METODOLOGÍAS DE IMPLEMENTACIÓN, SIMULACIÓN Y APRENDIZAJE DE LOS SISTEMAS DIGITALES

J. Terrón, J.M. Benítez y J. Moreno
Departamento de Ingeniería de Sistemas y Automática, Tecnología Electrónica y Electrónica. Universidad de Cádiz.
Polígono Río San Pedro s/n. 11510. Pto. Real (Cádiz).
Tfno: 956/470842, Fax: 956/470803
e-mail: julio.terron@uca.es

RESUMEN.- En este trabajo se pretende mostrar la posible integración de las distintas visiones que el alumno puede poseer (provenientes de distintas disciplinas) de los sistemas digitales con objeto de establecer una relación de equilibrio entre implementaciones hardware y software. El objetivo es potenciar la creatividad del alumno para que sea capaz de plantear sus propias soluciones y conseguir una mejora en el conocimiento profundo de los sistemas que estudia. La utilización en clase de herramientas de programación visual y orientada al objeto ofrece la posibilidad de que el alumno razone desde un punto de vista más natural (enfoque cognitivo) para la creación de representaciones y modelos empleados en la simulación. A partir de un ejemplo tipo se desarrolla la metodología expuesta y sus posibilidades educativas.

1.- INTRODUCCIÓN

En el estudio de los sistemas digitales convergen conocimientos provenientes de varios ámbitos (electrónica digital, programación, lógica matemática, representación de modelos, etc...) que son recibidos por los alumnos a través de distintas disciplinas, generalmente de forma aislada. Sin embargo, las soluciones creadas en hardware y software están cada vez más íntimamente relacionadas, existiendo un equilibrio entre ambas. En el ámbito educativo, como profesores, durante las explicaciones suelen transmitirse los conocimientos a los alumnos unidireccionalmente, y éstos los reciben la mayoría de las veces de forma pasiva, con las soluciones ya dadas y desde un único punto de vista.

Se priva así al alumno del esfuerzo creativo en el enfoque y en el rediseño personal del dispositivo existiendo una ausencia del aprendizaje por descubrimiento, que puede suponer negativas repercusiones pedagógicas. Todo ello le provoca algunas lagunas como *falta de comprensión profunda*, *poca habilidad de relacionar lo aprendido* con otras disciplinas (ausencia del aprendizaje significativo), *poca motivación y pasividad* ante el aprendizaje. Se pretende aportar, por una parte, una metodología que abarque una perspectiva multidisciplinar que produzca una integración del conocimiento en el alumno que palie el distanciamiento de las asignaturas y, por otra, lograr una comprensión más profunda del análisis de los elementos de los sistemas digitales incidiendo en los conceptos de objetivo, estructura, comportamiento y función. Para paliar los defectos indicados se realiza un análisis cognitivo de las soluciones implementadas en hardware y en software y se aprovechan las ventajas de las herramientas elegidas (modularidad, orientación al objeto, jerarquización...).

2. METODOLOGÍA

1) Se expone y estudia el sistema digital a implementar identificando objetivos, comportamientos, funciones y contextos, discutiendo en clase con los grupos las distintas formas de realización (orientado al circuito electrónico, a la implementación por software a partir de tablas, de funciones, de secuencia de operaciones, grafos de transición...), eligiendo cada uno los distintos módulos y sus posibles soluciones.
2) Supuesto conocido el manejo del software elegido (HP VEE, VISSIM...) y mínimos conocimientos de las estructuras de programación, se estudian en clase sus posibilidades y limitaciones de implementación de los modelos con respecto a las librerías disponibles. Como son lenguajes de programación visual y orientado a objetos, posee unas propiedades idóneas para lo que se pretende, ofreciendo la posibilidad de trabajar y pensar con una "realidad modularizada y jerarquizada".
3) Cada grupo implementa su solución procurando integrar de forma idónea las características de la programación visual (interface didáctico y avanzado, agradable al usuario) y de la programación orientada a objetos (modularidad, encapsulamiento, paso de mensajes, jerarquía, polimorfismo, herencia...). Posteriormente simula de forma progresiva el funcionamiento real de los sistemas y circuitos digitales, comprobando resultados en pantalla mediante instrumentación virtual, o bien conectándolos con el mundo exterior mediante tarjetas de adquisición de datos.
4) Con objeto de aumentar la creatividad y la capacidad de análisis, los alumnos van rotando por los distintos grupos, estudiando los diversos puntos de vista y las soluciones de cada uno.
6) Se discuten en clase las soluciones y, una vez consensuadas, se va creando una biblioteca de programas, de módulos digitales y de dispositivos virtuales desde donde se pueden simular y manejar variables físicas reales de entrada/salida que está a disposición de cualquier grupo.

