Gastón Alejandro Giménez

El aprendizaje de la Programación Multimedial en carreras artístico-tecnológicas

La aplicación didáctica de la Multimedia en la enseñanza de la Introducción a la Programación Multimedial

Tesis de Grado. Universidad Maimónides

ÍNDICE

Introducción. 7

Plan de Tesis. 11

Preguntas de Investigación. 11

Objetivos. 11

Hipótesis. 12

Capítulo 1. Educación. 13

1.1. Estudiar de memoria. 13
Un problema generalizado.

1.2. Profesores errados. 15
Dificultades para promover el aprendizaje.

1.3. Profesores exitosos. 16
Aprendiendo de los mejores.

1.3.1. El buen docente. 20
Generar interés y delegar el mando.

1.3.2. El aprendizaje profundo. 23
Un compromiso entre todos.

1.4. La evaluación. **27**
Niveles de motivación y exigencias.

1.5. Método Constructivista. **30**
Construyendo el aprendizaje.

1.6. Educación a distancia. **38**
Clases presenciales vs. Entornos virtuales.

1.6.1. El diseño formativo. **40**
Herramientas pedagógicas no presenciales.

1.6.2. Aplicación del diseño formativo. **43**
Pautas para su desarrollo.

Capítulo 2. Programación. **45**

2.1. Algorítmica. Pensar la programación. **46**

2.1.1. Tipos de Algoritmos. **49**

2.1.2. Lenguajes de Programación. **51**

2.2. Modelos de enseñanza de la Programación. **52**

2.2.1. Enseñanza a través de lenguajes profesionales. **55**

2.2.2. Enseñanza a través de lenguajes académicos. **60**

2.2.3. Conclusiones. Mercado vs. Aprendizaje. **68**

Capítulo 3. Multimedia. **71**

3.1. Contextualización histórica. Cambios culturales del S.XXI. **71**

3.1.1. La multimedia como herramienta integradora. **73**

3.1.2. Nativos Digitales. El perfil del alumno multimedia. **74**

3.2. Situación actual. Análisis de Casos. **76**

3.2.1. Planes de Estudios de Programación Multimedial. **78**

3.2.2. Planes de Estudios. Similitudes y Diferencias. **80**

3.3. Vocación del alumno multimedia. **82**

3.3.1. Encuesta para alumnos de multimedia. **84**

Capítulo 4. Propuesta Educativa. **86**

4.1. Análisis. **86**

4.2. Diseño y concreción. **93**

4.2.1. MarioCode. Una herramienta educativa. **94**

4.2.2. Sobre su desarrollo técnico. **105**

4.3. Desarrollo propuesta. **106**

4.3.1. Ejercitación. **106**

4.3.2. Teorización. **110**

4.3.3. Comunicación. **118**

4.3.4. Evaluación. **120**

4.3.5. Código de mensajes de error. **124**

4.4. Prototipo / Test / Evaluación. **126**

4.4.1. Objetivos de las entrevistas **126**

4.4.2. Metodología. **127**

4.4.3. Guia de evaluación para docentes. **127**

4.5. Plan de Estudios Propuesto. **128**

4.5.1. Objetivos. **128**

4.5.2. Metodología. **129**

4.5.3. Programa. **131**

4.5.4. Evaluación. **132**

Conclusiones Finales. **133**

Bibliografía. **138**

Referencias Electrónicas. **141**

INTRODUCCIÓN

Dentro del ámbito de toda carrera universitaria con vocación artístico-tecnológica, las asignaturas relacionadas con la Programación de Computadoras –o los lenguajes de Programación- suelen verse como las ***ovejas negras*** de la carrera, generando una cierta opinión negativa, y hasta antipática, de los alumnos hacia la disciplina en general.

Por una cuestión de practicidad y dinamismo, a lo largo de esta investigación se tomará como ejemplo la situación dispuesta en la carrera de Licenciatura en Tecnología Multimedial de la Universidad Maimónides.

Si bien allí se considera muy alto el porcentaje de alumnos aprobados en el primer año -menos del 5% de los estudiantes deben recursar la materia Introducción a la Programación Multimedial- , la realidad demuestra que su conocimiento básico sobre la programación, y su dominio en la lógica y en el diseño de los algoritmos es realmente limitado.

Generalmente se aburren en clase y tienden a **resolver problemas de memoria**, a bajar soluciones poco originales de Internet, e incluso a reutilizar el código de otros alumnos más *avanzados*.

Esta realidad fácilmente induce que los alumnos no fueron capaces de generar su propio conocimiento sobre los contenidos de la asignatura, sino que adoptaron desacertados hábitos y se acostumbraron a resolver problemas específicos preestablecidos sin analizarlos, o peor aún, simplemente se habituaron a no resolverlos y copiarlos de otro lado.

Para ellos, el concepto de programación está definitivamente ligado a las matemáticas, el álgebra, y las ciencias exactas, y son pocos los alumnos que asocian esta asignatura con el arte, la creatividad o el diseño. Sin ir mas lejos, un alumno de primer año, con apenas pocos meses en la carrera, ya tildó a la programación como "***un trabajo demasiado sofisticado y frío***".

Haciendo un rápido análisis contextual sobre el perfil de estos estudiantes, una primera explicación podría devenir del prejuicio existente en la comunidad sobre la naturaleza misma de sus contenidos. sin importar que francamente son desconocidos o poco profundizados por la mayoría de las personas.

De igual manera, estos constantes prejuicios son habitualmente potenciados por rígidos y poco inspiradores planes de estudios cuantitativos. impuestos por las autoridades académicas de turno. Que, sumados a las insuficientes y complejas herramientas pedagógicas actuales, las presiones sociales y comerciales por obtener una educación con rápida salida laboral, y las poco exitosas metodologías educativas implementadas, son incapaces de revertir esta realidad adversa.

Por otro lado, esta problemática no sólo dificulta el progreso del aprendizaje en los niveles iniciales, sino que el conflicto es arrastrado en los siguientes años de la cursada. Se ha visto que en materias más avanzadas de programación, el porcentaje de alumnos aprobados no logra superar el 50% del total, de los cuales tienen un promedio de calificación entre 6 y 7 puntos sobre 10, y donde el estudiante simplemente desea "***aprobar la materia y sacársela de encima***", tal como lo indicó un alumno del tercer año.

Esto genera una serie de datos curiosos y alarmantes, especialmente si se tiene en consideración que, ante una encuesta realizada entre alumnos y ex-alumnos de la universidad, casi el 70% de los encuestados admitió participar -o contaron sus ganas de participar- en algún proyecto relacionado con la programación.

Lejos de parecer un dato alentador, la realidad demuestra que con las buenas intensiones no alcanza, ya que la mayoría de los estudiantes son incapaces de trasladar sencillos problemas de la vida cotidiana a la lógica de los lenguajes algorítmicos, y lejos están de poseer las habilidades cognitivas básicas para comprender las soluciones de los problemas planteados.

En ese sentido, una de las fallas más recurrentes que se pueden observar en casi todos los cursos, corresponde a un grave problema de comprensión del enunciado. Alrededor del 90% de los alumnos observados demostró realizar los ejercicios planteados en los exámenes sin haber comprendido realmente el problema dado, llegando a conclusiones disparatadas y resultados poco satisfactorios que apenas lograban cumplir -en los casos donde cumplían- con la premisa planteada originalmente por el docente.

Por si fuera poco, este gravísimo defecto en la comprensión de los textos se da dentro de un inadecuado contexto social donde, según cifras oficiales surgidas en base a las pruebas PISA (*Program for International Student Assessment*) del año 2013, el 53,5% de los estudiantes argentinos no comprenden lo que leen [18], ubicando al país muy por debajo del promedio esperado por la OCDE (Organización para la Cooperación y el Desarrollo Económico).

Esta realidad parece alertar a las autoridades sobre el nivel educativo de los alumnos secundarios, tanto general como individualmente, ya que los bajos rendimientos en los exámenes también surgen de los resultados obtenidos de manera conjunta en matemáticas, literatura y ciencias.

Sin dudas, esta frecuente carencia en la educación secundaria repercute directamente en el nivel con que contarán estos mismos alumnos años más tarde, cuando les toque afrontar desafíos más serios en los niveles terciarios y universitarios de sus respectivas profesiones, y no cuenten con las herramientas epistémicas necesarias para hacerles frente.

Siguiendo estos datos, y teniendo en cuenta que el 9,3% de los alumnos porteños no entienden lo que leen, en el marco de esta exploración académica, también se considera imperativo cambiar la manera de encarar las clases –tanto para los docentes como para los alumnos- generando flexibles y entretenidos espacios de debate, reflexión y experimentación sobre los principios de la Programación Multimedial.

Así se demostrará su rica y vasta integración en el campo profesional y experimental de todo desarrollo interactivo, y se desterrará la noción de la disciplina como una ciencia exacta, para trasladarla a un terreno artístico, donde ganará –acertadamente- una opinión general objetiva, y permitirá así una mayor aceptación en los cursos, generando **una nueva generación de estudiantes creativos, motivados y comprometidos con su aprendizaje.**

Haciendo uso de las teorías del conocimiento y la enseñanza, -regidos especialmente por los conceptos del constructivismo pedagógico-, y mediante el adecuado empleo de las tecnologías multimediales y sus potenciales capacidades de comunicación bidireccional, se pretende la generación de una rica y activa metodología educativa de programación.

De esta forma, la presente investigación culminará así con la generación de un Plan de Estudios propuesto para la materia de Introducción a la Programación Multimedial, que sustentado en una herramienta didáctica interactiva, fomente la comprensión y asimilación de los dispositivos cognitivos aplicables con las disciplinas de la Programación, y especialmente la algorítmica, y genere el espacio formativo necesario para que cada alumno desarrolle el **autoaprendizaje y la construcción del conocimiento**, dentro de un marco que le otorgue la confianza y educación necesaria para experimentar y navegar libremente entre los límites mismos de la asignatura.

PLAN DE TESIS

PREGUNTAS DE INVESTIGACIÓN.

En las carreras artístico-tecnológicas, la introducción a la programación multimedial se basa en el uso de complejos programas comerciales, tales como ActionScript o Java, generando grandes dificultades para comprender, analizar y generar un propio conocimiento por parte de los alumnos en los niveles iniciales.

Dados estos inconvenientes, ***¿de qué forma puede una herramienta pedagógica interactiva ayudar a construir en los estudiantes una conceptualización efectiva de los conocimientos de la programación y la algorítmica, a la vez que estimule su interés, curiosidad y motivación?***

OBJETIVOS.

Generar una metodología alternativa de aprendizaje promovida por una aplicación didáctica que presente las ventajas de la multimedia como herramienta pedagógica, y permita la autogeneración de los conocimientos en cada alumno, permitiéndole comprender, asimilar y profundizar en los conceptos básicos de la programación estructurada y el diseño de algoritmos simples.

HIPÓTESIS.

1. El uso de lenguajes de pseudocódigo en los niveles iniciales permitirá absorber una mayor capacidad en la resolución de problemas mediante la programación, generando estructuras de conocimiento más profundas que permitirá pensar y diseñar algoritmos más eficientes y simples, evitando las soluciones "*de memoria*".

2. Reducir la cantidad de contenidos del programa actual, en función de un mejor desarrollo de los conceptos claves de la programación, potenciará la facilidad y adaptación de los alumnos frente a nuevos lenguajes de programación a futuro.

3. Generar espacios informales de debate y experimentación como una alternativa de las clases teórico-prácticas de los complejos software actuales generará curiosidad y motivará a los alumnos a mantenerse atentos, interesados y desafiados por los contenidos de la materia, a la vez que potenciará el desarrollo cognitivo de la disciplina y demostrará el potencial de la multimedia como herramienta pedagógica.

CAPÍTULO 1. EDUCACIÓN.

1.1. ESTUDIAR DE MEMORIA. UN PROBLEMA GENERALIZADO.

A principios de los ochenta, dos físicos de la Universidad Estatal de Arizona (EEUU) quisieron comprobar si la asignatura de introducción a la física cambiaba la manera de pensar de los estudiantes sobre el movimiento (Bain, 2007). Si bien esta investigación estuvo directamente relacionada con la materia de física de esa universidad, este ejercicio bien puede servir como parámetro para cualquier asignatura curricular, ya que su objetivo principal fue descubrir si los estudiantes cambian su manera original de pensar después de asistir a las clases, y éste ejemplo vale para la mayoría de las instituciones, entre ellos, la Universidad Maimónides y su carrera de Tecnología Multimedial.

En función de esto, los profesores Ibrahim Abou Halloun y David Hestenes idearon para los alumnos de introducción a la física un cuestionario con preguntas simples sobre el movimiento, pidiendo que los respondan el primer día de clase. Por supuesto, los resultados no fueron satisfactorios, ya que los alumnos respondían en base a su intuición y no poseían aún las herramientas teóricas correspondientes para dar una respuesta acertada.

Sin embargo, al finalizar el curso, los profesores nuevamente hicieron el cuestionario, descubriendo sorprendidos que la cursada sólo había producido cambios muy pequeños en la manera de pensar de sus alumnos. Descubrieron que los estudiantes habían memorizado fórmulas y también habían aprendido a poner los valores correctos en ellas, pero no habían logrado cambiar sus concepciones básicas sobre la materia, y

seguían interpretando el movimiento según el esquema intuitivo que habían traído con ellos al curso, y no en base a las teorías "*aprendidas*" que demostraban lo contrario. Este curioso hecho se daba incluso en aquellos alumnos que habían logrado aprobar la cursada con notables calificaciones.

```
/*
    Descubrieron que sus estudiantes habían memorizado
    fórmulas y también habían aprendido a poner los
    valores correctos en ellas, pero no habían logrado
    cambiar sus concepciones básicas sobre la materia.
*/
```

Desorientados, los profesores quisieron ir un poco más allá de este inquietante resultado, y organizaron reuniones personales con sus alumnos. Durante estas entrevistas, les hicieron preguntas teóricas sobre problemas elementales del movimiento, para que predigan lo que ocurriría en un experimento dado a continuación. Luego, realizaron el experimento práctico delante de ellos, demostrando el error en sus respuestas, y solicitándoles a los alumnos que expliquen las diferencias entre sus respuestas y el experimento. Pero lo que escucharon los dejó atónicos: la mayoría de los estudiantes seguían reacios a desechar sus ideas equivocadas sobre el movimiento, y en su lugar se excusaban diciendo que se trataban de casos especiales y que no se podía aplicar exactamente la ley del movimiento para todos los casos.

Así los investigadores llegaron a la inesperada conclusión que "***los estudiantes mantenían firmes sus creencias equivocadas, incluso cuando se confrontaban con fenómenos evidentes que contradecían esas creencias***". Si los profesores señalaban una contradicción, los estudiantes "*en primer lugar tendían a no cuestionar sus propias creencias, sino a mantener que el ejemplo observado era gobernado por alguna otra ley o principio, y que el principio que ellos utilizaban para demostrarlo era aplicable a un caso totalmente diferente*" (Halloum y Hestenes, 1985).

Este ejemplo presentado no es un hecho aislado, sino que es parte de un creciente conjunto de artículos que cuestiona si los estudiantes aprenden siempre tanto como tradicionalmente se pensó que aprendían. Los participantes en un congreso de 1987 sobre enseñanza de las ciencias observaron el mismo problema en matemáticas, concluyendo que "*aquellos que superan con éxito el cálculo frecuentemente no tienen una comprensión conceptual de la materia, ni una apreciación de su importancia debido a que los profesores confían en ejercicios del tipo 'enchufar y que funcione', que poco tiene que ver con el mundo real*" (McDonald, 1987).

Como resultado, los investigadores han encontrado que incluso algunos "*buenos estudiantes*" pueden que no progresen intelectualmente tanto como se creía. Han descubierto que algunas personas consiguen buenas calificaciones aprendiendo la técnica de "***enchufar y que funcione***", simplemente memorizando fórmulas o poniendo números en la ecuación correcta, pero comprendiendo muy poco cómo funciona, y **cuando terminan las clases, olvidan muy rápidamente lo que creían haber aprendido**.

1.2. PROFESORES ERRADOS. DIFICULTADES PARA PROMOVER EL APRENDIZAJE.

Ahora consideremos el caso de una profesora de anatomía que, entrevistada por el profesor Bain, insistía en que "*los alumnos deben aprenderse las cosas. Aquí no hay mucho para discutir. La estructura del cuerpo humano es bien conocida por los científicos, y los estudiantes no tienen más que absorber un montón de hechos. No es posible ninguna otra forma de enseñar que no sea plantarse delante de ellos y contarles esos hechos. No se puede discutir, como sí podría hacerse en una clase de literatura*". Esta profesora de dudoso éxito siempre habló de "*trasmitir*" conocimiento, insistiendo en que el objetivo principal de su curso era "*memorizar grandes paquetes de información*".

Ella decía que los estudiantes deben "*confiarlo todo a la memoria, almacenarlo y usarlo*". Sus exámenes reflejaban la misma línea de pensamiento, exigiendo a los estudiantes que reprodujeran todo lo que el profesor les había dado en clase. Sus alumnos, por lo tanto, confesaban que tenían dificultades para recordar la información pocos meses después de haber terminado el curso, mientras que el profesor simplemente lo atribuía tildándolos de "*estudiantes flojos que no estudian lo suficiente*".

Por el contrario, otra profesora del mismo curso siempre habló de estructuras de comprensión, de cómo se relacionan las partes individuales y del tipo de decisiones que los estudiantes debían ser capaces de tomar con el nivel de comprensión que desarrollasen. Habló de "*ayudar a los estudiantes a construir su entendimiento, y de aprender a utilizar la información para resolver problemas médicos*". En clase, a menudo explicaba cómo funcionan las cosas, intentando simplificar y aclarar conceptos básicos, pero también ponía problemas sobre casos clínicos para que los estudiantes se entusiasmen y se esforzaran con resolver esos ejercicios en base a todo lo aprendido previamente. Los estudiantes se encontraban con la información en un contexto en que debían primero enfrentarse a la comprensión, y luego a la aplicación de esa comprensión. Ella solía decir "*tengo que pensar en la razón por la que a alguien le gustaría recordar una información en concreto*".

En sus exámenes, pedía que los estudiantes debatieses casos clínicos, que desarrollasen y defendieran sus análisis, síntesis y evaluaciones. **Ellos seguían teniendo que recordar una enorme cantidad de información, pero también tenían que razonar sobre esos problemas, y no recordarlos y repetirlos de memoria.**

1.3. PROFESORES EXITOSOS. APRENDIENDO DE LOS MEJORES.

Según las conclusiones alcanzadas por las investigaciones del profesor Bain, se puede decir que generalmente los mejores

profesores son académicos, artistas o científicos expertos en sus disciplinas. Suelen tener un sentido inusualmente agudo de la historia de sus materias, incluyendo las controversias que se han agitado alrededor de ellas, y parece que esto les ayuda a reflexionar profundamente sobre la naturaleza de sus campos. Parecen utilizar esa capacidad extraordinaria de pensar sobre su propio razonamiento, y sobre su concepción de la disciplina como tal para entender cómo podrían aprenderla otras personas. Saben qué es lo que debe ir primero, y se dan cuenta dónde es más fácil que las personas encuentren dificultades a la hora de avanzar en su propia comprensión, simplificando asuntos que para otros resultan muy complejos de comprender y de explicar.

```
/*
    "Tengo que pensar en la razón por la que a alguien le
    gustaría recordar una información en concreto. Tendrán
    que recordar una enorme cantidad de información, pero
    aprenderán a razonarla para no repetirla de memoria."
*/
```

Los conceptos clave que utilizan los buenos profesores son (Bain, 2007):

1. El conocimiento es construido, no recibido.

El cerebro no es sólo una unidad donde se almacena nuestro conocimiento, por el contrario, es **una unidad tanto de almacenamiento como de procesamiento**. Nuestro sentido de la realidad se construye a partir de todas las entradas sensoriales que recibimos y conectamos, construyendo patrones que nos explican cómo funcionan las cosas. En algún momento, comenzamos a utilizar esos patrones disponibles para comprender nuevas entradas, y así es que cuando llegamos a la universidad, contamos con miles de modelos mentales que podemos utilizar para entender las clases.

2. Los modelos mentales cambian lentamente.

Para estimular a los estudiantes en un aprendizaje profundo, y que logren recordar los conceptos luego de rendir los exámenes, los alumnos deben:

1. Enfrentarse a una situación en la que su modelo mental no funciona, o no les sirva para realizar determinada tarea.

2. Asegurarse de que funciona lo suficientemente mal como para tener que detenerse y necesitar esforzarse a cambiar el asunto en cuestión.

3. Ser capaces de manejar el trauma emocional que en ocasiones acompaña al desafío de creencias erróneas mantenidas tanto tiempo.

Los mejores profesores a menudo hablan de **desafiar intelectualmente a los estudiantes**, buscando crear una situación en la que los modelos mentales existentes produzcan expectativas fallidas - fracaso de la expectativa -, provocando que sus estudiantes se den cuenta de los problemas a los que se enfrentan al creer lo que sea que crean.

Incluso estos profesores saben que los seres humanos se enfrentan a demasiadas expectativas fallidas en su vida como para preocuparse por todas ellas, por lo que los estudiantes pueden no comprometerse con la intensidad de razonamiento necesaria para construir modelos mentales nuevos, o simplemente pueden no saber cuál de todos sus esquemas los han conducido a los resultados fallidos, por lo que podrían corregir los que no deben.

Aquí es donde los estudiantes de física del capítulo 1.1 se equivocaban cuando veían experimentos en los que sus teorías no funcionaban y no podían admitir sus propios errores.

Estas ideas tienen implicaciones importantes para los profesores, ya que ellos llevan las clases y los asuntos propios del oficio, y deben proporcionar a sus estudiantes un lugar seguro en el que construyan ideas. Debido a que tratan de poner a los estudiantes en situaciones en las que sus modelos mentales no funcionaran, intentan entender esos modelos y la carga emocional unida a ellos. Escuchan las suposiciones de los estudiantes antes de desafiarlas, y en lugar de demostrarle el error a sus estudiantes, le hacen preguntas para ayudarlos a ver sus propios errores.

Para ellos, **aprender tiene poco sentido si no ejerce una influencia permanente en la forma en que posteriormente piensa, actúa o siente el estudiante.**

3. Las preguntas son cruciales.

Las preguntas ayudan a construir conocimiento. Apuntan a los huecos de nuestras estructuras de memoria y son vitales para indexar la información que retenemos, cuando desarrollamos una buena respuesta para esa pregunta. Incluso algunos científicos dicen que las preguntas tienen tanta importancia en nuestra educación, que no aprendemos hasta que nos hagan la pregunta indicada, ya que si la memoria no hace la pregunta, después no sabe dónde buscar la respuesta.

De esta manera, cuantas más preguntas hacemos, de más formas podemos indexar un conocimiento en la memoria. Un proceso de indexación mejor produce una mayor flexibilidad, un recuerdo más fácil y una compresión más rica.

Donald Saari, un matemático de la Universidad de California, dice que "*cuando podemos estimular con éxito a nuestros estudiantes para que se formulen sus propias preguntas, estamos justo en la base del aprendizaje*".

4. El interés es crucial.

La gente aprende mejor cuando responde a una pregunta importante que realmente tiene interés en responder, o cuando persigue un objetivo que desea alcanzar. Si no hay interés, no intentará reconciliar, explicar o integrar el nuevo conocimiento con el antiguo, y así no podrá recordar la respuesta un tiempo después. Las personas sólo podrán recordar la información un breve período de tiempo, el suficiente para llegar al examen, pero **sólo cuando su memoria genere preguntas interesantes estarán preparadas para cambiar sus estructuras de conocimiento**.

El interés, o la falta de interés, puede por ejemplo explicar por qué los alumnos de física del capítulo 1.1. completaban su curso con calificaciones sobresalientes, y aún así seguían sin comprender los conceptos básicos del movimiento. No aprendieron más que a colocar números en ecuaciones, sin experimentar una sola expectativa fallida de los universos que ellos imaginan en sus mentes. Quizás porque estaban más preocupados por las calificaciones en vez de por comprender el universo físico, no les importó lo suficiente como para tratar de vencer sus propias ideas y construir nuevos paradigmas de la realidad.

Las personas desarrollan un interés intrínseco que guía su búsqueda de conocimiento, y este interés puede disminuir al enfrentarse con recompensas y castigos extrínsecos que parezcan manipular su centro de atención.

```
/*
   Desafiando intelectualmente a los alumnos, y
   haciéndolos disfrutar de su propia educación, es más
   fácil que las personas estén preparadas a cambiar sus
   modelos mentales.
*/
```

1.3.1. EL BUEN DOCENTE. GENERAR INTERÉS Y DELEGAR EL MANDO.

Las ideas principales que animan a los mejores profesores tienen su origen en una observación muy básica: los seres humanos son animales curiosos. **La gente aprende de manera natural mientras intenta resolver los problemas que le preocupan** (Bain, 2007).

El profesor Hernán Cuevas, de la materia Comunicación Social, dice que "*es nuestro desafío hacer que los alumnos se interesen en la materia para que se queden con algo valioso. Si un alumno me dice que se está aburriendo, en vez de castigarlo trato de volver a integrarlo, haciendo un seguimiento personal hasta descubrir cuales son esas cosas que lo motivan*".

La historia universal demuestra que los grandes descubrimientos científicos de los últimos cinco siglos se produjeron tratando de resolver problemas cotidianos que los afectaban. Para comprender esta afirmación sólo basta con pensar en los avances tecnológicos en la cocción y conservación de los alimentos, en el traslado, la calefacción y la salud de las personas, y hasta en los infinitos avances en las telecomunicaciones.

Por lo tanto, **es más fácil que las personas disfruten y progresen en su educación si creen que están al mando de la decisión de aprender**, sin verse involucrados en redes burocráticas académicas que sólo las compriman en un objetivo institucional preestablecido y una calificación que muchas veces poco tiene que ver con el nivel de su aprendizaje.

De esta manera, los mejores profesores crean un entorno para el aprendizaje crítico natural, en el que incluyen las destrezas y la información que ellos quieren enseñar mediante trabajos que los estudiantes encontrarán fascinantes. Éstas son auténticas tareas que les provocarán curiosidad y motivarán a repensar sus supuestos, examinando sus modelos mentales de la realidad.

Estos profesores crean un **entorno seguro donde los estudiantes pueden probar, equivocarse, realimentarse y volver a intentarlo** cada vez con conceptos más amplios. Valeria Drelichman, docente de programación en el Instituto Universitario Nacional del Arte, asegura que "*en la materia los inducimos al error en clase, porque si no lo corrigen, se les queda grabado el error en vez de grabarse la solución correcta*".

