Alberto Yáñez-Castillo
ORCID 0000-0001-7322-1048

Ana L. Laureano-Cruces
ORCID 0000-0002-5451-0175

Gustavo de la Cruz-Martínez
ORCID 0000-0002-4446-7396

Visualización de conceptos abstractos en programación estructurada

Capítulo ***14*** **pp. 147-156**

De los métodos y las maneras
Número 6

Coordinador de la obra
José Iván Gustavo Garmendia Ramírez
Compilación y Diseño editorial
Sandra Rodríguez Mondragón
Diseño de portada
Martín Lucas Flores Carapia

México
Universidad Autónoma Metropolitana
Unidad Azcapotzalco
Coordinación de Posgrado de
Ciencias y Artes para el Diseño

Primera edición impresa: **2021**

Primera edición electrónica en pdf: **2021** **http://hdl.handle.net/11191/7926**

ISBN de la colección en versión impresa: **978-607-28-1322-9**
ISBN de la colección en versión electrónica: **978-607-28-1326-7**
ISBN No. 6 versión impresa: **978-607-28-2227-6**
ISBN No. 6 versión electrónica: **978-607-28-2229-0**

Universidad Autónoma Metropolitana
Casa abierta al tiempo Azcapotzalco

CyAD
Ciencias y Artes para el Diseño

Cordinación de Posgrado CyAD
http://cyadposgrados.azc.uam.mx/

De los métodos y las maneras

Visualización de conceptos abstractos en programación estructurada

Alberto Yáñez-Castillo & Ana L. Laureano-Cruces

Universidad Autónoma Metropolitana, unidad Azcapotzalco, CBI

Gustavo de la Cruz-Martínez

UNAM, Instituto de Ciencias Aplicadas y Tecnologías

Resumen

En este trabajo se tocan una serie de características que permiten comprender los problemas a los que se enfrentan los estudiantes de programación estructurada. Y se propone una novedosa metodología que permite la visualización de los conceptos abstractos que la constituyen. En este caso se utiliza la teoría de Forbus y su razonamiento cualitativo, a través del cual se pretende llevar el concepto abstracto a un plano físico. Con la llegada de la tecnología y las computadoras a principios de los años noventa, se inició la investigación del aprendizaje interactivo a través de la visualización. Lo que se presenta en este artículo es la investigación que tiene como objetivo general, desarrollar mini – teorías articuladas sobre el dominio de los conceptos abstractos de programación mediante el análisis de los modelos mentales del experto en programación y el razonamiento cualitativo, utilizando eventos de la vida cotidiana. Implementado con la ayuda de recursos didácticos como: el robot Sphero, un juguete tecnológico que puede ser programado (es decir, controlado) por medio de una computadora o dispositivo móvil utilizando la aplicación de programación visual Sphero Edu. Con la finalidad de lograr una experiencia interactiva, permitiendo una conexión entre lo real y abstracto de la programación de manera visual y tangible. Los conceptos abstractos de programación son fundamentales en los cursos de programación; siendo el pivote para diversas ingenierías y carreras relacionadas. Por lo anterior es importante que los alumnos comprendan adecuadamente los conceptos abstractos. A continuación, se definen a detalle los conceptos abstractos de programación, las posibles razones o factores que pueden intervenir en la comprensión de conceptos abstractos, la propuesta de entorno de programación visual (desarrollada por el MIT Media Laboratory y Lifelong Kindergarten Group) y una breve explicación de la Teoría cualitativa del proceso.

Palabras clave:

Programación estructurada
Conceptos abstractos
Mini-lenguajes
Razonamiento cualtiativo

No. 6, ISBN 978-607-28-2229-0

Introducción

Dentro del área de las Ciencias e Ingeniería en Computación existe la programación estructurada, la cual regularmente es un curso pivote para las ingenierías y carreras relacionadas con la computación. El término de **programación consiste en comprender un problema para resolverlo por medio de una computadora**, de forma paralela se debe comprender cómo la computadora podrá llegar a la solución (Levine, 2001, p. 306). Para otros, es el arte de construir programas por medio de un conjunto de instrucciones entendibles que son la solución a un problema y ejecutadas por la computadora (Trejos Buriticá, 2012). Dicho de otra manera, l**a programación es un proceso mental complejo, no se trata de aprender un procedimiento o memorizar conocimiento, sino de resolver un determinado problema por medio de una computadora y una serie de instrucciones entendibles**.

Para aprender programación estructurada es requerido cuando menos el **dominio de conceptos abstractos** relacionados con la misma programación y de ciertas capacidades básicas como son: el pensamiento lógico, el lógico matemático y el razonamiento. Así mismo, el estudiante debe contar con una buena cantidad de tiempo y esfuerzo mental (Levine, 2001, p. 308).