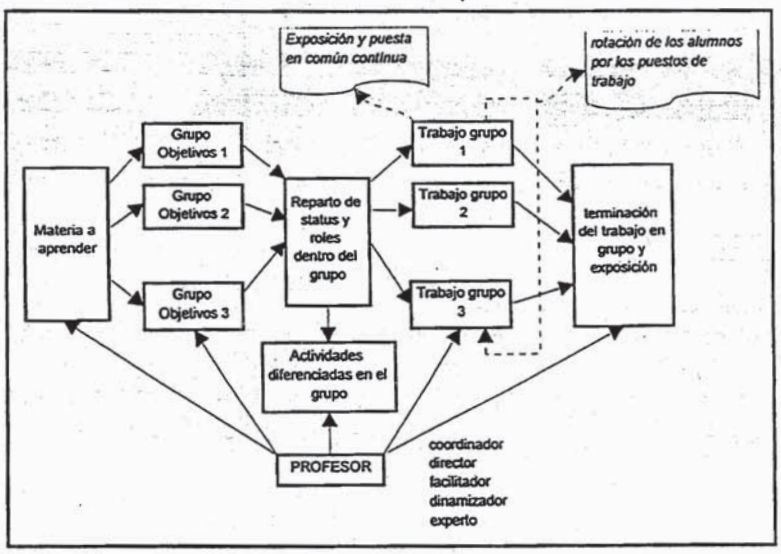

Figura 1.- Estructura y actividades de la dinámica de trabajo en grupo.

3. MEDIOS EMPLEADOS.

3.1.- Sistemas y elementos en el laboratorio

Básicamente el material utilizado en los puestos de trabajo está compuesto de entrenadores digitales que simulan el proceso y permiten la utilización de módulos físicos (multiplexores, contadores...), ordenadores PC con tarjetas de adquisición de datos, unidos en red local para compartir las librerías y los programas creados (Ver figura nº 2).

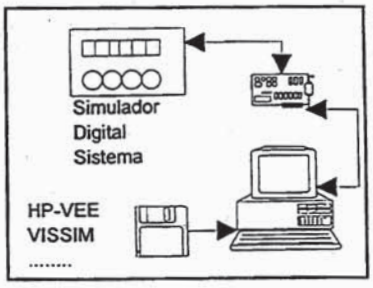

Figura 2.- Equipos usados.

3.2.- Herramientas de programación

Aunque existen muy buenas herramientas de diseño de sistemas digitales como Orcad Workbench... y lenguajes de descripción de circuitería como el VHDL, para el uso diario en formación presentan inconvenientes como la rigidez y automatismo que a veces imponen en el razonamiento por la utilización de bloques funcionales ya encapsulados o por su cercanía al código que los aleja, a veces, de la comprensión natural, ya que están eminentemente enfocados a la importante labor del diseño e implementación El objetivo es potenciar también las tareas de representaciones cognitivas y funcionales para otras labores importantes como las de diagnóstico y mantenimiento, trabajo en grupo... Para tratar de paliar todo esto se ha elegido voluntariamente herramientas visuales de programación general, para plantear los rediseños desde el origen y a pesar de sus pequeños inconvenientes nos permite una programación rápida y creativa de cualquier sistema desde distintos puntos de vista (hardware, software o combinación). En cuanto a las ventajas que nos ofrecen algunos de ellos como el HP VEE pueden destacarse el seguimiento del flujo de datos en la simulación, el paso de mensajes, la jerarquización de funciones, el encapsulamiento, la posible intercalación de objetos de código desarrollado en los lenguajes tradicionales (C, Basic, Pascal...), entre otras.

