Màster en **Formació del Professorat d'Educació Secundària Obligatòria i Batxillerat, Formació Professional i Ensenyament d'Idiomes**
Curs 2010 / 2011

Treball de fi de màster

Títol: Utilització del software constructiviste Etoys com a ajuda i complement als cicles formatius de GM i GS d'automatització Industrial. Unitat Formativa: Máquines Elèctriques

Cognoms: Mañas García

Nom: Santiago

Titulació: Màster en Formació del Professorat d'Educació Secundària Obligatòria i Batxillerat, Formació Professional i Ensenyament d'Idiomes

Especialitat: Formació Professional (FP)

Director/a: Mireya Fernández Chimeno

Data de lectura: 27 Juny 2011

Índice

Introducción

La verdadera revolución de los ordenadores, no ha comenzado todavía
Alan Kay, Turing Award. [3]

En cualquier lugar donde se mire, en los medios, en internet, por la calle, en conversaciones con amigos, se habla de revolución digital, de las tecnologías de la información y comunicación (TIC) , de las tecnologías de aprendizaje y conocimiento (TAC), de nativos y no nativos digitales. Todos estamos muy impresionados por la forma en que la informática ha cambiado y está cambiando nuestras vidas, nuestra forma de entender el mundo y la comunicación con los demás. Cambios que parecen abrir nuevos mundos, nuevos intereses, nuevas oportunidades educativas y de crecimiento...

Ante esta situación, parece de primera necesidad la integración de estas nuevas tecnologías en el aula. En primer lugar porque estas herramientas informáticas pueden ser una ayuda inestimable en el proceso de enseñanza-aprendizaje. En segundo lugar porque los alumnos de hoy van encontrar un mundo laboral que les exigirá no solo conocimientos, sino un dominio ágil de la caja de herramientas informática.

Si nos centramos en sus aspectos positivos, Internet es, a día de hoy, el referente de búsqueda de información. No solo eso, sino que dispone de espacios donde crear y compartir información (”nubes” como Dropbox, por ejemplo). Vídeos y fotos que cualquiera puede colgar o contemplar (Flickr y YouTube). Foros de opinión y especializados. Bitácoras personales o Blogs. Plataformas académicas (moodle). Áreas de visita virtual a museos y exposiciones. Suscripciones a noticias y novedades en áreas de interés específicas seleccionadas por el propio usuario (RSS). Redes sociales (Facebook, Twitter). Una oferta cursos a distancia (e-learning) cada vez más amplia y profesional. Y otras cosas que, seguramente he olvidado añadir.

Pero eso no es todo. Utilizando internet como dominio de búsqueda dirigido en el aula, pueden pilotarse experiencias formativas y creativas tan interesantes, a nivel educativo, como las WebQuest, que comenta Jordi Adell:

> En una WebQuest se divide a los alumnos en **grupos**, se le asigna a cada grupo un **rol** diferente y se les propone realizar una **tarea**, que culminará en un **producto** con características bien definidas. Para ello seguirán un **proceso** [...] planificado previamente por el profesor, durante el cual los alumnos realizarán una **amplia gama de actividades** como leer, comprender y sintetizar información seleccionada de la Internet o de otras fuentes, organizar la información recopilada, elaborar hipótesis, valorar y enjuiciar ideas y conceptos, producir textos, dibujos y presentaciones multimedia, objetos físicos, manejar aparatos diversos, entrevistar a sus vecinos, etc. Durante el proceso el profesor les propondrá el uso de diversos **recursos**, generalmente accesibles a través de Internet [...] y, cuando sea necesario, una serie de ayudas o **andamios** de recepción, transformación y producción de información que les ayudarán a asimilar y acomodar la nueva información y a elaborar el producto final. [1]

Metodología, metodología, metodología

El por qué de la necesidad de introducir las tecnologías de la información en las aulas parece estar muy claro. Lo que ya no lo está tanto es como llevarlo a cabo. Es en esta fase de realización práctica donde el proceso de integración de las TIC y las TAC en las aulas puede resultar en experiencias positivas y enriquecedoras o en un auténtico desastre.

> *What’s wrong with education, cannot be fixed by technology.*
> Steve Jobs

En primer lugar, la caja de herramientas informática no garantiza nada por si misma. Actúa, más bien, como un amplificador. Puede ayudar a mejorar y a facilitar los procesos de enseñanza-aprendizaje bien diseñados, pero también puede hacer mucho peores los diseños mediocres. Lo importante, antes y ahora, es la metodología, que debe tener en cuenta el medio empleado, pero que está siempre por delante del mismo.

Si el proceso de enseñanza-aprendizaje está bien diseñado para hacer un uso con sentido común de las TIC y las TAC, estas no harán más que ayudar, familiarizando, de paso, a los alumnos con la tecnología. Es el caso, por ejemplo, de las WebQuest bien planteadas.

Si, por el contrario, no hay una idea clara de como integrar o utilizar estas nuevas tecnologías, el mero hecho de introducir los ordenadores en el aula no mejorará nada. Es muy posible, incluso, que métodos tradicionales que, hasta entonces habían estado funcionando bien, dejen de hacerlo al verse desplazados por pretendidas mejoras que no son tales porque no hacen un uso adecuado, o que hacen, incluso, un uso involuntariamente pernicioso de la caja de herramientas informática. En opinión de Fernando Fraga y Adriana Gewerc, de la Universidad de Santiago de Compostela:

> En experiencias anteriores hemos constatado que los ordenadores producen una cierta "encantación" en profesores/as ya formados y en los que están en formación, creándose la falsa idea de que la sola utilización del aparato garantiza procesos de innovación en las aulas. Hemos tratado de luchar contra esta idea implantada con tanta fuerza, no siempre con éxito. Por eso, en las investigaciones que realizamos en las escuelas, con el objeto de indagar qué está sucediendo y en que medida la utilización de las TIC está significando un cambio de paradigma de enseñanza, nos devuelve una cruda realidad: **las propuestas son semejantes a las que se hacían con texto, pero ahora se hacen frente a una pantalla, y quizás con peores resultados**. [2]

Se llega a la paradoja de que algunos de los programas de ordenador incluidos en un proceso de enseñanza-aprendizaje que se precia de un enfoque constructivista, como es el nuestro, se parecen mucho a realizaciones, actualizadas utilizando ordenadores, de las máquinas "de enseñar" de B. F. Skinner, que, como es sabido, son más bien máquinas de "enseñar la respuesta correcta" en el más puro estilo conductista.

> En [...] 1954, Skinner ya señalaba que dividiendo las etapas de aprendizaje de una tarea en pequeños pasos, y que estos recibiesen reforzamiento contingente, podría incrementar la frecuencia de respuestas correctas [...]. Para ello sugirió que las "maquinas de enseñanza" podrían presentar ese material en pequeños pasos y proporcionar reforzamiento inmediato sobre cada respuesta del alumno. En aquellos momentos la tecnología de ordenadores no se había iniciado siquiera, sólo se utilizaban relés y mecanismos semi-eléctricos. [7]

Naturalmente, una buena metodología debe tener en cuenta las características del medio de transmisión empleado. No para adaptarse a esas características, sino, más bien para hacer un uso adecuado de las mismas. Para "sacarles todo el jugo" dicho en términos sencillos.

> *A good rule of thumb for curriculum design **is to aim at being idea based, not media based**. Every good teacher has found this out. Media can sometimes support the learning of ideas, but often the best solutions are found by thinking about how the ideas could be taught with no supporting media at all. Using what children know, can do, and are often works best. After some good approaches have been found, then there might be some helpful media ideas as well.*
> Alan Kay

Tampoco hay que perder de vista y este es un punto que considero particularmente importante, que el uso de las TIC y de las TAC en el mundo de la enseñanza, debe estar orientado a mejorar todo el proceso de enseñanza-aprendizaje. La adquisición de habilidades tecnológicas y el uso fluido de la caja de herramientas informáticas por parte de los alumnos es algo que se dará por añadidura. Es solo un medio. Aunque no un medio más, sino uno que, bien empleado puede resultar particularmente potente y mal utilizado especialmente pernicioso.

Por otra parte, el proceso de enseñanza-aprendizaje, sobre todo en las primeras etapas de primaria y educación secundaria obligatoria, no puede estar enfocado pensando únicamente en el mundo laboral, cuyas necesidades pueden cambiar de un día para otro, sino en construir personas, en el mejor sentido de la palabra, que sean capaces de adaptarse a esas condiciones cambiantes.

> *The school's real job is to focus on "how to make a life" which is quite different from "how to make a live"*
> Neil Postman

> *Childrens are not messages we send to the future. Childrens are the future we send to the future.*
> Alan Kay

Personas con sentido critico, con imaginación, capaces de hacer frente con lógica y sentido común a situaciones absolutamente nuevas e imprevistas. Personas con una curiosidad generalista, despierta e inquisitiva que les ayude a continuar formándose a lo largo de toda su vida.

De nada servirá que aprendan a utilizar las herramientas que hoy son la punta de lanza de la tecnología informática si no son capaces de adaptarse en el futuro a nuevas herramientas que quizá poco tengan que ver con las actuales.

> En el pasado la mayor parte de la gente dejaba el mundo apenas ligeramente diferente de como estaba cuando lo encontró. El rápido y acelerado cambio que marca nuestro tiempo significa que todas las personas, en periodos cortos de tiempo, verán cambios mayores que los que las generaciones anteriores vieron en toda su vida. Entonces esta es la elección que debemos hacer nosotros mismos, para nuestros hijos, para nuestros países y para nuestro planeta: adquirir las habilidades que se necesitan para participar comprendiendo la construcción de lo que es nuevo, o resignarse a una vida de dependencia. [4]

¿Pero quien es el dichoso Alan Kay?

Como menciona la cita inicial de este trabajo, según Alan Kay la "verdadera" revolución de los ordenadores no ha comenzado todavía".

¿Quien es Alan Kay?¿A que se refiere cuando dice esto?¿Cual sería la "verdadera" revolución de los ordenadores, que aún no se ha producido?

