/12/Rushkoff-Study-Guide.pdf*)]*

Referencias bibliográficas agrupadas por ítems temáticos.

A. Algunos artículos relativos al desarrollo de la didáctica de la informática en los últimos años (Computer Science Education en el mundo anglosajón)

1. Schwill, A. () *Computer Science Education Based on Fundamental Ideas.* ddi.cs.unipotsdam.de/didaktik/forschung/israel97.pdf.

2. Dowek, G. Et al. (2013) *L'enseignement de l'informatique en France, Il est urgent de ne plus attendre.* www.academiesciences.fr/activite/rapport/rads 0513.pdf.

3. Page, R. and Gamboa, R. (2013) *How Computers Work: Computational Thinking for Everyone.* Programming in Education 2012 (TFPIE 2012). EPTCS 106, pages 1-19.

4. Zendler, A. et al. (2013) *Semantic categorization of content and process concepts relevant to computer science education*. International Journal of Research Studies in Computing, 2:3-10.

5. Peyton Jones, S. et al. (2013) *Bringing Computer Science Back into Schools: Lessons from the UK.* SIGCSE'13.

6. Bradshaw, P. and Woollard, J. (2013) *Computing at School: An Emergent Community of Practice for a Re-Emergent Subject*. In: International Conference on ICT in Education (LNCS 7780).

7. Zendler, A., McClung, O. and Klaudt, D. (2012) *Content and process concepts relevant to computer science education: A cross-cultural study.* International Journal of Research Studies in Computing, 1:27-47.

8. Saeli, M. (2012). *Teaching Programming for Secondary School: a Pedagogical Content Knowledge Based Approach*. Dissertation, Technische Universiteit Eindhoven. [available online http://www.pgce.soton.ac.uk/IT/Curriculum/Resources/Programming/Papers/Mara_Saeli_Manuscript.pdf]

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

10. Zendler, A. and Spannagel, C. (2008) *Empirical Foundation of Central Concepts for Computer Science Education.* ACM Journal on Educational Resources in Computing, 8.

11. Wing, J. (2006) *Computational Thinking*, March 2006/Vol. 49, No. 3 COMMUNICATIONS OF THE ACM.

12. C. Holmboe, L. McIver, and C. E.George. (2001) *Research Agenda for Computer Science Education*. In G.Kadoda (Ed). Proc. PPIG 13, pp 207-223.

13. Dowek, G. (2012) *La place de l'informatique dans la classification des*

sciences Exposé au séminaire Philosophie de l'informatique, de la logique et de leurs interfaces, coordinné par Jean-Baptiste Joinet, le 30 janvier 2012, l'École normale supérieure.

B. Materiales sobre epistemología genética

1. García, R. (1997) La Epistemología Genética y la Ciencia Contemporánea. Barcelona, Gedisa, 325 pp. ISBN 84-7432-645-1.

2. García, R. (2000). El Conocimiento en Construcción. Barcelona: Gedisa, 252 pp. ISBN 9 788474-328110.

3. Inhelder B. and Cellérier, G. (1992). Le Cheminement des découvertes de l'enfant. Delachaux & Niestlé, 1992.

4. Piaget J., Greco P., Inhelder B., Matalon B. (1963). La Formation des Raisonnements Recurrentiels. Presses Universitaires de France.

5. Piaget, J. (1974) La Prise de Conscience. Presses Universitaires de France.
6. Piaget, J., Henriques, G. (1975) L'equilibration des Structures Cognitives, Probleme Central du Developpement. Presses Universitaires de France.

7. Piaget, J. (1978) Recherches sur la Generalisation. Presses Universitaires de France.

8. Piaget, J. (1978) Success and Understanding. Harvard University Press.

9. J. Piaget, J. and García, R. (1989). Psychogenesis and the History of Sciences. Columbia University Press, New York.

10. Piaget, J. and Beth, E. (1966). Epistemología Matemática y Psicología D.Reidel Publishing Company, Dordrecht-Holland.

11. Borel, M. J., (1987) Piaget's Natural Logic, in ''Piaget today'' Lawrence Erlbaum Associates Publishers, 65-75.
12. Cellérier, G. (1987). Structures and Functions, in ''Piaget today'' Lawrence Erlbaum Associates Publishers, 15-36.