4.- EJEMPLO TIPO

Como base para clarificar las ideas expuestas hemos tomado un ejemplo de E. Mandado [1] y cuyo enunciado dice así *"Una vía férrea con tráfico en ambos sentidos corta a una carretera en la cual se coloca una barrera gobernada por la salida Z de un autómata asíncrono. A 500 mts del punto de cruce se colocan dos detectores X1 y X2 respectivamente y a partir del estado inicial (Z=0) Z debe pasar a "1" al acercarse un tren en cualquier sentido al rebasar su máquina los 500 mts del cruce* y debe volver a "0" *cuando el último vagón se aleje más de dicha distancia independientemente de la longitud del tren*". Ver Figura 3.

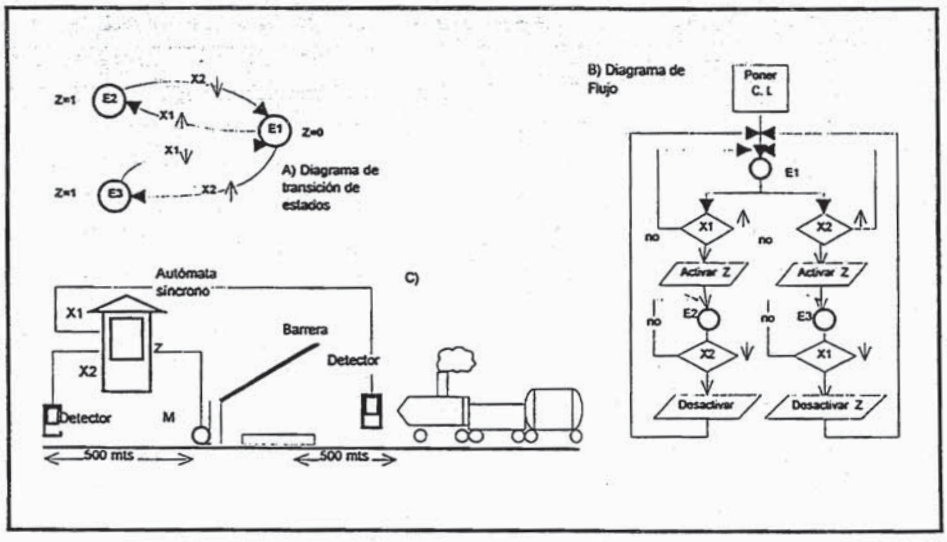

Figura 3.- Ejemplo [1] del control de la barrera de un paso a nivel y diagramas del autómata asíncrono.

Este ejemplo es interesante porque hay que implementar distintos elementos como: Biestables J-K activados por flancos, célula CAF de dos entradas y generalizada.

A partir del diagrama de transición de estados del problema (Figura 3a) y después de su tratamiento mediante tablas de Karnaugh se llega al autómata que resultante de la Figura 11 (siguiendo el modelado del diagrama de la Figura 3a pueden asemejarse las representaciones). La solución (Figura 11) se implementa mediante células CAF (dos de una entrada y una generalizada de dos entradas), pero si aplicamos los principios expuestos en este artículo este resultado puede diversificarse aún más. Las herramientas expuestas nos permiten desarrollar una CAF según su estructura interna física (Figura 8) o bien implementándola mediante estructuras de programación (Figura 7) siguiendo el ordinograma correspondiente de la Figura 3b. Si este algoritmo se somete a un tratamiento de un sencillo shell de sistemas expertos basado en reglas puede simplificarse y optimizarse el resultado (Figura 5). A su vez, y teniendo en cuenta la composición física de la CAF mediante biestables, los J-K síncronos pueden desarrollarse mediante programación de J-K asíncronos (Figura 6) incluyendo la detección de los flancos de subida de sus entradas de sincronización (Figura 9). Con esto y con la realización de la CAF generalizada de dos entradas a partir de CAF simples (Figura 10) se pone de manifiesto la posible jerarquización de los distintos módulos que componen la solución del problema; y si vamos mezclando las distintas implementaciones hardware y software (Figura 12) de cada uno se justifica el equilibrio entre ambos del que hablábamos al principio, escogiendo la representación que más se adapte a nuestras necesidades de docencia, así como al máximo rendimiento y compresión del sistema. Por otra parte, en los diagramas visuales de programación aparecen elementos y conexiones intrínsecos del propio lenguaje y otros (Qt, Qt+1,...) que son implementados debido a que en un sistema secuencial deben de guardarse los estados sucesivos.