En primer lugar, aunque Alan Kay no sea tan conocido a nivel popular como Bill Gates o Steve Jobs, puede afirmarse, sin temor a equivocarse, que se trata de uno de las padres fundadores, y uno de los principales, de la revolución digital. Su trayectoria profesional lo hace alguien digno de ser atendido con interés y escuchado con calma cuando se refiere a estos temas. Así que merece la pena que nos detengamos y prestemos atención a lo que dice:

> La sociedad de la información del futuro podría ser como la sociedad de la televisión de hoy en día, o incluso mucho peor. O podría ser tan buena que solo pudiese ser comparada con los cambios provocados por la aparición de la imprenta en la gran transición desde la Edad Media a la Edad Moderna. La primera ruta es fácil, y es deseada tanto por los intereses comerciales como por la mayoría de las personas. La segunda es más difícil y los sistemas educativos del mundo necesitarán cambiar drásticamente para alcanzar ese objetivo [3]

A mediados de la década de 1960 (la década prodigiosa, la de los hippies, los Beatles, Bob Dylan y el Mayo francés), Alan Kay fue el creador de uno de los primeros lenguajes de programación orientado a objetos: Smalltalk.

No solo creó ese lenguaje, sino, mucho más importante, acabó de perfeccionar el concepto de programación orientada a objetos basado en clases, métodos, herencia, polimorfismo y encapsulación. Ese nuevo paradigma, la programación orientada a objetos, se sigue utilizando hoy en día como la mejor forma conocida de encarar la programación de ordenadores.

En los años setenta siguiendo varios de los principios de Jerome Bruner, (padre de la psicología cognitiva del descubrimiento), sobre el aprendizaje y la acción, y en estrecha colaboración con Dan Ingalls y otros miembros del equipo del Palo Alto Research Center (PARC) en Xerox, desarrolló, tras cinco años de experimentación, de interacción y de prueba y error, **la interfaz gráfica de usuario, prácticamente tal como la conocemos hoy**: escritorio, ratón, representación gráfica de los documentos, ventanas móviles, redimensionables y deslizantes, copiar y pegar, la selección precede a la orden, lo que ve es lo que obtiene, etc.

Esa interfaz gráfica, o sus conceptos fundamentales, es la que comercializaron a principios de la década de 1980, Apple, primero con Lisa y después con el Macintosh, Atari con su ST, y Commodore con el Amiga. Hubo que esperar casi quince años más para que Microsoft

presentase al mercado Windows 95. La primera versión de Windows equiparable a lo que ya hacían, en su momento, los primeros ordenadores con interfaz gráfica. (Las versiones anteriores parecían una broma, más que un sistema operativo).

¡Para que luego haya quien piense que la revolución digital se la debemos a Bill Gates!

Como detalle curioso, que da una idea de la seriedad de la apuesta de los tres fabricantes pioneros, cabe indicar que cuando el resto de ordenadores personales utilizaba un *bus* de 8 bits, Lisa, Macintosh, ST y Amiga utilizaron ya uno de 32. Se saltaron directamente el paso intermedio por los 16 bits que los PCs basados en texto sí realizaron ¡algunos años después! Dieron un salto adelante tan absolutamente revolucionario, y que rompía de tal manera con todos los paradigmas anteriores de la informática, que aún hoy no ha sido superado. Nuestra actual revolución digital, sigue viviendo de aquellas rentas.

Como la gente y la sociedad en general presenta una inercia al cambio considerable, los saltos adelante, incluso los mejores como era el caso, pueden resultar peligrosos. De los cuatro ordenadores pioneros en venir dotados con sistema operativo de interfaz gráfica: Lisa, Macintosh, Amiga y ST, solo ha sobrevivido hasta nuestros días el Macintosh. En realidad lo que ha sobrevivido es el concepto, puesto que los Macs de hoy son mucho más potentes que los primeros Macs y, además funcionan sobre UNIX. Sin embargo, toda la interfaz ya estaba ahí desde los modelos de 1984. Era en blanco y negro y funcionaba sobre una pantalla de 9 pulgadas, pero se manejaba prácticamente igual que ahora.

> No es una exageración afirmar que la interfaz gráfica de usuario fue la invención más importante en computación personal, ya que ha permitido que **más de 2.000 millones de personas** puedan usar los ordenadores [3].

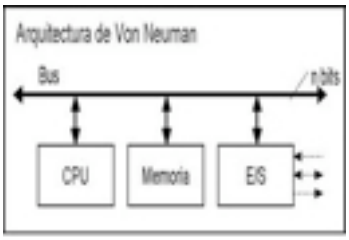
Y es, precisamente Alan Kay, uno de los padres de la interfaz gráfica de usuario, pionero de la revolución digital y alguien que dijo aquello de “la mejor manera de predecir el futuro es inventarlo”, entendiendo por “inventarlo” el hecho de “crearlo” como novedad absoluta. Es precisamente Alan Kay, quien tiene una visión más crítica y pesimista sobre el estado actual de nuestra tan traída y llevada sociedad digital, que tanto nos da que hablar a todos.

Cabría preguntarse, por un lado, qué es lo que Alan Kay entiende por “verdadera revolución de los ordenadores” y, en el caso de que esa sea una mejor alternativa a la situación actual, qué cambios deberían producirse en los sistemas educativos y, especialmente, si alguno de estos cambios se está produciendo ya. Si hay alguna experiencia prometedora en este campo.

Cedamos la palabra a Seymour Papert, creador del lenguaje de programación Logo. No está de más recordar que Seymour Papert trabajó con Jean Piaget en la Universidad de Ginebra desde 1959 hasta 1963, año en que fue invitado a unirse al MIT donde, junto con Marvin Minsky, fundó el Instituto de Inteligencia Artificial.

> En esa conferencia me quejé del efecto perjudicial que tiene sobre la cultura popular el uso del nombre “tecnología de la información”, para referirse a lo que más adecuadamente se debería llamar “tecnología digital”. [...] para la mayoría de la gente la palabra “información” tiene el significado popular de obtener algo que informa. Pero la mayor parte del uso que se da a las computadoras, nada tiene que ver con la información entendida en ese sentido. Piénsese en la construcción de una nave espacial. La tarea de diseñar el transbordador espacial sería muy compleja para que la mente humana la manejara sin la ayuda de computadores [...]
>
> En resumen, me gusta reconocer [...] dos aspectos de la tecnología digital: la tecnología como medio de información y la tecnología como medio de construcción. [...] Claro que los dos aspectos son de igual importancia, pero la percepción popular está dominada por el de información, porque es el que la gente ve y del que oye hablar incesantemente, y ese es el que refleja el papel predominante que los medios de información tienen en sus vidas.
>
> Permítanme ahora volver a la educación, para reconocer que esa unilateralidad en la percepción de la tecnología ha producido una profunda distorsión en lo que piensa la gente sobre su contribución a la educación. Esto ha sucedido porque la misma educación tiene dos variantes que también podrían llamarse de “información” y de “construcción”. Parte del aprendizaje consiste en obtener información, que puede provenir de leer un libro, o de escuchar a un profesor, o de visitar sitios en la Web. Pero eso es solamente una parte de la educación. La otra parte consiste en hacer cosas, producir cosas, construir cosas. Sin embargo también aquí se presenta un desequilibrio: en gran parte a causa de la ausencia de tecnologías adecuadas, el lado constructivo de la educación en las escuelas se ha quedado rezagado, y ocupa un lánguido segundo lugar frente al dominante lado informativo. [4]

Un poco de historia



Los ordenadores son máquinas conceptualmente sencillas constituidas básicamente por una unidad central de procesamiento, una memoria y periféricos de entrada y salida.

Sin embargo, son, a gran distancia, las máquinas creadas por la humanidad, capaces de realizar las tareas más complejas. Y eso por un solo motivo: son máquinas programables.

Los programas de ordenador son las estructuras mayores y más complejas que ha creado el genero humano. Tomando como unidad elemental por un lado el bit y por otro el ladrillo o la piedra, la inmensa mayoría de programas de ordenador resultan mucho más complejos que las mayores catedrales góticas [6].

La programación de ordenadores no es un trabajo sencillo. No es nada fácil explicarle a una máquina que básicamente solo saber sumar y comparar, lo que tiene que hacer. Como es natural, la máquina carece absolutamente de criterio. (Eso que intentamos fomentar en nuestros alumnos para que sean personas en todo el sentido de la palabra). Como la máquina es incapaz de decidir por si misma, el programador tiene que prever y plantear todas y cada una de las posibles contingencias que puedan presentarse y programar la respuesta de la máquina a todas y cada una de ellas.

Además, el programa de ordenador, que al fin y al cabo es una secuencia de instrucciones de funcionamiento para todas y cada una de las posibilidades de entrada y salida, debe estar constituido por bits, para que la máquina lo entienda.

Es posible realizar programas así, son los programas en código máquina. Este tipo de programación tiene graves inconvenientes, los programas son muy laboriosos de realizar y, como consecuencia, es casi imposible hacer programas largos y complejos. Pero hay algo que los hace aún mucho peores: presentan grandes dificultades a la hora de introducir modificaciones o mejoras.

Ya puede entenderse que es muy poco agradecido orientarse en un programa constituido únicamente por unos y ceros y aunque se pueda realizar un esfuerzo, más bien ímprobo, a la hora de crear el programa, enfrentarse a uno de estos programas para modificarlo sustancialmente, sobre todo si ha sido hecho por otra persona, o incluso por uno mismo unos meses antes, es misión imposible.

Aunque se dio un primer paso mediante la utilización de códigos mnemotécnicos en lo que se llamó lenguaje ensamblador, lo que permitió superar definitivamente estos obstáculos fue la creación de los lenguajes de programación de alto nivel.

En un lenguaje de alto nivel las instrucciones y estructuras de control se escriben utilizando un lenguaje con cierta similitud con un lenguaje natural (generalmente el inglés) que luego un programa especializado, el compilador, se encarga de traducir a código máquina. (Los primeros compiladores, como puede comprenderse, fueron escritos en ensamblador).