Los conceptos abstractos básicos que debe comprender el estudiante de programación estructurada son: **variable, secuenciación, selección simple, iteración condicional, e iteración fija**. Algunas explicaciones o definiciones dadas en curso para dichos conceptos son:

- **Variable:** se debe de pensar en esta como una posición de memoria, donde su contenido puede variar mediante instrucciones de asignación (Sánchez, 2009). Es también un nombre simbólico de la dirección inicial de un grupo de celdas de memoria que contiene un valor (Levine, 2001).
- **Secuenciación:** es "ejecutar las acciones indicadas, una después de otra." (Levine, 2001, p. 313), es decir, seguir de forma disciplinada una secuencia sucesiva de pasos ordenados para lograr describir, modelar o resolver una situación.
- **Selección simple:** se evalúa una condición la cual puede tener el valor de verdadero o falso, una vez evaluada la condición y conociendo el valor obtenido, se ejecuta la acción 1 sí el valor es verdadero o bien la acción 2 para el valor de falso. Aclaremos que la acción 1 o acción 2 están representadas por un conjunto de acciones.
- **Iteración condicional:** se evalúa una condición la cual puede tener el valor de verdadero o falso, caso parecido al de la selección simple, pero esta vez sí el valor resulta ser verdadero, se ejecuta una o varias acciones repetidamente mientras el valor de la condición siga siendo verdadero; sí el valor resulta falso la ejecución de la acción o acciones terminan o quizá el ciclo nunca se inicie porque desde el comienzo el valor resultó ser falso (Levine, 2001).
- **Iteración fija:** en este caso se ejecuta la (las) acción repetidamente durante el un cierto número determinado de veces, bien podría decirse que el valor del número es la condición por cumplir para detener la iteración fija.

Además, dentro de estos conceptos abstractos básicos se encuentran las estructuras fundamentales de control (secuenciación, selección simple, iteración condicional y fija) con ellas es posible escribir cualquier programa. Todas las estructuras de control operan sobre proposiciones, es decir una oración o enunciado con un valor informativo que puede resultar verdadero o falso (solo un resultado, no ambos a la vez), y cuando se tiene el resultado (verdadero o falso) de estas proposiciones se producen las acciones resultado de la proposición.

Razones y factores que pueden intervenir en la comprensión de conceptos abstractos

Es nuestro punto de vista, que la comprensión de conceptos abstractos es importante para el estudiante de programación en primer lugar por ser parte del curricular de su carrera, en segundo lugar, porque el mundo de hoy tiene necesidades tecnológicas que

pp. 147-156; recibido 2019-12-30; aceptado 2020-05-31

alguien debe cubrir, y en último lugar la programación contribuye a desarrollar un pensamiento computacional en los estudiantes, útil para pensar recursivamente, resolver problemas, pensar en paralelo (Sáez y Cózar, 2017)

¿por qué a algunos estudiantes de programación se les dificultad comprender los conceptos abstractos de programación y estructuras de control?

Para tratar de responder la pregunta anterior, se expondrá el juicio de diversos autores a la problemática que han observado en estudiantes de programación.

De acuerdo con Ruíz Velasco (1990), las matemáticas y el lenguaje de informática (programación) son de gran importancia por la aplicación que pueden tener en cualquier dominio o actividad, es decir, "ayudan a la gente a pensar en su vida, a organizar sus conocimientos y a desarrollarse social, emocional e intelectualmente" (Ruíz Velasco, 1990, p. 2); sin embargo, los estudiantes encuentran obstáculos para dominar ciertos conceptos abstractos, probablemente causados porque "no existe una relación entre la utilización y la manipulación de estos conceptos y las situaciones vividas cotidianamente" (Ruíz Velasco, 1990, p. 2).

Carlos Chesñevar (2000) asegura que muchos alumnos desde el nivel de secundaria egresan con una formación deficiente y esto les "dificulta organizar nuevos conceptos de una manera ordenada", aunado a ello programar en algún lenguaje de programación demanda un esfuerzo considerable, así como "varias competencias y habilidades, que involucran básicamente la capacidad de manipular un conjunto de abstracciones interrelacionadas entre sí" (Chesñevar, 2000).

Por otra parte, en Insuasti (2016) se exponen razones por las cuales el alumno no logra aprender programación, entre ellas: los conceptos de programación, la carga cognitiva implicada en el aprendizaje de programación y la falta de habilidades cognitivas propias para solucionar problemas (p. 236); dichas razones han sido presentadas en investigaciones publicadas a través de la ACM (*Association for Computing Machinery*). A continuación, se citan algunas de estas razones, tal como las identificaron sus autores.

En lo que se refiere a la tarea cognitiva de programación, Baldwin y Kuljis (2001) identificaron que aprender programación demanda **habilidades cognitivas** como: la planificación, razonamiento y resolución de problemas (citados en Insuasti, 2016). Y según Insuasti (2016) dicha demanda también se extiende al desarrollo de **habilidades de pensamiento** como: la capacidad de atracción, la facilidad de análisis y la destreza para la síntesis, tales habilidades son un complemento de la resolución de problemas; en general si estas habilidades no han sido desarrolladas podrían ser parte de la dificultad que tienen los alumnos con la tarea de programar.

Así mismo, existen factores como: a) la incapacidad del alumno para saber lo que ocurre con la ejecución de las instrucciones, cálculos y resultados que el programa realiza, b) la falta de motivación para programar, y c) la dificultad para comprender la lógica compuesta (Insuasti, 2016). En este trabajo dichos factores se consideran de menor importancia. Por otro lado, un factor esencial, es la falta de motivación para programar si es de suma importancia; debido a que hoy en día los estudiantes son parte de una generación de nativos digitales que pasa la mayor parte de su tiempo interactuando con tecnología y no tienen motivación para comprender la programación, entonces como se podrá tener un desarrollo de su pensamiento computacional con solo ser espectadores.