Para respetar al máximo las diagramaciones (estructura física, ordinogramas, diagramas de circuitos, diagramas de estados) desde una óptica cognitiva (la comprensión de algo y su percepción deben de estar unidas)hay que tener en cuenta la posición y la distribución óptima de los objetos respetando su simetría y unificando las vistas de los distintos enfoques, sus conexiones, la nomenclatura empleada, los tamaños relativos, las posibles entradas salidas, el sombreado para resaltar los elementos principales y bucles de decisión...

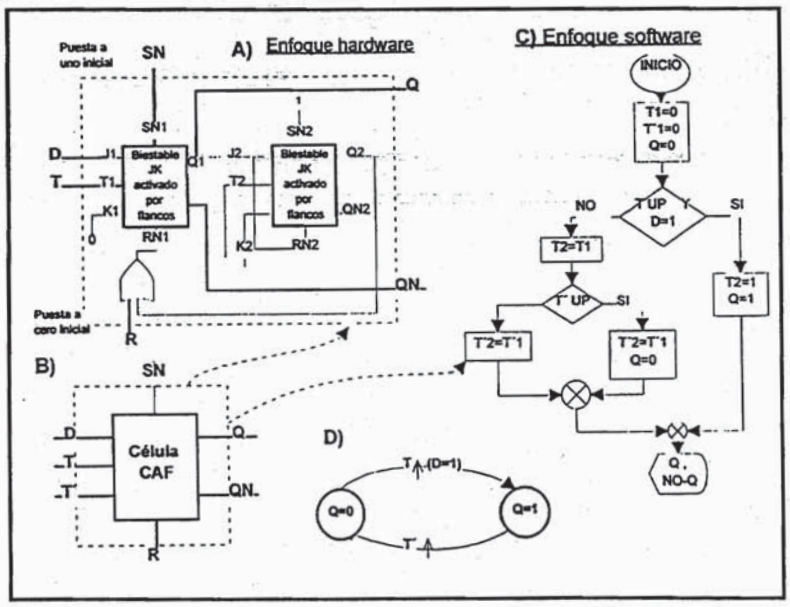

Figura 4.- Distintos enfoques de diseño de una CAF.

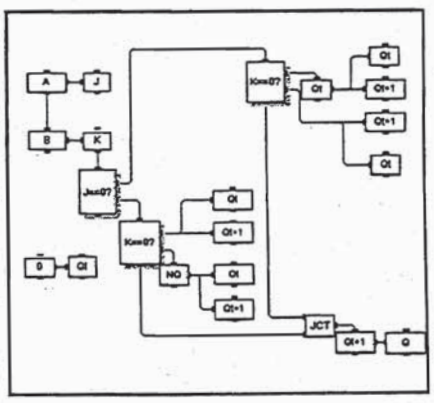

Figura 5.- JK software óptimo.

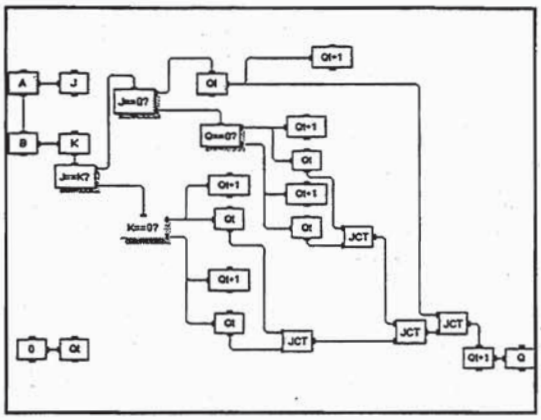

Figura 6.- J-K mediante estructura interna de programación (no óptima).