Para ellos, este entorno es crucial para el correcto aprendizaje, y se arma en base a cuatro preguntas principales (Bain, 2007):

1. ¿Qué deberían ser capaces de hacer intelectual, física o emocionalmente mis alumnos como resultado de su aprendizaje?

2. ¿Cómo puedo ayudarlos y animarlos de la mejor manera para que desarrollen esas habilidades y los hábitos mentales y emocionales para utilizarlas?

3. ¿Cómo podemos mis estudiantes y yo entender mejor la naturaleza, la calidad y el progreso de su aprendizaje?

4. ¿Cómo puedo evaluar mis intentos de fomentar el aprendizaje?

Los mejores profesores siempre planifican hacia atrás: comienzan con los resultados que esperan fomentar. Se preguntan si quieren que los estudiantes recuerden, comprendan, apliquen, analicen, sinteticen o evalúen. Se preocupan en que los alumnos deseen y crean posibles llegar a esos mismos resultados. Se esfuerzan en ayudar al estudiante a entender toda la belleza y el disfrute de la empresa que les precede. Esperan más que la mera memorización de respuestas correctas, quieren saber cómo ayudar a los estudiantes a razonar sobre cada pregunta y todas sus respuestas, y esto lo aplican tanto dentro como fuera del aula.

Los alientan a generar sus propias preguntas y hacerse cargo de su propia educación, tratándolos incluso como colegas y no como alumnos, siempre actuando como moderadores del conocimiento y determinando en conjunto los contenidos y objetivos curriculares. Debido a que son conscientes que todos los estudiantes pueden aprender algo nuevo, los animan a involucrarse y a pensar en voz alta, y crean una atmósfera no amenazadora donde los alumnos puedan libremente pelear con sus pensamientos sin ser calificados o criticados. Ken Bain concluye que "***si los estudiantes no pueden aprender a juzgar la calidad de su propio trabajo es porque en realidad no han aprendido***".

1.3.2. EL APRENDIZAJE PROFUNDO. UN COMPROMISO ENTRE TODOS.

Algunos profesores de poco éxito están convencidos que el camino para congeniar sin problemas con un curso está pavimentado con expectativas de bajo nivel, poca exigencia y nula recepción por parte de los alumnos. De esta manera, anticipan un rechazo a los altos resultados académicos, logrando calificaciones mediocres sin crearse demasiados problemas, ni para los alumnos ni para los docentes mismos. Sin embargo, también existen profesores que, igualmente equivocados, intentan el camino contrario, retando a sus estudiantes con una cantidad exagerada de trabajo, y que a la larga acaba resultando en notas más bajas y quizás menos aprendizaje aún, debido a que los alumnos terminan exhaustos y ofendidos por el abultamiento de información desordenada que se les proporciona, y perdieron de vista sus objetivos de aprendizaje y la motivación con la que habían arrancado el curso.

Por otra parte, los mejores profesores generan altas expectativas y suelen cumplirlas con grandes satisfacciones guiados por una compleja red de creencias, concepciones, actitudes y prácticas. La fortaleza de cada hebra de esta red depende de las demás, y sólo funcionan en conjunto. Primero, los buenos profesores tienden a apreciar el trabajo individual de cada estudiante.

No los separan entre malos y buenos alumnos, sino que buscan las cualidades individuales de cada uno, y confían enormemente en la capacidad de cada estudiante de lograr sus objetivos. Paul Baker, un exitoso profesor de arte dramático de Texas, dice que "***cada estudiante es único y proporciona contribuciones que nadie más puede aportar***" (Bain, 2007). Así, los estudiantes mantendrán su ilusión mediante expectativas positivas que sean genuinas, estimulantes pero realistas, y que tomen en serio su trabajo, evitando el lenguaje de las exigencias y proporcionando a los estudiantes una sensación de control sobre su propia educación.

```
/*
    "No vamos a intentar meterlos en ningún molde,
    al contrario, estamos intentando ayudarlos a que salgan
    de él". Paul Baker
*/
```

Hay profesores que se esfuerzan increíblemente por explorar el aprendizaje de sus estudiantes, para analizar cuidadosamente su trabajo, para reflexionar en profundidad sobre qué y cómo pueden aprender las distintas personas, e incluso para diseñar tareas individuales ajustadas a las necesidades, los intereses y las capacidades reales presentes en cada alumno. Viviana Polo, profesora de programación en la Universidad Maimónides, asegura que en sus clases intentan "*fomentar la conducta de la práctica, tanto individual como en forma grupal, para que el alumno que ya entendió les pueda explicar el tema a sus compañeros*".

Incluso, mientras los profesores poco exitosos suelen poner el énfasis en la cantidad de trabajo, y el cumplimiento rígido de los objetivos académicos, éstos profesores se focalizan en la capacidad de razonar correcta y cuidadosamente, de comprender asuntos complejos, de recoger y utilizar evidencias para resolver problemas, y de todo aquello que cualquiera de los mejores eruditos, profesionales y artistas puedan hacer en sus propios campos, y sirvan de ejemplos para sus alumnos.

De igual manera, la confianza, el rechazo del poder y el armado de estándares que representen objetivos reales en lugar de tareas *escolares* son cosas que están muy presentes en las clases de los mejores profesores. Allí, el instructor no sólo deja muy claras las promesas y oportunidades que ofrece el curso a sus alumnos, sino que les explica lo que debe hacer cada uno para conseguir esas promesas. Es un diálogo en el que tanto estudiantes como instructores exploran la forma de entender el aprendizaje, ajustándose constantemente sobre la marcha, y evaluando al fin del curso la naturaleza de lo aprendido. Paul Baker decía con frecuencia a sus estudiantes "*el principal objetivo del curso es el desarrollo de personas creativas, proporcionándoles confianza en ellas mismas.* ***No vamos a intentar meterlos en ningún molde, al contrario, estamos intentando ayudarlos a que salgan de él***".

Las cualidades llevadas a práctica en estos programas descansan en la sencilla idea de que el aprendizaje involucra tanto al desarrollo personal como al intelectual, y que ni la capacidad de pensar ni la calidad de una persona madura son inmutables. **Las personas pueden cambiar, y esos cambios representan aprendizaje auténtico**. Esto lleva al modelo de educación en el que los que aprenden hacen algo más que almacenar información, tal como decía la poco exitosa profesora de anatomía del capítulo 1.2.

Estos alumnos llevan a cabo cambios en profundidad, transformaciones que afectan tanto a las costumbres emocionales y los hábitos de pensamiento, como a la capacidad para continuar creciendo. "*Todo lo que aprendes influye en quién eres y lo que puedes hacer*", decía Ralph Lynn de la Baylor University. Los profesores generan así entornos creativos de aprendizaje, para tomar decisiones correctas en cualquier aspecto docente, y para responder a los problemas con creatividad y eficacia. **El éxito alimenta al éxito**. Como los métodos, y los estudiantes consiguen sus metas, también aumenta su fe en los instructores, y esa confianza se convierte en sus mismas fuerzas. Al final, ninguno de estos factores está solo, y todos se alimentan de todos.

A modo de resumen, se presentan a continuación (TABLA 1) las principales diferencias entre las formas de pensar de los profesores que hemos determinado como buenos y malos, según el criterio que se viene manejando en esta investigación.

TABLA 1. COMPARACIÓN ENTRE MALOS Y BUENOS PROFESORES.

CONCEPTOS	MALOS PROFESORES	BUENOS PROFESORES
El conocimiento	Se transmite	Se ayuda a construir
Promueven	La memoria	El razonamiento
El problema es	Estudiantes flojos	Estudiantes desmotivados
El cerebro es	Una unidad de almacenamiento	Una unidad de procesamiento
El profesor	Corrige errores	Ayuda a corregir errores
Los alumnos son	Alumnos	Colegas
Separan los cursos en	Buenos y malos alumnos	Cualidades propias de cada alumno
Suelen dar mayor importancia a	Gran cantidad de trabajos y fechas de entrega	Ejemplificar cada caso y motivar el autoaprendizaje
El buen docente debe	Encontrar las capacidades de sus mejores alumnos	Fomentar las capacidades de todos sus alumnos
Los problemas se solucionan	Practicando mecánicas y memorizando fórmulas	Reflexionando sobre los conceptos e imaginando nuevos escenarios

1.4. LA EVALUACIÓN. NIVELES DE MOTIVACIÓN Y EXIGENCIAS.

Durante los últimos cuarenta años, los psicólogos han estudiado lo que pasaría si alguien tiene mucho interés en hacer algo y otro le ofrece una recompensa extrínseca para reforzar su interés intrínseco, y lo que sucedería si más tarde se les quita ese premio.

La realidad dice que el interés disminuye considerablemente. Edward Deci ha teorizado que las personas pierden mucha de su motivación si creen que están siendo manipuladas por la recompensa externa, es decir, si pierden lo que los psicólogos llaman su sentido de locus de causalidad de su comportamiento.

Él formuló que "l*os sujetos investigados perdieron parte o toda su fascinación intrínseca una vez que desaparece el motivador extrínseco, porque generaron dependencia hacia el motivador y se olvidaron lo que los motivaba en principio*" (Deci, 1970). En otras palabras, si la gente ve determinada conducta como un medio para conseguir algo en especial, entonces se dedicarán a estas actividades sólo cuando deseen esas recompensas específicas, y cuando crean que ese premio sólo llegará el comportamiento correspondiente. Y, por el contrario, **si no desean ese beneficio en concreto, o si la posibilidad de recompensa se elimina posteriormente, perderán todo tipo de interés en esa actividad**, aunque haya sido en principio la fuente motivadora del alumno. Por el contrario, como expresó Deci, "*el esfuerzo verbal, la aprobación social, y cosas así son más difíciles que sean percibidas por las personas como reguladores del comportamiento*". La clave entonces parece estar en cómo el sujeto considera la recompensa.

Los investigadores han descubierto también que tanto la motivación como el resultado pueden ser peores cuando los sujetos creen que otros tratan de controlarlos. Si los alumnos estudian sólo porque quieren sacar buenas notas, o porque les prometieron una recompensa especial, no les irá tan bien como si estudiasen porque tienen interés propio y genuino en la materia.

Quizás obtengan altas calificaciones, pero a la larga no resolverán problemas con tanta eficacia, no analizarán tan bien los resultados, ni se plantearán la misma clase de desafíos que los otros. A menudo optarán por problemas más sencillos, mientras que los que trabajan a partir de motivaciones intrínsecas escogerán tareas más ambiciosas. **Pueden convertirse en aprendices estratégicos, que se centran principalmente en que les vaya bien en los exámenes**, evitando cualquier desafío que pueda dañar su reputación académica, y sin desarrollar una comprensión en profundidad. Estos estudiantes se sienten únicamente inteligentes cuando evitan estas actividades que son precisamente las que con mayor probabilidad les ayudarían a aprender, centrándose exclusivamente en aquellas tareas que creen poder resolver sin tanta dedicación ni esfuerzo.

El profesor Bain expone un ejemplo personal de sus inicios como docente, donde tenía dudas sobre cómo preparar el examen a sus alumnos, y solicitó la asistencia de un colega más experimentado (Bain, 2007). Juntos compusieron inteligentes enigmas para confundir a los estudiantes, porque buscaban obtener altos estándares entre los alumnos. Sin embargo, el resultado no fue el esperado, ya que unos días antes del examen, sus alumnos estaban más preocupados por la clase de preguntas que iban a tener, que por el conocimiento en sí. Por lo tanto, los resultados de esos exámenes poco dijeron sobre los logros intelectuales de sus alumnos, ni mucho menos sobre sus logros como docente.

En cambio, reforzó el aprendizaje estratégico en lugar del aprendizaje profundo. Años más tarde dijo " *fallé como tantos otros profesores a la hora de entender que* ***examinar y calificar no son actos de importancia menor*** *que llegan con el final de las clases, sino aspectos muy poderosos de la educación, y ejercen una influencia enorme en todo el proceso de ayudar y animar a los estudiantes a aprender*". Sin una evaluación adecuada, ni profesores ni estudiantes pueden comprender el progreso que están haciendo los que aprenden, y los instructores pueden averiguar muy

poco sobre si sus esfuerzos son los más adecuados para sus estudiantes y sus objetivos. "*Muchos exámenes pueden captar la capacidad de los estudiantes para hacer ciertos tipos de pruebas, pero son un pobre reflejo de su forma de pensar*", concluyó.

Por otro lado, muchos profesores de dudoso éxito tenían en claro que las calificaciones son para "*separar las ovejas de las cabras*". Bajo esta idea, la escolaridad es principalmente una forma de catalogar, de elegir a los mejores y más brillantes, y no de ayudar a todos los estudiantes a aprender mejor. Así, estos fallidos profesores creen internamente que su principal responsabilidad está en encontrar la capacidad de unos pocos, en lugar de fomentar el desarrollo intelectual de todos por igual.

```
/*
    Todos los estudiantes, incluso los de mejor calificación,
    pueden convertirse en aprendices estratégicos si no se les
    ayuda a razonar sobre el objetivo real de su aprendizaje.
*/
```

Contrariamente, los mejores profesores opinan que **el objetivo primario de la calificación es ayudar a los estudiantes a razonar sobre su propio conocimiento**, de forma que puedan utilizar los estándares de la disciplina para reconocer las deficiencias y corregir sus razonamientos sobre la marcha. Esto no es clasificar a los estudiantes, ya que cada calificación representa un nivel de logro personal directamente articulado con los objetivos de cada uno. Tampoco tienen valor para estos profesores aquellos alumnos que consumen su tiempo de preparación intentando adivinar lo que puede preguntar el profesor. Paul Travis, profesor de la Universidad de Texas, dijo "*quiero que mis estudiantes se preparen intelectualmente, se concentren en lo que entienden y en cómo razonan con lo que comprenden. No quiero que pierdan el tiempo sonsacándome qué cosas podría exigirles que memoricen. Si entienden la materia, ellos saben qué información merece ser recordada. Si no, yo he hecho mal mi trabajo*".

Incluso, muchos profesores directamente anticipan las principales preguntas del examen final en los primeros días de clase. En matemáticas, y otras asignaturas orientadas a los problemas lógicos, esto implica ayudar a los estudiantes a comprender conceptos que les permitirán resolver los problemas, en lugar de poner interés únicamente en practicar la mecánica propia de la resolución de problemas. En vez de usar fórmulas de memoria, y hacer cálculos delante de los alumnos semana tras semana, para luego no exigirles más que el mero proceso de los ejercicios diarios, Donald Saari los ayuda a aprender a cada uno a inventar su propio cálculo.

De igual manera, muchos profesores extraordinarios hacen exámenes globales y acumulativos, de manera de cada prueba es igual a la anterior, pero más sofisticada. Ralph Lynn decía que "*la meningitis es tan importante al principio como al final de año.* ***No aprendemos algo solo para darle un beso de despedida una vez terminado el examen***".

De esta forma, haciendo cada examen progresivo, los buenos profesores dejan en claro a sus estudiantes que el aprendizaje es permanente, y no sólo para cubrir una instancia de examen. Los preparan para que hagan determinados tipos de trabajo intelectual, y no para que sean buenos haciendo exámenes. Si el objetivo del curso es desarrollar suficiente entendimiento para resolver problemas, la calificación no puede depender de lo bien que recuerdan la información en un tiempo limitado.

1.5. MÉTODO CONSTRUCTIVISTA. CONSTRUYENDO EL APRENDIZAJE.

Por este motivo, es necesario evaluar herramientas y métodos que contribuyan al desarrollo de las habilidades que los alumnos requieren para comprender y absorber los contenidos dados en cada materia. Puesto que los estudiantes necesitan de estos conocimientos para enfrentar los retos que le depara una sociedad

que cada vez exige mayor competitividad, es indiscutible generar las habilidades que tendrán que ver con el aprovechamiento de la capacidad creativa, el razonamiento crítico, la inventiva, la moral autónoma, etc., como la única forma de lograr la construcción de una nueva generación consciente de sus saberes y responsable en sus decisiones (Burón Orejas, 2002).

Puesto que el conocimiento no es resultado ni copia de la realidad, sino que se da mediante un proceso dinámico e interactivo que transforma las estructuras internas del cerebro, construyendo redes cada vez más complejas, es que surge como respuesta un nuevo paradigma educativo: el Constructivismo Pedagógico. (Barreto Tovar, Gutiérrez Amador, Pinilla Díaz y Parra Moreno, 2006).

Sus características principales son (Ramírez Toledo, 2007):

1. Se apoya en la estructura conceptual de cada estudiante: parte de las ideas y preconceptos que el estudiante trae consigo sobre el tema de la clase.

2. Anticipa el cambio conceptual que se espera de la construcción activa del nuevo concepto y su repercusión en la estructura mental.

3. Confronta las ideas y preconceptos afines del tema de la enseñanza, con el nuevo concepto científico que enseña.
4. Aplica el nuevo concepto a situaciones concretas y lo relaciona con otros conceptos de la estructura cognitiva con el fin de ampliar su transferencia.

El constructivismo plantea que "*cada alumno estructura su conocimiento del mundo a través de un patrón único, conectando cada nuevo hecho, experiencia o entendimiento en una estructura que crece de manera subjetiva y que lleva al aprendiz a establecer relaciones racionales y significativas con el mundo*". [3]

La enseñanza constructivista considera que el aprendizaje es siempre una construcción interior, aún en el caso de que el profesor realice una clase magistral, pues **no podrá ser significativa si sus conceptos no encajan ni se insertan en los conceptos previos de los alumnos**. Y por ese motivo, el propósito de la acción constructivista es precisamente facilitar y potenciar al máximo ese procesamiento interior del alumno con miras a su desarrollo.

Por supuesto que la culpa de no absorber y comprender los nuevos conocimientos no debe recaer exclusivamente en el alumno. Contando con la ayuda de un conocido refrán que promulga "*la culpa no la tiene el chancho, sino quien le da de comer*", caemos en consciencia de que el instructor tiene un grado mucho mayor de responsabilidad en la forma de transmitir sus conocimientos hacia los alumnos, que ellos en la forma de absorber y construir el conocimiento en base a lo recibido. Generalmente existen buenos profesores y malos estudiantes, pero como hemos visto anteriormente, el caso inverso también es posible.

Por ello, la creciente tendencia de volcar las decisiones educativas hacia un enfoque constructivista, habla de una sana búsqueda donde el propio alumno será libre de desarrollar y reconstruir sus propias experiencias subjetivas de la realidad, para construir así un nuevo conocimiento y su propia sabiduría.

A mediados del siglo XX, el sociólogo Alfred Schütz expone que "*todo nuestro conocimiento del mundo, tanto en el sentido común como en el pensamiento científico, supone construcciones, es decir, conjuntos de abstracciones, generalizaciones, formalizaciones e idealizaciones propias del nivel respectivo de organización del pensamiento. En términos estrictos, los hechos puros y simples no existen*" (Schütz, 1995). Lo que constituye la realidad no es la estructura ontológica de los objetos, sino la interacción entre los sujetos y esos objetos.

Más recientemente, Ernst von Glasersfeld aclara que el constructivismo no niega la posibilidad de conocer, sino que propone otros términos para explicar estos procesos afirmando que "*el constructivismo es una teoría del conocimiento activo, no una epistemología convencional que trata al conocimiento como una encarnación de la verdad que refleja al mundo 'en sí mismo', independiente del sujeto cognoscente*". Sobre este enunciado se entiende que el conocimiento no se recibe pasivamente, ni surge meramente por la acción de los sentidos, ni por medio de la comunicación, sino que es construido por el sujeto cognoscente.

En síntesis, **el conocimiento no es más que una propuesta que responde a una forma de situarse frente a la experiencia** (López Pérez, 2003).

El constructivismo contiene una ética de la convivencia con especial reconocimiento para la tolerancia. La argumentación es mejor recurso que la fuerza y las tradiciones heredadas tienen que pasar el examen de la reflexión crítica. La tarea es buscar colectivamente la mejor solución, aunque no sea posible alcanzar la verdadera. Así se crean acuerdos y se postulan valores que sin ser definitivos, mantienen un alto significado dentro de las condiciones que se han creado.

Esto presupone la generación de objetivos cortos pero concretos, destinados a generar nuevas realidades, que sin llegar a ser una verdad absoluta, se tratará de una realidad objetiva, comprobable únicamente en la conciencia de cada uno, puesto que los hombres son incapaces de reconocer los objetos que están fuera de su percepción sensorial, y ésta no garantiza una aprehensión de las cosas tal como son. La percepción revela lo que aparece, pero no tenemos jamás testimonio directo de lo que es. De esta manera, si la naturaleza de las cosas no puede ser conocida, no existe una referencia sólida para decidir la certeza del conocimiento.

En resumen, **el conocimiento es una construcción**, y como tal, refleja el tipo de dilemas que los seres humanos han enfrentado en el transcurso de sus vidas. El conocimiento expresa orientaciones y posee por tanto un importante valor de uso, puesto que está en conexión con las distintas maneras de actuar y de cumplir objetivos, y rápidamente se separa de sus creadores y comienza a ser parte del mundo. Se convierte a continuación en parte de la interacción y pasa a tener repercusión sobre la vida de sus propios creadores. Hay una retroalimentación permanente entre los hombres y sus construcciones.

Para Piaget, el conocimiento está unido a la acción, a las operaciones, es decir, a las transformaciones que el sujeto realizo sobre el mundo que le rodea. Así, el conocimiento resulta de la interacción entre sujeto y objeto. El origen del conocimiento, entonces, no radica en los objetos, sino en la interacción entre ambos. Es la suma de la acción, la asimilación y la acomodación de los datos recogidos. Sin embargo, Piaget opina que la estructura intelectual que caracteriza al sujeto en determinado estadio de desarrollo no únicamente posibilita la comprensión de un determinado rango de fenómenos, sino que también limita lo que el alumno puede comprender y aprender (Villar, 2003).

Así, los efectos del aprendizaje en el sujeto (informando verbalmente, dejándole ver los resultados de la deducción que habría tenido que hacer, etc.), pueden acelerar hasta cierto punto algunos logros que el sujeto adquirirá por sí mismo posteriormente, pero en general provocan poco cambio en el pensamiento lógico. Lo que se enseña al sujeto sólo es verdaderamente asimilado cuando da lugar a una reconstrucción activa o incluso a una reinvención por parte del alumno.

Sobre este punto, Piaget afirma que "***cada vez que se le enseña prematuramente a un alumno algo que habría podido descubrir solo, se le impide a ese sujeto inventarlo, y en consecuencia, entenderlo completamente.***" [5].

/*

"Cada vez que se le enseña prematuramente al alumno algo que habría podido descubrir solo, se le impide inventarlo, y en consecuencia, entenderlo completamente". Jean Piaget.

*/

Siguiendo esta línea, Kamii propone que los objetivos generales de la educación han de encaminarse hacia la promoción de la autonomía intelectual y moral de los alumnos, pasando de estados menos autónomos a estados más autónomos. Por ejemplo, intelectualmente, el alumno pasa de estar dominado por las apariencias y por las cualidades perceptivas de los objetos a no dejarse llevar por lo accesorio y discernir entre las propiedades sensoriales y las propiedades lógicas de esos objetos. De un pensamiento intuitivo se pasa a un pensamiento sistémico que funciona a partir de la formulación y comprobación de hipótesis.

De manera similar, del egocentrismo que supone tener en cuenta un único punto de vista sobre una situación, se pasa a la coordinación de múltiples perspectivas sobre las situaciones (Kamii, 1983).

Kamii entiende esta autonomía en relación a la educación en al menos dos sentidos:

1. Un alumno aumenta su autonomía cuando se potencia su pensamiento independiente y creativo. Se trata de orientar la educación hacia la promoción de la curiosidad, de las ganas de descubrir y de inventar en el alumno.

2. Un alumno aumenta su autonomía cuando se hace más capaz de tomar en cuenta y coordinar puntos de vista diferentes y es capaz a partir de esa competencia de pensar críticamente su propio punto de vista, ya sea este intelectual o moral.

De esta manera, el desafío de establecer la autonomía como el objetivo por excelencia de la educación implica, entre otras cosas:

- **Ofrecer un espacio a los estudiantes para que puedan construir sus propias ideas, pensamientos y actitudes morales,** con independencia de que nos gusten o no, o de que sean más o menos correctas desde nuestro punto de vista. Si por el contrario, nos limitamos a ofrecer "*conocimientos acabados*" a los alumnos, correremos el riesgo primero de que no los entiendan, después de que puedan aprovecharse del valor de sus propios errores.

- Reducir el poder coercitivo del alumno, de manera que el alumno tenga confianza en sus propias posibilidades y competencias, pueda tomar decisiones, y por otro lado, no se impongan valores arbitrarios por medio de sanciones o castigos. Se trata de crear alumnos libres, no serviles.

- **Potenciar la interacción y el intercambio de puntos de vista, de igual a igual, entre el alumno y los profesores, y el alumno y sus compañeros**, de manera que se desarrollen habilidades de cooperación y negociación, que por otra parte, contribuirán al propio desarrollo personal.

Para hablar del papel del profesor en la educación constructivista, algunas interpretaciones extremas sostienen que su papel en la enseñanza y aprendizaje simplemente no existe, ya que es el alumno quien ha de aprender y descubrir por sí mismo. Sin embargo, la gran mayoría de los investigadores sí le dotan de una importante función en este proceso.

Dentro de la postura del constructivismo en sentido estricto, investigadores como Furth y Wachs [5] se inclinan por un profesor que interviene poco y nada en las situaciones de aprendizaje del alumno. En lugar de diseñarlas y exponerlas a los alumnos, desde esta posición la tarea del profesor simplemente ha de

asegurar un entorno rico en estímulos que le de las posibilidades para que el alumno, trabajando a su propio ritmo, sea capaz de construir nuevas estructuras cognitivas. Así, es el desequilibrio provocado por la propia actividad espontánea del estudiante y de su funcionamiento cognitivo lo que asegura el progreso, más que un desequilibrio provocado "*desde afuera*" por un profesor que imponga actividades desafiantes. La principal crítica que se le puede hacer a esta postura es que no basta poner en contacto al alumno con un entorno adecuado para que actúe en manera efectiva con ellos, ya que el riesgo de que se estanquen es muy elevado.

Por el contrario, desde una postura menos extrema del constructivismo se opina que la tarea del profesor es primero el diagnosticar el nivel de desarrollo cognitivo de sus alumnos, para después proponer actividades apropiadas a ese nivel de desarrollo. Las actividades apropiadas serán aquellas que planteen un desafío, un conflicto a las estructuras de conocimiento que el alumno utiliza para interpretar la realidad. Sin embargo, este nivel no debe ser tan alto como para que el conflicto provocado sea tan grande que, simplemente, sea imposible de asimilar.