Insuasti (2016) complementa lo anterior con algo muy importante, la "dificultad que enfrentan los estudiantes de programación, es la incapacidad de imaginar y comprender términos abstractos que no tienen equivalente en la vida real", por ejemplo, ¿cómo podrían imaginar o relacionar una variable con un objeto de la vida real? o tal vez ¿se necesite de una analogía o metáfora para relacionar los conceptos abstractos de programación con algún objeto de la vida real?

En concreto, algunos factores o razones que pueden intervenir en la comprensión de conceptos abstractos de programación se ilustran y resumen en la Figura 1.

No. 6, ISBN 978-607-28-2229-0

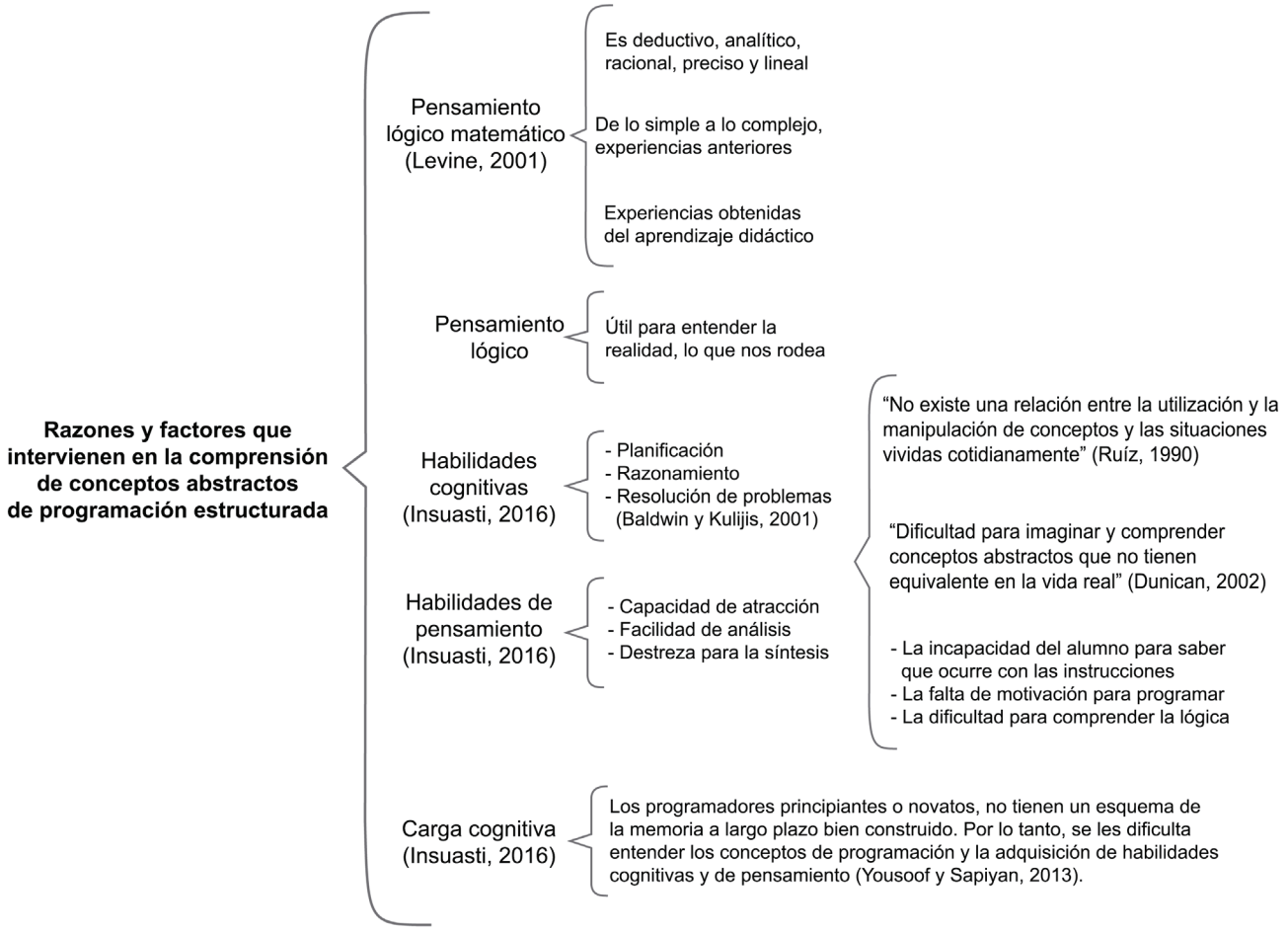

Figura 1. Razones y Factores que pueden intervenir en la comprensión de conceptos abstractos de programación (Fuente: elaboración propia)

Algunas propuestas para aprender a programar

Las propuestas más comunes y actuales para enseñar programación a los estudiantes se derivan en dos vertientes, en algunas propuestas éstas se fusionan en una sola. La *primera vertiente* es el uso de mini – lenguajes[1], y la *segunda* son objetos físicos, robots o artefactos diseñados a medida con su propio lenguaje de programación. En el caso de *una sola vertiente* es cuando un robot o artefacto utiliza como lenguaje de programación un mini – lenguaje previamente desarrollado por otro.