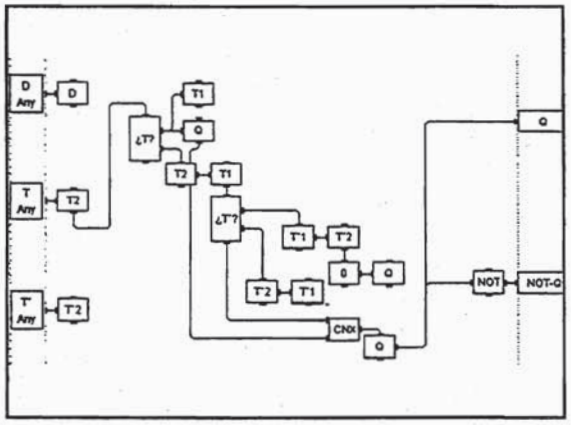

Figura 7.- Diseño de una CAF desde la perspectiva de la programación.

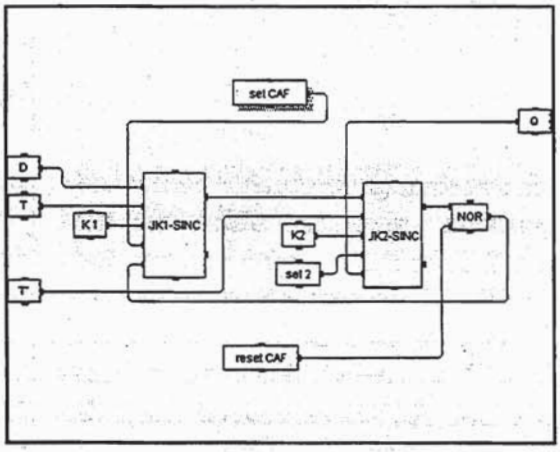

Figura 8.- Célula CAF con diseño orientado al hardware mediante J-K en HP-VEE.

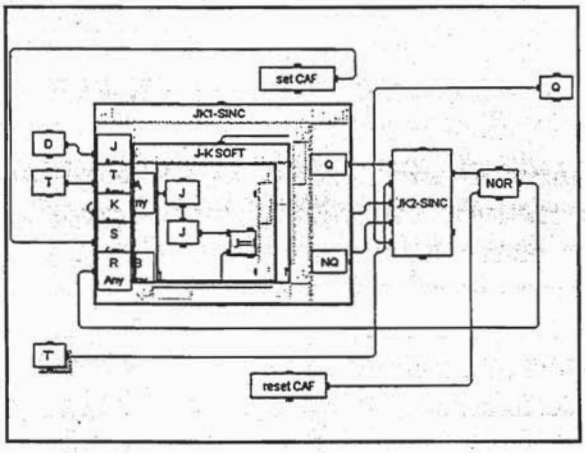

Figura 9.- Modularización jerárquica basada en objetos de la célula CAF.

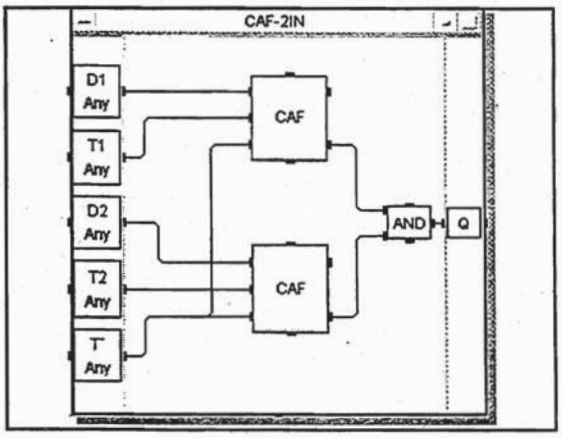

Figura 10.- Célula CAF generalizada.

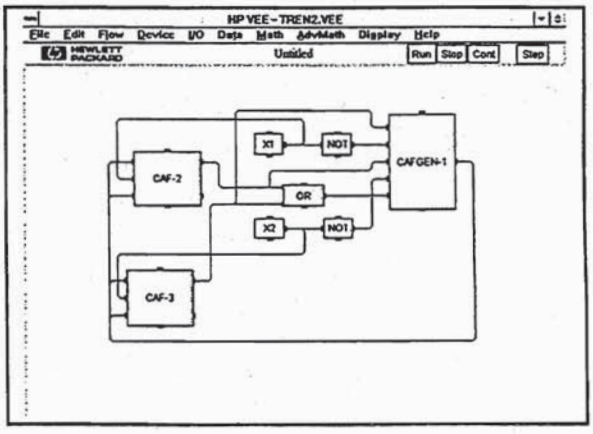

Figura 11.- Autómata asíncrono del ejemplo del paso a nivel.