*/**
Los problemas más valiosos, aquellos que van a fomentar estructuras más amplias y profundas de conocimiento, han de tener sentido para el alumno, para que pueda elaborar sus propias soluciones creativas.
**/*

La profesora Delia Lerner, intentando unificar ambas posturas, comenta que los problemas más valiosos desde un punto de vista educativo, aquellos que con mayor probabilidad van a provocar la movilización de las competencias cognitivas existentes y su crecimiento para fomentar estructuras más amplias y profundas, son los que presentan dos características (Lerner, 1996):

- **Han de ser problemas que tengan sentido para el alumno y, al mismo tiempo, que han de ir un poco más allá de los esquemas que ya poseen.**

- **Han de ser problemas que se expongan de forma abierta, en los que los alumnos tengan que tomar decisiones y puedan elaborar de forma creativa soluciones por ellos mismos.**

De esta manera, es necesario repensar la educación y los programas de enseñanza, pues el objetivo de estas nuevas herramientas educativas es demostrar a los alumnos que es posible la aplicación del conocimiento a la vida diaria, y que **el aprendizaje puede ser un proceso divertido, que puede producir en ellos cambios significativos**, no sólo en términos de conocimientos, sino, en el logro de nuevas habilidades que los ayuden a insertarse de manera más productiva en el contexto social en el que crecen.

1.6. EDUCACIÓN A DISTANCIA. CLASES PRESENCIALES VS. ENTORNOS VIRTUALES.

Desde sus orígenes, la educación a distancia se ha ocupado de la profesionalización en los campos de la formación inicial y continuada. Los programas específicos de este tema han sido vinculados a la realidad emergente y a las necesidades del mercado de trabajo, y han contribuido en gran medida al desarrollo social y económico en el contexto correspondiente (Oliveira y Rumble, 1992).

Sin embargo, también se ha cuestionado su capacidad para desarrollar adecuadamente una formación orientada a la práctica profesional, prejuicio derivado de un **supuesto pedagógico bastante arraigado en la cultura occidental que formula que la educación pasa únicamente por la presencia física**, entendida como coincidencia en el espacio y el tiempo (Duart y Sangrà, 2000).

El problema comprende también la naturaleza de los contenidos, debido a las limitaciones del medio, ya que se presupone un mayor volumen de contenidos teóricos que prácticos en este tipo de educación.

Por lo tanto, defender que el aprendizaje es factible en un contexto no presencial pasa por cuestionar el supuesto de la coincidencia espacio temporal como condición única de posibilitar una acción educativa de calidad, y en caso de hacerlo, debe limitarse a incluir exclusivamente contenidos teóricos en las clases. Yendo a casos extremos, este punto de vista no es completamente infundado si se tiene en cuenta que los modelos tradicionales de educación a distancia por correspondencia dificultan un verdadero proceso de relación educativa. La rigidez y homogeneización del modelo postal hacen que el grado de interacción entre profesor y estudiante quede considerablemente diferido en el tiempo, y se convierte en el primer obstáculo para que uno pueda aprender del otro debido a la imposibilidad de mantener un diálogo fluido entre las partes. Recordemos que el instructor debe poder crear un entorno de aprendizaje idóneo para que el alumno adquiera progresivamente una posición intelectual profunda con respecto a un contenido específico. También debe prepararlo para permitirle adquirir por sí mismo una perspectiva, un punto de vista, una vía de acceso a la cultura y a la realidad de su disciplina.

Debe, por lo tanto, **enseñarle al estudiante a profundizar en la información**, a analizar por qué se han seleccionado determinados hechos, por qué son importantes, cómo tienen que ser interpretados y cómo se pueden rechazar. Así, el estudiante aprende a analizar la realidad desde diferentes ángulos, bajo diferentes condiciones, lo que le permite imaginar varias alternativas y posibilidades (Schön, 1992).

Por otra parte, el campo experimental de cualquier profesión viene determinado por una formación discursiva que se manifiesta en cada situación. Una práctica diferente para

cada situación con elementos comunes a todas, en virtual de los cuales cada uno elabora su propia experiencia y saber posibles. **La profesionalización consiste en hacer posible una individualización de la práctica discursiva, rechazando las respuestas estandarizadas**. Para que esto sea posible, no hay que ceder a la presión por la inmediatez y eficacia de las respuestas, y situar la teoría y la práctica en una única dimensión estructural que haga posible una reelaboración de cada situación educativa (Duart y Sangrà, 2000). Según este planteamiento, la simple presencia no es una garantía de aprendizaje. Por supuesto esto no significa que no tenga su valor agregado, pero sí se cuestiona el supuesto que el estudiante ya adquiere conocimiento por el mero hecho de estar presente en una clase.

Los entornos virtuales, en cambio, añaden nuevos elementos que sí aseguran los factores elementales de este proceso: la aptitud para formular, discutir y adoptar perspectivas diferentes de práctica profesional. Además, lo desarrollan incorporando la proximidad, e incluso la presencia virtual específica, inmediata y regular, que a través del avance de la tecnología, adquieren tanto profesor como alumno en estos nuevos entornos formativos. Por lo tanto, **la elaboración de entornos virtuales tiende a fomentar la interactividad, facilitando el acceso no lineal a la información, y permite la bidireccionalidad en las comunicaciones, con la finalidad de vencer el aislamiento, de monitorizar el proceso de aprendizaje y de fomentar el trabajo en grupo** (Schön, 1992).

1.6.1. EL DISEÑO FORMATIVO. HERRAMIENTAS PEDAGÓGICAS NO PRESENCIALES.

Los materiales didácticos para garantizar la adecuación y la eficacia del aprendizaje en un entorno no presencial responden a una efectiva aplicación del diseño formativo, traducida en herramientas multimediales que integran en un único producto un amplio abanico de elementos que hasta ahora sólo podían ser tratados separadamente por tecnologías diversas.

Si bien la definición data del año 1997, su concepto prevalece impecable en el tiempo. "*El diseño formativo es el elemento más importante del proceso de definición y de elaboración de una acción formativa. A pesar de que el resto de elementos que la componen haya sido conceptualizado correctamente y muy bien elaborado, si el diseño formativo no se realiza de forma adecuada, la acción formativa no será válida ni responderá a los objetivos académicos para los cuales ha sido definida*" (Phillips, 1997).

A menudo también se utiliza para describir el proceso en que:

1. Se analizan las necesidades de aprendizaje y el entorno donde se manifestarán.

2. Se definen los objetivos de la formación.

3. Se escogen los recursos más adecuados teniendo en cuenta los procesos de aprendizaje y el contexto correspondiente.

4. Se desarrollan los contenidos y las actividades.

5. Se diseña la evaluación.

El Applied Research Laboratory de la Penn State University define el concepto de diseño formativo como *"el desarrollo sistemático de una acción formativa basado en las teorías del aprendizaje. Es el proceso global de* ***análisis de necesidades educativas, de determinación de objetivos de aprendizaje y la definición del soporte y los medios en el desarrollo de la acción****.*

Incluye también el desarrollo de materiales didácticos y actividades de aprendizaje, así como el proceso de evaluación, tanto del material en sí mismo como del proceso de aprendizaje de los alumnos, y la aplicación sistemática de las estrategias y técnicas derivadas de las teorías del conocimiento" (Duart y Sangrà, 2000).

Pero, independientemente del modelo, del nombre y de las teorías, el aprendizaje siempre se produce a partir de una combinación de múltiples factores más o menos estándares, tales como la motivación, la activación de los conocimientos previos, las actividades de aprendizaje, los materiales, las habilidades, los procesos, las actitudes, el entorno de interacción, la orientación, la reflexión y la evaluación.

Algunos de los factores estrechamente relacionados con el diseño formativo son:

1. **El estudiante aprende de maneras diferentes.**

   Es necesario ofrecer entornos, recursos y herramientas adecuadas que le ayuden a aprender de manera activa e individualizada, y que le permitan experimentar, discutir y compartir en grupo, construir y progresar.

2. **El aprendizaje es un proceso dinámico.**

   El conocimiento lo va configurando y validando el mismo alumno sobre la marcha, y toma significado a través del entorno de interacción. Por lo tanto, se habla de un aprendizaje activo centrado en el estudiante.

3. **El aprendizaje requiere un rol activo del estudiante.**

   El aprendizaje será más efectivo si el alumno tiene unos objetivos específicos que le interesan, si tiene acceso a la información cuando la requiere, si se siente responsable de aquello que aprende y lo controla, si siente el aprendizaje como un proceso que se produce continuamente, y si se interesa por recibir más información.

1.6.2. APLICACIÓN DEL DISEÑO FORMATIVO. PAUTAS PARA SU DESARROLLO.

El éxito del diseño e implementación de los materiales didácticos para la formación no presencial será adecuado y eficaz si se desarrolla en seis etapas fundamentales (Duart y Sangrà, 2000):

1. **Análisis y definición.**

   Se busca adecuarla al contexto social donde será aplicada, se combina con los estudios y programas que la institución ofrece, se proponen los objetivos de aprendizaje, se presenta un programa de contenidos adecuados para la disciplina y se integra en un entorno multimedial que unifique y contextualice la acción formativa.

2. **Diseño y concreción.**

   Se genera un diseño apropiado teniendo en cuenta el perfil de los estudiantes, con motivaciones y necesidades diferentes, con el fin de darles apoyo, completar la información y dinamizar el proceso de aprendizaje. También se determina la arquitectura general de la acción formativa, se diseña la estructura y funcionalidad de todos los elementos multimediales, se proponen los recursos metodológicos que utilizarán y se diseñan las actividades de aprendizaje que activen los conocimientos previos, motiven al estudiante a participar activamente y permitan evaluar sus rendimientos.

3. **Desarrollo de la propuesta.**

   Es necesario desarrollar los contenidos y las actividades para cada uno de los objetivos propuestos, proponer las tareas relacionadas con la evaluación y seleccionar los materiales

publicados que se necesitarán. También será necesario definir los estándares técnicos, por lo que tanto diseñadores gráficos como programadores y expertos en herramientas multimediales deberán trabajar en conjunto para añadir el máximo valor agregado posible al producto.

4. Prototipo.

Será necesario para comprobar que las pautas se aplican correctamente, o bien se utilizará como propuesta para definir pautas nuevas. El prototipo sirve de prueba para garantizar el buen desarrollo del programa, detectar errores técnicos y/o conceptuales y proponer nuevos elementos de mejora.

5. Implementación.

Se realiza la formalización del encargo y la recepción del material implementado y revisado debidamente. También se proporcionan las guías, pautas e indicaciones necesarias para el correcto desarrollo del proyecto.

6. Evaluación.

Es la fase final de validación del proyecto y servirá a la vez para realimentar el diseño formativo de forma global, con la finalidad de recoger las experiencias y proponer nuevos elementos de mejora, de desarrollo y de evolución en los recursos, en las estrategias y en los enfoques que se hayan aplicado.

Se puede completar también con una encuesta entre profesores y alumnos, obteniendo valoraciones de los contenidos, actividades propuestas, dinámica del producto y sentimientos generales de satisfacción con respecto a su utilización.

CAPÍTULO 2. PROGRAMACIÓN.

La programación es un Arte*... el arte de hacer exhaustiva y explícita la solución de un problema. En este sentido requiere de talento y creatividad. La Programación es también una actividad, es un trabajo práctico que se sirve de una serie de técnicas y herramientas con un objetivo particular y diferente cada vez, que requiere de mucha práctica y paciencia. Es una actividad difícil... implica saber escuchar, interpretar, analizar, encontrar la mejor manera de resolver un problema. Muchas formas correctas, desde luego, pero de mayor o menor complejidad, de mayor o menor elegancia, de mayor o menor rapidez, de mayor o menor creatividad...*" [17].

Más allá de esta elocuente declaración, y en términos más formales, se podría definir a la programación como "*el proceso de diseñar, codificar, depurar y mantener programas computacionales, escritos en un lenguaje de programación*" (Mongui Naranjo, 2010).

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

- El desarrollo lógico para resolver un problema en particular.
- Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).
- Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.
- Prueba, depuración y documentación del programa.

Como toda disciplina curricular, **la programación implica destreza, conocimientos y dominio sobre el objeto de estudio**, y también requiere un análisis global y objetivo de cada proyecto.

Para evitar una futura generación de programadores monótonos, desinteresados y opacos, cuya única función sea la de "copiar y pegar" -en referencia a la acción reutilizar código obsoleto- , se debe presentar la asignatura Programación dentro de las disciplinas artísticas, induciendo –y por supuesto, enseñándole- a los alumnos a utilizar la imaginación para encontrar soluciones creativas y eficaces para cada problema dado.

```
/*
    Para evitar una futura generación de programadores
    'copy-paste', la programación requiere de compromiso,
    práctica, dedicación y, por supuesto, pasión para
    comprender y razonar el diseño de algoritmos.
*/
```

Al igual que sucede con todas las otras disciplinas universitarias, la Programación requiere de compromiso, práctica, dedicación y por supuesto, pasión. Por lo tanto, estudiaremos y analizaremos a los algoritmos, puesto que son considerados como el corazón de la computación (Bowman, 1999).

2.1. ALGORÍTMICA. PENSAR LA PROGRAMACIÓN.

La palabra algoritmo toma su nombre de Abu Abdullah Muhammad Bin Musa, quien tomó como seudónimo Al-khôwarizmi (780-850), nombre de su ciudad y uno de los centros de saber y cultura de Asia Central en la Edad Media. Fue un matemático y astrónomo del siglo IX que escribió un tratado sobre manipulación de números y ecuaciones, "***el Kitab al-jabr w´almugabala***", que se usó en gran medida como la base de lo que hoy se conoce como algoritmo [8].

Como sus estudios eran de fácil comprensión, su principal valor no fue el de crear nuevas corrientes de pensamiento, sino simplificar las matemáticas a un nivel comprensible para todo el público. Señaló muchas virtudes del sistema decimal indio -contrario al sistema tradicional árabe-, y explico también que mediante una especificación clara y concisa de cómo calcular sistemáticamente se podrían definir algoritmos que fueran usados en dispositivos mecánicos en vez de las manos, dando lugar a la posterior creación de los ábacos, y las primeras computadoras [11].

Un algoritmo es una secuencia finita bien definida de tareas concretas, cada una de las cuales se pueden realizar con una cantidad de recursos finitos. Se dice que una tarea está "bien definida" cuando se sabe de manera precisa las acciones requeridas para su realización. Aunque los recursos que debe utilizar cada tarea deben ser finitos, estos no están limitados, es decir, si una tarea requiere una cantidad inmensa -pero finita- de algún recurso para su realización, dicha tarea puede tranquilamente formar parte de un algoritmo. Además, se dice que una secuencia de tareas está "*bien definida*" si se sabe el orden exacto de ejecución de cada una de ellas. Por lo tanto se concluye que **un algoritmo es una serie de instrucciones que ejecutan una tarea determinada en un lapso de tiempo finito**, y que culmina después de ejecutar la última instrucción.

Ejecutar un algoritmo es realizar las tareas indicadas por el mismo, en el orden especificado, y utilizando los recursos disponibles. Dada una cantidad de datos de entrada de un algoritmo, se dice que la cantidad de un recurso usado por dicho algoritmo para su ejecución determina la complejidad del algoritmo respecto a tal recurso. Cuando se implementa un algoritmo en un computador, los recursos con los que se cuenta son tiempo de proceso y memoria.

Por lo tanto, a un algoritmo implementado se le pueden calcular sus complejidades temporales y espaciales.

En resumen, todo algoritmo debe cumplir en forma eficiente estas cinco condiciones esenciales [2]:

- **Finitud**

  El algoritmo debe acabar tras un número finito de pasos. Es más, es casi fundamental que sea en un número razonable de pasos para no generar códigos muy extensos que perjudiquen la performance de un programa.

- **Definibilidad**

  El algoritmo debe definirse de forma precisa para cada paso, es decir, hay que evitar toda ambigüedad al definir cada paso. Puesto que el lenguaje humano es impreciso, los algoritmos se expresan mediante un lenguaje formal, ya sea matemático o de programación para un computador.

- **Entrada**

  El algoritmo tendrá cero o más entradas, es decir, cantidades dadas antes de empezar el algoritmo. Estas cantidades pertenecen además a conjuntos especificados de objetos. Por ejemplo, pueden ser cadenas de caracteres, enteros, naturales, fraccionarios, etc. Se trata siempre de cantidades representativas del mundo real expresadas de tal forma que sean aptas para su interpretación por el computador.

- **Salida**

  Son los resultados arrojados por el proceso al finalizar la ejecución de todas las instrucciones. El algoritmo tiene una o más salidas, en relación con las entradas.

- **Efectividad**

  Se entiende por esto que una persona sea capaz de realizar el algoritmo de modo exacto y sin ayuda de una máquina en un lapso de tiempo finito.

A menudo los algoritmos requieren una organización bastante compleja de los datos, y es por tanto necesario un estudio previo de las estructuras de datos fundamentales. El uso de estructuras de datos adecuadas pueden hacer trivial el diseño de un algoritmo, o un algoritmo muy complejo puede usar estructuras de datos muy simples [8].

/*
Un algoritmo es un conjunto prescrito de instrucciones bien definidas, ordenadas y finitas que permite realizar una actividad mediante un estado inicial para obtener una solución concreta.
*/

2.1.1. TIPOS DE ALGORITMOS

Existen dos tipos de algoritmos, llamados así por su naturaleza: **cuantitativos y cualitativos**.

Los cuantitativos son aquellos en los que se utilizan cálculos numéricos para definir los pasos del proceso.

Los cualitativos son aquellos en los que describen los pasos utilizando palabras.

Entre los distintos tipos de algoritmos cualitativos se pueden encontrar:

- **Diagramas de flujo.**

  Utiliza símbolos gráficos para su resolución y pueden resultar fácilmente identificables para los alumnos. Sin embargo presentan los siguientes problemas:

  1. Cualquier mínima modificación en el diagrama nos obliga a reorganizarlo de nuevo.

  2. Utiliza una técnica lineal, en desuso en día.

  3. El proceso de recorrer el diagrama desde el principio al final puede resultar complejo y propicia la omisión de una cierta combinación poco frecuente, pero posible, de condiciones bajo las cuales el algoritmo no arroja el resultado esperado.

- **Tablas de decisión.**

  Tabulan todas las posibles situaciones que se pueden presentar en el programa, y las correspondientes acciones para cada una de ellas.

- **Pseudocódigo.**

  Describe un algoritmo utilizando una mezcla de frases en lenguaje común - generalmente en el mismo idioma verbal de los estudiantes-, instrucciones de lenguaje de programación y palabras claves que definen las estructuras básicas de control.

  Los conceptos básicos de pseudocódigo, así como su importancia en el aprendizaje de la programación multimedial, se desarrollarán en profundidad en los capítulos 2.2.2 y 2.2.3.

2.1.2. LENGUAJES DE PROGRAMACIÓN.

Los lenguajes de programación se utilizan para escribir programas. Los programas de las computadoras modernas constan de secuencias de instrucciones que se codifican como secuencias de dígitos numéricos que podrán entender dichas computadoras. El sistema de codificación se conoce como lenguaje máquina – o lenguaje de bajo nivel- que es el lenguaje nativo de una computadora.

Como la escritura de programas en lenguaje máquina es una tarea difícil, ya que sus instrucciones son secuencias de 0 y 1 heredadas del sistema de pulsos eléctricos -patrones de bits, tales como 11110000, 01110011, etc. - que son muy difíciles de recordar y manipular por las personas, **se necesitan lenguajes de programación amigables que permitan escribir los programas para enviar instrucciones inteligibles a las computadoras**. Sin embargo, las computadoras "*sólo entienden las instrucciones en lenguaje máquina, por lo que será preciso traducir los programas resultantes a lenguajes de máquina antes de que puedan ser ejecutadas por ellas*" (Aguilar, 2005).

Los lenguajes de programación de alto nivel han sido definidos como una solución intermedia entre los lenguajes naturales humanos y los lenguajes máquina de los computadores. Están bien definidos, en el sentido de que la tarea que se puede expresar con ellos no es ambigua y por lo tanto pueden ser traducidos en un programa máquina concreto, de forma automatizada y por el propio computador que va a realizar la tarea.

Existen muchos lenguajes de alto nivel de propósito general, sus principales diferencias se encuentran en que poseen un conjunto de órdenes más adecuado para expresar tareas de un tipo concreto de problema o porque corresponden a distintos niveles de evolución de los computadores.

Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o verbal. Una característica relevante de los lenguajes de programación es precisamente que más de un programador pueda usar un conjunto común de instrucciones que sean comprendidas entre ellos para realizar la construcción del programa de forma colaborativa sin perder la calidad final del programa [12].

2.2. MODELOS DE ENSEÑANZA DE LA PROGRAMACIÓN.

Según un artículo publicado por la Asociación Colombiana de Facultades de Ingeniería (ACOFI), las alternativas que se discuten para la enseñanza de la programación son: programación estructurada, programación orientada a objetos (POO), o programación funcional. Los dos primeros casos forman parte del paradigma imperativo, mientras que el último, del modelo declarativo (Timarán Pereira; Jiménez Toledo; Checa Mora; Ordóñez Erazo y Colunge, 2009).

Allí se indica que en la Universidad Mariana de Colombia se viene utilizando el modelo imperativo, comenzando con programación estructurada y continuando con programación orientada a objetos, utilizando lenguajes como C y Java. Se ha encontrado que en promedio el 34% de los estudiantes reprueban las materias relacionadas con el área de programación, y dentro de los que aprueban, el 70% no ha desarrollado la lógica necesaria para dar una solución computacional a un problema específico.

```
/*
   En la Universidad Mariana de Colombia, el 34% de
   los estudiantes reprueban las materias relacionadas con
   la programación, y de los que aprueban, el 70% no ha
   desarrollado los conceptos esperados.
*/
```

El paradigma imperativo se caracteriza por escribir los programas mediante sentencias que manipulan variables de memoria y cambian el estado del programa. Es decir, son conjuntos de instrucciones que le indican al computador cómo realizar una tarea. Se dice que la programación imperativa se centra en cómo encontrar la solución a un determinado problema.

El problema de este modelo consiste en que generalmente el estudiante desarrolla estructuras mentales muy secuenciales, y, **está más enfocado en el conocimiento del lenguaje que en la solución del problema dado**. Comprender adecuadamente el problema y diseñar una solución acorde exige conocimiento y creatividad, mientras que la codificación en un lenguaje de programación es un proceso mecánico (Timarán Pereira, Jiménez Toledo, Checa Mora, Ordóñez Erazo y Colunge, 2009).

Por lo tanto, implementar este modelo de programación en las materias de introducción, utilizando a la vez lenguajes de programación profesionales, es enseñarles a los alumnos a preocuparse por la implementación del problema, más que en su solución. Se los estructura con complejos modelos de datos y lenguajes secuenciales que sólo los distrae del objetivo concreto del curso: razonar en la lógica de los algoritmos y en las soluciones más funcionales.

Por otra parte, el modelo declarativo es un paradigma de programación de alto nivel basado en el desarrollo de programas especificando un conjunto de condiciones y ecuaciones que describen el problema y detallan su solución. Se suele pensar que los lenguajes declarativos tienen la ventaja de ser razonados matemáticamente, lo que permite el uso de mecanismos matemáticos para optimizar el rendimiento de los programas [15].

Son fiables, elegantes y expresivos, y facilitan la escritura de relaciones y funciones.

Están considerados como lenguajes "*de más alto nivel*", ya que el programador trabaja con conceptos en sí, y no con representaciones abstractas de memoria física.

Entre las principales diferencias con respecto al paradigma imperativo, en el modelo declarativo se cuentan [16]:

- Los lenguajes declarativos están orientados a buscar la solución del problema, y no tanto en la forma de conseguirla.

- Los programas están formados por un conjunto de definiciones o ecuaciones, las cuales describen lo que debe ser calculado, no en si la forma de hacerlo.

Sin embargo, la programación funcional tiene una negativa serie de contraindicaciones muy importantes:

- **Mayor dificultad inicial.**

  La gran variedad de conceptos complejos a tener en cuenta hace difícil dominar el lenguaje. La ausencia de variables de estado dificulta enormemente su comprensión, especialmente en los niveles iniciales del aprendizaje donde la abstracción del problema resulta el mayor desafío para los alumnos.

- **Falta de recursos.**

  Al estar tan poco extendidos, falta gran cantidad de recursos -librerías, frameworks, tutoriales, etc.-. Si bien parece algo trivial, la realidad demuestra que el alumno multimedia tiene muy arraigada la tarea de buscar en internet ejemplos e instructivos del tema de estudio, y el no contar con esa enorme ayuda puede provocarle sentimientos negativos y frustraciones hacia la materia y su aprendizaje.

Por lo tanto, estas características negativas hacen que el paradigma funcional pierda terreno ante el modelo imperativo, tan vasto de opciones y documentación, y deja el juego abierto a dos grandes grupos: los lenguajes de programación profesionales y los lenguajes de programación académicos, ambos pertenecientes al modelo estructurado.

2.2.1. ENSEÑANZA A TRAVÉS DE LENGUAJES PROFESIONALES.

Este modelo de enseñanza es el más difundido y aceptado entre las instituciones educativas de la actualidad, ya que son muchos los profesores que se deciden por un lenguaje de programación comercial y profesional como herramienta de enseñanza en los primeros años de las carreras tecnológicas.

Si bien la mayoría los eligen por una cuestión de demanda y urgencias del mercado, otros como el profesor Alejandro Piscitelli lo deciden por fines meramente científicos. Él mismo indica su predilección a los lenguajes complejos, diciendo que "*por suerte el universo computacional hace rato que, a través de ejercicios como la programación orientada hacia objetos, ha cambiado nuestra visión ingenua de la programación —casi ningún programa interesante de hoy sigue los lineamientos secuenciales del Basic—, favoreciendo la organización de los objetos de aprendizaje y sus propiedades, en círculos concéntricos o espiralados en expansión permanente*" (Piscitelli, 2010).

Sin entrar en mayores detalles sobre los distintos lenguajes de programación, puede decirse para simplificar que **la programación puede ser definida en dos partes esenciales: la tecnología y su fundamento científico**. La tecnología consiste en las herramientas, lenguajes, técnicas prácticas y estándares que permiten hacer un programa. El fundamento científico consiste en la parte teórica permitiendo entender la programación.

Enseñar programación correctamente implica enseñar ambas partes: tecnología -herramientas actuales- y ciencia -conceptos fundamentales-. Conocer las herramientas prepara al estudiante para el presente, y conocer los conceptos lo prepara para la evolución futura (Pérez, López, 2007).