Por ejemplo, en el caso de mini – lenguajes se tiene a Scratch un entorno de programación visual, creado por MIT *Media Laboratory y Lifelong Kindergarten Group.* El entorno Scratch permite construir un programa utilizando los bloques de instrucciones como se observa en el lado izquierdo de la Figura 2, y anima al estudiante a visualizar su programa por medio de animaciones como se observa en el lado derecho de la Figura 2. Lo que permite no solo ver u obtener un resultado matemático.

Además, en Scratch es posible construir programas haciendo uso de conceptos fundamentales como: secuenciación, selección simple e iteración condicional.

1 El termino mini – lenguajes fue introducido por Brusilovsky, Calabrese, Hvorrecky, Kouchnireko, y Miller (1997) para referirse a lenguajes de programación con sintaxis simple. Los mini – lenguajes son visualmente intuitivos, simples y poderosos. Están construidos sobre metáforas y tienen la cualidad de involucrar y atraer visualmente. Su característica principal es que se basan en bloques y de esa forma se eliminan los problemas de sintaxis, permitiendo a los estudiantes concentrarse en otros aspectos.

pp. 147-156; recibido 2019-12-30; aceptado 2020-05-31

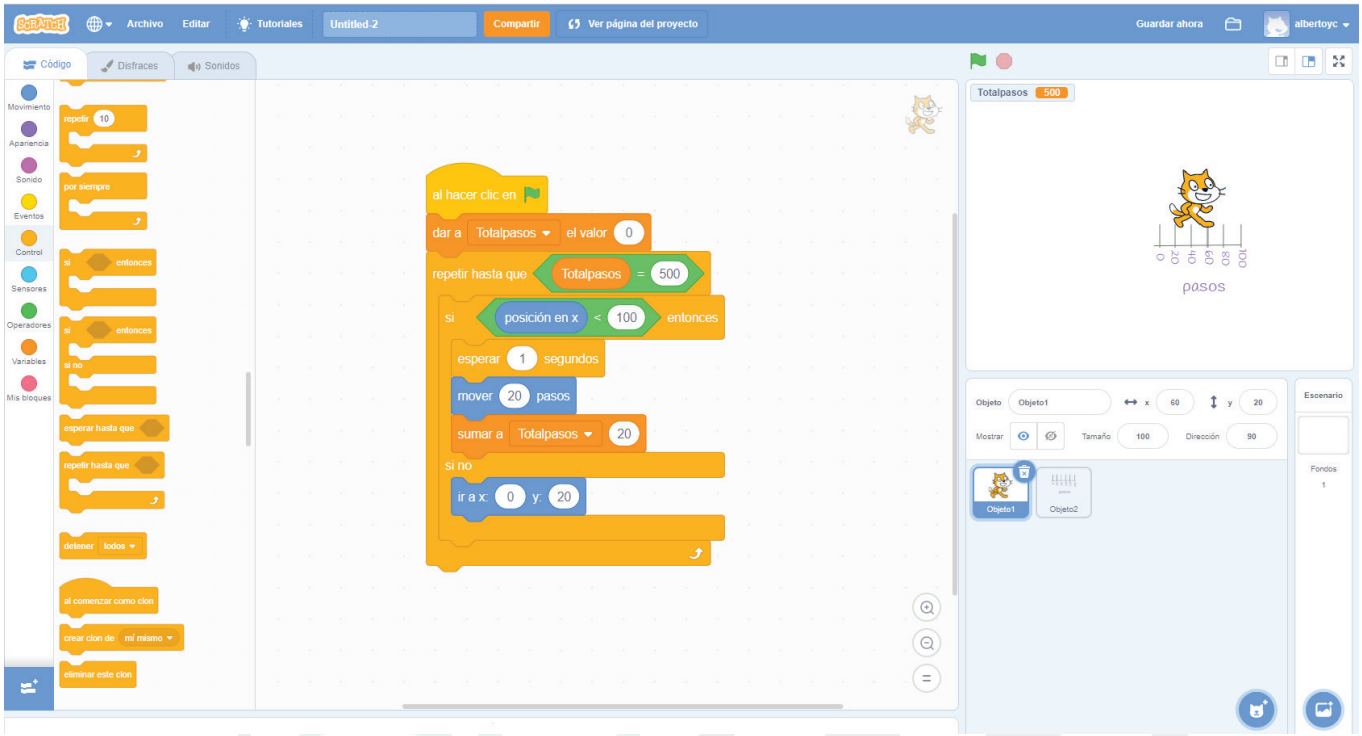

Figura 2. Captura de pantalla del entrono de programación visual Scratch ("Scratch - Imagine, Program, Share," s.f.)

En la Figura 3 en la parte superior se observan dos bloques correspondientes a la selección simple (si<>entonces) y en la parte inferior de la Figura 3 los bloques correspondientes a una iteración condicional (repetir hasta que <>) e iteración fija (repetir_nVeces).