Estos lenguajes de programación tienen, forzosamente, una morfología y unas reglas de escritura muy precisas que es imprescindible respetar para que el compilador acepte nuestro programa escrito en ese lenguaje de programación, (se dice que está escrito en código fuente), y lo convierta a código máquina. Aún así, queda por comprobar que el programa ya compilado haga todo aquello que queríamos que hiciese y tal y como queríamos que lo hiciese, pero esa es ya otra historia...

Desde luego es mucho más fácil hacer programas de ordenador utilizando algún lenguaje de alto nivel que mediante código máquina o ensamblador. Los programas pueden ser, consecuentemente, mucho más complejos y el mantenimiento mejora sustancialmente. No obstante, sigue siendo un trabajo donde hay que hilar muy fino. No es apto para usuarios casuales.

Los primeros lenguajes de programación de alto nivel, aparecieron a en la transición de la década de 1950 a la de 1960 y fueron FORTRAN, especializado en programas de cálculo, y COBOL, para gestión y bases de datos. Como dato curioso, los primeros programadores eran renuentes a utilizar FORTRAN porque pensaban que el código generado por su compilador no sería nunca tan compacto y eficaz como el código máquina hecho a mano. Realmente tenían razón, pero, finalmente, la facilidad de uso y mantenimiento acabó imponiéndose.

A finales de 1960 surgió el concepto de programación estructurada, cuya intención era que los programas realizados siguiendo esos criterios fuesen más claros, más fáciles de seguir y más fáciles de modificar. Los criterios de programación estructurada se basan en que todo programa puede realizarse mediante solo tres estructuras: secuencia, selección e iteración. Se evita así el uso de las instrucciones de salto que convierten los programas que las utilizan en prácticamente ilegibles a menos que sean extremadamente simples. Los lenguajes de programación, más conocidos, nacidos bajo este concepto y que, hasta cierto punto, fuerzan su uso, son Pascal y C.

Es interesante destacar, que los propios lenguajes de programación han ido creciendo a medida que transcurren los años y no lo han hecho añadiendo instrucciones, puesto que el conjunto de las mismas ya era completo cuando se crearon, sino añadiendo secuencias de programa ya probadas que permiten realizar funciones muy usuales y comunes a muchos otros programas. Estos programas *pret-a-porter* ya incluidos junto a la mayoría de compiladores se denominan librerias (a causa de una deficiente traducción del vocablo inglés *libraries*) y el compilador puede integrarlos (el término utilizado es el anglicismo “linkarlos”) en los nuevos programas.

Al conjunto formado por un editor de textos, para crear el código fuente, el compilador para obtener el código máquina y el conjunto de programas *pret-a-porter* o librerías, se le llama, generalmente, entorno de programación.

Aparece BASIC

Durante la década de 1980, y debido principalmente al auge de los primeros ordenadores personales, se hizo muy popular el lenguaje de programación BASIC.

BASIC es un lenguaje de programación de alto nivel con la particularidad de que la compilación se realiza instrucción a instrucción. La principal ventaja de esta enfoque es que hace la prueba y modificación de los programas mucho más inmediata y sencilla. Este tipo de compilador instrucción a instrucción se denomina “intérprete”, o también compilador JIT (*Just-In-Time*). El precio a pagar por estas comodidades es la velocidad de ejecución del programa.

BASIC no era un lenguaje estructurado en el sentido de que no forzaba ese tipo de programación, pero un buen programador podía aplicarla por su cuenta. Lo más destacable de BASIC, dada su facilidad para realizar experimentos de prueba y error fue el gran número de aficionados que generó, muchos de los cuales acabaron haciéndose profesionales y derivando hacia lenguajes de programación más potentes y rápidos.

Por primera vez un lenguaje de programación se hacía más o menos accesible a los aficionados. Hay que destacar que a estos aficionados, los primeros usuarios de ordenadores personales, la posibilidad de programar sus máquinas para realizar tareas a su medida les interesaba y que, por regla general, eran personas con cierta formación, afición o propensión técnica.

Merece la pena detenerse aquí un momento. Por primera vez, la programación de ordenadores, empezaba a ser algo accesible a los aficionados. Solo medianamente accesible, puesto que seguía y sigue exigiendo una sintaxis rigurosa que no admite un uso casual, pero mucho más accesible por el hecho de que su facilidad de modificación instantánea permitía y permite un **aprendizaje, por acción, por prueba y error, eficaz y divertido**. ¿No suena esto mucho a constructivismo?

Indudablemente lo es. Y, además, ha demostrado rendir excelentes resultados desde el momento en que muchos de los artífices de la revolución digital que disfrutamos hoy se iniciaron así. Se me ocurre poner como ejemplo, pero es solo uno entre muchos, a Mitch Kapor, el inventor de la hoja de cálculo (VisiCalc), creador de la compañía Lotus y, más recientemente (2001) fundador de la *Open Source Applications Foundation*. También ha sido director (2003) de la fundación Mozilla. Otro ejemplo podría ser el filantrópico Paul Allen, cofundador de Microsoft.

Por estas fechas, aunque algunos programas, especialmente los juegos, empleaban un modo gráfico, más o menos rudimentario, la interacción de los usuarios con sus ordenadores se hacía, básicamente, mediante cadenas de caracteres y teclas de función, a través de teclado y pantalla, como hacemos ahora, pero sin ningún dispositivo de apuntamiento como el ratón.

Lejos quedaban los llamados procesos *batch* donde toda la información se cargaba a la vez sobre el ordenador (generalmente por tarjetas perforadas) y donde este devolvía los resultados sobre interminables listados en papel impreso.

Sin embargo, la interacción con la máquina, aunque era “conversacional”, lo que significa que los programas pedían a los usuarios los datos necesarios desde la pantalla y devolvían los resultados del mismo modo, seguía siendo muy primitiva si la comparamos con la interfaz gráfica actual.

Eso tenia inconvenientes evidentes. Solo los aficionados les veían todas las gracias a los ordenadores personales. El común de la gente los consideraba poco menos que juguetes caros que no servían para gran cosa. Sin embargo, estos aficionados, eran entonces **verdaderos dueños y señores de sus máquinas**.

¿Seguimos siendo ahora dueños y señores de nuestros ordenadores? Es evidente que no. Los verdaderos dueños y señores, no solo de nuestras máquinas sino de toda la tecnología digital en general, son las grandes compañías (Apple, Microsoft, Sony, Hewlett Packard, Dell, Compaq, Adobe, Google y un muy largo etcétera) que supieron, en su momento, llevarse el gato al agua y allí lo mantienen hasta hoy.

Claro que, a cambio, disfrutamos de aplicaciones profesionales que nos permiten, como usuarios y solo como usuarios, sacarle partido a nuestros ordenadores.

La verdadera revolución de los ordenadores

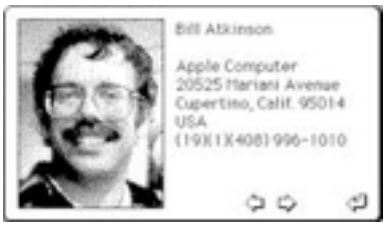
De las dos facetas de la revolución digital que comentaba Seymour Papert, más arriba, nos hemos quedado con la informativa y hemos reducido la constructiva a la de meros usuarios de programas que nos permiten crear documentos de texto, presentaciones, hojas de cálculo, alguna que otra base de datos y, para los profesionales, y usuarios “extremos” programas de CAD, de gráficos en 3D y de diseño gráfico.

No es que hayamos perdido con el cambio. No es, en absoluto, despreciable la utilidad que nos ofrecen hoy nuestros ordenadores. Solo que, quizá seducidos por ello, hemos olvidado una faceta especialmente interesante de nuestros ordenadores que es no solo es usarlos como máquinas más o menos polivalentes sino utilizarlos como simuladores que permiten experimentar con la realidad y la teoría de las ciencias, convertidos en potentes y personales herramientas de aprendizaje.

Pero para ello, naturalmente, hemos de ser capaces de convertirnos en autores. De programar nuestras máquinas, de una u otra forma, para experimentar y aprender de una forma parecida aunque preferiblemente más abierta a la que tantas personas realizaron con BASIC.

No se trata de volver a BASIC, menos aún de utilizar alguna encarnación más reciente, orientada a un entorno gráfico como pudiera ser Visual Basic. Estas últimas versiones que sirven para realizar programas basados en interfaz gráfica vuelven a ser demasiado complejas para usuarios casuales. Han perdido la sencillez y frescura del BASIC original.

HyperCard



En 1987, HyperCard, creado por Bill Atkinson y Dan Winkler, con su orientación hipermedia y su lenguaje de programación HyperTalk, interprete orientado a objetos, representó un fenómeno parecido al del BASIC inicial con la fundamental diferencia de que las aplicaciones funcionaban directamente en un entorno gráfico. HyperCard, gracias a la insistencia de Bill Atkinson que impuso esa condición a Apple, se entregaba gratuitamente con todos los ordenadores Macintosh y tuvo inmediatamente un éxito extraordinario: prácticamente todos los usuarios de Mac nos lanzamos a realizar nuestras propias pilas, que así se llamaban los archivos creados por HyperCard. En HyperCard nacieron juegos como Cosmic Osmo o incluso la primera versión del popular Myst. También se utilizó, incluso en la propia Apple, como base de desarrollo rápido de prototipos de nuevas aplicaciones. Su gran inconveniente era que solo funcionaba en ordenadores Macintosh. No podía ser de otra forma ya que, por aquellas fechas, los PCs no disponían de sistemas operativos con interfaz gráfica lo bastante potente.

Lo menciono aquí como merecido homenaje a Bill Atkinson y a Dan Winckler, pero no vamos a dedicarle más espacio. Solo como curiosidad, me gustaría comentar que HyperCard sirvió de inspiración, varios años más tarde, a los creadores de la World Wide Web y que el cursor en forma de mano con el índice extendido, usual en la web, procede también de HyperCard.

LOGO

LOGO nació en una época tan temprana como 1967. Muy anterior a BASIC y a HyperCard. Fue el primer lenguaje de programación de alto nivel especialmente diseñado como herramienta educativa. Basado en LISP, uno de los primeros lenguajes de inteligencia artificial,

Seymour Papert, uno de sus diseñadores, hizo uso de su experiencia de tres años de trabajo en la universidad de Ginebra con Jean Piaget para crear un entorno de programación constructivista asequible a los niños. La idea de Papert era y es la de presentar a los niños retos intelectuales que puedan ser resueltos mediante el desarrollo de programas en LOGO y, más importante, de su posterior comentario y discusión entre los niños y el profesor, ejerciendo las funciones de tutor.