Las instituciones que deciden utilizar lenguajes de programación profesionales dentro del currículo lo hacen en relación a un punto importante: la profesionalización. Si bien reconocen que podrían dedicarle más tiempo al aprendizaje profundo, ellos **privilegian el rápido acceso a la información profesional y una pronta salida laboral**. Infieren que el mercado actual no posibilita las chances de que un alumno se desarrolle profundamente sólo con los contenidos académicos, y apoyan la idea que la profundización y asimilación de los contenidos dados se realizará a futuro, cuando el alumno se encuentre gozando de su vida laboral. Por lo tanto, ratifican su decisión de prepararlos con los lenguajes profesionales que utilizarán luego en sus trabajos, dandoles la oportunidad de conocer de antemano con qué se encontrarán fuera de la universidad.

Algunos de los lenguajes que se utilizan en la actualidad como parte del curso introductorio en la programación en la mayoría de las instituciones educativas son:

- **JavaScript**

  Es un lenguaje de programación utilizado para crear programas encargados de realizar acciones dentro del ámbito del nevagador web. JavaScript permite crear efectos especiales y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones JavaScript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

JavaScript es el siguiente paso, después del HTML, que puede dar un programador de la web que decida mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de programación bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica.

- **C / C#**

Es un lenguaje de propósito general, uno de los más rápidos y potentes que existen, e incluso ha demostrado ser extremadamente eficaz, tanto como para crear sistemas operativos como Linux, que fue creado bajo este lenguaje, y tiene las siguientes características:

1. **Es un lenguaje Compilado.**

A diferencia de otros lenguajes, que son lenguajes interpretados, los cuales necesitan del código fuente para funcionar -por ejemplo Basic-, C es un lenguaje compilado esto quiere decir que convierte el código fuente en un archivo independiente, que luego es enlazado con las librerías necesarias dando lugar a un archivo ejecutable.

2. **Es un lenguaje de Nivel medio.**

Esto quiere decir que combina elementos de lenguaje de alto nivel con la funcionalidad de los lenguajes de bajo nivel. Es decir, trabaja a un nivel cercano al computador, y sin embargo ofrece las posibilidades de construir estructuras de datos sofisticadas, equivalentes a los que manejan los lenguajes de alto nivel.

3. **Es un lenguaje Estructurado.**

Esto quiere decir que permite crear procedimientos en bloques dentro de otros procedimientos, y trabajar bajo conceptos de abstracción y tipos abstractos de datos (TAD).

4. **Es un lenguaje Portable.**

Este lenguaje permite utilizar el mismo código en diferentes equipos y sistemas informáticos, o sea que es independiente de la arquitectura de cualquier hardware.

5. **Es un lenguaje relativamente pequeño.**

Este es un lenguaje económico en cuanto a expresiones se refiere, se puede describir en poco espacio y es fácil y rápido de aprender.

6. **Tiene abundancia de Operadores y Tipos de Datos.**

Este lenguaje prácticamente posee un operador para cada una de las posibles operaciones en código de máquina.

- **Pascal**

Es un lenguaje de programación de alto nivel y de propósito general que ha derivado del ***ALGOL-60*** y fue diseñado para enseñar técnicas de programación estructurada. Sin embargo con el tiempo su utilización excedió el ámbito académico para convertirse en una herramienta para la creación de aplicaciones de todo tipo.

Es de alto nivel porque su repertorio de instrucciones lo hacen próximo a los lenguajes humanos y a los procesos humanos de pensamiento. Sus instrucciones o sentencias se componen de expresiones de apariencia algebraica y de ciertas palabras inglesas como *BEGIN, END, WRITE, IF, THEN, REPEAT, WHILE, DO*, etc, que permiten manipular la ejecución del código.

A diferencia de otros lenguajes, Pascal contiene algunos rasgos singulares que han sido diseñados para estimular el uso de la programación estructurada, logrando un entorno ordenado y disciplinado de la programación que conduce a la obtención de programas claros, eficientes y libres de errores.

Con este lenguaje, no sólo se dispone de un lenguaje de programación, sino que además se adquiere una metodología para el diseño y escritura de programas. Por ello, Pascal se utiliza ampliamente en la enseñanza de la informática.

- **ActionScript 3**

Es un lenguaje de programación asociado al software Adobe Flash, y con el paso del tiempo ha ido variando y mutando notablemente desde su concepción original. A medida que ha pasado el tiempo, el lenguaje se ha hecho más robusto, versátil y eficiente.

ActionScript 3 es el lenguaje de programación que utiliza Flash desde su versión CS3 en adelante. Aunque se trata de un lenguaje orientado a objetos (POO), inicialmente ActionScript era un lenguaje orientado a eventos, pero poco a poco y debido al potencial de la orientación a objetos esto ha ido cambiando y mejorando. Con él podemos estructurar el código de nuestras aplicaciones de forma más clara y sencilla, facilitando la búsqueda de errores y la lectura del programa, además de realizar programas más complejos y dinámicos.

2.2.2. ENSEÑANZA A TRAVÉS DE LENGUAJES ACADÉMICOS.

Por peculiar que suene, en comparación con el capítulo anterior, es menor el número de docentes que deciden volcarse a un lenguaje puramente académico, que tiene por único fin contribuir a la educación del alumno. Sin embargo, en todos los casos, se ha encontrado una clara predilección por utilizar un lenguaje de pseudocódigo como motor del aprendizaje.

En esta línea, el profesor madrileño Miguel Ángel Rodríguez Almeida, indica que el pseudocódigo "***intenta sentar las bases de la programación estructurada para todas aquellas personas que quieran aprender a programar***, *con el fin de que por sí mismo sea capaz de resolver cualquier problema que se le pueda presentar. En definitiva, aprende a pensar desde el punto de vista de la Informática*" (Rodríguez Almeida, 1991).

Por su parte, el profesor Nahuel Gonzalez, de la Universidad Nacional Tecnológica (UTN), afirma que "*el pseudocódigo es beneficioso para el aprendizaje porque permite al alumno pensar el problema en términos que él puede conocer de la vida cotidiana*".

```
/*
    "El pseudocódigo intenta sentar las bases de la
    programación estructurada para aquellos que quieren
    aprender a programar, sin perder el rigor científico de la
    materia". Miguel Ángel Rodríguez Almeida.
*/
```

Ambos profesores están de acuerdo en que, si bien el lenguaje trata de ser lo más sencillo posible, eso no quita en absoluto el rigor científico que requiere la materia, y hasta permite crear sólidas bases para todas aquellas personas que quieren aprender a programar, y que deseen luego especializarse en un lenguaje comercial determinado.

- **Pseudocódigo**

  Considerado como un lenguaje falso, el pseudocódigo es un lenguaje intermedio entre nuestro lenguaje verbal y el lenguaje simple de programación de computadoras, debido a que quien lo utiliza se guía por una serie de normas estructuradas, pero sin llegar a usar una estructura tan rígida como la del lenguaje de programación tradicional. Su aplicación didáctica dentro de esta investigación se desarrollará con mayor profundidad en el capítulo 4.2.

  Su objetivo es que quien lo pone en práctica se centre más en la solución del algoritmo o el diseño de un software que en el programa que utiliza para crearlo. Y esto es posible porque es más fácil de manipular ya que no tiene que tener en mente el lenguaje en sí y además, más fácil de codificar. Por lo tanto, parecería ser el más indicado para aquellas personas que deseen comenzar a programar desde cero.

  De esta manera, al ser un lenguaje intermedio, no tiene una composición estandarizada por lo que no todos los programadores utilizan la misma sintaxis con exactitud. Pero a la vez, como es una herramienta que está un paso previo al lenguaje formal de programación, es fácil de transformar al que será ejecutado en la computadora.

- **LPP**

  Este lenguaje de programación fue creado como proyecto de graduación del Ingeniero Iván Deras. LPP es un lenguaje de programación para principiantes, el cual fue diseñado con la idea de facilitar el proceso de enseñanza-aprendizaje de un lenguaje de programación en pseudocódigo totalmente en idioma español, que contiene la mayoría de instrucciones que tienen los lenguajes de programación.

- **PseInt**

  En resumidas cuentas, se trata de un intérprete de pseudocódigo de programación que contiene las instrucciones más básicas necesarias para aprender a programar, crear cualquier programa y está completamente en español.

  Es ideal para alguien que no sabe absolutamente nada de programación, pues a través de este programa, comenzará a entender (y a experimentar) los conceptos más básicos como "algoritmo", "iteración", "condición", etc. Además, cuenta con varias características muy interesantes, entre ellas, la capacidad de generar un diagrama de flujo del código que se está escribiendo, lo cual sin duda facilita muchísimo el aprendizaje y nos ayuda a visualizar mejor el funcionamiento del programa.

  Lo dice su creador, Pablo Novara: "*PSeInt está pensado para asistir a los estudiantes que se inician en la construcción de programas o algoritmos computacionales. El pseudocódigo se suele utilizar como primer contacto para introducir conceptos básicos como el uso de estructuras de control, expresiones, variables, etc., sin tener que lidiar con las particularidades de la sintaxis de un lenguaje real. Este software pretende facilitarle al principiante la tarea de escribir algoritmos en este pseudolenguaje presentando un conjunto de ayudas y asistencias, y brindarle además algunas herramientas adicionales que le ayuden a encontrar errores y comprender la lógica de los algoritmos.*"

- **SL Pseudocódigo**

  Es un lenguaje diseñado para apoyar la formación profesional de estudiantes de Informática. Las construcciones del lenguaje fueron cuidadosamente seleccionadas para que el alumno se concentre en la búsqueda de las soluciones algorítmica

apropiadas, obviando detalles de implementación que seguramente tendrá ocasión de estudiar en otras etapas de su aprendizaje.

El lenguaje presenta características que lo hacen apropiado para expresar algoritmos de las etapas iniciales del aprendizaje, pero simultáneamente reúne un rico conjunto de construcciones que posibilitan el tratamiento de tópicos más avanzados de estructuras de datos y programación modular.

SL está en uso desde hace ya unos años en varias universidades del Paraguay (entre ellas UNA y UCA) como lenguaje de introducción a la programación.

- **Code Academy**

Es una novedosa y popular plataforma interactiva online que le ofrece al usuario un curso interactivo y gratuito para poder aprender a programar diversos lenguajes, como JavaScript, PHP, Ruby, etc. Cuenta con varios temas, distribuidos desde un nivel muy básico a uno avanzado, en los que se irán completando los niveles, según se resuelvan correctamente los ejercicios propuestos en cada uno.

Las ventajas de este portal son muchas: resulta entretenido aprender un lenguaje de programación aquí e incluso tiene la capacidad de atraer al usuario para avanzar en las lecciones.

Por otro lado, no hay que instalar nada para comenzar, ya que su acceso es a través de internet. Simplemente se siguen las lecciones y si el usuario desea guardar su progreso, puede registrarse de manera gratuita para almacenar sus datos, revisar sus lecciones, y consultar todos los “trofeos” que fueron ganando con el aprendizaje.

- **Visual Da Vinci**

  Es un lenguaje de pseudocódigo de alto nivel, con ambiente visual destinado a la enseñanza de la programación en el curso de ingreso y en el primer año de las carreras de Informática en la Universidad Nacional de La Plata.

  Este lenguaje presenta varias particularidades que lo hacen muy accesible. La principal, se trata de mover un robot por una cuidad que ta compuesta de 100 calles y 100 avenidas, donde este se podra mover, girar, tomar papeles, depositar papeles, tomar flores, depositar papeles y tambien sobre pasar obstaculos.

  El conjunto de acciones que el robot puede realizar es muy reducido. Cada una de estas acciones corresponde a una instrucción específica y entendible para que la máquina la interprete correctamente, y un significado único, a fin de poder verificar que el resultado final de la tarea se corresponde con lo requerido. De esta manera se desplaza, recoge, deposita, evalúa algunas condiciones sencillas y hasta puede visualizar información.

  Otra particularidad es q la sintaxis se encuentra en castellano, permitiendo asi al programador saber usar las herramientas que brinda este lenguaje.

  Otra punto muy distinguida es la sintaxis, ya que este programa es muy estricto con las mayusculas y los espacios, donde estos serian los casos de mayor error que presentan los programadores.

  Su interfaz está compuesta por una barra de herramientas principal, el mapa de la ciudad por donde se moverá el robot, la ventana de coordenadas y el editor de código.

Las acciones del robot primitivas son:

1. Se mueve una cuadra a la vez.

2. Sólo gira 90 grados en el sentido de las agujas del reloj.

3. Está capacitado para recoger o depositar dos formas de objetos ubicados en las esquinas de la ciudad: flores y papeles. Puede transportarlos en forma ilimitada.

4. Puede realizar cálculos aritméticos simples.

5. Puede informar los resultados obtenidos.

- **Scratch**

Es un entorno de aprendizaje que ayuda, tal como lo promueve su eslogan, a *programar, jugar y crear*.

Es un programa destinado principalmente a los alumnos jóvenes y les permite explorar y experimentar con los conceptos de programación de ordenadores mediante el uso de una sencilla interfaz gráfica. Scratch está escrito en Squeak, una implementación libre de Smalltalk-80. Es un entorno de programación que facilita el aprendizaje autónomo.

Bajo el nombre de Scratch, la filosofía del programa se basa en el conocido juego de Logo, esto significa, apilar bloques para que se relacionen e interactúen una vez en funcionamiento.

En Scratch todos los objetos, gráficos, sonidos y secuencias de comandos pueden ser fácilmente importados a un nuevo programa y combinados en maneras permitiendo a los principiantes a conseguir resultados rápidos y estar motivados para intentar más.

- **Logo**

Logo funciona como un instrumento didáctico que permite a los alumnos, sobre todo a los más pequeños a construir sus conocimientos. Su creador, Seymour Papert, desarrolló Logo en base a las teorías constructivistas de Piaget, colega suyo en la Universidad de Ginebra. De hecho, se cuenta que en una ocasión Piaget dijo que nadie entendía sus ideas tan bien como Papert.

Es una potente herramienta para el desarrollo de los procesos de pensamiento lógico-matemáticos. Para ello, construyó un robot llamado la "*tortuga de Logo*" que permitía a los alumnos resolver problemas.

También provee un ambiente donde los estudiantes asumen el rol de maestros. Y, como maestros, ellos deben:

1. Entender el conocimiento que debe ser enseñado.

2. Planear un método para impartir este conocimiento.

3. Dividir el conocimiento en trozos pequeños y entendibles.

4. Saber cómo comunicar el conocimiento claramente.

5. Establecer este nuevo conocimiento como fundamento para aprendizaje futuro.

6. Estar al tanto y construir sobre el conocimiento que el aprendiz ya posee.

7. Ser receptivo a explorar nuevas ideas mientras van apareciendo.

8. Responder a los malentendidos y errores del aprendiz.

Este proceso de revisión manual de los errores contribuye a que el alumno desarrolle habilidades metacognitivas al poner en práctica procesos de autocorrección.

Los estudiantes logran esto usando Logo así:

1. Experimentando con los comandos de Logo para entenderlos y alcanzar confianza es su uso.

2. Planeando su trabajo y organizándolo sus varios componentes.

3. Escribiendo un conjunto de instrucciones para realizar cada pequeña tarea.

4. Construyendo un programa para realizar todas las tareas en el orden correcto.

5. Evaluando su programa al localizar y corregir errores o reestructurando el método utilizado.

- **PilatoX**

Es un software diseñado para construir y analizar algoritmos. Permite crear algoritmos de programación estructurada a partir de las instrucciones en español dispuestas que para éste propósito. Después de haber ingresado el algoritmo representado por el pseudocódigo, podrá ejecutarlo, analizarlo y depurarlo en un entorno interactivo diseñado para éste fin.

La interfaz gráfica de PilatoX, facilita en gran medida el trabajo con ambientes gráficos de aprendizaje, simula la sintaxis del Turbo Pascal y evita la prueba de escritorio en hojas de papel; lo que conducía a resultados imprecisos y tediosos.

2.2.3. CONCLUSIONES. MERCADO VS. APRENDIZAJE.

Siguiendo los lineamientos anteriores, el debate sigue instalado entre los profesores y estudiosos del tema, y hoy en día continúa suscrito en el ámbito de la educación universitaria en las ciencias de la computación, y más precisamente, en las disciplinas de introducción a la programación, que lleva varios años sin una respuesta clara: ***¿Qué lenguaje es el mejor para aprender a programar?***

La respuesta a este interrogante descubre un enorme universo de variantes y posibilidades que deben tenerse en cuenta, y se va desde el nivel educativo con el que los estudiantes ingresan a las universidades, hasta la profesión que desarrollará el alumno al finalizar sus estudios. Por lo tanto, no será lo mismo enseñar programación a un futuro desarrollador de software y sistemas informáticos, que transmitirle conocimientos a un diseñador multimedial, quien de antemano supone otro tipo de búsqueda más interesada en los fines comunicacionales, interactivos y –por qué no- lúdicos, y no tanto en los aspectos técnicos y funcionales del proyecto.

Por nombrar sólo un ejemplo, mientras en el primer caso se preparará al alumno para desarrollar sistemas contables y administrativos –con todas las condiciones y exigencias invariables que ello implica-, en el segundo caso su tarea estará más ligada a una cuestión artístico-funcional de algún desarrollo interactivo menor –menor en términos de exigencia funcional, no necesariamente de relevancia o alcance tecnológico.

Por tanto, la metodología de enseñanza en la programación de computadoras estará fuertemente ligada a los objetivos académicos de cada caso, y sus funciones deberían suscribirse a la simple absorción y desarrollo de dichos contenidos.

Sin embargo, existen varios puntos de similitud entre todos los grupos de desarrollo. Sin importar la orientación académica, la profesión o las intensiones de cada alumno, es menester encontrar cuáles de las existentes metodologías corresponden a cada caso particular, sin perder de vista todos los factores técnicos y cognitivos que interactúan en cada caso. La búsqueda de estas metodologías, entonces, será la búsqueda del mejor paradigma de programación.

Según un artículo publicado por la Universidad Abierta de Cataluña [1], la mayoría de los institutos educativos de formación multimedial eligen como lenguajes de programación para sus niveles iniciales, en primer medida aquellos que tienen una buena aceptación dentro del mercado, y en segunda medida aquellos que den muestras de su facilidad y/o flexibilidad para con el alumno y los objetivos académicos.

Esto fácilmente hace concluir que la prioridad en la elección la posee el mercado laboral, en contra del aprendizaje efectivo del alumno, quien **deberá someterse a las complejas herramientas del sistema informático que se encuentre de moda en ese momento, atentando directamente contra su propio proceso de formación**, en virtud de generar una fábrica de programadores que lejos de alcanzar altos niveles en calidad, aprehensión y dominio de conceptos, se asemeje más al modus operandi de un *fast-food* de programadores listos para lanzar al mercado antes del próximo receso vacacional.

Esto sin dudas trae profundas consecuencias en el proceso de aprendizaje, adaptación y formación básica del futuro programador, quien se verá desconcertado, perdido e incluso intimidado por la compleja herramienta que se le está imponiendo aprender a urgencias ilógicas, sin caer en cuenta que los software del mercado no están pensados para aprendices, sino justamente para profesores de la programación que sabrán hacer el mejor uso posible de sus indudables cualidades.

Por otro lado, existen lenguajes de programación muy básicos y sencillos, analizadas en el capítulo 2.2.2., que seguramente resultarían más sencillos de comprender y percibir para el estudiante, al contar con interfaces gráficas mucho más simples y sin tantas herramientas que desconcentren o pierdan el punto focal del objeto de estudio. Así, el uso de la multimedia y las nuevas tecnologías está al servicio de la educación, dentro de un entorno pedagógico donde el alumno pueda completar su educación con contenidos y materiales interactivos ricos en nuevas experiencias.

Sin embargo, la vorágine del mercado actual, sumado a las urgencias y necesidades de cada alumno, obliga a los institutos a descartar una mejor educación en función de una rápida salida laboral, premiando con unas pocas monedas toda una nueva generación de programadores mediocres.

Está claro que el mercado se impone, y en cualquier ámbito –no sólo en la programación- un profesional egresado debe cumplir con sus exigencias y destacarse por su capacidad de resolución.

No obstante, **ofrecer el mismo software de desarrollo tanto al principiante como al profesional, es desprestigiarlos a ambos**. El primero, porque se sentirá amenazado, intimidado y rápidamente generará una serie de vicios peligrosos tanto para su profesión, como para la integridad del proyecto que integre. ¿O acaso nunca se oyó decir a un programador “no sé cómo funciona, lo bajé de internet”? El segundo, por otro lado, habrá crecido en su formación con estos vicios tan arraigados en su concepción de la disciplina, que harán tambalear su superficial base en todo momento, acortando no sólo su capacidad de resolver futuros problemas, sino también matando su instinto de curiosidad y experimentación de nuevas y originales soluciones. Un instinto que ningún profesional que se precie de serlo debería perder jamás.

CAPÍTULO 3. MULTIMEDIA.

3.1. CONTEXTUALIZACIÓN HISTÓRICA. CAMBIOS CULTURALES DEL S.XXI.

Al igual que la imprenta del siglo XIV y la fotografía en el XIX tuvieron un impacto revolucionario sobre el desarrollo de la sociedad y la cultura moderna, hoy nos encontramos en medio de una nueva revolución mediática, que supone el desplazamiento de toda la cultura hacia formas de producción, distribución y comunicación mediante el ordenador (Manovich, 2006).

Estos nuevos medios de comunicación representan la convergencia de dos recorridos históricamente separados, como son las tecnologías informáticas y mediáticas. Ambas arrancan en la década de 1830 con la máquina analítica de Babbage y el daguerrotipo de Daguerre. Con el tiempo, a mediados del siglo XX, se desarrolla un moderno ordenador digital que efectúa cálculos más eficaces con datos numéricos, y que sustituye a los numerosos tabuladores y calculadoras mecánicas utilizadas por los gobiernos y grandes empresas desde principios del siglo. El resultado son los nuevos medios: gráficos, imágenes en movimiento, sonidos, formas, espacios y textos digitales, etc.

Dentro de este contexto aparecen las Tecnologías de la Información y la Comunicación (TIC), que hacen referencia al conjunto de avances y desarrollos tecnológicos que favorecen y benefician a la comunicación y el intercambio de información en el mundo actual. Entre las tecnologías que se encuentran principalmente en el foco de crecimiento podemos nombrar a la informática, Internet y las telecomunicaciones.

Los nuevos entornos de aprendizaje que posibilitan las TIC están favoreciendo la incorporación de estudios en modalidad a distancia en la oferta formativa universitaria. Sin embargo, para conseguir una docencia de calidad en estos nuevos entornos **es necesario que el docente posea tanto el dominio técnico como pedagógico para la incorporación de las últimas herramientas disponibles**, y aplicar estrategias para motivar al alumnado (Llorente, 2006).

En este sentido, la actividad formativa apoyada en los entornos digitales debe centrarse en el asesoramiento de los estudiantes desde una doble perspectiva: individual, a través del seguimiento de cada estudiante mediante tutorías virtuales o presenciales, así como mediante el entorno virtual que facilita la gestión y evaluación de los estudiantes; y grupal, fomentando el trabajo colaborativo aprovechando el potencial de algunas herramientas informáticas tales como aplicaciones interactivas, foros, chats, listas de correo, etc.

El empleo de las TIC también promueve la tendencia del desarrollo de la denominada Sociedad de la Información, en la que l**as tecnologías que facilitan la creación, distribución y manipulación de la información juegan un papel primordial en las actividades sociales, culturales y económicas**. Este tipo de sociedad es llamada también Sociedad del conocimiento, Sociedad del aprendizaje y Sociedad de la inteligencia. (Marquès Graells, 2000).

Esta sociedad se ve caracterizada por dos sucesos interrelacionados, como lo son por un lado los ya mencionados avances tecnológicos, y por el otro, la globalización. De esta forma, la rapidez y el caudal de la recepción de la información aumentan sin cesar y de forma exponencial, de una manera que no está equilibrada con el ritmo de pensamiento y de comprensión existente en la naturaleza humana, como así también en los distintos estratos y grupos sociales, conllevando en muchos casos que la capacidad de innovación tecnológica vaya un paso adelante de la valoración de sus riesgos y repercusiones sociales.

3.1.1. LA MULTIMEDIA COMO HERRAMIENTA INTEGRADORA.

El Cambridge International Dictionary define la multimedia como "*el uso de una combinación de imágenes estáticas y móviles, sonido, música y palabras, especialmente en ordenadores o entretenimiento*". En 1994, Tony Feldman, en su libro titulado Multimedia, describía el concepto nuclear de su obra como "*una integración sin fisuras de datos, texto, imágenes de todo tipo y sonido en un único entorno digital de información*" (Feldman, 1994).

La multimedia pasa entonces a constituir una parte importante de la formación de los estudiantes, convirtiéndose en un entorno rica en experiencia y potencial, donde el estudiante puede desarrollar su aprendizaje con perfecta sintonía con sus objetivos académicos y personales. Con el uso de la multimedia, los estudiantes poseen mayor autonomía y participación en el proceso formativo por lo que deben ser capaces de adquirir las competencias necesarias para gestionar su propio aprendizaje (Marín y Reche, 2011).

Por otro lado, la multimedia permite generar herramientas comunitarias como los foros virtuales, que cada vez se están configurando más como una poderosa herramienta de comunicación y trabajo colaborativo. Estos espacios de trabajo y diálogo proporcionan la posibilidad de participación de una forma reflexiva, frente a otras herramientas de comunicación y trabajo de carácter sincrónico, donde la inmediatez supone un obstáculo a la reflexión y el análisis.

En base a esta importancia, el papel de coordinador, moderador, desempeñado en muchos casos por el docente, cobra un papel de gran relevancia, pues será quién reconduzca, haga reflexionar, proponga nuevas orientaciones, etc., dentro del espacio del foro. Así mismo, se tratará de ofrecer algunas sugerencias o propuestas didácticas para poner en práctica, orientadas sobre todo a la aplicación en las ciencias sociales, pero con posibilidades de adaptación a otros ámbitos educativos [6].

3.1.2. NATIVOS DIGITALES. EL PERFIL DEL ALUMNO MULTIMEDIA.

En la sociedad de la información aparece una nueva forma de cultura, el nacimiento de una nueva generación multimedial, compuesta por los llamados ***nativos digitales*** -jóvenes nacidos a partir de los años '90-, la cual da lugar al surgimiento de nuevas costumbres, nuevos hábitos y cambios culturalmente radicales a los que se venían desarrollando anteriormente. Estos individuos se han desarrollado a la par del creciente avance tecnológico y la aparición de los diversos fenómenos arrolladores como Internet, los cuales ampliaron claramente sus formas de comunicación.