Por otro lado, en los objetos físicos, robots o artefactos diseñados a medida con su propio lenguaje de programación como son: micro:bit, circuit playground express, LEGO Mindstroms Education EV3, Chibi chip, todos con distintas formas de poder programarse pero también teniendo en común la plataforma Microsoft MakeCode ("Microsoft MakeCode," s.f.) para iniciarse en la programación. Estas opciones se muestran en la Figura 4).

La lista de robots que permite ser programados por medio de un entorno de programación visual es interminable, muchos de ellos son de costos bastante altos y por lo tanto de difícil acceso para los estudiantes, pero uno que resulta bastante practico y económico es Sphero mini como se muestra en la Figura 5; un pequeño robot con forma de esfera, se conecta por medio de bluetooth a dispositivos móviles con sistema operativo Android, iOS, Kindle y a computadoras con bluetooth y sistema operativo Mac o Windows ("Sphero Mini," s.f.).

Para cada uno de los sistemas operativos está disponible la aplicación Sphero Edu app la cual permite construir programas e interactuar con el robot por medio de un entorno de programación visual, dicha aplicación cuenta con distintos bloques de programación, entre ellos las estructuras de control: selección simple e iteración, como se muestran en la Figura 6. Este es similar a Scratch, además cuenta con una variante de iteración repetir indefinidamente, es decir, no tiene condición para ejecutar o no la acción.

Razonamiento cualitativo

Dentro de la inteligencia artificial se encuentra (una subdisciplina) la física cualitativa, su objetivo es la descripción de los sistemas físicos, representa y explica su comportamiento basándose en el sentido común utilizado por el ser humano para analizar cualitativamente

No. 6, ISBN 978-607-28-2229-0

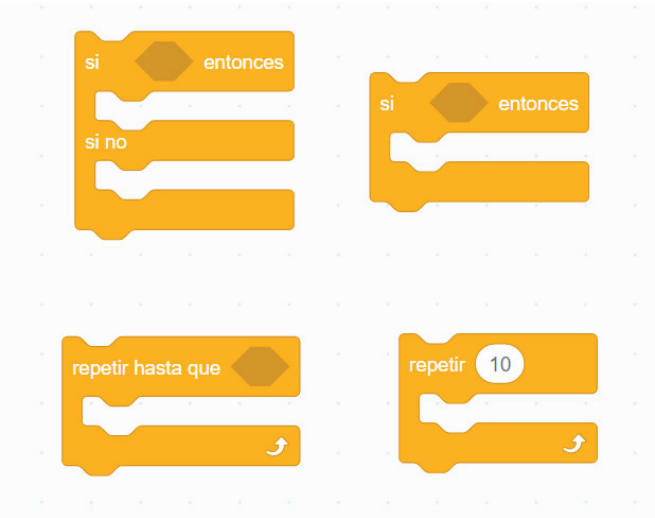

Figura 3. Estructuras de control en Scratch ("Scratch - Imagine, Program, Share," s.f.)

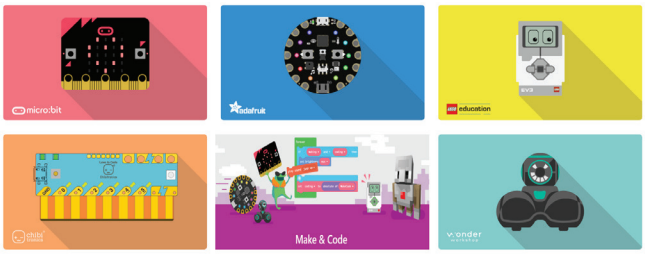

Figura 4. Variedad de artefactos que utilizan MakeCode ("Microsoft MakeCode," s.f.)

Figura 5. Sphero mini ("Sphero Mini," s.f.)

el entorno y en el conocimiento científico utilizado por los ingenieros (Bobrow, 1984). Este objetivo de predecir, describir y explicar el comportamiento de los sistemas físicos sin tener que recurrir a las matemáticas, es decir, hacerlo más simple y en términos cualitativos, forma parte del razonamiento cualitativo, dicho razonamiento es ampliamente utilizado por los científicos e ingenieros. (Klenk, Bobrow, De Kleer, Hanley y Janssen, 2012).

El razonamiento cualitativo se considera un dominio de la inteligencia artificial desde 1984 con la publicación de un número especial de la revista Artificial Inteligence. De acuerdo con Forbus, el razonamiento cualitativo crea representaciones para los aspectos continuos del mundo, como el espacio, el tiempo y la cantidad. Dicho razonamiento se sostiene sobre dos observaciones: 1) las personas obtienen conclusiones útiles e ingeniosas sobre el mundo físico sin el uso de ecuaciones diferenciales, es decir, en la vida diaria las personas pueden descubrir lo que ocurre o acontece a su alrededor y cómo cambiarlo (u afectar), trabajando con menos datos o poca información. 2) aparentemente, tanto científicos como ingenieros utilizan el razonamiento cualitativo para comprender en inicio un problema. (K. Forbus, 2004).