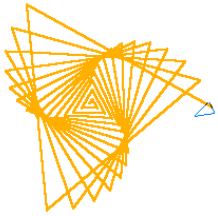
> Programar un ordenador no significa ni más ni menos que comunicarse con este en un lenguaje que tanto la máquina como el usuario humano puedan "comprender". Y aprender lenguajes es una de las cosas que mejor hacen los niños.
> Seymour Papert

Papert, llegando un poco más allá del concepto de constructivismo de su maestro Piaget acuñó un nuevo término: construccionismo. El construccionismo de Papert sostiene que, para que se produzca aprendizaje, el conocimiento debe ser construido, o reconstruido, por el propio alumno a través de la acción.

El construccionismo considera que las actividades de construcción de artefactos son facilitadoras del aprendizaje. Estos artefactos pueden ser desde simples dibujos a programas de ordenador, pasando por cualquier otra construcción humana, como castillos de arena, o poemas.

El construccionismo involucra a los estudiantes y les anima a sacar sus propias conclusiones a través de la experimentación.

Puedo dar fe, personalmente, de la validez de este enfoque a través de mis hijos que se desarrollaron, desde muy temprana edad, “constructivamente” haciendo artefactos de Lego®, muchos de ellos extraordinariamente complejos, diseñados y construidos por ellos mismos sobre la marcha.



En cuanto al lenguaje LOGO en si, se trata de un lenguaje poco exigente, sintácticamente hablando, lo que facilita su uso a los niños, para los que fue creado. La salida de LOGO sobre la pantalla del ordenador son gráficos generados por una tortuga virtual a la que el niño da instrucciones. La tortuga al desplazarse sobre la pantalla puede dejar un trazo de su trayectoria generando así diversas figuras geométricas. Sus instrucciones son sencillas, como *baja el lápiz*, *avanza 100*, *gira 60*... así es muy fácil comenzar, pero, además y como todo lenguaje de programación, LOGO incorpora también bucles y condicionales.

Las figuras geométricas generadas por la tortuga, como quien no quiere la cosa, introducen al alumno en conceptos matemáticos y físicos como signo, magnitud, ángulo, velocidad y aceleración, pero no solo eso, cedamos la palabra a Seymour Papert:

> No es verdad que la imagen que desarrollaré aquí sobre la reacción del niño con el ordenador vaya mucho más allá de lo que es corriente en las escuelas actuales. Mi imagen no va más allá: va en dirección opuesta.[...] En la mayoría de situaciones educativas contemporáneas en las que los niños están en contacto con ordenadores, el ordenador se utiliza para poner al niño a realizar tareas, para proveer ejercicios de un nivel apropiado de dificultad, para proveer retroalimentación, y para dar información. El ordenador programa al niño. En el ambiente LOGO la relación se invierte: El niño, aun a edades preescolares, está en el control: El niño programa el ordenador. Y, al enseñar al ordenador cómo pensar, los niños se embarcan en la exploración de cómo ellos mismos piensan. La experiencia puede ser inquietante: Pensar sobre el pensamiento propio convierte al niño en un epistemólogo, una experiencia no conocida ni siquiera por la mayoría de adultos.

“En la mayoría de situaciones educativas contemporáneas en las que los niños están en contacto con ordenadores, el ordenador se utiliza para poner al niño a realizar tareas, para proveer ejercicios de un nivel apropiado de dificultad, para proveer retroalimentación, y para dar información. El ordenador programa al niño.”

¡Que fácil resulta caer es esta forma negativa de introducir la revolución digital en las aulas!

Como docentes debemos poner todo nuestro empeño en evitar estas situaciones en que el alumno puede llegar a convertirse en servidor de la máquina.

La situación presenta paralelismos con la que se da en el mundo laboral. Hay un uso positivo de los ordenadores como máquinas polivalentes que ayudan en el desarrollo de una actividad profesional. El ordenador es aquí un auxiliar inestimable al servicio de la persona que lo utiliza.

Pero es posible también, y de hecho se produce cada vez con más frecuencia, un uso negativo y perverso cuando el ordenador se utiliza por parte de la dirección como instrumento de control de sus trabajadores.

Squeak, Etoys, Scratch, Alice

¿Como definir Squeak? El concepto de Squeak es tan original que se hace difícil catalogarlo por comparación con cosas más conocidas. Squeak es un mundo, una caja de sorpresas, algo con lo que podrías pasar horas y más horas y siempre acabarías descubriendo algo nuevo. Es un entorno de programación a varios niveles, el más potente directamente en SmallTalk, el más adecuado para enseñanza mediante creación gráfica de objetos y construcción de guiones por ladrillos apilables.

Squeak es Open Source, funciona, como Java, sobre una máquina virtual (VM) lo que permite que un programa de Squeak creado en una plataforma determinada pueda ejecutarse sin problemas en cualquier otra plataforma sin más condición que tener instalada la máquina virtual de esa plataforma.

Es un entorno multimedia que permite la integración de audio, video, fotografías, dibujos y texto. Dispone de herramientas como reproductor MIDI, reproductor MPEG, reproductor y grabador de sonidos, conexión a cámaras de video o editor de formas de onda.

> Todo esto viene favorecido por tratarse de una herramienta que se ha creado bajo el prisma del constructivismo, de la teoría piagetiana, de la visión del aprendizaje con ordenadores de Papert, las investigaciones de J. Bruner, las investigaciones en los 60 y 70 en Xerox-PARC, etc. Y cristaliza en el lenguaje de programación Small-Talk y en el desarrollo de Squeak. [2]

Tiene también vocación de sistema operativo. Dentro del propio Squeak pueden funcionar los programas creados por el propio usuario, pero también hay aplicaciones de correo electrónico, navegadores de red, reproductores multimedia, juegos, calculadora, reloj...

Squeak es, además, absolutamente modificable lo que permite (a programadores expertos en este caso) convertirlo en otras aplicaciones aparentemente diferentes. Es el caso de Scratch, creado por un equipo dirigido por Mitchel Resnick en el MIT o el de Alice, del extinto Randy Paush, creado para la experimentación en 3D en el aula.

Scratch es, quizá, más fácilmente accesible que Squeak original y su programación en bloques está mejor estructurada ayudándose en el color y la forma de estos bloques. Por otro lado, sus posibilidades son más limitadas y la pantalla de trabajo presenta la mitad de superficie útil en pixels que la de Squeak original.

Resulta muy interesante asomarse a la página oficial de Scratch y comprobar la multitud de aplicaciones que han colgado allí miles de “scratchers” de todas las edades:

http://scratch.mit.edu/

Etoys es la versión de Squeak adaptada especialmente para su uso en los *laptops* del proyecto *One Laptop per Child* (OLPC), aunque funciona perfectamente en cualquier otra plataforma.

Squeak es un mundo de objetos. estos objetos pueden tomarse de una libreria de objetos o pueden crearse mediante herramientas de dibujo similares a Paint que Squeak lleva incorporadas. Squeak permite también importar gráficos en multitud de formatos y tratarlos como objetos.

Todos los objetos llevan asociadas una serie de propiedades (color, posición, orientación, dimensiones, sombra, borde, etc). Tanto las acciones que realizan los objetos como sus propiedades pueden ser programados por medio de guiones. Cada objeto puede tener tantos guiones como se desee y estos guiones pueden, a su vez, ser controlados desde otros objetos (como botones), para lanzar determinadas acciones. Las acciones presentes en LOGO, avanzar, girar, uso del lápiz, están también presentes en Squeak, solo que aquí podemos usar cuantas tortugas (objetos) queramos.

> Por nuestra parte, teniendo en cuenta la variada información disponible y desde una orientación pedagógica, entendemos Squeak como:
>
> 1.- Ambiente matemático: Squeak conecta directamente con las experiencias desarrolladas con el lenguaje Logo por Seymourt Papert en la línea de su matemalandia como ambiente alternativo al mundo real. El propio Alan Kay reconoce que la visión que aporta este autor sobre el trabajo con ordenadores condiciona totalmente el desarrollo de Squeak.
>
> El construccionismo (Papert, 1995, 1999) está muy presente y se palpa en muchas de sus propuestas. Aunque el ambiente matemático no se tiene que trabajar de forma directa aflora a medida que el aprendizaje del alumno se desarrolla. En estadios iniciales de trabajo podemos trabajar propuestas que se centren en otras características del *software* sin que sea explícito el trabajo matemático.
>
> 2.- Multimedia: permite la integración de diferentes medios en los proyectos y en el desarrollo de ideas: audio, vídeo, fotografías, dibujos o texto pudiéndolos organizar en diferentes formatos. Para ello dispone de herramientas como un reproductor MPEG, reproductor y grabador de sonidos, conexión a dispositivos de vídeo o editor de ondas.
>
> 3.- Medio de comunicación: posee herramientas orientadas específicamente para el trabajo colaborativo como Chat, Chat de Audio, el Servidor Nebraska o compartición de proyectos mediante

un repositorio alojado en un Swiki y la exploración de internet como Scamper, un navegador web, o Celeste, un lector de correos electrónicos.

El Servidor Nebraska permite que se comparta un escritorio entre dos usuarios pudiendo desarrollar un trabajo conjunto. Swiki es la denominación usada para un Servidor Squeak Wiki Wiki. Un Servidor "Wiki Wiki" es una invención de Ward Cunningham que permite que un servidor de páginas web sea modificado por cualquier usuario actualizándose su contenido de forma automática sin mediación de un webmaster. El desarrollo en Squeak se debe a Mark Guzdial de la Georgia Tech con lo que pasó a denominarse swiki.

Los proyectos compartidos son accesibles para su utilización mediante un navegador web con un plugin específico.

4.- Entorno de programación: podemos entenderlo en un doble sentido. El ambiente de Squeak permite que todos los elementos de los proyectos se puedan programar o manipular mediante el uso de un visor por el que se accede a comandos pero también permite el desarrollo e implementación de programas en lenguaje Smalltalk.