Estos nuevos usuarios enfocan su trabajo, el aprendizaje y los juegos de nuevas formas: absorben rápidamente la información multimedia de imágenes y videos, igual o mejor que si fuera texto; consumen grandes cantidades de datos simultáneamente de múltiples fuentes; esperan respuestas instantáneas; permanecen comunicados permanentemente y crean también sus propios contenidos. Forman parte de **una generación que ha crecido inmersa en las Nuevas Tecnologías**, desarrollándose entre equipos informáticos, videoconsolas y todo tipo de artilugios digitales, convirtiéndose los teléfonos móviles, los videojuegos, Internet, el email y la mensajería instantánea en parte integral de sus vidas y en su realidad tecnológica. Navegan con fluidez; tienen habilidad en el uso del ratón; utilizan reproductores de audio y video digitales a diario; toman fotos digitales que manipulan y comparten entre sus amigos; y usan, además, sus ordenadores para crear videos, presentaciones multimedia, música, blogs, etc. A los nativos digitales les encanta hacer varias cosas al mismo tiempo: son multitarea. Afrontan distintos canales de comunicación simultáneos, prefiriendo los formatos gráficos a los textuales.

Utilizan el acceso hipertextual en vez del lineal. Comparten sus vidas a diario con amigos y desconocidos por igual. Funcionan mejor trabajando en red. Y prefieren los juegos al trabajo serio. Destacan la inmediatez en sus acciones y en la toma de decisiones.

Sin duda, su actividad con la tecnología configura sus nociones sobre lo que es la comunicación, el conocimiento, el aprendizaje e, incluso sus valores personales. Y lejos de ser una moda temporal, los nativos digitales **parecen ser un fenómeno que abarca el conjunto de una generación y que crece firmemente y que llegó para quedarse**. [13].

/*
Los nativos digitales utilizan el acceso hipertextual en vez del lineal. Comparten sus vidas a diario con amigos y desconocidos por igual. Funcionan mejor trabajando en red, prefieren los juegos al trabajo serio, y lejos de ser una moda temporal, han llegado para quedarse.
*/

De esta forma, vemos como los progresos han ido impactando en la sociedad, rompiendo paradigmas y, casualmente, dando lugar al nacimiento de una nueva sociedad, influida por esta nueva cultura popular que cimentaba sus bases en el ilimitado potencial de los medios de comunicación, los nuevos lenguajes y la multiplicidad de las expresiones sociales, incentivando la amplificación de los sentidos y la potenciación del aprendizaje.

Por lo tanto, estamos haciendo frente a una nueva generación con una estructura en continua formación y crecimiento. Varios autores plantean dos posiciones ante el gran cambio:

La primer opción es la posibilidad de fusionar la diversidad cultural actual con las bases de la cultura anterior, donde ésta nueva generación multimedia tiene la responsabilidad de entender y manejar la complejidad del lenguaje, acompañados por los inmigrantes digitales, quienes se ven obligados a descubrir las sutilezas de las narrativas ***transmedias***, a multiplicar las preguntas, a suspender su creencia en los conceptos y conocimientos dentro de los que se formaron y que tanto les ha costado dominar, "***para poder enseñar lo viejo con ojos nuevos***".

La opción restante queda en optar por la aparición de una brecha generacional, donde lamentablemente se perderían las esencias y riquezas que las generaciones fueron construyendo a lo largo del tiempo en cuanto a conocimiento, crecimiento y progreso. Es un arduo trabajo el cual busca la integración de las dos realidades para fomentar el crecimiento, el desarrollo y el progreso por parte de las nuevas tecnologías, las cuales nos provean el correcto uso de los medios de comunicación y explotar las posibilidades que esto conduce, especialmente en la generación y consumo de nuevos e interesantes productos educativos multimediales.

3.2. SITUACIÓN ACTUAL. ANÁLISIS DE CASOS.

"*Las tecnologías tienen la posibilidad de resolver los viejos problemas educativos como las clases superpobladas y la sobrecarga de trabajo de los docentes, en síntesis, según esta perspectiva* ***las tecnologías por sí mismas permitirán mejorar la calidad del sistema educativo en su conjunto*** *(...). A la segunda perspectiva, los autores la denominan 'la computadora como herramienta'. Desde aquí se plantea que las computadoras no son ni buenas ni malas, sino que dependen de los usos que los actores sociales les asignen.*

Así, toda la responsabilidad otorgada en la primera perspectiva a las tecnologías se traslada a las personas (...) Pero la provisión de computadoras no es suficiente, también la inversión debe tender hacia el desarrollo de nuevos entornos de enseñanza y aprendizaje que permitan aprovechar las posibilidades que brindan las tecnologías de la información y la comunicación." (Landau, 2001).

Aunque aquí se refiera al modelo educacional proyectado por el Estado argentino para la educación primaria, es muy interesante la observación que se refiere al hecho de generar nuevos contenidos ricos en materiales pedagógicos, y el no asignarle a la tecnología la responsabilidad intrínseca de educar a los alumnos por el mero hecho de ser algo novedoso e intangible.

Por el contrario, el Ministerio de Educación es consciente en su reflexión de la necesidad de proveer a las máquinas con software que potencie las habilidades y disciplinas planteadas en clase, no como un substituto del modelo educativo actual, sino como un poderoso complemento que pueda continuar la labor pedagógica desde otro ángulo, al cual no se puede acceder con los métodos tradicionales de enseñanza.

Siguiendo este camino, y sólo por nombrar un ejemplo cercano que a su vez es pionera en el país, la Universidad Maimónides mantiene una metodología concreta en la formación de profesionales en la Multimedia: la correcta aprehensión de los contenidos formativos en los alumnos se da exclusivamente por la implementación en paralelo de las cuatro áreas en que dividen las asignaturas –diseño, programación, comunicación y marketing, y sólo mediante su completa integración a lo largo del tiempo y los proyectos realizados, se dan bien los profesionales multimediales en su máximo esplendor, manteniendo un completo dominio sobre todas las ciencias en que se compone la multimedia en su función de comunicadora interactiva.

A lo largo de los cuatro años que dura la carrera, el alumno adquiere sólidos conocimientos de Diseño, Comunicación, Programación, Animación, Fotografía tradicional y digital, Edición de imagen y sonido, Arte Digital, Historia del Arte, Metodología de la Investigación, Dirección de Proyectos, Marketing, Investigación de Mercados y Dirección Empresarial, todo ello en un ámbito interdisciplinario. Para ello, además de un plantel docente formado por los mejores especialistas en cada materia, cuenta con máquinas, equipos e instalaciones de última generación que se renuevan constantemente.

En su sitio web, la Universidad Maimónides ilustra que "*la gran demanda de personal idóneo que exige en la actualidad el mercado multimedial no se cubre ni siquiera en forma parcial, no sólo por las características propias de la actividad, sino también, por la carencia*

casi absoluta de una sólida formación metodológica del escaso personal especializado disponible. La Licenciatura en Comunicación Multimedial, concebida en forma específica para cubrir esta carencia del mercado, tiene como objetivo primordial formar profesionales con sólidos conocimientos en el campo de la integración de equipos y programas, a fin de lograr un importante dominio de la actividad que desarrollan los diferentes especialistas que participan en el desarrollo de productos multimediales e interactivos." [22]

3.2.1. PLANES DE ESTUDIOS DE PROGRAMACIÓN MULTIMEDIAL.

A continuación se describen algunas de las universidades latinoamericanas que en actualidad ofrecen la multimedial, o la programación de computadoras, como carreras oficiales de grado:

Licenciatura en Tecnología Multimedial - Universidad Maimónides

La carrera está dividida en cuatro áreas bien definidas: Creatividad: dedicada a Software Creativo y Arte Digital. Diseño: inicia con los Fundamentos del Diseño tradicional para luego trabajar sobre diseño de interfaces y luego diseño de páginas web y CDs. Imagen y Sonido: se capacita al alumno en la obtención y tratamiento de imágenes tradicional y digital. Luego las Técnicas Audiovisuales lo forman en el área de video y su edición para abordar temas referidos a especializaciones de audio. Programación: inicia con Introducción a la Programación Multimedial, donde se trabaja sobre programación orientada a objetos, para profundizar luego en programación web y a partir de tercer año abordar temáticas vinculadas a programación de bases de datos y outsourcing. Negocios: se incluyen materias de Marketing, Investigación de Mercado, Derecho Empresarial y de Recursos Humanos con el objetivo de formar un profesional con un perfil orientado a los negocios, capaz de desarrollar su liderazgo en diferentes contextos organizacionales [7].

Técnico de Nivel Superior Diseñador y Programador Multimedia - Universidad Tecnológica de Chile (INACAP)

Su plan de estudio tiene una duración de 4 semestres académicos con un total de 1.773 horas, distribuidas en 23 asignaturas lectivas y prácticas. Considera una práctica de introducción al mundo del trabajo y como requisito para la obtención del título, un Examen Final de Competencias. La carrera comprende las áreas formativas de: especialidad, conformada por las líneas curriculares de programación e integración, diseño y comunicación, producción y edición; disciplinas básicas, gestión y formación general que promueve valores y competencias para el desarrollo integral, propios del sello del alumno INACAP.

Este técnico recibirá una formación eminentemente práctica, en la que utilizará una variada gama de herramientas de diseño y programación de última generación orientada a la Web, las que aplicará durante toda la carrera a través del desarrollo de proyectos integrales, fomentando en el alumno una actitud organizativa, creativa y emprendedora.[4]

Ingeniería en Sistemas de Información - Universidad Tecnológica Nacional (UTN)

El ingeniero en Sistemas de Información es un profesional capacitado para desarrollar sistemas de ingeniería y tecnología afines a los existentes y producir innovaciones, capaz de analizar y evaluar requerimientos de procesamiento de información y, sobre esta base, diseñar, desarrollar, organizar, implementar y controlar sistemas informáticos al servicio de múltiples necesidades de información de las organizaciones y de todas las profesiones con las que deberá interactuar con versatilidad y vocación de servicio interdisciplinario.[10]

Tecnicatura en Programación - Universidad Nacional de Quilmes (UNQ)

Busca formar técnicos capaces de elucidar e implementar soluciones en un amplio espectro de problemas asociados a las tareas de diseño/programación dentro del desarrollo de software, en un alcance razonable para un egresado/a pre-universitario, siendo capaces de aprovechar los conceptos aprehendidos en la carrera para pensar y resolver situaciones concretas, y basados en una amplia experiencia práctica obtenida durante el recorrido de la carrera. [21]

Tecnicatura en Programación de Sistemas - Universidad de Ciencias Empresariales y Sociales (UCES)

Aporta a los estudiantes las herramientas y metodologías más utilizadas en el mercado, necesarias para incidir eficazmente en el proceso de producción de software. Se complementa con una sólida base teórica y una práctica intensiva en laboratorio que garantizan el aprendizaje de las más modernas técnicas y con un sistema web para el apoyo virtual sobre las clases presenciales, que permite compartir información y extender la relación Alumno-Docente más allá del aula.

De esta manera, UCES busca preparar a los estudiantes en las nuevas tecnologías, para que transformen sus conocimientos en productos y servicios, asegurando su competitividad en el mercado laboral y profesional como capital humano valioso. [20]

3.2.2. PLANES DE ESTUDIO. SIMILITUDES Y DIFERENCIAS.

En este punto se busca establecer un rango de similitudes y diferencias de acuerdo a la visión que cada universidad, de acuerdo a sus propios objetivos y su orientación académica, establecen sobre la Multimedia, y especialmente la Programación.

Es interesante ver que los resultados se dan en tres grupos claramente definidos: las dos primeras universidades hablan de la programación como una parte integradora con el diseño y el arte, tal como lo establece la propia definición de Multimedia previamente analizada. Las universidades tecnológicas (UTN y UNQ) hacen mayor hincapié en el desarrollo intrínseco del software como objetivo esencial de la programación. La última establece sus pretensiones de integrar la programación con el mundo de los negocios, y la rápida inserción comercial. Y, aunque cada grupo tiene su propia visión de lo que representa la programación, todas comparten el interés por formar técnicos capaces de desarrollar programas de computación mediante lenguajes de programación. Lo relevante de esta comparación es concluir en que, de una forma u otra, tanto en un proyecto artístico, comercial, o tecnológico, la programación es una disciplina altamente requerida y necesaria para afrontar los desarrollos y la puesta en marcha del proyecto multimedial en sí.

En conclusión, la tecnología posibilita la creación de nuevos paradigmas educativos en todas las áreas y niveles, desde el preescolar hasta los niveles universitarios y posgrados, siempre y cuando esté acompañada de una sólida herramienta pedagógica con base y fundamentos fuertemente analizados y logrados. **La enseñanza de la programación mediante entornos virtuales permite diseñar e implementar nuevas y originales soluciones que complementen el aprendizaje en forma clara y amena.** Por otro lado, brindarán una herramienta que permita un adecuado seguimiento y control de la evolución personal y grupal de los alumnos. Por lo tanto, es menester comprender cuáles son los factores que hoy en día generan tanto rechazo e indiferencia frente al alumnado, y incluso cómo fomentar su afición y facilitar el proceso de aprehensión en todos los alumnos, tanto los interesados en la materia, como aquellos que traen consigo los prejuicios negativos hacia la programación, visto y considerando que es una herramienta sumamente interesante y esencial en todo tipo de proyecto multimedial.

3.3. VOCACIÓN DEL ALUMNO MULTIMEDIA.

La vocación no se define como algo innato, sino que va desarrollándose en el plano de la acción, el conocimiento y la convivencia (González Maura, 1999). Durante la marcha, se van adquiriendo una serie de experiencias de modo consciente e inconsciente que diariamente convencen -o no- al sujeto de la posibilidad de elegir por si mismo lo que quiere realizar durante su vida, como mejor reflejo de sus anhelos y sueños, para desenvolverse en el espacio social, académico y, tiempo después, profesional.

Además, en este caso también intervienen las denominadas "*experiencias familiares*", que son las expectativas que los padres muestran hacia los hijos, o lo inculcado por medio de situaciones, experiencias o tradiciones en el ámbito familiar.

La psicóloga Alicia Arrillaga indica que estos elementos observables que componen profunda e intrínsecamente a la persona, emergen a lo largo del tiempo y van dando forma a la identidad. Se la puede definir como ***un mero reflejo de nosotros mismos***, como la imagen que se va construyendo con todas aquellas cosas que tomamos, adaptamos y nos apropiamos para luego transformarlas en herramientas útiles, habilidades, aptitudes que nos caracterizan y nos permiten determinar qué nos gusta, en qué nos desempeñamos mejor, cuáles son las características que nos definen.

Es entonces cuando se logra identificar una vocación, aquello que anhelamos y aspiramos a ser; cuando el sujeto encuentra todo aquello con lo que se quiere identificar. Aquí se halla el motor motivacional que impulsa al sujeto a integrarse en la sociedad por medio del ejercicio de lo que se educó y aprendió, y positivamente su actuación participante en el progreso y desarrollo profesional dentro de la comunidad.

Hoy en día, podríamos referirnos claramente a la situación cultural como un factor exponencial y de real importancia a la hora de elegir qué carrera seguir. El impacto de la multimedia en nuestra realidad, generó los cambios suficientes para que surjan nuevos "caminos" y estudios profesionales que ponen en uso y práctica las herramientas tecnológicas, brindando así una nueva forma de educación que pretende integrar los elementos "***para poder enseñar lo viejo con ojos nuevos***" (Piscitelli, 2009). Muchos se adhieren al cambio y la diversidad cultural, mientras tanto otros, prefieren seguir instando por las formas de la cultura analógica.

El alumno con vocación en multimedia, entonces, se encuentra ante un vasto universo de posibilidades, ya que en la actualidad existe una enorme diversidad de oferta académica para la formación de alumnos universitarios. Y entre ellas, se cuentan cada vez en mayor medida aquellas ligadas a la innovación tecnológica, sus herramientas y aplicación en la realidad actual, sumado a los fenómenos que generan cambios en nuestra sociedad y en su comportamiento.

La existencia de un perfil individual pertinente a éste nuevo tipo de vocación multimedial converge también a un perfil colectivo, inmerso en una sociedad tecnológica que influirá en mayor o menor medida en la toma de nuevas decisiones, amparados en el continuo avance de las tecnologías de la información.

Consecuentemente las preferencias y objetivos de cada persona pueden coincidir entre pares que transitan un camino de desarrollo similar, y con esto ha de referirse a que la "toma de decisiones" individualmente en un "sujeto A" con seguridad tiene algún punto en común con la de un "sujeto B".

Dado que la misma integra y utiliza múltiples medios de expresión y comunicación -físicos o digitales- en común para transmitir información.

3.3.1. ENCUESTA ALUMNOS MULTIMEDIA.

A fin de conocer en mayor profundidad la relación de la vocación de los alumnos multimedia con la programación, entre los meses de febrero y marzo de 2014 se realizó una encuesta online para alumnos y ex-alumnos de la carrera Tecnología Multimedial.

Los resultados, obtenidos entre más de sesenta entrevistados, determinaron que **un 67% de alumnos actualmente trabaja, o le gustaría trabajar en un futuro próximo en algún proyecto relacionado con la programación** (FIGURA 1).

FIGURA 1. ¿TRABAJAS O TE GUSTARÍA TRABAJAR EN EL ÁREA DE PROGRAMACIÓN?

Entre las razones de esa decisión, el 47% valoró el alto índice de salida laboral que posee el área, mientras que el 33% habló de pasión y gustos concretos por la disciplina (FIGURA 2).

Con respecto al nivel universitario alcanzado, se puede advertir que el 37% de los alumnos encuestados no se encuentra en condición regular, ya que han finalizado sus estudios pero adeudan materias o finales pendientes (FIGURA 3).

FIGURA 2. ¿QUÉ SIGNIFICA LA PROGRAMACIÓN PARA VOS?

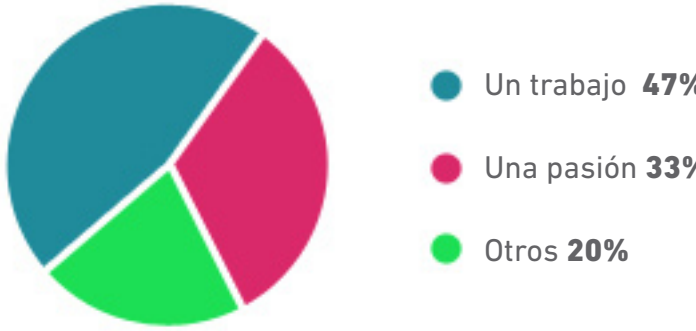
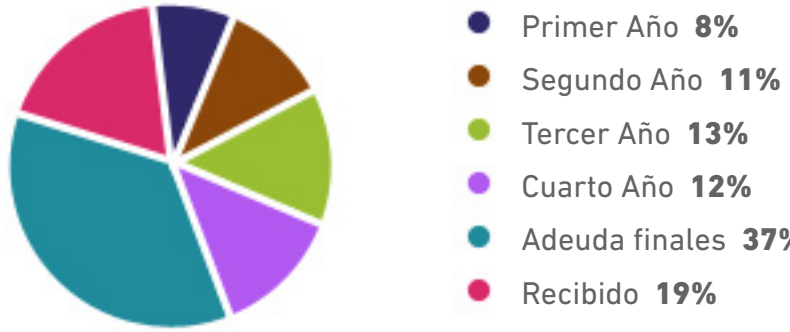


FIGURA 3. ¿QUÉ AÑO ESTÁS CURSANDO ACTUALMENTE?



Por otro lado, entre el 33% de los estudiantes que no trabajarían en programación, se destacan las razones de un alumno de primer año que habló de la programación como "***un trabajo demasiado sofisticado y frio***". En la misma frecuencia, otro alumno de cuarto año admitió que la programación es "*un area de conocimientos muy interesantes (para las que no me da la mente)*".

CAPÍTULO 4. PROPUESTA EDUCATIVA.

Siguiendo la línea pedagógica desarrollada por Duart y Sangrà para el diseño formativo, analizada previamente en el capítulo 1.6.1 de la presente investigación, la siguiente Propuesta Educativa incluye el desarrollo de una herramienta multimedia para el aprendizaje de la programación multimedial, realizada en seis etapas: análisis, diseño y concreción, desarrollo y propuesta, prototipo y evaluación.

Por otro lado, en el capítulo 4.6 se incluye el Plan de Estudios propuesto para la materia Introducción a la Programación, aplicable a cualquier universidad o institución educativa con perfiles artísticos y tecnológicos.

4.1. ANÁLISIS.

Como se manifestó a lo largo de esta investigación, el principal concepto de esta propuesta educativa corresponde con el ***constructivismo***, que como se menciona en el capítulo 1.5, se trata en su postura menos extrema de una tendencia filosófica que asume la realidad como una construcción humana. Probablemente la forma más directa para expresar el sentido del constructivismo fue acuñada por Gregory Bateson cuando dijo que "l*a realidad es cosa de fe*". La frase no nos deja dudas, es la intervención humana la que le otorga existencia, y la experiencia la genera y la define.

La idea de una realidad que está allí, sin depender de nuestra voluntad, no tiene cabida en esta concepción, ya que las cosas existen sólo cuando uno lo experimenta (López Pérez, 2003).

Para el constructivismo, la enseñanza no es una simple transmisión de conocimientos, es en cambio la organización de métodos de apoyo que permitan a los alumnos construir su propio saber (Ramírez Toledo, 2007). No aprendemos sólo registrando en nuestro cerebro, aprendemos construyendo nuestra propia estructura cognitiva.

```
/*
   El profesor tomará al alumno como un colega más,
   destruyendo así la vieja figura vertical de la relación
   profesor-alumno.
*/
```

Por lo tanto, el aprendizaje deberá ser realizado por y para el alumno mismo, con la guía continua de un profesor que tomará al estudiante como un colega más, como alguien con quien compartir una conversación sobre un determinado objeto de estudio, destruyendo así la vieja figura vertical de la relación profesor-alumno, en la que el primero es la autoridad absoluta que le debe transmitir conocimientos al segundo, y éste limitarse únicamente a recibir la información del primero.

Otro concepto tratado es el de ***metacognición***, que por definición se trata de un "*área nueva de estudio que se viene expandiendo con rapidez desde hace años y ha suscitado una inquietud vibrante por diseñar sistemas didácticos para enseñar a los alumnos a hacer del estudio un ejercicio de la inteligencia y no simplemente de la memoria mecánica*" (Burón Orejas, 2002).

Debido a que la presente investigación tiene como objetivo la obtención de una nueva metodología educativa para la formación de futuros programadores multimediales, dentro del análisis de esta propuesta no pueden quedar ajenos ciertos conceptos relativos al ámbito de la programación, y más específicamente, al diseño y desarrollo de los ***algoritmos*** necesarios para cualquier producción multimedial.

Desde el punto de vista pedagógico, la algorítmica ayuda a desarrollar el pensamiento computacional y compromete a los estudiantes en la consideración de varios aspectos importantes para la solución de problemas: decidir sobre la naturaleza del problema, seleccionar una representación que ayude a resolverlo, monitorear sus propios pensamientos y las **diversas estrategias para analizarlo, y realizar las conclusiones pertinentes a cada solución.**

Este tipo de abstracción no sólo mejorará su capacidad de resolver problemas de programación, sino que potenciará su capacidad de resolución de problemas generales, ya sean pedagógicos, profesionales, e incluso personales. No debemos olvidar que solucionar problemas con ayuda del computador puede convertirse en una excelente herramienta para adquirir la costumbre de enfrentar problemas de manera rigurosa y sistemática, evitando las soluciones de memoria como hacían los alumnos de física del capítulo 1.1, aun cuando no se utilice un computador para solucionarlo.

```
/*
   Luego de un maduro proceso bien guiado por los profesores,
   los alumnos generan conocimiento sobre sus propias ideas,
   convirtiéndolas en nuevos saberes.
*/
```

Por lo tanto, estas nuevas teorías basan sus fortalezas en la experiencia de los alumnos como disparadores de nuevas ideas, que luego de un maduro y enfocado proceso bien guiado por los instructores, genere conocimiento propio y se conviertan en nuevos saberes del alumno. Éste conocimiento no será efímero, ocasional o superficial, ni se borrará de la memoria luego de aprobado el examen. Por el contrario, este aprendizaje nacerá de las decisiones propias de cada alumno, y estará impregnado en su ADN, guiando sus decisiones y sus pensamientos de aquí en más.

Por eso, vale sin dudas la pena el esfuerzo por adoptar estas nuevas metodologías educativas que están más enfocadas en la calidad que en la cantidad, puesto que **el criterio de enseñanza debe siempre privilegiar la absorción de pocos conocimientos bien aprendidos**, antes que el simple cumplimiento de un amplio plan de estudios mal enfocado y poco correspondido.

A la hora de realizar la presente Propuesta, se contó también con el importante aporte de varios docentes universitarios de esta materia. Todos los profesores entrevistados dejaron en claro, en forma directa o indirecta, que su principal labor como instructores siempre es generar curiosidad en el alumno, motivándolo a través de resultados que el propio alumno pueda admirar y anhelar.

Tanto Mariano Makedonsky como Fabián Mandelbaum, profesores de programación de primer y segundo año de la Universidad Maimónides, utilizaron la palabra pasión para describir el principal combustible que mueve sus clases. **Pasión para brindar una clase. Pasión para resolver un problema. Pasión para adquirir nuevos conocimientos.**

Por otro lado, todos ellos resaltaron la necesidad, casi obligatoria, de acompañar el ritmo de las clases con una buena práctica no presencial. Esto significa que ***el alumno debe practicar mucho en su casa para mantenerse activo y no perder los contenidos de la clase siguiente***. Darío Saeed, experimentado profesor de primer año, hace una buena analogía sobre este punto, resumiendo que la programación es como una escalera que el alumno debe subir a paso firme, sin saltearse escalones. Por cada clase que falte, o cada trabajo que no entregue, el saltearse esos escalones le requerirá un esfuerzo adicional al alumno, y muchos no estarán preparados para realizarlo. Por ello, continuar el aprendizaje fuera del espacio curricular es indispensable para la correcta fijación de los conocimientos, puesto que al tratarse de contenidos tan abstractos, la enseñanza de la programación requiere que el alumno esté enteramente dispuesto a practicar, equivocarse, y

seguir practicando por su cuenta. Aunque para ello, es forzoso contar con la orientación cercana del docente, que lo guiará a través de todas las soluciones posibles, con el objetivo de presentarle la mejor.