Cuando los objetos se mueven, chocan, calientan, enfrían, estiran, comprimen y otros cambios que pueden ocurrir en dichos objetos con el tiempo son caracterizados por procesos. En la física formal, dichos procesos se caracterizan por ecuaciones diferenciales que describen como cambian los parámetros de los objetos en el tiempo; pero lo que ocurre durante un proceso físico, comúnmente lo podemos averiguar o concluir teniendo a la mano una poca de información. Por ejemplo, si calentamos agua en un recipiente cerrado, el agua hierve después de cierto tiempo, y sabemos que, si se sigue calentando el recipiente el agua continuará hirviendo hasta que muy probablemente el recipiente explote. Entonces, para comprender el razonamiento de sentido común físico, es necesario entender cómo razonar cualitativamente sobre los procesos, por ejemplo, cuándo ocurrirán, cuándo se detendrán y sus efectos de tales procesos; para ello nos sirve la teoría cualitativa del proceso (K. D. Forbus, 1984).

pp. 147-156; recibido 2019-12-30; aceptado 2020-05-31

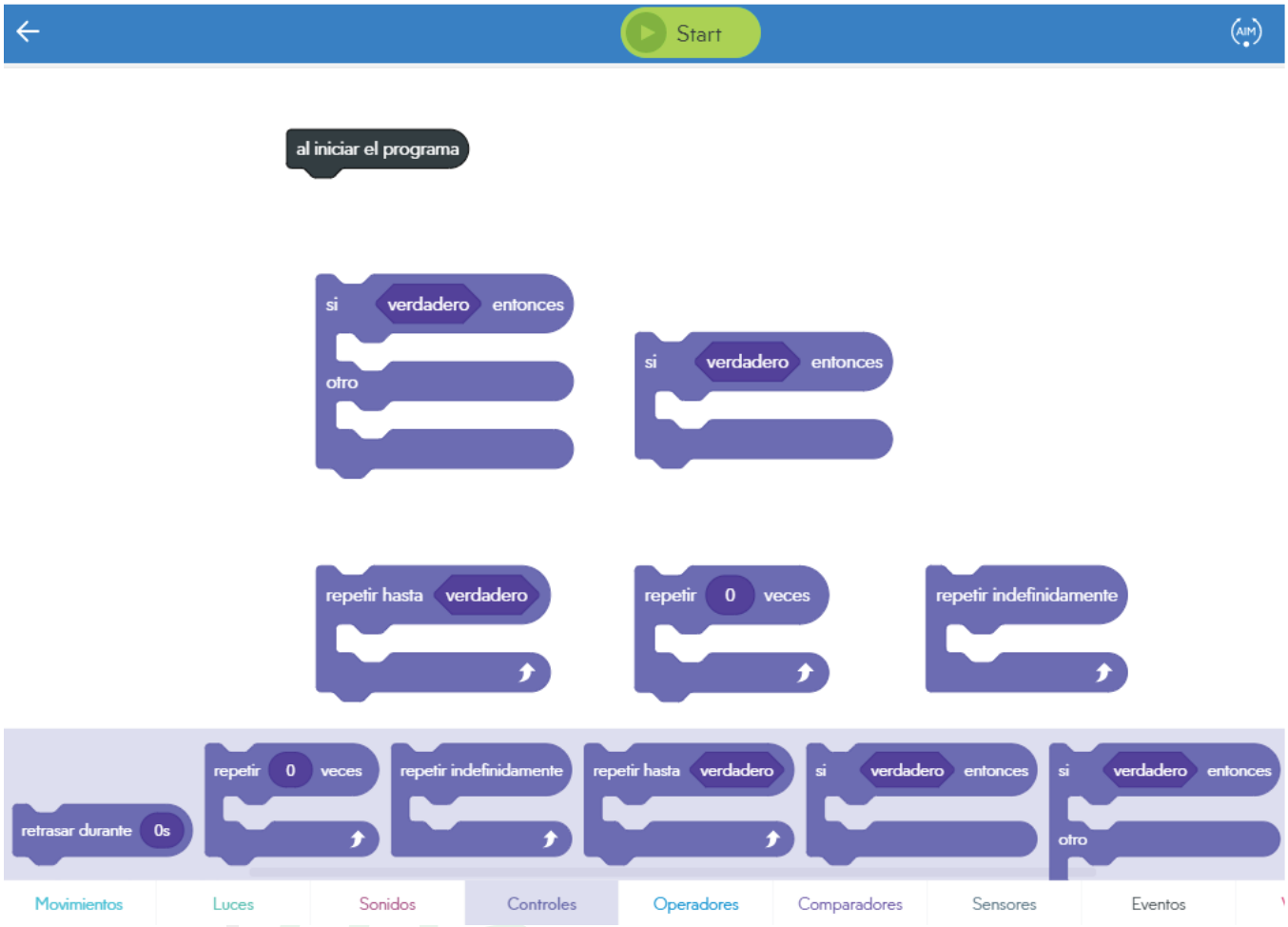

Figura 6. Estructuras de control en la aplicación Sphero Edu ("Sphero Mini," s.f.)