Para completar, si cabe un poco más, la aportación anterior, señalamos a continuación la clasificación de los instrumentos cognitivos que se presentan en el trabajo con ordenadores (Martín, Beltrán y Pérez, 2003) tomando a su vez como referencia a Jonassen (2000):

1.- Instrumentos para la exploración del conocimiento: Internet.

2.- Instrumentos para la construcción del conocimiento: Hipermedia.

3.- Instrumentos para la organización del conocimiento: Bases de datos.

4.- Instrumentos para la representación del conocimiento: Mapas conceptuales.

5.- Instrumentos para la comprensión del conocimiento: sistemas de experto y micromundos.

6.- Telecomunicación y aprendizaje cooperativo.

De las seis categorías propuestas vemos que cinco se ven directamente reflejadas en Squeak, incluyendo las bases de datos gracias a *Pila*, a excepción del cuarto punto. Esto nos hace ver la flexibilidad de adaptación a diferentes intenciones educativas.

Creemos que queda claro que el objetivo fundamental del uso de este programa es avanzar en los procesos de aprendizaje que convierte al alumno en el verdadero protagonista. Siguiendo a Papert (1995:176) y tomando una clasificación clásica: aprender con el ordenador, aprender del ordenador y aprender sobre el ordenador, es una forma de reflejar tres posibles visiones frente a la dualidad que presentan las dos primeras (Martín, Beltrán y Pérez, 2003).

Se trata de concebir el aprendizaje pensando en la programación como una manera de enseñar cosas sobre ordenadores concibiendo el ser capaz de programar como sinónimo de haber aprendido cómo funciona. Como el mismo autor explica se trata de convertir a los alumnos en productores en vez de consumidores de *software* educativo. "Niños y adultos se convierten en usuarios y creadores de propiedad intelectual disponible bajo dominio público"(Cathleen Galas, 2001). [2]

Comparto completamente los expuesto más arriba por Fernando Fraga y Adriana Gewerc, excepto en un detalle: Squeak está en continuo crecimiento y es muy posible que los mapas conceptuales no estuviesen incorporados por las fechas en que publicaron este informe. Ahora lo están.

Definición y contexto del problema

Squeak y LOGO

Squeak comparte con LOGO, del que es heredero, no solo la visión constructivista sino las ordenes y acciones típicas de la tortuga de LOGO. Podríamos decir sin temor a equivocarnos que LOGO, desde la aparición de Squeak, ha quedado reducido a un subconjunto incluido en el propio Squeak. Todos y cada uno de los objetos de Squeak pueden programarse, mediante guiones, como tortugas de LOGO.

En otras palabras, visto desde el punto de vista de LOGO, Squeak es un entorno de programación que permite la convivencia de múltiples tortugas. Las tortugas “a la Squeak” son objetos de las más diversas características, lo que aumenta exponencialmente la riqueza de posibilidades.

Área tradicional de uso de Squeak

Tradicionalmente, en los ambientes Squeak, los protagonistas son los alumnos de Primaria o Secundaria. Raramente los de Bachillerato y nunca, que yo sepa, los de Ciclos Formativos.

Esta situación me parece un error.

Está bien que los alumnos utilicen Squeak al estilo LOGO en Primaria y Secundaria para familiarizarse con los conceptos matemáticos y físicos, pero Squeak es mucho más que LOGO, aunque comparta su filosofía.

Squeak no solo introduce “a la LOGO” los conceptos primarios de las matemáticas, sino que permite una interacción del alumno con el ordenador mucho más rica, potente, variada y creativa.

Los alumnos de los cursos superiores como Bachillerato y Ciclos Formativos, que ya tienen un bagaje matemático integrado como parte de los conocimientos adquiridos, pueden utilizar Squeak de una forma mucho más “potente” como simulador de los procesos físicos con los que entran en contacto en las aulas y laboratorios de prácticas.

Creo, sinceramente, que están en perfectas condiciones para sacar un gran partido de Squeak.

Nuestra propuesta de uso de Squeak en Ciclos Formativos

En primer lugar ¿que Ciclos Formativos pueden beneficiarse de la realización de simulaciones de procesos físicos? Básicamente, los pertenecientes a las familias más directamente relacionadas con la tecnología:

Edificación y Obra Civil
Electricidad y Electrónica
Energía y Agua
Fabricación Mecánica
Imagen y Sonido
Informática y Comunicaciones
Instalación y Mantenimiento
Química
Transporte y Mantenimiento de Vehículos

Los alumnos de estos Ciclos Formativos tienen una base matemática, principalmente de trigonometría, de la que los estudiantes de Primaria carecen y con la que los alumnos de Secundaria están solo empezando a familiarizarse.

Squeak es un mundo de uso relativamente sencillo pero que incluye entre sus instrucciones funciones matemáticas avanzadas.

Estas funciones matemáticas avanzadas, combinadas con las funciones gráficas “ a la tortuga LOGO” permiten representar sobre el mundo de Squeak, en una gráfica amplitud-tiempo, las tres fases de una corriente trifásica senoidal, por ejemplo. Y en el caso más general, cualquier

función matemática que se nos ocurra. “Solo” debemos crear el guión correspondiente para que el objeto “trazador” dibuje la función en el mundo de Squeak. En la creación por parte del alumno de ese guión de representación gráfica está la base de su aprendizaje por construcción.

Esta representación que escaparía al uso de Squeak por parte de alumnos de Primaria resulta especialmente interesante para que los alumnos de Bachillerato y Ciclos Formativos jueguen (en el sentido más constructivo de la palabra) y experimenten con conceptos matemáticos mucho más avanzados. Sean capaces de hacer que Squeak trace funciones matemáticas complejas y de analizar y probar conceptos que, a falta de Squeak, requerirían la realización de un acto de fe.

En otras palabras, está muy bien y me parece muy adecuado el uso de Squeak en Primaria y Secundaria, puesto que es perfectamente accesible a la experimentación por parte de los alumnos en este nivel. Pero me parece un error no utilizar todas las potencialidades de Squeak en niveles de enseñanza superiores.

Squeak convertirá así en autores y en dueños y señores del ordenador y de su propio proceso de aprendizaje a los alumnos de Ciclos Formativos.

Aún llegaría más lejos, si se me permite. Squeak pude utilizarse también como una buena base de pruebas y de realización de prototipos para un uso profesional. Sobre todo realizando guiones directamente en Smalltalk. Pero eso sale fuera de los propósitos de este trabajo.

Un ejemplo

En la figura-1 puede verse un ejemplo de lo que acabamos de comentar realizado, esta vez, con Etoys. Se trata de analizar como tiene lugar un movimiento lineal de vaivén utilizando aceleraciones senoidales. El movimiento resultante es un movimiento cicloidal y tiene la particularidad de que, para un periodo de vaivén determinado, el estrés mecánico que se produce sobre la pieza móvil es el menor posible. Con cualquier otro perfil de aceleración, será mayor.

Es una opinión personal que no he visto escrita en ningún sitio, pero estoy convencido de que la elegancia de los saltos de los gatos y otros felinos se debe a que realizan de forma natural este tipo de movimientos.

Hay cuatro objetos principales con forma circular. el circulo negro marca el paso del tiempo y realiza las funciones de cursor. El círculo amarillo traza el perfil de la aceleración, el azul el de velocidad y el verde el de la posición.

Figura-1 Simulación de un movimiento lineal de aceleración senoidal

En el lado derecho, se representa el movimiento mediante un carro móvil (rectángulo verde) que se desplaza sobre una guía (rectángulo amarillo) las lineas roja y azul que parten del centro del carro, representan su aceleración y velocidad instantáneas. El carro se posiciona “copiando” la coordenada Y del objeto circular verde que traza la gráfica de posición.

Lo más interesante de este ejemplo es que el trazado y cálculo de la velocidad instantánea se realiza mediante integración gráfica de la aceleración, es decir, el guión que calcula el valor de la velocidad va sumando, con su signo, los valores instantáneos de la aceleración.

Lo mismo sucede con el cálculo de la posición, que se realiza mediante integración gráfica de la velocidad. En la figura-2 puede verse el guión que realiza esta integración gráfica.

De esta forma, modificando únicamente el perfil de la aceleración, sea haciéndolo, triangular, trapezoidal o rectangular podemos ver, directamente y sin más modificaciones, las consecuencias directas que cada perfil de aceleración tiene sobre la gráfica de la velocidad y la de la posición.

Para los alumnos, el hecho de comprobar visualmente que una integración es una suma y que esa suma equivale al área abarcada por la función a integrar, reforzará y les ayudará a entender mejor el concepto de integral.

Figura-2. Guión de trazado de la posición

Descripción de la solución

Regalarles Squeak

Se trata de entregar a los alumnos de Ciclos Formativos una herramienta de programación potente y de uso relativamente fácil, como Squeak o Etoys para que se la hagan suya.

Hablo de regalársela en el sentido de suministarles el conocimiento suficiente para que puedan hacerse suya esta joya de herramienta de programación. La joya en si es gratis. Se la pueden descargar desde internet. Pero sin una ayuda previa dificilmente llegarán a utilizarla.

También hablo de regalo en el sentido de que no van a realizar ningún examen de aprovechamiento de Squeak. La herramienta de programación va a ser suya y la podrán utilizar para todas las aplicaciones que se les ocurran y cuando quieran.

Squeak es divertido y crea adicción, pero, seguramente, no todos los alumnos van a aficionarse a él.

Podríamos hablar de dos tipos de diversión, la diversión pasiva y la diversión activa. Es diversión pasiva ver una película o contemplar un deporte que nos gusta. La diversión activa por el contrario, implica acción por nuestra parte y puede ser tan o más divertida que la pasiva. Sería diversión activa hacer una película o jugar a un deporte que nos guste.

Squeak es diversión activa. Exige hacer trabajar la imaginación, probar, equivocarse, corregir, modificar, volver a probar...