Siguiendo ese camino, y aunque se traten de disciplinas diferentes, el profesor de videojuegos Joaquín Ungaretti planea sus clases presentando primero la teoría, y luego dando una evidencia manifiesta de lo explicado. Así, preparaba una clase entera explicando las características principales de los distintos géneros de videojuegos, para luego mostrar ejemplos de títulos que refuercen y evidencien lo expuesto. Valería Drelichman, profesora de programación de primer año en el Instituto Universitario Nacional del Arte (IUNA), va un poco más lejos en esta idea diciendo que, más allá del docente, **el alumno está obligado a ser curioso por naturaleza, puesto que sin la curiosidad, y sin pensar en el "*por qué*" nunca va a aprender**. Saeed vuelve sobre este punto al decir que los alumnos que más se equivocan son los que más aprenden, ya que son conscientes del error en términos mucho más profundos con respecto a aquellos alumnos que no hicieron el ejercicio, e incluso en comparación con aquellos que lo resolvieron correctamente. En sus clases, continuamente los inducía al error, para enseñarles luego un nuevo método, más eficiente u original, de resolver determinado problema. La investigadora María Ledesma mejora esta línea de pensamiento aconsejando que las clases de lógica y cálculo "*debieran comenzar primero con la ejercitación, para fomentar en primer lugar la práctica, el error y el análisis profundo de cada alumno, y luego completar la solución con la información teórica y la explicación del caso*".

Otro punto importante que destacan la mayoría de los profesores entrevistados, tiene que ver con la tolerancia mínima que le tendrán a cada alumno con sus correspondientes evaluaciones. Mientras que Hernán Cuevas, profesor de Comunicación Social, acepta que en su materia no tiene grandes expectativas debido al poco tiempo disponible, y que se conforma con generar un mínimo

de opinión crítica en cada alumno, Mandelbaum se muestra firme en su postura inalterable de aprobar sólo aquellos que cumplan con los altos niveles aceptados por su cátedra. Motivado por una fuerte impronta ética acerca del valioso compromiso del docente hacia la comunidad en general, no se permite formar profesionales mediocres, y cree sólidamente en la idea de nivelar siempre los estándares académicos para arriba.

```
/*
    "Al reemplazar el pseudocódigo por JavaScript, el alumno
    hoy está más lejos de comprender la profundidad del
    problema". Valeria Drelichman.
*/
```

En ese punto, Drelichman reconoce que en la actualidad, su cátedra decidió bajar sus patrones de excelencia en búsqueda de resultados más evidentes, medibles y alcanzables. Al reemplazar el pseudocódigo por JavaScript, ella es consciente que hoy el alumno está más lejos de comprender la profundidad del problema, aunque finalmente termine por resolverlo, utilizando alguna fórmula básica predeterminada. La diferencia radica, según su opinión, en que **antes el alumno tenía mejores herramientas para comprender metódicamente la solución del problema, mientras que ahora realiza soluciones técnicamente más profesionales**, aunque las ejecute mecánicamente de memoria.

Aunque afirma que programar no es copiar código, y se esfuerza firmemente por fomentar eso en todas sus clases, ella acepta que debió negociar la búsqueda de un alto estándar de excelencia en beneficio de calmar las ansiedades de sus alumnos, que en reiteradas oportunidades expresaron su desagrado por lo "*aburrido del pseudocódigo*" y que querían trabajar más con "*proyectos reales*". Saeed, con cierta resignación, reconoce que para el alumno, es más importante decir que "sabe ActionScript", antes que decir que "sabe pensar", puesto que de ese modo puede presentar en el mercado un currículum más competitivo.

Por otro lado, indica que gracias al cambio de lenguajes y metodologías pedagógicas, sus estudiantes han alcanzado hoy niveles más elevados en la resolución de los ejercicios, tanto con JavaScript como con ActionScript, y que esto se da precisamente por el hecho de estar en contacto por más tiempo con el lenguaje.

Sin embargo, ambos profesores reconocen que esos ejercicios suelen ser "*menos creativos pero más completos que antes*", y que esta decisión está fuertemente inspirada por una cuestión puramente comercial de las instituciones a la que responden, y son partidarios de la idea que **la programación es un arte que debe aprenderse en forma pausada y profunda**, permitiendo que los alumnos asimilen perfectamente los conocimientos dados, y que actualmente con el plan de estudios de hoy, no pueden garantizar que así suceda.

A modo de conclusión, destaco la confrontación planteada por estos profesores, acerca de lo que es mejor para los alumnos: ***¿saber ActionScript, o saber pensar?***

Si bien, la primera responde a una necesidad individualista de posicionamiento objetivo dentro de un mercado por demás competitivo, lo cierto es que no hay punto de comparación. El "*saber pensar*", abre las puertas infinitas del conocimiento, y no sólo estará presente en la vida académica de los alumnos, sino que lo acompañará el resto de su vida personal y profesional, potenciando cada una de sus decisiones a niveles máximos de excelencia. Por el contrario, ActionScript no es más que es la herramienta popular de turno, una más del montón, y aunque valiosa, no se acerca bajo ninguna circunstancia al desarrollo innato del aprendizaje profundo de una profesión.

Es tarea del docente, entonces, alentar esta idea, para reducir la ansiedad lógica del alumno inexperto, y aclararle que si aprende a pensar, y desarrolla las capacidades cognitivas correspondientes con la disciplina, la herramienta va a aprenderse por sí sola.

4.2. DISEÑO Y CONCRECIÓN.

Los jóvenes de hoy se mueven en un mundo de ágil dinamismo y continua estimulación, donde todo es simultáneo e inmediato, y viven experimentando una constante sensación de impaciencia y "*presentismo*" (Morduchowicz, 2008).

Son la primera generación que ha conocido desde su infancia un universo mediático extremadamente variado en medios: radio, TV, videojuegos, videocasetera, DVDs, iPods, MP3, Internet, Smartphones, etc. Las pantallas de televisión, de la computadora y del celular son para ellos una parte natural de su cotidianeidad. Les resultan simples, entretenidos y cómodos, y por allí pasan sus círculos sociales, laborales y también educativos. Son la cultura del ***zapping***, de los videojuegos en red, del chat móvil y del sms como principal vía de comunicación.

Teniendo esto en cuenta, es permisible suponer que **el uso de herramientas multimediales que dialoguen en su idioma natal potenciará su aprendizaje** desde niveles mucho más profundos a los que pudieran aspirar aquellos planes de estudio antiguos, desactualizados y poco conjugados con sus objetivos de aprendizaje.

Si bien se han investigado y analizado muchos programas académicos de enseñanza en la programación, se ha visto que todos ellos están centrados casi exclusivamente en la creación y edición de la sintaxis y la codificación de los algoritmos, perdiendo de vista otros aspectos fundamentales para el alumno multimedia: diseño, usabilidad, manipulación, arquitectura de información, etc.

Por otro lado, muchos de ellos tampoco cuentan con una versión en idioma español que permita a los alumnos pensar y plantear cada problema como una manifestación lógica que deban resolver en la vida real con sus propias palabras.

Por lo tanto, para el completo desarrollo cognitivo de la materia, se desarrollará un entorno virtual educativo propio, siguiendo los lineamientos de Schön descritos en el capítulo 1.6. El mismo deberá ser capaz de satisfacer las necesidades de aprendizaje de este segmento específico de estudiantes de multimedia, los nativos digitales, ya analizados en el capítulo 3.1.2.

4.2.1. MARIOCODE. UNA HERRAMIENTA EDUCATIVA.

Bajo este contexto se realizó la herramienta ***MarioCode***, un entorno multimedial de enseñanza que tiene como único objetivo facilitar el aprendizaje de la algorítmica y la programación estructurada para alumnos universitarios con vocación técnica y artística.

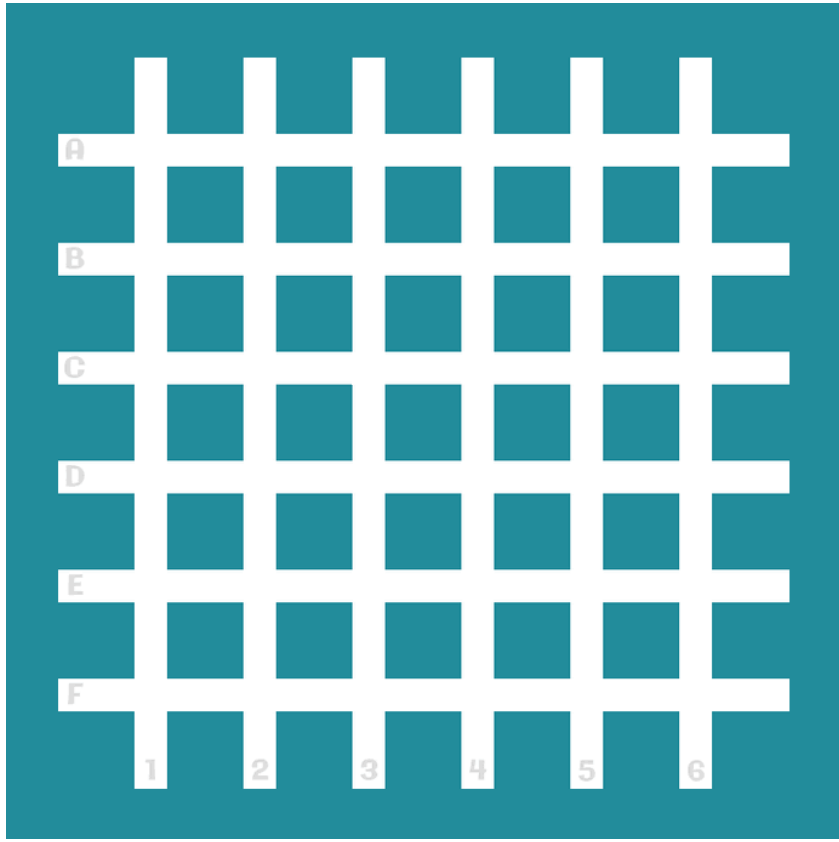
Se trata de una aplicación web de libre acceso, disponible en forma gratuita desde ***www.mariocode.com.ar***. Cuenta con una interfaz simple y ordenada, de estilo moderno y limpio, que se ve potenciada por la inclusión del Mapa, un gráfico 2D que representa una ciudad por donde el alumno deberá desplazar al personaje principal para cumplir los objetivos educativos (FIGURA 4).

Sabiendo que el nativo digital pertenece a la nueva generación que heredó la cultura de los videojuegos, y como se describió anteriormente, privilegia los videojuegos al trabajo serio, se utiliza la figura de Super Mario Bros, un personaje ficticio de arcade diseñado en 1985 por el japonés Shigeru Miyamoto para la compañía Nintendo.

En el juego original, Mario es un plomero italiano que debe moverse a través de varias plataformas, esquivar todos los obstáculos y combatir a sus enemigos para rescatar a la princesa Peach. A partir de su aparición en videojuegos, películas y series televisivas, Mario se ha convertido en el icono emblemático de Nintendo, y es considerado **el personaje de videojuegos más famoso y reconocido de todos los tiempos**. [14].

MarioCode es la adaptación de esta historia lúdica como una nueva plataforma educativa. En esta versión, al igual que en las originales, Mario es el héroe que debe llegar hasta la ubicación de la Princesa sorteando los enemigos y obstáculos que se encuentren en su camino. La complejidad de los ejercicios irá creciendo a medida que el alumno avance, ya que contará con las capacidades cognitivas para resolverlos en forma satisfactoria.

FIGURA 4. MAPA DE MARIOCODE.

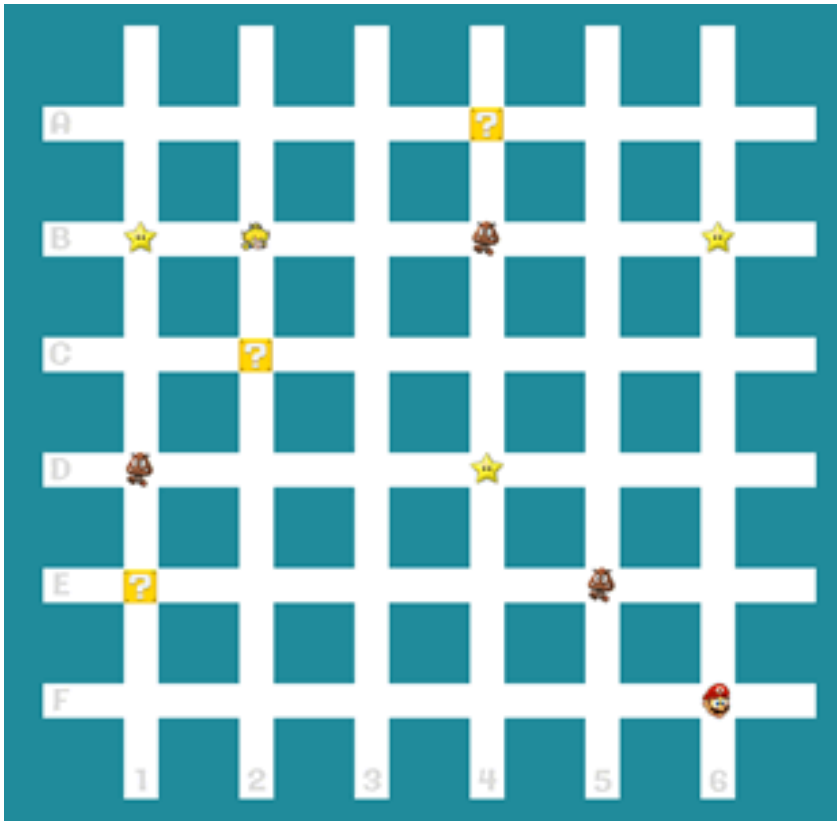


Los actores y elementos involucrados en la herramienta para el desarrollo de los ejercicios son (FIGURA 5):

1. Mario.

Es el personaje principal. El alumno deberá darle instrucciones precisas para que llegue hasta la ubicación de la Princesa cumpliendo todos los objetivos dados por cada ejercicio.

FIGURA 5. MARIO, PRINCESA, ESTRELLAS, MONEDAS Y ENEMIGOS.



2. **Princesa.**

   Es la dama en peligro. El alumno debe programar a Mario para que lleguen siempre a reencontrarse, sin importar su posición ni la cantidad de obstáculos o enemigos que los separen.

3. **Enemigos.**

   Siempre están en movimiento horizontal, y se deberán evitar. Con ellos se busca complejizar los ejercicios, permitiendo al alumno aprender a anticiparse a los problemas externos, en este caso, al movimiento previsible de cada enemigo.

4. **Bloqueos.**

   Son problemas adicionales que deberá resolver el alumno en cada ejercicio. Los bloqueos no se pueden atravesar y el alumno deberá programar su algoritmo de forma tal que siempre los rodeen sin salir del Mapa. A diferencia de los Enemigos, los bloqueos no se mueven nunca.

5. **Estrellas.**

   Son indicadores de calidad del código. Cuantas más estrellas recoja el alumno, más completo será su algoritmo final.

6. **Monedas.**

   Al igual que las estrellas, otorgan mayor puntaje y ayudan a complejizar los ejercicios. Aunque las monedas están siempre ocultas en bloques misteriosos y el alumno deberá consultar si hay monedas ocultas en todos los que encuentre en su camino.

El alumno deberá mover a Mario a través del mapa utilizando instrucciones de programación y lenguaje de pseudocódigo. Estas sentencias, ingresadas en la línea de comandos de MarioCode (FIGURA 6), facilitarán su comprensión de la estructura secuencial de los algoritmos, sustentada por la demostración gráfica de cada instrucción ejecutada.

Al ver a Mario moverse a través del mapa, el alumno tendrá un método fácil e intuitivo de seguir visualmente cada instrucción de su código, sin recurrir a pruebas de escritorio, diagramas de flujos, ni cualquier otra técnica conocida para debugear el código. Con esto se busca preparar al futuro programador a realizar códigos limpios y claros, donde pueda comprender y analizar con exactitud cada instrucción realizada.

En este sentido, el profesor de programación de cuarto año Hernán Beati, al evaluar la herramienta, aseguró que una de sus principales ventajas es que "***mappearles visualmente las consecuencias de cada instrucción puede resultar muy útil para el aprendizaje, ya que vuelve visible un proceso abstracto***".

FIGURA 6. LÍNEA DE COMANDOS E INSTRUCCIONES.

Entre las instrucciones primitivas de pseudocódigo que posee Mario, se encuentran:

- **Arriba**

  Mueve a Mario un casillero hacia arriba.

- **Abajo**

  Mueve a Mario un casillero hacia abajo.

- **Izquierda**

  Mueve a Mario un casillero hacia la izquierda.

- **Derecha**

  Mueve a Mario un casillero hacia la derecha.

La sintaxis del lenguaje empleado utiliza siempre cada instrucción en minúsculas, seguida por un punto y coma, y una por cada línea de código.

Por ejemplo:

```
arriba;
abajo;
izquierda;
derecha;
```

Sin embargo, para lograr una mayor profundidad en los conceptos correspondientes a la programación, también se emplearán las estructuras de control que permiten realizar algoritmos condicionales.

Las estructuras condicionales básicas son:

- **Si.**

  Permite agregar una instrucción del tipo ***IF***, donde se afirma una condición, y en caso de ser verdadera, se ejecutan las instrucciones correspondientes.

  Su sintaxis respeta la forma:

```
si(condición){
    instrucciones
}
```

- **Si / Sino.**

  Permite agrega una instrucción del tipo ***IF ELSE***, donde se afirma una condición, y en caso de ser verdadera se ejecuta una instrucción determinada, y en caso de ser falsa, se ejecuta otra instrucción distinta.

  Su sintaxis utiliza la forma:

```
si(condición){
    instrucción 1
}
sino{
    instrucción 2
}
```

También se incluyen las estructuras iterativas, para completar el corpus de conocimientos requerido por la asignatura:

- **Repetir.**

  Se utiliza para realizar la misma operación en una cantidad establecida de repeticiones, en forma automática por parte de la herramienta. Utiliza la clásica instrucción del tipo ***FOR***.

  Se escribe de la siguiente forma:

```
repetir(cantidad){
    instrucciones
}
```

- **Mientras.**

  La instrucción del tipo ***WHILE*** es una combinación entre las estructuras iterativas y condicionales, ya que las instrucciones se ejecutarán reiteradas ocasiones, siempre y cuando cada repetición cumpla con una condición verdadera.

  Su sintaxis es:

```
mientras(condición){
    instrucciones
}
```

Por último, para completar la acción educativa correspondiente a los objetivos de la materia, se incluyen otras instrucciones básicas que están presentes en todo lenguaje de programación estándar y son la base de la programación estructurada:

- **Variables.**

  Declara un espacio de memoria destinado a almacenar valores alfanuméricos o booleanos (verdadero o falso), al que se accede a través de un nombre clave. Puesto que la herramienta está desarrollada bajo entorno JavaScript, las variables siempre son declaradas como tipos de datos primitivos: cadena de caracteres, números o booleano, aunque no es necesario indicar cada tipo correspondiente en cada caso.

  Se declaran de la siguiente forma:

```
var cadena = “valor”;
var numero = 123;
var boolean = true;
```

- **Funciones.**

  Una función es un conjunto de instrucciones declaradas bajo un nombre clave. En esta versión de la herramienta, las funciones pueden aceptar parámetros por valor, pero no devuelven valores (***void***), cualidad que en algunos lenguajes profesionales se conoce como procedimientos.

  Una función se declara de la siguiente forma:

```
funcion mi_funcion2(parámetros){
    instrucciones
}
```

  Y se invoca la función siempre con paréntesis, reciba o no parámetros:

```
mi_funcion();
mi_funcion2(parámetros);
```

- **Imprimir.**

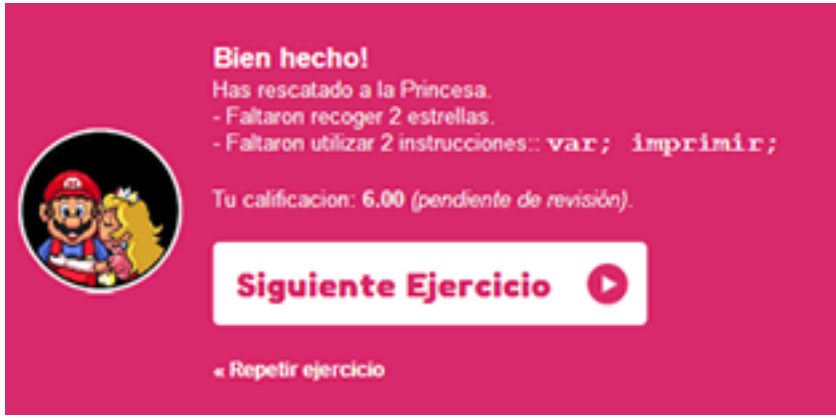
  La función ***imprimir*** muestra en pantalla el contenido de una variable o de un dato determinado que recibe como parámetro. Es una función primitiva propia del lenguaje.

  Se utiliza de la siguiente forma:

```
var texto = “hola mundo”;
imprimir(texto);
```

Al rescatar a la Princesa, **la herramienta evaluará el nivel de eficiencia del código**, dando un puntaje numérico que será calculado en base a cuatro condiciones esenciales - si encontró a la princesa, si utilizó correctamente las dimensiones del Mapa, si recogió todas las estrellas y monedas ocultas, y si no superó la cantidad máxima de instrucciones- y será empleado para medir y cualificar el nivel del código programado (FIGURA 7).

FIGURA 7. MENSAJE EXITOSO CON CALIFICACIÓN Y SUGERENCIAS.



Así, el alumno deberá utilizar todas las combinaciones posibles de estas instrucciones para llegar hacia la ubicación de la Princesa.

Si el objetivo se cumple, se pasará al siguiente ejercicio, donde aumentará la dificultad y se agregarán nuevos conceptos de programación estructurada, por lo tanto crece la complejidad en la forma de llegar al resultado correcto.

En caso de no alcanzar la eficiencia esperada, de contar con algún error en el algoritmo, o de no llegar hasta la ubicación de la Princesa, la herramienta indicará que el objetivo no fue cumplido, y enseñará el error correspondiente, a modo de enseñarle al alumno los motivos de su error o de su baja puntuación (FIGURA 8).

La descripción completa de la evaluación que hace la herramienta podrá verse en profundidad en el capítulo 4.3.4, y el detalle de los errores comunes en la codificación de MarioCode se desarrolla en el capítulo 4.3.5.

FIGURA 8. MENSAJE Y CÓDIGO DE ERROR CON LINK A LA EXPLICACIÓN DETALLADA DEL ERROR.

4.2.2. SOBRE SU DESARROLLO TÉCNICO.

MarioCode es un desarrollo online - disponble libremente en ***www.mariocode.com.ar -*** generado bajo tecnología libre LAMP (*Linux, Apache, MySQL, PHP*) como soporte del lado del servidor, y tecnología JavaScript en el lado del cliente. Más específicamente, el cliente utiliza la librería jQuery, que es un software libre y de código abierto con Licencia Pública General de GNU. Esta licencia es la más ampliamente usada en el mundo del software y garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el programa, permitiendo su uso en proyectos libres y privativos de toda índole [9].

Entre sus características fundamentales, se incluye la manipulación de elementos DOM (*Document Object Model)*, la creación de eventos y animaciones, las ejecuciones asincrónicas en el servidor (*AJAX*), y también garantiza la compatibilidad con los navegadores más populares de la actualidad: Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+, incluyendo además una alta variedad de navegadores predeterminados de smartphones y tablets. Estas características fueron las razones fundamentales que justifican la elección de esta tecnología para el desarrollo de MarioCode, ya que el hecho de contar con una licencia de código abierto, que asimismo garantiza la compatibilidad en la mayoría de los dispositivos del mercado -tanto de escritorio como móviles-, ayuda a su crecimiento y su rápida expansión.

Al no requerir instalación alguna, ni poseer plugins externos -como el caso de Flash Player o Java Virtual Machine (JVM)-, se supone que su distribución no será un problema -específicamente al tratarse de un sitio web online accesible en todo el mundo-, certificando además que los alumnos puedan utilizar la herramienta en todo momento y desde cualquier lugar, potenciando sus motivaciones de aprendizaje sin generarles ninguna frustración en su utilización.

4.3. DESARROLLO PROPUESTA.

A fin de garantizar el análisis de todos los aspectos relacionados con la enseñanza de la programación, la herramienta contará de cuatro unidades, cada una ligada a un objetivo específico del aprendizaje: ejercitación, teorización, comunicación y evaluación.

Como referencia de su aplicación multimedial pedagógica, se tomaron los conceptos cognitivos básicos utilizados en otros desarrollos análogos, tales como ***Logo, Visual Da Vinci, Scratch*** y ***Code Academy***. Todos fueron ya desarrollados en el capítulo 2.2.2.

4.3.1. EJERCITACIÓN.

Como se mencionó con anterioridad en el capítulo 1.5, la experimentación propia de cada alumno es la condición fundamental en el proceso de formación. Es la madre de todo conocimiento, la que se asegura la fijación profunda del aprendizaje y la que permite a la memoria indexar y combinar los distintos conceptos estudiados. El método constructiva de educación sugiere que **el profesor simplemente ha de asegurar un entorno rico en estímulos que le de las posibilidades para que el alumno, trabajando a su propio ritmo, sea capaz de construir nuevas estructuras cognitivas.**

Tal como dijo Fabián Mandelbaum, profesor de programación en Universidad Maimónides, "*hacer las clases prácticas suple un poco la deficiencia que tiene el alumno cuando sale del aula y no repasa la clase. Y los alumnos sobresalientes, los que realmente logran cumplir los objetivos de la materia, son los que también se sentaron en sus casas a seguir la materia*". Haciendo las clases prácticas, se motiva al alumno a practicar la materia, y el hacerlo en forma pausada, le da el tiempo necesario para tantear y asimilar los contenidos en forma individual y grupalmente.

Por lo tanto, se diagramarán diez ejercicios prácticos, correspondientes a diez clases presenciales, donde los alumnos deberán resolver individual o colectivamente cada problema planteado, con la asistencia del profesor que deberá guiarlos ingeniosamente a través de la curiosidad y las motivaciones de cada uno, para que sean los mismos alumnos los que resuelvan cada ejercicio sin su ayuda.

```
/*
    "Los alumnos sobresalientes son los que también se sentaron
    en sus casas a seguir practicando". Fabián Mandelbaum.
*/
```

Los ejercicios planteados serán siempre progresivos, es decir, cada uno contendrá pautas y conceptos aprendidos por el anterior. Así se fomentará el continuo repaso de las nociones ya vistas en clases anteriores, siguiendo por caso el ejemplo del profesor norteamericano Ralph Lynn cuando hablaba de la importancia de las meningitis a lo largo de todo el año, en el capítulo 1.4.