Tareas de razonamiento

Dentro de la teoría cualitativa del proceso se describen diferentes tareas (o estilos) de razonamiento que pueden ser apropiados para resolver diferentes clases de problemas. Forbus (1984) propone las siguientes tareas de razonamiento:

- Actividad determinante: deducir que está sucediendo en una situación, durante un momento particular. Proporciona respuestas directas a una clase de preguntas ("qué está sucediendo aquí"), siendo una operación básica en otras tareas de razonamiento.
- Predicción: deducir lo que sucederá en el futuro de alguna situación. Al trabajar con información incompleta, solo se pueden obtener descripciones de futuros posibles en lugar de un solo futuro.
- Postdicción: deducir cómo un particular estado de cosas podría haber surgido. Este tipo de razonamiento es más difícil que la predicción debido a la necesidad de postular individuos. Y de acuerdo con Laureano, Terán, y De Arriaga (Laureano y de Arriaga, 2000; Laureano, Terán, y De Arriaga, 2003) es más difícil de considerar este razonamiento por su dificultad para encadenar (hacía atrás) lo ocurrido y teniendo información incompleta.

No. 6, ISBN 978-607-28-2229-0

- Análisis escéptico: determinar si la descripción de una situación física es consistente.
- Interpretación de la medición: dada una descripción parcial de los individuos en una situación y algunas observaciones de su comportamiento, inferir si existen otros individuos y que más está sucediendo.
- Planeación experimental: dado un conocimiento de lo que puede ser observado y lo que se puede manipular, planificar acciones que produzcan más información sobre la situación.
- Razonamiento causal: considerar una descripción del comportamiento que genera cambios a partes particulares de la situación. La relación causa – efecto (causalidad), puede ser una herramienta para dar crédito a la hipótesis sobre una conducta observada.

Metodología

El procedimiento de investigación ha sido en el siguiente orden: 1) exploración de la literatura, trabajos de investigación y entrevistas con expertos en relación al tema. 2) se realiza el planteamiento del problema, hipótesis, justificación y objetivo general, propios del protocolo de investigación. 3) se recopila información y analiza. 4) posteriormente se lleva un analizas sobre las opciones que se tienen para cumplir los objetivos y la solución al problema. Las posteriores etapas del procedimiento de investigación que permitirán concluir son: 5) diseño y desarrollo del producto final, 6) ejecutar las pruebas necesarias, 7) realizar las correcciones o modificaciones pertinentes, y por último 8) conclusiones con base al análisis, evaluación y resultados.

La propuesta de este trabajo

La idea principal es transformar un concepto abstracto en un evento físico que permita ser percibido a través de los sentidos. El razonamiento cualitativo y su realización a través de las mini – teorías, ocupa un lugar muy destacado entre las ayudas posibles a facilitar a los alumnos para la descripción de procesos y solución de ciertos problemas en un dominio concreto (Laureano, Terán, De Arriaga, y El Alami, 2003).

Las mini – teorías serán diseñadas para los diferentes conceptos abstractos, creando escenas donde se descubran las propiedades necesarias para la comprensión de los conceptos involucrados.

Un ejemplo de una mini- teoría articulada para le concepto de **selección simple** es:

Comprensión del concepto de **selección simple**

> **Objetivos:** definir selección simple con base en la elección del robot Sphero por un camino u otro, decidido por una condición verdadera o falsa.
> **Marco:** en un micro – mundo, el robot *Sphero* toma un camino que le permite encontrar pistas sobre sus tareas, pero al llegar a cierto punto se encuentra en una encrucijada, la pregunta, ¿debe ir a la derecha o a la izquierda?, ¿qué condición se debe cumplir para que tome el camino de la derecha o la izquierda?
> **Actividad determinante:** tomar una decisión en un instante dado con base en el resultado de una expresión lógica.

Recordar que, con base en las tareas de razonamiento, la actividad determinante se debe deducir que esta sucediendo en una cierta situación, para obtener la respuesta.

Reflexiones finales

Hasta el momento se ha realizado un estudio desde la perspectiva pedagógica en la que se descubren las dificultades que se les presentan a los estudiantes de programación. Lo anterior ha movido a la comunidad científica para descubrir nuevas formas de enseñanza a través de diferentes tipos de visualización de estos conceptos abstractos.

Este trabajo es probablemente uno de los primeros en extrapolar la teoría cualitativa del proceso a los conceptos de las estructuras de control que son parte de lo que llamamos conceptos abstractos de programación.

La propuesta de este trabajo de investigación se basa en el proceso cognitivo que subyace a la programación de aquí, el énfasis en que los alumnos comprendan los conceptos abstractos y no en aprender a programar.

pp. 147-156; recibido 2019-12-30; aceptado 2020-05-31

Reconocimientos: este trabajo forma parte del proyecto de investigación que lleva a cabo J. Alberto Yáñez Castillo para obtener el grado de Doctor en Diseño y Visualización de la Información de la Universidad Autónoma Metropolitana-Azcapotzalco; asi como del proyecto divisional *Diseño de interfaces inteligentes para la simulación de conductas de organismos vivos o animados: sección de visualización de la información;* de la misma Universidad.