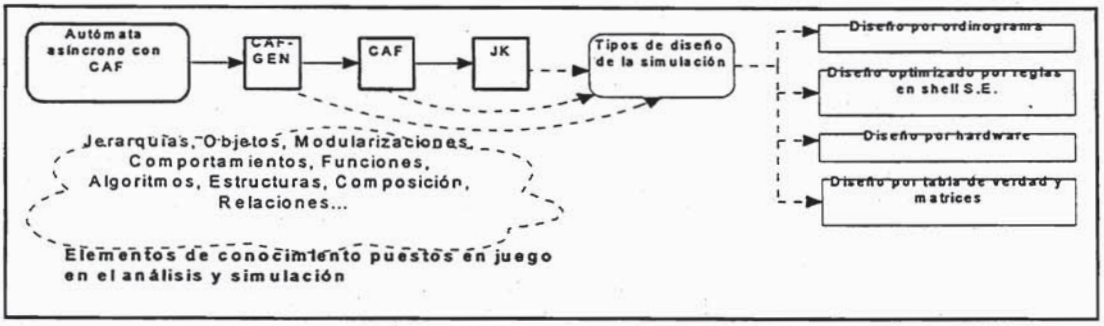

Figura 12.- Diversidad de representaciones del análisis y la simulación.

CONCLUSIONES

La *integración del conocimiento* que consigue el alumno al abordar el problema de forma personal y desde *perspectivas distintas*. En cuanto al aprendizaje se trabajan con los dos paradigmas mas importantes en la actualidad, es decir el aprendizaje por *descubrimiento y significativo*. El alumno trabaja con conceptos (función, objetivos, jerarquías, herencias...) y métodos, que facilitan el *conocimiento profundo* (el que permanece más tiempo en su memoria a largo plazo) de los sistemas digitales y que logran un *equilibrio entre las soluciones* implementadas en software y en hardware de los sistemas programables. Se trabajan los diagramas de flujo y la modularidad en la solución de los problemas. La *creatividad* de las clases crece (desarrollo de la capacidad crítica del alumno) y *la motivación*, ya que el alumno va realizando su propia biblioteca de objetos y sus propias soluciones. Asimismo, se potencia la participación y el trabajo en grupo.

El alumno se va adaptando a las últimas tendencias de análisis y desarrollo de la ingeniería del software y del hardware aprendiendo a *pensar en términos de objetos,* lo que le ayudará al manejo de la complejidad y le proporcionará una mayor *flexibilidad* y rapidez de pensamiento de cara a sus desarrollos. Por otra parte, adquiere un *enfoque multidisciplinar* y, al trabajar con esquemas y jerarquías a distintos niveles de complejidad, se le facilita, mediante simulación, las labores de mantenimiento y diagnóstico, ya que podrá identificar mejor los módulos, funciones y los comportamientos esperados en cada nivel de la implementación. Se podrán crear por programación simuladores educativos para los sistemas digitales.

BIBLIOGRAFÍA.-

[1] E. Mandado, "*Sistemas Electrónicos Digitales*", Ed. Marcombo, 5ª Ed, 1984.
[2] J. Terrón, "*Metodologías de Aprendizaje Basadas en el Conocimiento Experto y la Hipermedia. Aplicaciones Navales*", Facultad de Ciencias Náuticas, Universidad de Cádiz, Marzo, 1995.
[3] J. D. Novack, D. B. Gowin, "*Aprendiendo a Aprender*", Ed. Martinez Roca, 1988.
[4] S. de la Torre, "*Manual de la Creatividad. Aplicaciones Educativas*", Ed. Vicena Vives, 1991.
[5] Robert Hesel, "*Graphical Programming*", Ed. Prentice Hall, 1995.
[6] Visual Solution Incorporated, "*VisSim. User' s Guide*", 1994.
[7] R. M. Ibáñez, *"El Arte de Relacionar", Manual de la Creatividad. Aplicaciones Educativas*", Ed. Vicena Vives, pp. 244-247, 1991.