Se buscará así cubrir la comprensión total de los aspectos fundamentales de la materia, logrando que el alumno complete el curso con conocimientos sólidos sobre la programación estructurada y el diseño de algoritmos.

Por otro lado, al completar cada ejercicio, los alumnos deberán crear y resolver un problema propio, inventado por ellos mismos, y aplicando todos los conocimientos que lleven aprendido hasta ese momento. Así, al finalizar el cuatrimestre, cada alumno habrá resuelto en forma individual más de veinte ejercicios prácticos con la herramienta, contando los propios y ajenos.

Los ejercicios que deberán resolver los alumnos a través de MarioCode, en forma obligatoria a lo largo del cuatrimestre son:

Ejercicio 1. Introducción al lenguaje.

Presentación de la interfaz. Programación secuencial simple.

Utiliza las instrucciones `arriba; abajo; izquierda; derecha;` para llevar a Mario hasta la ubicación de la Princesa.

Ejercicio 2. Estructuras secuenciales.

Recolección de estrellas.

Utiliza las instrucciones `arriba; abajo; izquierda; derecha;` para llevar a Mario hasta la ubicación de la Princesa, recogiendo en el camino todas las estrellas que aparecen.

Ejercicio 3. Estructuras secuenciales -2da parte-

Evitar caminos bloqueados por obstáculos.

Utiliza las instrucciones `arriba; abajo; izquierda; derecha;` para llevar a Mario hasta la ubicación de la Princesa, recogiendo en el camino todas las estrellas que aparecen en el mapa, evitando los caminos bloqueados.

Ejercicio 4. Variables.

Creación de variables .Operadores aritméticos. Impresión en pantalla.

Utiliza las instrucciones `var; imprimir;` para rescatar a la Princesa, contando la cantidad de pasos necesarios y mostrando esa cantidad en pantalla.

Ejercicio 5. Variables -2da parte-

Evitar caminos bloqueados por enemigos.

Utiliza las instrucciones `var; imprimir;` para rescatar a la Princesa, evitando a los enemigos y mostrando en pantalla el promedio de estrellas recogidas sobre pasos realizados.

Ejercicio 6. Estructuras condicionales.

Uso del SI.

Utiliza las instrucciones `si; haymoneda; var; imprimir;` para rescatar a la Princesa, buscando y contando las monedas escondidas en las cajas que se encuentran por el camino.

Ejercicio 7. Estructuras condicionales -2da parte-

Uso del SI/SINO. Operadores relacionales. Concatenación de caracteres.

Utiliza las instrucciones `si; sino; haymoneda; var; imprimir;` para rescatar a la Princesa, contando la cantidad de cajas vacías que se encuentran, imprimiendo en pantalla "Hay xx cajas con estrellas y XX cajas sin estrellas."

Ejercicio 8. Estructuras iterativas.

Uso del REPETIR.

Utiliza la estructura `repetir` para rescatar a la Princesa utilizando sólo tres instrucciones en tu código.

Ejercicio 9. Estructuras iterativas -2da parte-

Uso del MIENTRAS.

Utiliza la estructura `mientras` para llevar a Mario hasta la ubicación de la Princesa, contando sólo cuatro monedas encontradas, y utilizando sólo nueve instrucciones en tu código.

Ejercicio 10. Programación estructurada.

Abstracción y creación de funciones.

Utiliza la estructura `funcion` para generar una función que mueva a Mario por toda una fila de cuadras, contando la cantidad de monedas que encuentre. Debe rescatar a la Princesa tras recorrer todo el mapa.

4.3.2. TEORIZACIÓN.

Luego de realizado cada ejercicio práctico, el profesor explicará las distintas resoluciones profundizando en el marco teórico de la materia, utilizando los conceptos teóricos de la programación y la algorítmica. Esto se dará dentro de un marco de aprendizaje constructivista que permita al alumno construir y analizar su propia experiencia, tal como lo desarrolla Kamii en el capítulo 1.5.

Estas definiciones también estarán distribuidas en diez clases teóricas a las que el alumno podrá acceder en todo momento a través de la herramienta online.

Las clases estarán directamente relacionadas con cada ejercicio, y tratarán todos los conocimientos aprendidos en ellos:

Clase 1. Introducción al lenguaje y la herramienta.

Los programas escritos en MarioCode utilizan una forma de codificación denominada pseudocódigo. En pocas palabras, el pseudocódigo es un lenguaje de programación de alto nivel, de simple comprensión por estar diseñado directamente para la lectura humana. El objetivo de este lenguaje es proporcionar herramientas sencillas de asimilar y comprender para el aprendizaje de la programación multimedial.

¿Qué son las instrucciones?

Las instrucciones son las órdenes que el usuario le dará a Mario para controlarlo y hacerlo desplazar a través del mapa. En MarioCode, todas las instrucciones primitivas -`arriba, abajo, izquierda y derecha`- se escriben en una línea cada una, separadas por salto de línea y finalizan con un punto y coma que determina la separación de instrucciones. La ejecución del código –también llamado algoritmo, programa, etc. – finalizará al realizar la última instrucción dada por el programador, por lo cual, el alumno deberá escribir todas las instrucciones necesarias para realizar la tarea requerida por el ejercicio.

¿Qué son los errores de sintaxis?

Son instrucciones mal escritas, con faltas de ortografía, de puntuación, o simplemente inexistentes. El alumno debe controlar que todas las instrucciones no lleven nunca mayúsculas, signos de puntuación ni caracteres especiales.

Todas las instrucciones primitivas finalizan siempre en punto y coma y se escriben una instrucción por línea.

Clase 2. Estructuras secuenciales.

¿Para qué sirven las estrellas?

Las estrellas son elementos dentro del mapa que el alumno deberá recoger mientras resuelve el ejercicio. Están dispuestos en cualquier ubicación de la ciudad, y cuantas más estrellas recoja Mario, mayor será la puntuación del ejercicio.

Clase 3. Estructuras secuenciales – 2da parte-

¿Qué son los obstáculos?

Son sitios específicos del mapa por donde Mario no podrá pasar, y el alumno deberá programar su rodeo para evitarlos. Si Mario choca contra un bloqueo, inmediatamente se corta la ejecución del programa.

¿Cómo evitar a los enemigos?

Los enemigos son personajes que se deberán evitar, de accionar similar a los bloqueos, con la diferencia de que ellos están en continuo movimiento. Un enemigo siempre se estará moviendo de izquierda derecha, de un paso a la vez.

Clase 4. Variables.

¿Qué son las variables?

Las variables son espacios lógicos en la memoria donde se puede guardar información. Todas las variables tienen un nombre, que permite accederla para asignarle y consultar valores, y un tipo de datos, que indica si el contenido es un número, un texto o un valor booleano (verdadero o falso).

En MarioCode no hace falta declarar el tipo de dato, ya que se da en forma automática al momento de asignarle un valor.

Clase 5. Variables -2da parte-

¿Cómo concatenar una cadena de caracteres?

Como se explicó, las variables son espacios de memoria que pueden contener datos. Estos datos pueden ser números, caracteres, booleanos, e incluso variables mismas. Si se desea concatenar varios datos dentro de una variable, se utiliza el comando "+".

Por ejemplo:

```
var nombre = "gaston";
var apellido = "gimenez";
var resultado = nombre + " " + apellido;
```

En este caso, se utilizan tres variables. La primera almacena el nombre, la segunda el apellido, y la tercera concatena el nombre con un espacio en blanco, seguido del apellido. Hay que tener en cuenta que la concatenación se hace sólo entre cadenas de caracteres. En cambio, si se ejecuta el mismo código utilizando variables numéricas, el resultado será la suma de dicha instrucción.

Por ejemplo:

```
var uno = 1;
var dos = 2;
var resultado = uno + dos;
```

En este caso, en la variable resultado se guardaría el valor 3, ya que es el resultado de la suma (1+2=3).

¿Cómo imprimir el contenido de una variable?

Se utiliza el comando imprimir() para mostrar en pantalla la variable o el dato que se incluya entre los paréntesis.

Por ejemplo:

```
var nombre = “gaston”;
imprimir (nombre);
```

Clase 6. Estructuras condicionales.

Permiten ejecutar o no una serie de instrucciones en base al cumplimiento o no de una condición. Representa la toma de decisiones del algoritmo.

Por ejemplo:

```
var numero = 10;
si(numero<20){
    imprimir(“el numero es menor a 20”);
}
```

En este caso, se declara una variable con el valor 10 y se pregunta si el contenido de esa variable es menor a 20. En caso que se cumpla esa condición, es decir, si el número es menor a 20, se ejecuta la instrucción que le sigue, en este caso, se imprime en pantalla “el numero es menor a 20”.

Clase 7. Estructuras condicionales -2da parte-

La forma avanzada de la toma de decisiones incluye una alternativa en caso que la condición sea falsa.

Por ejemplo:

```
var numero = 30;
si (numero<20) {
     imprimir (“el numero es menor a 20”);
}
sino{
     imprimir (“el numero es mayor/igual a 20”);
}
```

En este caso, se pregunta si el número es menor a 20. Al no cumplirse esa premisa, es decir, al ser falsa la condición, se ejecuta la instrucción alternativa, que en este caso imprime en pantalla “el número es mayor o igual a 20”.

Clase 8. Programación iterativa.

¿Qué es la instrucción REPETIR?

Es un comando que permite ejecutar una o más instrucciones una cantidad indicada de repeticiones (bucles).

Por ejemplo:

```
repetir (5){
    imprimir (“hola”);
}
```

Tendría un efecto similar a ejecutar:

```
imprimir(“hola”);
imprimir(“hola”);
imprimir(“hola”);
imprimir(“hola”);
imprimir(“hola”);
```

La diferencia principal radica en la cantidad de instrucciones necesarias para realizar la misma tarea. Supongamos que en lugar de repetir 5 veces la instrucción, necesitemos repetirla 100 veces: deberíamos escribir 100 veces el comando "`imprimir`..." en lugar de simplemente indicar el número de repeticiones en la instrucción "`repetir`".

Clase 9. Programación iterativa -2da parte-

¿Qué es la instrucción MIENTRAS?

Es un comando que ejecuta continuamente una o más instrucciones mientras se cumpla una condición. Representa la toma de decisiones en combinación con el retorno de los datos hasta estados de tiempo previos o posteriores.

Por ejemplo:

```
var cantidad = 1;
mientras(cantidad<10){
    imprimir(cantidad);
    cantidad = cantidad + 1;
}
```

En este ejemplo se declara una variable cantidad inicializada en 1. La instrucción "`mientras(cantidad<10)`" se lee como "***ejecutar mientras cantidad sea menor a diez***", y lo que se ejecuta son las instrucciones que están dentro de la sentencia.

Entonces, "`imprimir(cantidad)`" mostrará primero el valor 1, después el 2, y así sucesivamente hasta llegar al 9, ya que la instrucción "`cantidad = cantidad + 1`" se ejecutará 8 veces antes de salir por la condición de falso.

Clase 10. Programación estructurada.

¿Qué son las funciones?

Las funciones describen una secuencia de instrucciones que hacen una tarea específica. Se pueden crear distintas funciones para encapsular distintos comportamientos, y su utilización queda a expreso criterio del programador.

Por ejemplo:

```
funcion girar360(){
    izquierda;
    derecha;
    arriba;
    abajo;
}

repetir(2){
    girar360();
}
```

En este ejemplo, se aprecia el potencial de la programación estructurada, que utilizando funciones y estructuras de control permite realizar en pocas y prolijas instrucciones un algoritmo que de otra forma sería:

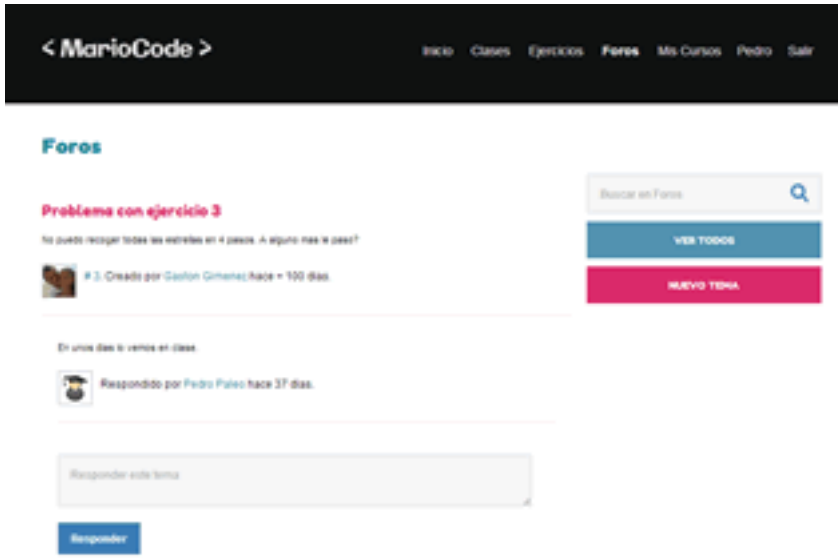
```
izquierda;
derecha;
arriba;
abajo;
izquierda;
derecha;
arriba;
abajo;
```

4.3.3. COMUNICACIÓN.

Debido a la dependencia casi absoluta que tienen los nativos digitales hacia las comunicaciones virtuales, dentro de la herramienta **se incluye un espacio comunitario de consulta y asistencia continua**, donde los alumnos podrán ponerse en contacto tanto con sus compañeros, como con sus profesores.

De esta manera, cuentan con un foro de consulta que los asista en todo momento, disponible en caso de surgir dudas ante un ejercicio o un concepto determinado (FIGURA 9). MarioCode incluye una opción muy sencilla para crear un nuevo tema dentro del foro (FIGURA 10), y lo convierte en una poderosa herramienta de consulta y análisis para resolver ejercicios, ya sea grupal como individualmente.

FIGURA 9. FORO DE CONSULTAS.



Por otro lado, una de las principales características de los foros virtuales y que definen su carácter es la asincronía, concepto inserto en la propia definición.

Los foros son herramientas que se pueden utilizar y consultar en cualquier momento, sin que sea necesario pactar una hora concreta, sino que las aportaciones de los demás participantes quedan recogidas permanentemente, y pueden ser respondidas en el momento en el que se desee. Este carácter asincrónico trae consigo aparejada otra gran característica de los foros, y es que son herramientas que permiten un mayor grado de reflexión de lo aportado por los demás participantes [6].

FIGURA 10. NUEVA ENTRADA DE FORO.

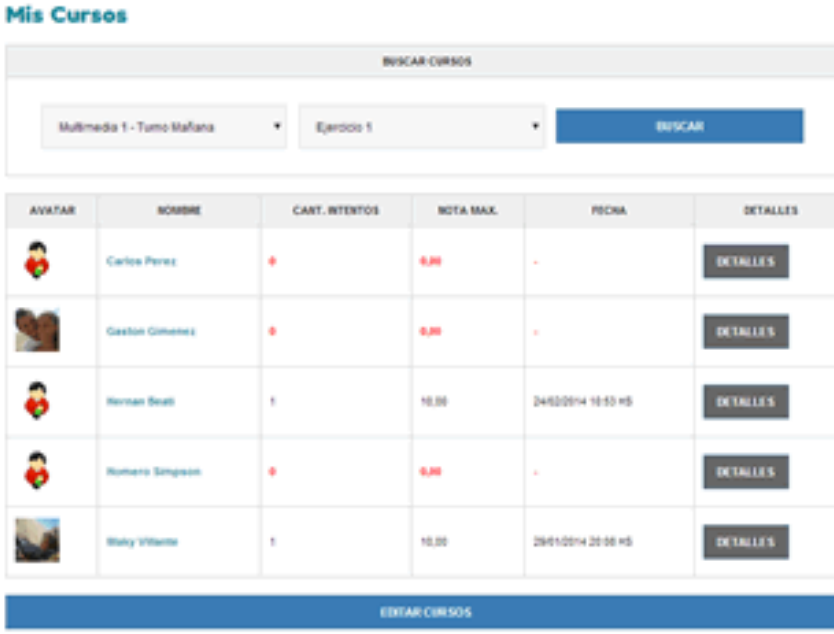
De esta forma, los alumnos contarán con más tiempo para analizar y descubrir las soluciones propias, y para reflexionar sobre las opiniones de los demás participantes. El docente estará encargado de moderar los comentarios, así como de guiar las comunicaciones en función del proceso de aprendizaje estipulado.

4.3.4. EVALUACIÓN.

La herramienta contará con un **área privada exclusiva para los profesores, donde podrán crear sus propios cursos, y consultar la información y el estado de aprendizaje individual de cada alumno** (FIGURA 10).

Todo lo que los estudiantes realicen a través de MarioCode quedará registrado, de manera que cada instructor pueda realizar un seguimiento cercano de la evolución de sus alumnos, teniendo así herramientas útiles que le permitan medir el nivel general y particular del curso.

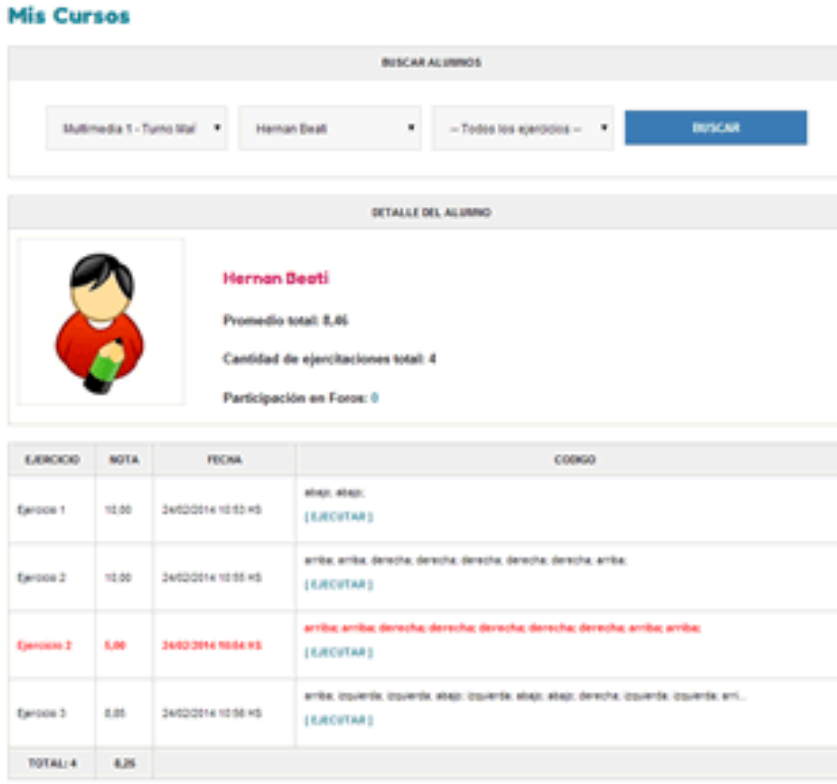
FIGURA 10. ÁREA DE PROFESORES. LISTADO DE ALUMNOS POR CURSO.



En su ***área privada***, los profesores podrán visualizar el detalle de cada alumno, junto con el promedio total de todas sus notas, la cantidad de ejercicios realizados hasta la fecha, e incluso el total de participaciones del alumno en el foro (FIGURA 11).

Así podrá acceder al estado de los ejercicios que realizó, para obtener una visión complementaria del estado general y particular del curso, tal como aconsejan las ideas evaluativas del profesor Bain, que se desarrollan en el capítulo 1.4.

FIGURA 11. ÁREA DE PROFESORES. DETALLE DEL ALUMNO.



Incluso, el profesor podrá verificar paso a paso qué tipos de errores tuvo el alumno, y cuántas pruebas necesitó antes de dar con el resultado final de cada ejercicio.

De esta manera, no sólo podrá evaluar el nivel y crecimiento personal de cada estudiante, sino también podrá medir los modelos de enseñanza, y realizar los cambios que consideren necesarios en sus clases.

También contará allí con la calificación conceptual de cada ejercicio, dada por el nivel de eficiencia de los algoritmos que será calculada en forma automática por la herramienta, en base a cuatro puntos esenciales.

Los dos primeros puntos son excluyentes para dar por aprobado el ejercicio. Es decir, aunque el alumno logre encontrar a la Princesa, o logre esquivar todos los obstáculos sin salir del mapa, **deberá cumplir obligatoriamente con ambos aspectos en simultáneo, o de lo contrario el ejercicio se dará por desaprobado.**

Los últimos dos puntos sirven para medir la calidad del algoritmo, y por lo tanto, para aumentar o disminuir la calificación conceptual del ejercicio. Sin embargo, su condición no es excluyente, por lo que el alumno tendrá aprobado el ejercicio aunque no logre juntar ninguna estrella, o no cumpla con la cantidad máxima de instrucciones requerida por cada ejercicio.

Esta calificación será promediada con las apreciaciones conceptuales del profesor sobre cada alumno, sumado a los trabajos en clase, la entrega de esquicios y la calificación de los exámenes, tal como se desarrolla en el Plan de Estudios propuesto en el capítulo 4.6.

Los puntos que evaluará la herramienta para calificar cada ejercicio son los siguientes:

1. **Encontrar a la princesa.**

   Si al finalizar la ejecución del algoritmo, Mario no logra llegar hasta la ubicación exacta de la Princesa, el código no es efectivo y el objetivo de aprendizaje no se cumplió.

2. **Empleo correcto del Mapa.**

   Si en determinado punto del algoritmo, Mario sale del mapa, si no logra esquivar un bloqueo, o si se choca con un enemigo, no se cumple el objetivo dado.

3. **Juntar estrellas.**

   Si Mario no logra pasar por todos los casilleros donde se encuentre una estrella, el ejercicio igualmente se da por aprobado, ya que se considera válido el rescatar a la princesa, pero el nivel de eficiencia del código no será el más adecuado.

4. **Cantidad de instrucciones.**

   El usuario deberá cumplir con todos los objetivos planteados anteriormente, en la cantidad de instrucciones indicada por cada ejercicio. La cantidad indica el número máximo de instrucciones que deberían ser necesarias para cumplir los objetivos, y el nivel eficiencia se basa promediando la cantidad de instrucciones realizadas por el alumno en relación con la cantidad estimada.

4.3.5. CÓDIGO DE MENSAJES DE ERROR.

Como todo software intérprete de código, MarioCode ofrece a los usuarios una serie de errores comunes ya identificados y explicados, a fin de que los alumnos puedan comprender fácilmente las causas de sus potenciales errores.

A continuación se describe el listado de los códigos de mensajes de error más comunes que pueden surgir a la hora de programar en MarioCode:

#1. Error de sintaxis vacío.

Este error surge porque el campo "***Ingresar Código***" de la herramienta no fue completado, y el programa no puede procesar el algoritmo por encontrarse sin ningún contenido.

#2. Error de sintaxis.

Has ingresado una instrucción que no existe, una variable que no fue declarada, o una palabra clave reservada por el sistema.

Verifica que esté bien escrito el código fuente, que no contenga errores de ortografía o caracteres especiales.

#3. Error en algoritmo. No encuentra princesa.

Ha finalizado la ejecución del algoritmo, pero Mario no ha llegado hasta la ubicación de la Princesa. Verifica el código fuente haciendo las modificaciones necesarias para que Mario termine encontrandose con la Princesa.

#4. Error en algoritmo. Salió del Mapa.

Mario ha salido del mapa. Has las modificaciones necesarias en el código para que él se encuentre visible en todo momento.

#5. Error de ejecución desconocido.

El sistema no pudo procesar el código. Puede deberse a una falla en la sintáxis, o en el armado del algoritmo. Verifica atentamente el código asegurando que funcione correctamente mediante pruebas de escritorio.

También debe asegurarse que no se realicen calculos matemáticos dentro de la función `imprimir`, si es asi procura realizarlos fuera de la función.

#6. Error de sintaxis no compilada.

No está bien estructurado el código.Asegurate que no hayan abiertas paréntesis, corchetes o comillas sin cerrar.

#7. Error en algoritmo. Obstáculo encontrado.

Mario ha chocado contra un bloqueo del Mapa. Has las modificaciones en el algoritmo para rodear los obstáculos sin cruzarlos durante la ejecución del código.

#8. Error en algoritmo. Enemigo encontrado.

Mario ha chocado contra un enemigo que se cruzó en su camino. Has las modificaciones en el algoritmo para esquivar los enemigos, teniendo en cuenta que están en constante movimiento.

4.4. PROTOTIPO / TEST / EVALUACIÓN

La aplicación se encuentra en fase de prototipo y está disponible para su testeo en ***www.mariocode.com.ar*** en forma libre y gratuita para cualquier usuario con acceso a Internet.

Para medir el potencial alcance cognitivo de la herramienta, se desarrolló una serie de entrevistas entre los docentes de Programación Multimedial I, II y III de la Universidad Maimónides, como primer aproximación de evaluar la aplicación.

4.4.1. OBJETIVOS DE LAS ENTREVISTAS.

Las entrevistas se realizaron para obtener una primera evaluación técnica y cognitiva que sirva como modelo crítico de corrección, ajuste de conceptos y mecánicas, y detección de fallas.

Sin embargo, se deberá tener en consideración que los resultados arrojados no representan una demostración empírica de las hipótesis, ya que una encuesta realizada fuera de todo contexto no puede nunca reflejar el comportamiento real de un curso, ni mucho menos, el nivel de aprendizaje alcanzado por ellos en un futuro hipotético no inmediato.

Por lo tanto, los resultados se basan en aproximaciones, sugerencias y pura suposición de las personas entrevistadas, quienes de buena fe expresaron sus opiniones en base a su experiencia como docentes universitarios.

Futuras investigaciones deberían profundizar esta exploración, utilizando encuestas generales, focus groups, prácticas reales con cursos universitarios, y otras técnicas que se consideren pertinentes, y que den la posibilidad de afirmar o rechazar las hipótesis planteadas en el presente trabajo.

4.4.2. METODOLOGÍA.

Se realizó una breve guía de evaluación con cinco preguntas de referencia, dirigidos a una muestra representativa de la población de docentes de programación en la carrera de Licenciatura en Tecnología Multimedial en la Universidad Maimónides.

Se efectuó la técnica de administración personal con los encuestados, y en los casos en que no fue posible establecer contacto directo con ellos, se optó por realizar una técnica de autoadministración digital, donde cada uno recibió un e-mail con las pautas de la evaluación y las preguntas de referencia que le sirvieran de base para emitir sus opiniones profesionales.

4.4.3. GUIA DE EVALUACIÓN PARA DOCENTES.

1. ¿Encuentra en la herramienta MarioCode puntos en común con el método que usa actualmente en sus clases, o con otros métodos que conozca de sus colegas?

2. ¿Le parece que esta herramienta supera alguna deficiencia de otros métodos educativos? Por el contrario, ¿le parece que perjudica en alguna medida el aprendizaje de los alumnos?