Referencias

Bobrow, D. G. (1984). Qualitative reasoning about physical systems: an introduction. Artificial Intelligence, 24(1–3), 1–5. https://doi.org/10.1016/b978-0-444-87670-6.50004-2

Brusilovsky, P., Calabrese, E., Hvorrecky, J., Kouchnireko, A., y Miller, P. (1997). Mini languages: Away to learn programming principles. Education and Information Technologies., 2(1), 65–83. https://doi.org/10.1023/A:1018636507883

Chesñevar, C. I. (2000). Utilización de los mapas conceptuales en la enseñanza de la programación (Vol. 3, p. 11). Vol. 3, p. 11. Recuperado de http://lidecc.cs.uns.edu.ar/~cic/2000/2000-jornadas-mapas/2000-jornadas-mapas.pdf

Forbus, K. (2004). Qualitative reasoning. In A. B. Tucker (Ed.), Computer Science Handbook, Second Edition (Segunda, p. 2752). https://doi.org/10.1201/b16812

Forbus, K. D. (1984). Qualitative Process Theory. Artificial Intelligence, 24, 85–168. https://doi.org/10.1016/B978-1-4832-1447-4.50016-X

Klenk, M., Bobrow, D. G., De Kleer, J., Hanley, J., y Janssen, B. (2012). Placing Qualitative Reasoning in the Design Process. In Proceedings of the 26th Annual Workshop on Qualitative Reasoning. Recuperado de http://matthewklenk.com/papers/QR12-Design.pdf

Laureano, A. L., y de Arriaga, F. (2000). Reactive Agent Design for Intelligent Tutoring Systems. Cybernetics and Systems, 31(1), 1–47.

Laureano, A. L., Terán, A., y De Arriaga, F. (2003). Un enfoque didáctico-cognitivo del análisis de los conceptos de los sistemas de un grado de libertad. Revista Digital Universitaria, 4(7), 1–29. Recuperdo de http://www.revista.unam.mx/vol.4/num7/art23/art23.htm

Laureano, A. L., Terán, A., De Arriaga, F., y El Alami, M. (2003). La importancia de las estrategias cognitivas en el diseño del curricula didáctico. XVI Congreso Nacional y II Congreso Internacional de Informática y Computación de La ANIEI., I, 35–41.

Levine, G. G. (2001). Computación y programación moderna. Perspectiva integral de la informática. (J. L. Vázquez, Ed.). México: Pearson Educación de México.

Microsoft MakeCode. (s.f.). Recuperado diciembre 6, 2019, from https://www.microsoft.com/en-us/makecode

Ruíz Velasco, E. (1990). La informática como medio de enseñanza y objeto de aprendizaje. Perfiles Educativos, 37–43.

Sáez, J. M., y Cózar, R. (2017). Pensamiento computacional y programación visual por bloques en el aula de Primaria. Educar, 53(1), 129–146. https://doi.org/10.5565/rev/educar.841

Sánchez, G. M. de L. (2009). Sistemas de Aprendizaje Inteligente con Objetos de Aprendizaje "ProgEst". (Universidad Autónoma Metropolitana). Recuperado de http://ce.azc.uam.mx/profesores/clc/02_publicaciones/tesis_dirigidas/tesis_final_lsgnov09.pdf

Scratch - Imagine, Program, Share. (s.f.). Recuperado diciembre 6, 2019, de https://scratch.mit.edu/

Sphero Mini. (s.f.). Recuperado diciembre 6, 2019, de https://www.sphero.com/sphero-mini

Del autor:

Licenciatura en Ingeniería en Computación, por la Universidad Autónoma Metropolitana – Unidad Azcapotzalco

Otros estudios:

Maestría en Diseño y Visualización de la Información, Posgrado de Ciencias y Artes para el Diseño en la Universidad Autónoma Metropolitana – Unidad Azcapotzalco. Título de la tesis: De lo efectivo a lo afectivo: Diseño de elementos interactivos visuales con la finalidad de favorecer la interacción entre adultos mayores y la computadora personal. 8 de diciembre 2016

Participación en congresos:

• Septiembre de 2014: Participación en la sesión simultanea IV con el tema: "La ergonomía cognitiva, diseño emocional y semiótica pueden converger en el diseño inclusivo de la interfaz gráfica de usuario en la computadora" en el IV Congreso Internacional de Avances de las Mujeres en las Ciencias, las Humanidades y todas las Disciplinas. Creatividad e Innovación.

• Julio de 2014: Ponencia del trabajo "Diseño de elementos interactivos visuales con base en ergonomía cognitiva, diseño emocional, semiótica e usabilidad, para reemplazar iconos de la metáfora de escritorio utilizada por adultos mayores." en las Terceras Jornadas de Investigación del Departamento de Sistemas en la Universidad Autónoma Metropolitana - Unidad Azcapotzalco.

• Abril de 2014: Ponencia "Diseño de elementos interactivos visuales para reemplazar iconos de la metáfora de escritorio utilizada por adultos mayores" en el Primer Coloquio De los métodos y las Maneras en la Universidad Autónoma Metropolitana – Unidad Azcapotzalco.

• Octubre de 2013: Ponencia "La metáfora de escritorio desde la semiótica y diseño emocional" en el XXVI Congreso Nacional y XII Congreso Internacional de Informática y Computación en la Universidad de Ixtlahuaca CUI.

Publicaciones:

• Noviembre de 2015: "La ergonomía cognitiva, diseño emocional y semiótica en el diseño inclusivo de la interfaz gráfica de usuario para computadoras" publicado en el Libro: Avances de las Mujeres en las Ciencias las Humanidades y todas las Disciplinas (Vol. 1). (2015). México: UAM. pp. 252-265

pp. 147-156; recibido 2019-12-30; aceptado 2020-05-31