> [...] *a lot of people thought about Logo and might think this about Squeak* ***"this is really hard.' That is right. It is hard but it is also fun. And, the kids say it is hard and the kids say it is fun.*** *Both Alan and Seymour both talk about hard fun and soft fun. Here is an example, soft fun is watching people play baseball. Hard fun is playing baseball. And, soft fun is watching someone play the violin and listening to a concert and hard fun is you playing the violin.* ***That is how we see Logo and Squeak, as constructive activities. It is harder. You have to use your head. It is not just putting other people's stuff together****. Frankly, a lot of people stop using Squeak because it is hard and it takes time to learn how to use Squeak well. You don't learn to play a musical instrument in a couple of days, it takes time. It is a deeper learning environment. It is kind of anti what much of our culture and society is doing and learning and expecting these days. It is fast food, fast every thing. Quickly achieving your goals. It is not working a long time and getting a deep understanding. It is hurry up and getting the end of that chapter because someone expects us to be at the end, at a certain point of learning, at a certain time. Squeak runs contrary to a lot of currents in our culture. This is what we are doing. We are not going to say 'Forget it if the majority of the world thinks it is too hard.*' [5]

Así que no todos los alumnos van a acabar convertidos en “squeakers”. Los más pasivos encontrarán que Squeak les exige “demasiado” esfuerzo. Si no son capaces de realizar ese esfuerzo inicial que les permita descubrir la caja de sorpresas que hay en Squeak y toda la diversión activa que pueden conseguir, dificilmente podrán seguir más allá. En Squeak solo se puede ser protagonista, no es para dejarse llevar, es para actuar.

A pesar de todo, mi opinión personal es que vale la pena realizar la experiencia para beneficio de los más inquietos, curiosos e imaginativos que si sean capaces de realizar ese esfuerzo inicial.

Vivimos en la sociedad de la información, pero esta no sirve de nada (más bien puede convertirse en desinformación) si la persona no es capaz de convertirla en conocimiento.

> *Knowledge is power, but information is not.*
> David Lewis

Un programa como Squeak permite ensayar y ayuda a crear y resolver conflictos cognitivos. A crear conocimiento a partir de nueva información.

Profesorado

Aunque Squeak se aprende con relativa facilidad, tiene sus idiosincrasias y es importante que el profesor o profesora tenga una cierta práctica en su uso.

No pasa nada si, en un momento dado, tiene que aprender algo sobre la marcha, incluso de los propios alumnos. Esto sería más bien positivo, todos estamos aprendiendo siempre. Pero tiene que dominar muy bien como crear un proyecto, como crear un objeto mediante el pintor y haber trasteado con el catálogo de objetos lo suficiente como para tener una idea de lo que es un "campo de juego", un "libro", una "pila" y muchos otros objetos "especiales" presentes en Squeak. Es esta multitud de objetos "especiales" lo que convierte a Squeak en una verdadera caja de sorpresas.

Para ello existe una comunidad muy activa a la que es fácil acceder vía internet. Squeak se utiliza en Estados Unidos, en Japón, en varios países de Sudamérica y en Alemania. En España, varias Comunidades Autónomas están utilizando Squeak. Durante a preparación del material que utilicé durante el prácticum, he estado en contacto mediante correo electrónico con alguno de ellos, concretamente de la comunidad de Extremadura, que me han suministrado, además, una imagen de Squeak traducida por ellos mismos al castellano y a la que han añadido una serie de objetos muy interesantes por su cuenta. Creo que en Galicia y en Murcia hay también profesores que son "squeakers".

En cuanto a internet, aparte del sitio oficial de Squeak y el de Etoys, es muy interesante el de las islas de Haiti donde puede seguirse gratuitamente un curso a distancia, en diez lecciones. Cada sesión dura algo menos de una hora. El curso es en inglés.

http://www.waveplace.com/

Sitio oficial de Squeak:

http://www.squeak.org/

Sitio oficial de Etoys (disponible en castellano):

http://www.squeakland.org/

Metodología

Aunque Squeak es de uso relativamente sencillo se requiere un cierto aprendizaje previo para poder arrancar y, a partir de ahí, por experimentación y por prueba y error el alumno sigue aprendiendo por su cuenta. Al fin y al cabo es lo que hemos hecho todos los "squeakers".

No se trata de hacer del aprendizaje de Squeak un nuevo módulo, sino de integrarlo como herramienta en alguno de los módulos ya existentes. Por ejemplo, en el Ciclo Formativo de Grado Medio *Instal·lacions Elèctriques i Automàtiques*, podría introducirse en el módulo de Máquinas Eléctricas Rotativas.

El uso de Squeak se extendería durante todo el curso escolar aprovechando las nociones del currículo que se presten a ello para hacer las simulaciones en clase y en grupos de dos alumnos por ordenador.

La primera vez que se introduzca Squeak los alumnos van a ir siguiendo en sus ordenadores lo que el profesor vaya haciendo sobre la marcha desde el proyector. Se trata aquí de introducir los rudimentos del uso de Squeak, sus menús, como crear y guardar proyectos, los objetos y sus halos, las pestañas, el catálogo de objetos, etc.

En esta primera sesión puede ya crearse algún guión que mueva los objetos utilizando las instrucciones " a la LOGO" y animar a los alumnos a probar, probar, probar...

Un poco más adelante ya pueden empezarse a introducir simulaciones de fenómenos físicos. Por ejemplo, algunos de los conceptos de las máquinas eléctricas rotativas no son intuitivos ni fácilmente comprensibles. Tal es, por ejemplo, el concepto de campo magnético rotativo creado por corrientes de tipo senoidal, trifásicas o bifásicas.

Esta sería una buena “excusa” para la introducción de simulaciones con Squeak, y es la que yo he utilizado a título de ejemplo y ensayo durante el Prácticum, pero no es, ni mucho menos, la única.

Durante el Prácticum debido a la escasez de tiempo disponible para cubrir las materias curriculares no ha sido posible realizar realmente una introducción a Squeak, se necesitaría todo el curso escolar para ello. Las simulaciones las he hecho yo y simplemente las he visualizado en clase.

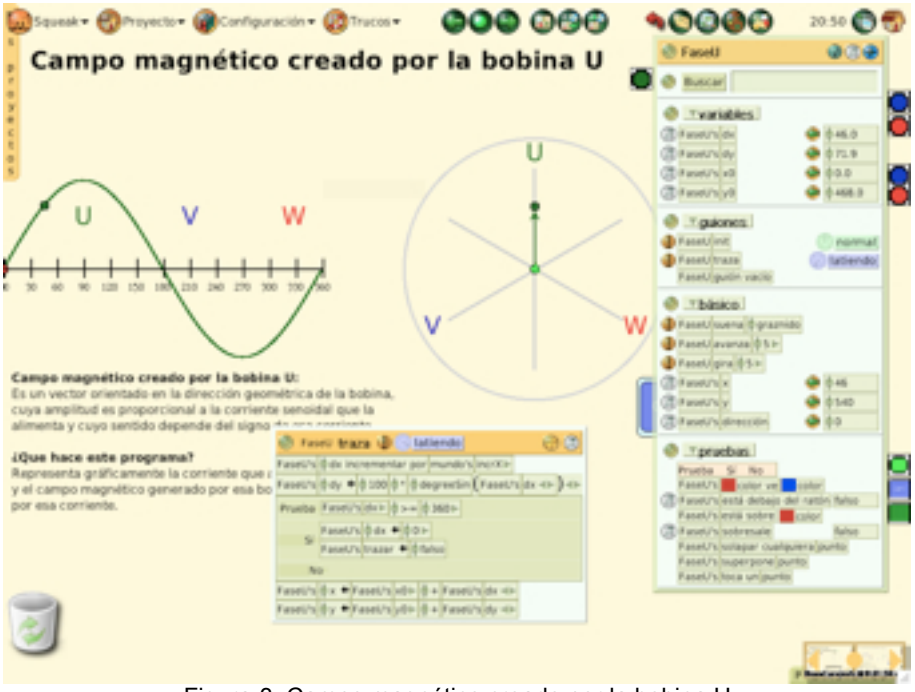


Figura-3. Campo magnético creado por la bobina U

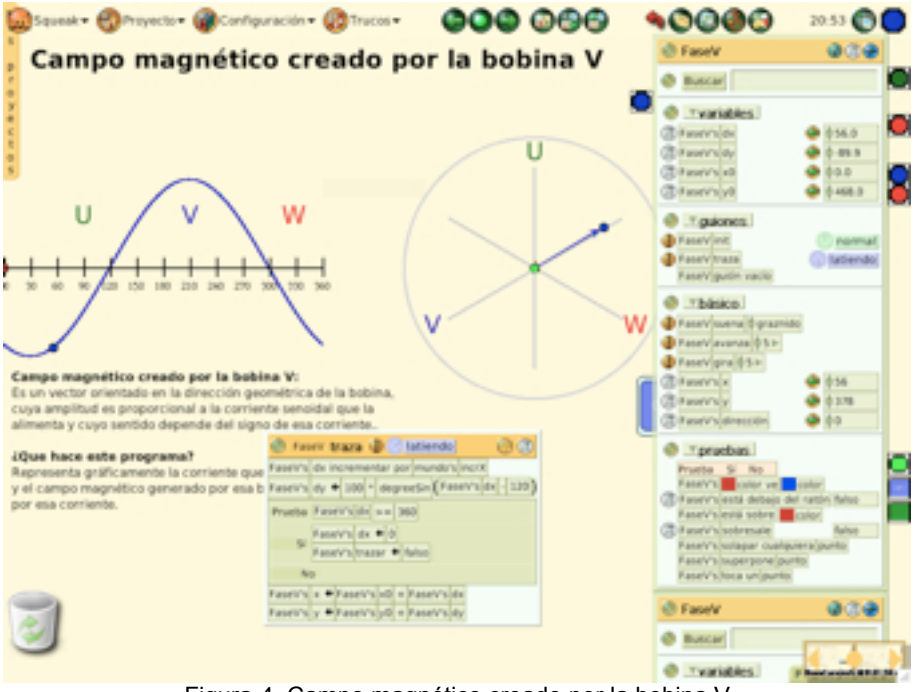


Figura-4. Campo magnético creado por la bobina V

No se trata de esto, naturalmente. Hay que hacer las simulaciones en clase y el profesor debe actuar un poco como director de orquesta, como motivador, como animador. Naturalmente habrá ensayado antes una posible solución a la simulación a realizar, pero debe estar dispuesto a cambiar el rumbo si en el grupo aparecen nuevas ideas interesantes.