3. ¿Cómo piensa que reaccionarían los estudiantes ante un método de éstas características? ¿Por que?

4. ¿Cómo piensa que sería el nivel de los alumnos al finalizar este curso? ¿Presume que en los años siguientes se perdería menos tiempo en repasar aspectos básicos de la programación, antes de comenzar con los contenidos nuevos de la materia?

5. ¿Supone que al finalizar sus estudios, los alumnos tendrían un nivel similar, inferior o superior en comparación con los alumnos del método actual?

4.5. PLAN DE ESTUDIOS PROPUESTO.

El siguiente Plan de Estudios está planeado para cualquier universidad o instituto educativo con perfil artístico y tecnológico. Se deberá realizar en el marco de una clase universitaria práctica, con aulas preparadas para la ocasión.

La duración del curso será cuatrimestral, y contará con dos exámenes nivelatorios a fin de evaluar el desempeño de los alumnos.

4.5.1. OBJETIVOS.

La materia de Introducción a la Programación deberá formar los **conceptos básicos de programación estructurada básica y lógica algorítmica**, generando una profunda base de conocimientos sobre la disciplina. Con ello se busca contar con sólidas herramientas cognitivas sobre las cuales los alumnos podrán especializarse durante los siguientes años de la carrera.

Para cumplir estos objetivos, se empleará el uso de lenguaje de ***pseudocódigo*** que, tal como se analiza en el capítulo 2.2, permite una completa asimilación de los conocimientos, y un entorno más propenso para el autoaprendizaje.

Entre los conceptos que el alumno deberá dominar al finalizar el curso, se cuenta las estructuras secuenciales, condicionales e iterativas a través del lenguaje de pseudocódigo.

Por otro lado, el curso deberá promover las ventajas de la programación multimedial como herramienta indispensable de todo desarrollo interactivo, destacando sus virtudes y demostrando su máximo potencial, motivando así a los alumnos a profundizar y perfeccionarse en la materia.

4.5.2. METODOLOGÍA.

Se tratarán de clases teórico-prácticas donde los alumnos deberán resolver ejercicios progresivos relacionados con la programación, utilizando el entorno virtual MarioCode. Esta materia se propone como un espacio de análisis, consulta, reflexión y producción de pensamiento crítico correspondiente con la disciplina.

Como una herramienta interactiva por sí sola nunca podrá crear las condiciones necesarias para que el alumno desarrolle su curiosidad y potencie su creatividad, las clases serán prácticas con la herramienta, y también serán teóricas y presenciales. Así se presenta una **combinación de métodos**, entre los ejercicios realizados por la herramienta y las explicaciones teóricas del profesor, aprovechando los beneficios de ambos tipos de enseñanza.

Teniendo en cuenta las observaciones de Darío Saeed, docente de Universidad Maimónides, las clases serán divididas en cuatro etapas progresivas de aprendizaje, cada una destinada a profundizar los conocimientos aprendidos por el alumno:

1. **Tomar conocimiento.**

   La primera hora del curso será destinada a solucionar en papel un ejercicio dado por el profesor, que los alumnos podrán realizar en conjunto entre varios, contando únicamente con una breve introducción de cada tema por parte del profesor. De esta manera, se presenta el problema dejando al alumno un tiempo a solas para que lo analice y comience a resolverlo, movilizando la autogeneración de ideas y soluciones propias, fomentando así la creatividad y originalidad entre los alumnos.

2. Fijar conocimiento.

La hora siguiente se utilizará para analizar las respuestas obtenidas, y testear su funcionamiento a través de la herramienta MarioCode. Allí se buscará comentar, teorizar y profundizar sobre los conceptos técnicos utilizados para solucionar el ejercicio dado. El alumno que realizó correctamente el ejercicio podrá compararlo contra otros modos de resolverlo, y el que no pudo completarlo, podrá analizar lo que hicieron sus compañeros y tenerlo presente para el próximo caso.

También se pueden generar pequeños cambios en los problemas, de modo que el alumno aprenda la capacidad de adaptar su código a cualquier escenario dado.

3. Evidenciar conocimiento.

Al finalizar cada clase, se pedirá a los alumnos que generen cada uno un ejercicio propio y lo resuelvan para la clase siguiente. Estos trabajos serán tratados como esquicios, y tendrán una calificación conceptual en base a la complejidad y la creatividad en su resolución. De esta manera, también se da a los alumnos la posibilidad de controlar el ritmo de la clase, profundizando o simplificando los ejercicios en la medida que ellos se sientan más cómodos y seguros con la materia.

4. Compartir conocimiento.

Por otro lado, la herramienta poseerá un formato del tipo Foro, donde los alumnos podrán compartir todo tipo de consultas relacionadas con los ejercicios y la disciplina en sí, para centralizar un polo de conocimiento virtual que les sirva de asistencia continua y dinámica.

4.5.3. PROGRAMA.

Se plantea la materia como un curso formativo cuatrimestral, con una carga horaria de noventa minutos semanales. Contará con **diez clases teóricas-prácticas obligatorias**, más dos clases destinadas a realizar ejercicios de evaluación en clase, y dos clases adicionales de repaso y/o recupero de contenidos.

El ritmo y la velocidad de cada clase serán evaluados por el profesor en correspondencia con las exigencias y necesidades de cada curso, para potenciar las virtudes de cada grupo de alumnos.

Sin embargo, deberá mínimamente respetar los contenidos de las siguientes unidades, pudiendo profundizar más o menos en ellos:

Unidad 1. Introducción a la programación.

Presentación de la materia. Características del pseudocódigo. Introducción a la herramienta.

Unidad 2. Programación secuencial.

Declaración de variables. Cálculos aritméticos. Operadores. Instrucciones lineales.

Unidad 3. Programación condicional.

Estructuras de control. Funciones y parámetros.

Unidad 4. Programación iterativa.

Bucles y estructuras repetitivas. Programación estructurada.

4.5.4. EVALUACIÓN.

A mitad del cuatrimestre se realizará un examen práctico nivelatorio que no llevará calificación. Su objetivo será únicamente analizar y evaluar el estado general del curso, para verificar el nivel del aprendizaje obtenido, despejar las dudas que surjan entre los alumnos y efectuar los ajustes correspondientes en el programa.

Al finalizar el cuatrimestre, los alumnos tendrán un examen parcial obligatorio, donde deberán resolver cinco ejercicios seleccionados al azar, de los creados previamente por sus compañeros. La aprobación de este examen será condición suficiente para dar la materia por aprobada, sin necesidad de realizar un examen final.

La calificación de cada alumno estará dada por un promedio realizado entre la nota del examen final, las calificaciones de los esquicios y trabajos en clase, las apreciaciones conceptuales, y el nivel de aprendizaje y progreso que el profesor haya notado en cada alumno a lo largo del curso.

CONCLUSIONES FINALES

Todos los docentes consultados se mostraron satisfechos con la metodología propuesta y las novedades creativas que se presentan en esta investigación. Incluso, algunos llegaron a certificar que el nivel de aprendizaje obtenido a través de MarioCode sería mucho mayor al compararlo con el de las técnicas convencionales.

Entre los puntos favorables, los entrevistados destacaron el clima informal de la herramienta, el modelo lúdico en que se desarrolla el aprendizaje, el empleo del pseudocódigo como lenguaje de programación, y especialmente, la cualidad visual de las instrucciones ejecutadas. Todos ellos afirmaron que visualizar la ejecución del código es beneficioso porque permite a los alumnos comprender con mayor profundidad las consecuencias de las decisiones en sus propios algoritmos, así como el reconocer los errores que pudieran surgir, y el modo correcto de corregirlos.

Bajo este modelo de pensamiento, el profesor de cuarto año Hernán Beati llegó a asegurar que "***sin dudas los alumnos tendrán un alto índice de aprendizaje de lo básico de la programación con esta herramienta, mayor al logrado con métodos no visuales",*** destacando especialmente esta original forma de contrarrestar la intrínseca naturaleza abstracta de la programación.

Por su parte, Raúl Drelichman, coordinador pedagógico de la carrera de multimedia, indicó que "***MarioCode es parecido al Logo, programa con el que mis hijos aprendieron programación***", haciendo alusión a su método informal y lúdico, que se diferencia en gran medida de las actuales herramientas de enseñanza en la programación, que son valiosas en rendimiento comercial, pero escasas de valor pedagógico para el estudiante multimedia.

Al respecto, Darío Saeed aumentó este análisis al declarar que "***el que viene a estudiar multimedia no quiere estudiar sistemas***. *Le gusta el diseño y tiene un perfil más artístico, y no le gustan las matemáticas ni las clases formales*". Por eso, se destaca el modo lúdico-educativo de la herramienta, que permite aprender los conceptos básicos de la programación, mientras "*jugás a ver si Mario puede rescatar a la Princesa y conseguir todas las monedas*".

De la misma manera, el profesor de cuarto año Pedro Pérez León, considera que "***está muy acertada la decisión de proponer un acercamiento a la programación a partir del pseudocódigo, y no desde un lenguaje específico***". Él opina que esto termina siendo de gran beneficio y ayuda para todos los alumnos en tres aspectos:

1. El lenguaje de programación está muy cercano al lenguaje habitual del alumno.

2. Al ser las sentencias en castellano, se evita cualquier inconveniente que pudiera haber en alumnos que no han desarrollado aún un nivel de idioma inglés mínimo, algo bastante común en alumnos de los niveles iniciales.

3. La posibilidad del pseudocódigo evita además las posibles confusiones que se pudieran dar en cursos en los que se suele modificar de un año al otro el lenguaje de programación introductorio, lo que suele ocurrir mientras la mayoría de los alumnos aún no han terminado de incorporar los conceptos básicos de programación.

Por otro lado, la profesora Valeria Drelichman afirma que *"presentar al curso los ejercicios que van a terminar resolviendo al finalizar las clases, hace que se ganen tu confianza, y se motiven aunque estén viendo pseudocódigo, y no entiendan del todo para qué funciona eso*". Así, concentran sus esfuerzos en resolver los problemas de la mejor forma posible, y no pierden tiempo tratando de adivinar lo que van a tener que *estudiar* para los exámenes.

También reflejó buenas críticas la inclusión de un Foro de Consultas. Drelichman manifiesta que "*hay una inmediatez en la corrección que es muy útil. Si se está equivocado en algo, en programación es crucial el momento de corregirlo. Porque si no lo corrigen, se les queda grabado el error.*" Los foros permiten esta vía de comunicación dinámica, donde tanto el profesor como los compañeros que se encuentren en línea, muy rápidamente pueden asistir al alumno, teniendo -a diferencia del chat- el tiempo adecuado para dar una respuesta convincente y efectiva.

De igual manera, Viviana Polo subraya el tema de que los alumnos puedan compartir el conocimiento entre ellos. Esto genera una mirada más grupal y objetiva, y es la base del conocimiento asimilado. En sus palabras, "*no importa si van a dedicarse a programar, pero sí que puedan saber qué solución merecería cada escenario que se les presente, para que puedan entenderse con otros miembros del equipo, ya que* ***la aplicación práctica de nuestra carrera implica una actividad interdisciplinaria***". En este sentido, la participación activa en el foro estaría motivando el empleo de conocimientos ya asimilados, puesto que son los mismos alumnos quienes se apoyan entre sí para resolver cada ejercicio planteado.

Entre los puntos negativos de la metodología, algunos profesores indican que el método planteado por sí solo podría no alcanzar para conseguir un nivel de programación intermedio o avanzado, y que en la práctica podría limitarse simplemente a un mero repaso de los conceptos más esenciales de la programación, dejando de lado otros aspectos muy importantes y necesarios del área.

En este sentido, Pérez León sugiere que esta metodología podría únicamente funcionar con cursos iniciales que se enfrentan por primera vez a un lenguaje de programación. Y que el principal desafío a resolver en este caso, sería el de encarar estos cursos no homogéneos donde seguramente aparezcan "***alumnos más avanzados, que no se sientan exigidos por la materia y se terminen desmotivando con la disciplina en general***".

Para estos casos, él propone que "*podría resultar de gran ayuda la interfaz gráfica y su componente lúdico, combinando esto con* ***ejercicios específicos orientados a los alumnos avanzados, donde puedan resolver problemas más interesantes***". Así, estos estudiantes pueden mantener un ritmo cómodo y estimulante en las clases, y convertirse a la vez en ejemplos para otros alumnos.

Sin embargo, Saeed observa que, según su propia experiencia como docente, la realidad demuestra que "*para un estudiante de primer o segundo año, es más valioso conocer y tener experiencia con un lenguaje profesional -por ejemplo, JavaScript-, que limitarse simplemente a programar en pseudocódigo"*. Si bien, estas diferencias ya se expusieron en el capítulo 2.2.3, una vez más reflota la problemática del aprendizaje versus las posibilidades laborales que ofrece el mercado actual.

En relación a ello, es el mismo docente quien afirma que "*esto pasa en muchos otros ámbitos, donde la inmediatez y las ganas de conseguir trabajo te llevan a -por ejemplo- priorizar el aprendizaje del Illustrator, antes que tener buen criterio de la tipografía. Por eso, toda metodología tiene sus pros y sus contras. Y si uno se toma medio cuatrimestre para que los alumnos vayan metiendose de a poco en el tema, pero en realidad apenas están programando, te perdiste medio año desde el punto de vista metodológico, y tal vez podrías haber aprovechado el tiempo de otra manera*".

De todas formas, él reconoce que para el alumno graduado, una materia que lo ayude a pensar y reflexionar, será mucho más valiosa que una materia netamente técnica. Ya que mientras la última puede generar brillantes operadores, la primera sin dudas permitirá la aparición de sabios líderes de proyectos, con la "***mente abierta en la disciplina***". Por el contrario, el operador, por más práctica y habilidades que posea con su herramienta, no podrá igualar jamás el nivel de conocimiento de aquel preparado que, como dice Ken Bain, "***responde a los problemas con creatividad y eficacia, teniendo siempre confianza en sus decisiones***".

En relación a este tema, Hernán Beati afirma que "*no creo que el nivel de los alumnos se altere por el uso de una herramienta u otra. Por el contrario, creo que el nivel logrado es fruto exclusivamente de la ejercitación y el esfuerzo por conceptualizar cada vez mayores niveles de abstracción. En ese sentido, creo que* ***la herramienta MarioCode ayuda a un nivel introductorio básico, previo al de programar en un lenguaje, pero no reemplaza a ese lenguaje***".

Al respecto, se declara que el pseudocódigo no busca reemplazar otro lenguaje, sino que simplemente ayuda a pensar un problema dado, desde un punto de vista computacional. De hecho, la sintaxis de MarioCode está escrita en idioma español, para que el alumno se sienta identificado con la tarea planteada, razone el problema criteriosamente, y no se sienta desmotivado por un enunciado y una resolución técnica poco amigable. Especialmente, si se tienen en consideración las conclusiones aportadas por las pruebas PISA 2013, que pregonan la falta de comprensión de los estudiantes argentinos en los enunciados dados por sus profesores.

En este sentido, Pérez León reflexiona que "*lamentablemente, es cada vez más común encontrarse con alumnos de cursos avanzados que no manejan conceptos básicos de programación, sobre todo en estructuras iterativas o condicionales. A veces esto se debe a que los cursos iniciales -y algunos más avanzados también- consisten en introducir al alumno a la programación a través de herramientas graficas -IDEs- en las que el conocimiento de los conceptos elementales de la programación se ven postergados por priorizar la producción por parte de los alumnos de ejercicios visuales desarrollados únicamente por la interfaz gráfica de estas IDEs.*"

El mismo docente termina por concluir que para corregir este defecto "***sin dudas resultaría de mucho mayor beneficio el uso de herramientas como MarioCode, donde sí existe un componente grafico destacado, pero donde la prioridad está más enfocada en la incorporación de los conceptos de programación por el alumno".***

Por ese motivo, a lo largo de esta investigación se insistió continuamente en el desarrollo del razonamiento y la construcción del aprendizaje, evitando los casos ejemplificados que motivan las soluciones estandarizadas y las respuestas memorizadas.

En conclusión, la metodología planteada presenta una serie de ventajas muy importantes para el alumno, que están directamente arraigadas en su propia debilidad: **programar en pseudocódigo no es programar. Es sentar las bases cognitivas necesarias para programar. Es la acción de pensar, analizar, imaginar y probar escenarios reales en un entorno no real**.

Como se aclaró anteriormente, la resolución de problemas utilizando herramientas de programación comerciales y/o profesionales, no son ambición de esta investigación, y deberán ser parte de los objetivos de futuras asignaturas de cada alumno.

Se dispone de esta manera, porque esta propuesta tiene como único fin, el introducir al alumno en el diseño de los algoritmos y de los lenguajes de programación, enseñandole a diferenciar y fragmentar los distintos problemas computacionales para resolverlos eficientemente uno a la vez. **No se busca sacar programadores, sino preparar futuros programadores.**

Sin embargo, como hemos visto, no es sencillo combinar todas las variables pedagógicas para lograr el escenario ideal, donde los alumnos se sientan cómodos para razonar, analizar y experimentar con los contenidos de la asignatura.

Tampoco será fácil contar con docentes capaces de guiarlos a traves de un aprendizaje profundo, ni es un objetivo que se pueda realizar en el corto plazo. Requiere tiempo y dedicación de ambas partes, que a la larga terminarán dando sus frutos, formando **jóvenes profesionales talentosos, curiosos, seguros, y especialmente, motivados en continuar aumentando sus propios conocimientos sobre la disciplina.**

BIBLIOGRAFÍA

- Aguilar, L. (2005). «Fundamentos y lenguajes de Programación». McGraw Gill. Segunda edición.

- Bain, K. (2007). «Lo que hacen los mejores profesores de universidad». España. Universidad de Valencia. Segunda edición.

- Barreto Tovar, C., Gutiérrez Amador, L., Pinilla Díaz, B. y Parra Moreno, C. (2006). «Limites del constructivismo pedagógico». Educación y Educadores. Vol. 9.

- Bowman, C. (1999). «Algoritmos y estructuras de datos». Oxford University Press.

- Burón Orejas, J. (2002) «Enseñar a Aprender: Introducción a la metacognición». Ediciones Mensajero.

- Deci, E. (1970) «Effects of Externally Mediated Rewards on Intrinsec Motivation». Journal of Personality and Social Psychology. Vol. 18, pág. 105-115.

- Duart, J. y Sangrà, A. (2000) «Aprender en la virtualidad». Edicions de la Universitat Oberta de Catalunya. Primera edición.

- Feldman, T. (1994) «Multimedia». Edición Reimpresa, BluePrint.

- Ferreira Szpiniak, A. y Rojo, G. (2006). «Enseñanza de la programación». Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología. Vol. 1, pág. 71-78.

• González Maura, V. (2004) «La orientación profesional y curriculum universitario. Una estrategia educativa para el desarrollo profesional responsable». Laertes.

• Grijalva, A. (1999). «Reflexiones sobre Pedagogía Universitaria Transformar o reflejar las realidades andinas ». La educación en el siglo XX. Bulletin de l'Institut Français d'Études Andines. Instituto Francés de Estudios Andinos (IFEA).

• Halloum, I. y Hestenes, D. (1985) «Common Sense Concepts about Motion». American Journal of Physics 53. Pág 1056-1065.

• Kamii, C. (1983). «El Conocimiento físico en la educación preescolar». Siglo XXI de España.

• Landau, M. (2001). «Los proyectos nacionales de integración de las TIC en el sistema educativo». Ministerio de Educación, Ciencia y Tecnología de la República Argentina.

• Lerner, D. (1996). «La enseñanza y el aprendizaje escolar. Alegato contra una falsa oposición». Paidos.

• Llorente, M. (2006). «El Tutor en e-learning: aspectos a tener en cuenta». Edutec. Revista Electrónica de Tecnología Educativa N20.

• López Pérez, R. (2003). «Idea de Constructivismo». Instituto de la Comunicación e Imagen. Universidad de Chile.

• Manovich, L. (2006). «El lenguaje de los nuevos medios de comunicación. La imagen en la era digital». Paidos.

• Marín, V. Reche, E. (2011). «Universidad 2.0: actitudes y aptitudes ante las TIC del alumnado de nuevo ingreso de la escuela universitaria de magisterio de la UCO». Píxel-Bit. Revista de Medios y Educación, Vol. 40, pág. 197-211.

• Marquès Graells, P. (2000) «Funciones de los docentes en la sociedad de la información». Revista Sinergia, Vol. 10, pág. 5-7.

• McDonald, K. (1987). «Science and Mathematics Leaders Call for Radical Reform in Calculus Teaching». Chronicle of Higher Education. Pág. 1.

• Mongui Naranjo (2010). «Guía de Aprendizaje en programación». Servicio Nacional de Aprendizaje SENA. CIMM Sogamoso.

• Morduchowicz, R. (2008). «La generación Multimedia. Significados, consumos, y prácticas culturales de los jóvenes». Paidos.

• Oliveira, J. y Rumble, G. (1992) « Vocational education at distance. International Perspectives» Kogan Page.

• Pérez, P. y López, M. (2007) «Multiparadigma en la enseñanza de la programación». Universidad Nacional del Comahue.

• Phillips, R. (1997) «The Developer Handbook Interactive Multimedia» Kogan Page.

• Piscitelli, A.; Adaime, I. y Binder, I. (2010) «El proyecto Facebook y la posuniversidad. Sistemas operativos sociales y entornos abiertos de aprendizaje». Colección Fundación Telefónica.

• Ramírez Toledo, A. (2007) «El constructivismo pedagógico». Educar Chile.

• Rodríguez Almeida, M. (1991) «Metodología de la programación a través de pseudocódigo». McGraw-Hill.

• Schön, D. (1992) «La formación de profesionales reflexivos. Hacia un nuevo diseño de la enseñanza y el aprendizaje de las profesiones». Paidos.

• Schutz, A. (1995) «El Problema de la Realidad Social». Amorrortu.

• Timarán Pereira, R., Jiménez Toledo, A., Checa Mora, J., Ordóñez Erazo, H. y Colunge, C. (2009). «Un cambio de paradigma en la enseñanza de fundamentos de programación en Ingeniería de Sistemas» Asociación Colombiana de Facultades de Ingeniería, Revista Educación en Ingeniería. Vol. 7.

• Villar F. (2003). «El enfoque constructivista de Piaget». Psicología Evolutiva y Psicología de la Educación. Cap. 5.

REFERENCIAS ELECTRÓNICAS.

1. Aprendiendo a programar en Flash – Universitat Oberta de Catalunya. http://mosaic.uoc.edu/2007/09/20/aprendiendo-a-programar-en-flash consultado el 01/05/2012.

2. Algoritmia – Introducción a los Algoritmos. http://www.algoritmia.net/articles.php?id=30 consultado el 03/09/2013

3. Constructivismo –Universidad Virtual del Sistema Tecnológico de Monterrey. http://www.cca.org.mx/profesores/cursos/cep21-tec/modulo_2/constructivismo.htm consultado el 10/06/2012

4. Diseño y Programación Multimedia – Universidad Tecnológica de Chile (INACAP). http://www.inacap.cl/tportalvp/?t=230&i=2&cc=13162&tm=2 consultado el 25/04/2012

5. El enfoque constructivista de Piaget – Perspectiva Constructivista de Piaget. http://docencia.uagro.mx/sites/default/files/Piaget%20Vigotski%20Y%20Maturana%20-%20

Constructivismo%20A%20Tres%20Voces_8.pdf consultado el 30/05/2012

6. El Foro Virtual como espacio educativo – Universidad de Salamanca. http://www.quadernsdigitals.net/datos_web/hemeroteca/r_1/nr_662/a_8878/8878.html consultado el 12/01/2014

7. Estudiá Multimedia – Universidad Maimónides. http://estudiamultimedia.maimonides.edu/perfil-profesional consultado el 25/04/2012

8. Fundamentos de Programación – Instituto Tecnológico de Tijuana. https://sites.google.com/site/fundprog11211268/3-mapa-mental/home consultado el 04/01/2014

9. GNU General Public Licence – Wikipedia. http://es.wikipedia.org/wiki/GNU_General_Public_License consultado el 07/01/2014

10. Ingeniería en Sistemas – Universidad Tecnológica Nacional (UTN). http://www.frd.utn.edu.ar/ingenieria-en-sistemas-de-informacion consultado el 25/04/2012

11. Introducción a los algoritmos – Universidad Nacional de Colombia. http://aplicaciones.virtual.unal.edu.co/drupal/files/Introduccion%20a%20los%20Algoritmos_Programacion%20de%20computadores.pdf consultado el 03/07/2012

12. Lenguajes de Programación – ECURed. http://www.ecured.cu/index.php/Lenguaje_de_Programaci%C3%B3n consultado el 10/01/2014

13. Nativos Digitales y modelos de aprendizaje – Universidad de País Vasco / Euskal Herriko Unibertsitatea (UPV/EHU). http://spdece07.ehu.es/actas/Garcia.pdf consultado el 15/01/2014

14. Mario Bros (personaje) – Wikipedia. http://es.wikipedia.org/wiki/Mario_(personaje) consultado el 15/01/2014

15. Paradigmas de Programación – Instituto Tecnológico de Celaya. http://www.iqcelaya.itc.mx/~vicente/Programacion/Paradigmas.pdf consultado el 10/06/2013

16. Paradigma Declarativo –Departamento de Computación, Universidad de Coruña. http://lpr31.blogspot.com.ar/2006/11/paradigma-declarativo.html consultado el 10/06/2013

17. Programación de Computadoras – Universidad Tecnológica Nacional (UTN). http://www1.frm.utn.edu.ar/pcomputadoras/ consultado el 04/01/2014

18. Pruebas PISA: es la hora de comprender lo que pasa – Diario Clarín. http://www.clarin.com/sociedad/hora-comprender-pasa_0_1042095852.html consultado el 26/01/2014

19. Teorías del aprendizaje: Origen del construccionismo como Teoría del aprendizaje OLPC. http://www.olpcmexico.org/2011/10/teorias-del-aprendizaje-origen-del.html consultado el 26/02/2014

20. Tecnicatura en Programación de Sistemas – Universidad de Ciencias Empresariales y Sociales. http://www.uces.edu.ar/grado/ciencias_empresariales/programacion/programacion.php consultado el 25/04/2012

21. Tecnicatura Universitaria en Programación Informática – Universidad Nacional de Quilmes (UNQ). http://www.unq.edu.ar/layout/redirect.jsp?idSection=3282 consultado el 25/04/2012

22. Tecnología Multimedial – Universidad Maimónides. http://www.maimonides.edu/carreras_ficha.asp?car=8&conte=1 consultado el 25/04/2012