13. Sadovsky, P. (2000). *La Teoría de Situaciones Didácticas: un marco para pensar y actuar la enseñanza de la Matemática*. [available online: http://www.buenosaires.gob.ar/areas/educacion/cepa/teoria situaciones.pdf]

C. Algunas publicaciones de la responsable del proyecto y otros participantes.

1. da Rosa, S. (2016) *Preconceptions of novice learners about program execution*, Proceedings of the 27th Psychology of Programming Interest Group Workshop, Cambridge, UK.
2. da Rosa, S. (2015) *The construction of knowledge of basic algorithms and data structures by novice learners*,Proceedings of the 26th

Psychology of Programming Interest Group Workshop, Bournemouth, UK.

3. da Rosa, S. (2010) *The Construction of the Concept of Binary Search* Algorithm. Actas del 22th Psychology of Programming Workshop, 100-111.

4. da Rosa, S. (2007) The *Learning of Recursive Algorithms from a Psychogenetic Perspective.* Actas del 19th Psychology of Programming Workshop, pages 201-215.

5. da Rosa, S. (2005) *The Learning of Recursive Algorithms and their Functional Formalization*, tesis doctoral, Technical Report 05-12, PEDECIBA Informática, Instituto de Computación Universidad de la República, Uruguay, 2005 [available online: www.fing.edu.uy/~darosa]

6. da Rosa, S. (2004) *Designing Algorithms in High School Mathematics,* Lecture Notes in Computer Science, vol. 3294, Springer-Verlag.

7. daRosa, S. (2003) Psychogenesis of the Concept of Recursion, Actas del Congreso Iberoamericano de Educación Superior en Computación.

10. da Rosa, S. Gómez, F. (2010) An educational methodology based on the student's work en Clei Electronical Journal Vol. 13, Nr. 2 [available online http://www.clei.cl/cleiej/volume.php]

11. da Rosa, S., Gómez, F. y Vigo, S. (2012) Un Software para Introducción de Algoritmia en Cursos de Matemática Actas de Latin American Conference on Learning Objects and Technologies (Laclo 2012), Ecuador.

D. Publicaciones del NI FCC

1. Chmiel, A., da Rosa, S. y Gómez, F. (2016) *Philosophy of Computer Science and its Effect on Education - Towards the Construction of an Interdisciplinary Group,* Clei Electronical Journal [available online: http://www.clei.org/cleiej/], Volume 19 : Number 1

2. da Rosa, S., Chmiel, A. (2012) *A Study about Students' Knowledge of Inductive Structures*, Proceedings of the 24th Psychology of Programming Interest Group Workshop, London, UK.

3. da Rosa, S., Chmiel, A., (2011) *Sobre la Construcción del Concepto de Inducción*, Actas del Congreso Iberoamericano de Educación Superior en Computación, Ciesc.

E. Algunos artículos sobre métodos formales de miembros del proyecto

1. Luna, C. *Taller de Especificación, Construcción y Verificación Formales de Programas – Propuestas y experiencias.* [available online: http://sedici.unlp.edu.ar/bitstream/handle/10915/22418/117_.pdf;jsessionid=7

2. Luna, C. (2006) *Enseñando métodos formales en Coq.* [available online: http://teyet-revista.info.unlp.edu.ar/wp-content/uploads/2016/06/07_Ensenando_metodos_formales_con_Coq-1.pdf]

3. Calderón, G. et al. (1996) *Type Theory and Functional Programming: A Work Proposal*. In Proceedings of 1st Latin American Workshop of Functional Programming; 25th Argentine Meeting of Informatic and Operational Research. Buenos Aires, Argentine.

4. Calderón, G. (1996) *Implementing Martin Lof's Theory of Types in a Higher-Order Logic Programming Language.* Extended Abstract in the Conference report of Volume 4, Number 3, of the Journal of the Interest Group in Pure and Applied Logics (IGPL).

5. Calderón, G. (1993) *A Resolution Rule for Martin Löf's Logical Framework*. Master Thesis. Technical Report 93-03 of the InCo, Uruguay.