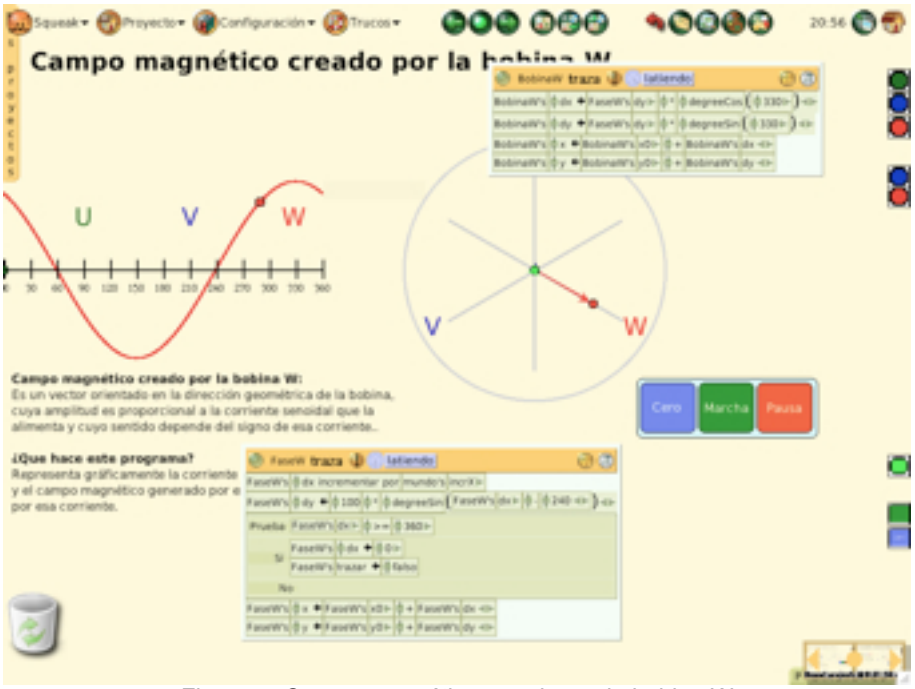


Figura-5. Campo magnético creado por la bobina W

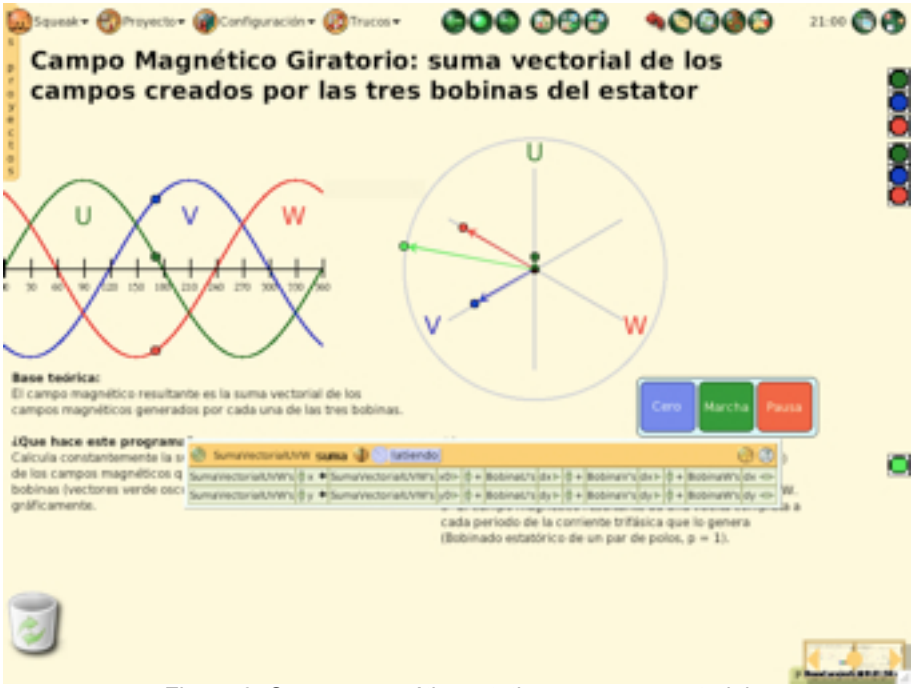


Figura-6. Campo magnético rotativo por suma vectorial

En el caso de la creación de campos magnéticos rotativos puede procederse por fases bien diferenciadas.

Primero representaremos los campos magnéticos creados por cada una de los tres bobinados dispuestos a 120° entre si. Primero el bobinado U, después el V y, finalmente el W. Paso a paso. (Figuras-3, 4 y 5)

Veremos que el campo creado por cada fase es alternativo y que puede representarse mediante un vector que se mueve sobre una linea situada a 120° de las otras dos fases.

Crearemos un guión que realice la suma vectorial de los vectores de los campos magnéticos de las tres bobinas y veremos que el resultado obtenido coincide con lo que dice la teoría. (Figura-6).

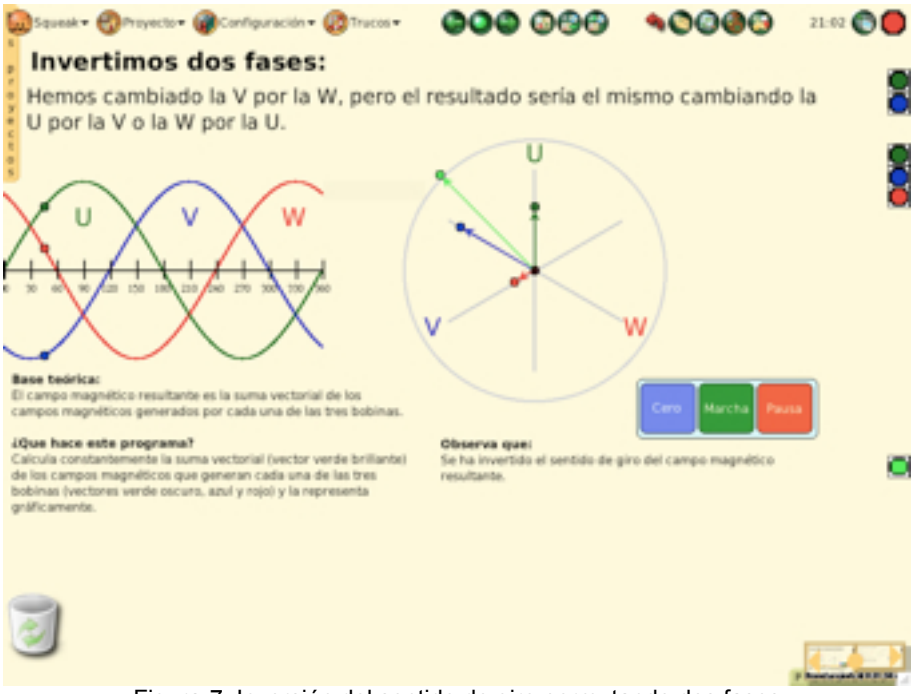


Figura-7. Inversión del sentido de giro permutando dos fases

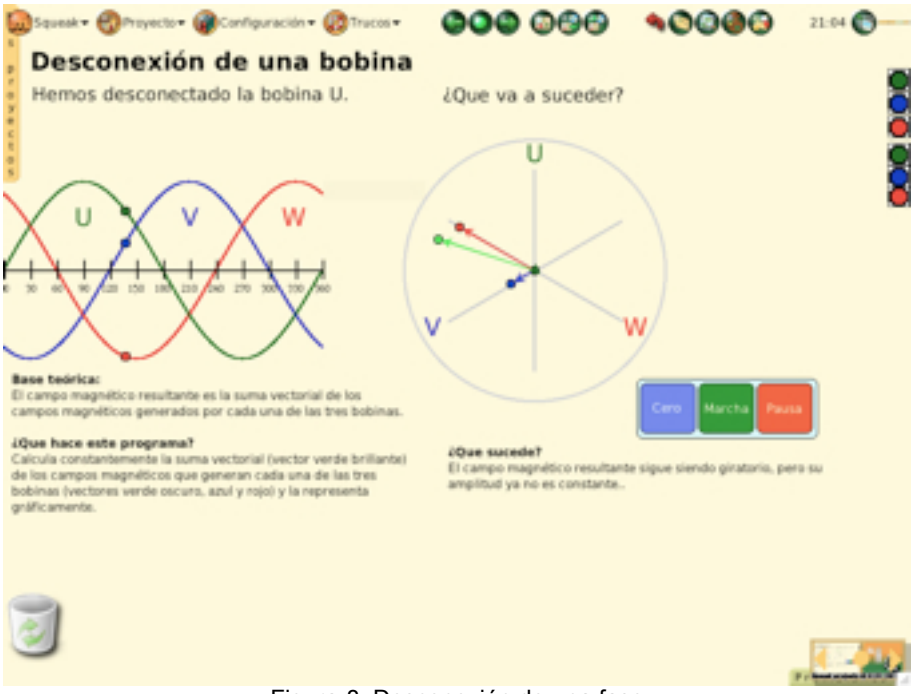


Figura-8. Desconexión de una fase

Y, una vez hecho esto, se trata de “sacarle todo el jugo al proyecto”

—¿Que pasa si invertimos una fase? (Figura-7)

—¿Y si ahora desconectamos una fase? (Figura-8)

Así comprobarán visualmente que cambiar el orden de las fases invierte el sentido de giro y que si desconectamos una fase seguimos teniendo un campo magnético rotativo, de igual frecuencia que con las tres fases conectadas, pero cuya amplitud ha dejado de ser constante para variar en un rango de 1 a 3.

— Y la posición de los valores máximos y mínimos de este campo ¿Tiene alguna relación con la fase que hayamos desconectado?

— ¡Pruébalo! Desconecta otra fase distinta...

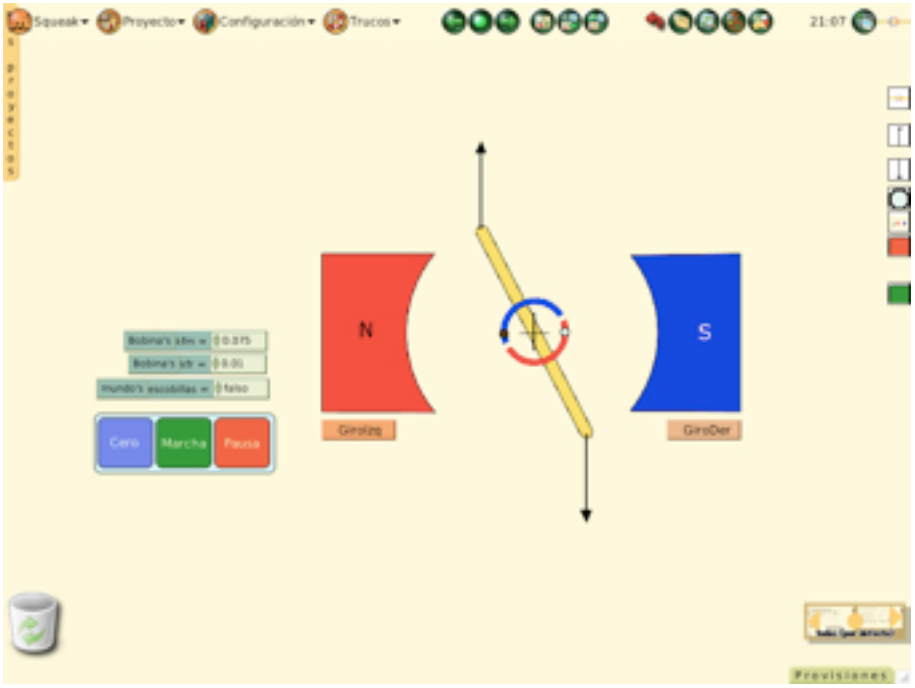


Figura-9. Motpr elemental de corriente continua

Otra de las simulaciones preparadas durante el Prácticum ha sido la del motor elemental de corriente continua que puede verse en la figura-9.

Este motor consta de una bobina rectangular (rectángulo amarillo con extremos redondeados) que puede girar sobre un eje, perpendicular a la pantalla, que aquí se ha representado como una cruz.

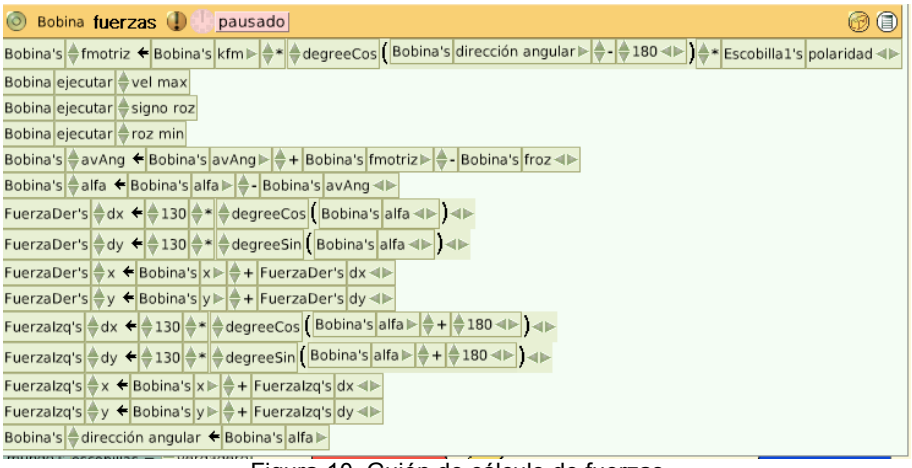


Figura-10. Guión de cálculo de fuerzas

Cuando “se hace pasar corriente” por la bobina se crean unas fuerzas perpendiculares al sentido de la corriente y al sentido del campo magnético. Parte de estas fuerzas, las que se proyectan perpendicularmente al rectángulo que representa la bobina crean un par que tiende a hacer girar el motor. Un guión calcula la magnitud y el sentido de estas fuerzas en función de la posición angular de la bobina y hace girar la bobina en consecuencia. (Figura-10).

Primero se simula que no hay colector de delgas ni escobillas y que la corriente circula siempre en el mismo sentido y se comprueba que la bobina se comporta como un péndulo. Si no introducimos rozamiento (kfr=0) el péndulo no se detendrá nunca. Al introducir un valor mayor que cero como coeficiente de rozamiento kfr el guión correspondiente (figura-11) lo tiene en cuenta y la amplitud del movimiento pendular va disminuyendo hasta que la bobina se detiene alineada con la linea neutra.

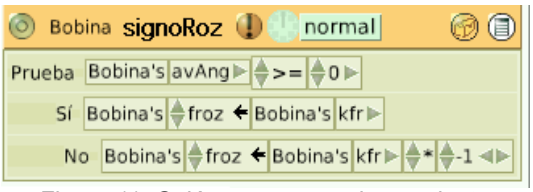


Figura-11. Guión que opone el rozamiento

Es interesante aprovechar la circunstancia para introducir el concepto de linea neutra, que es aquella posición de la bobina donde la fuerza motriz se hace cero.

A continuación, se hace que justo al pasar por la linea neutra la polaridad de la corriente se invierta con lo que las fuerzas cambian de sentido y el motor empieza a actuar como tal girando cada vez a mayor velocidad.

Esta conmutación hace uso de una característica muy interesante de Squeak: la posibilidad de detectar si un área de un objeto, de un color determinado, solapa con determinado color de otro objeto. Por eso cada delga se ha dibujado en un color distinto. Cuando el objeto circular blanco que hace de escobilla “ve” que la delga que tiene debajo es azul “hace circular la corriente” en un sentido. Cuando es roja, en el contrario. Se entenderá mejor viendo el guión. (Figura-12).

Figura-12. Conmutación de las escobillas

Resultados

Como no ha sido posible probar una verdadera introducción al uso de Squeak con la metodología explicada más arriba no puedo disponer de resultados. No se ha llegado a probar el método. Así que lo único que podemos hacer es conjeturar.

Suponiendo que se hubiese realizado esa introducción durante todo el curso escolar, se les habrá entregado una herramienta que puede dar excelentes resultados en las manos de los alumnos que se hayan aficionado. Es una herramienta de aprender a aprender, de probar, de jugar, de construir.

La naturaleza absolutamente orientada a objetos de Squeak se presta especialmente bien para este tipo de simulaciones. No hay un programa principal responsable de la simulación, solo hay objetos. Es el comportamiento de cada objeto y su interacción con los demás objetos lo que se programa mediante guiones que pertenecen al propio objeto.

Por otra parte, el hecho de que su compilación sea JIT (*Just In Time*) permite realizar modificaciones en cualquier momento, incluso sobre guiones que están ejecutándose y ver el resultado de esas modificaciones de forma inmediata.

Tanto la orientación a objetos, como la inmediatez de los cambios, facilitan un enfoque de prueba y error que, una vez superada la primera fase de familiarización, no hace más que favorecer e incentivar la iniciativa del alumno.

¿Que porcentaje de alumnos podemos esperar que se interese?
Dependerá mucho del grupo de clase, de la experiencia y del estilo que tenga el profesor o profesora y del gusto que pueda tener por el tema. Es muy dificil despertar interés por algo que a ti mismo no te interese. por lo que el entusiasmo del profesor por la materia es aquí de vital importancia.

Conclusiones

De forma análoga a lo que ocurre con los resultados, no pueden extraerse conclusiones fundadas, pues no ha sido posible realizar la experiencia con los alumnos creando sus propias simulaciones. Ni siquiera ha sido posible analizar con ellos como están realizadas las simulaciones preparadas por mi.

Sin embargo, las positivas experiencias realizadas con Squeak en Primaria y Secundaria, en Estados Unidos, Japón, Alemania, Uruguay, Perú... y en España en las Comunidades Autónomas de Extremadura, Galicia y Murcia, permiten aventurar el éxito de su introducción en Ciclos Formativos de familias tecnológicas. Más teniendo en cuenta el bagaje de trigonometría de que ya disponen los alumnos de estos Ciclos, especialmente los de Ciclos Superiores.

Bibliografía/Webgrafía

[1] Adell, Jordi. *Internet en el aula: las WebQuest*. Edutec Revista Electrónica de Tecnología Educativa.
http://www.uib.es/depart/gte/edutec-e/revelec17/adell_16a.htm

[2] Fraga, Fernándo y Gewerc, Adriana. *Una experiencia interdisciplinar en Educación Primaria mediante el uso de Squeak*. Universidad de Santiago de Compostela.
http://squeak.usc.es/USCSqueak/uploads/3/Una%20experiencia%20interdisciplinar.PDF

[3] Kay, Alan. *Entrevista tras su investidura como doctor Honoris Causa por la Universidad de Murcia*, Traducción de Jesús García Molina.
http://redi.um.es/campusdigital/entrevistas/4441-alan-kay-informatico-nuevo-doctor-honoris-causa-de-la-universidad-de-murcia-.html

[4] Papert, Seymour. *¿Que es Logo?¿Quien lo necesita?* Eduteka.
http://www.eduteka.org/modulos/9/288

[5] Rose, Kim. *Hard Fun... squeak!*.
http://www.mime.indiana.edu/squeak/

[6] Shore, John. *El Algoritmo Sachertorte y Otros Antídotos contra la Ansiedad que provoca el Ordenador*. Alianza Editorial.

[7] Valero Aguayo, Luis. *Máquinas de enseñanza de Skinner*. Grupo Contextos. Facultad de Psicología. Universidad de Málaga.
http://www.conducta.org/articulos/maquinas_ens.htm

Consultados, pero sin menciones explícitas en el texto del trabajo:

Ducasse, Stéphane. *Squeak: Learn Programming with Robots*. Apress®

Keller, Laurie. Preece, Jeny. Stolk, Hans, *Human-Computer Interaction*, Selected Readinggs. Prentice Hall.

Wirth, Niklaus. *Algoritmos más estructura de datos, igual programas*. Castillo ediciones.