TEMA 70: *Control programado: tipos, elementos y características.*

Esquema: **Autor: Jose Sebastián Trocoli**

1. INTRODUCCIÓN

Los primeros sistemas de control estaban basados en sistemas mecánicos, como el controlador de velocidad centrífugo de la máquina de vapor de Watt.

La utilización de actuadores eléctricos propicia la aparición de controladores de esta naturaleza y, por fin, los dispositivos electrónicos, y más concretamente los dispositivos electrónicos digitales, son hoy día el mecanismo más común en los sistemas automáticos de control.

Estas tecnologías de fabricación de los controladores han soportado adecuadamente las tareas fijas para las que habían sido diseñados, permitiendo, en general, pequeñas o nulas variaciones en dichas tareas, por lo que cuando se producían variaciones en el proceso controlado, había que cambiar el controlador por otro adecuado al nuevo proceso.

Si la aparición de los microprocesadores fue un factor muy importante

en el desarrollo de las computadoras, no lo fue menos en el campo de los sistemas de control, al posibilitar la programación de los controladores construidos con estos dispositivos.

Los sistemas automáticos de control programable están basados en controladores cuyo funcionamiento depende de un programa almacenado en su memoria.

En el caso de un cambio en el proceso controlado bastará con cambiar el programa de control (reprogramar el controlador) para obtener el nuevo funcionamiento.

En la Figura 70.1 se presenta el esquema general de un sistema de control, donde se han indicado los componentes que serían sustituidos por el controlador programable.

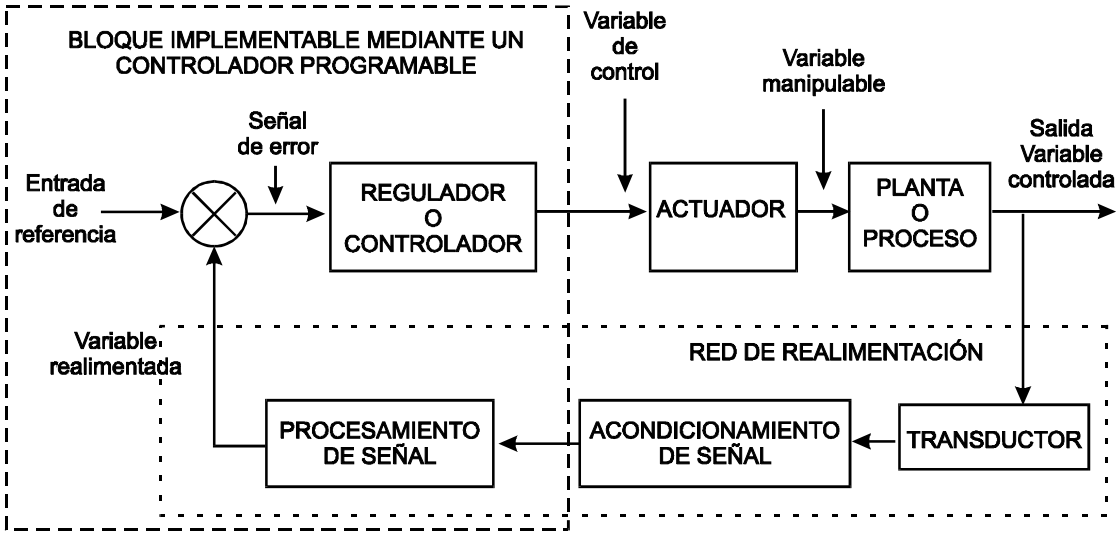

Figura 70.1

En el ámbito escolar, son muchas las soluciones de este tipo que han aparecido, y que permiten que los alumnos se familiaricen con este tipo de control, lo cual permite a los alumnos que los integren en el desarrollo de sus proyectos, así como familiarizarse con los principios básicos de programación.

2. TIPOS DE CONTROL PROGRAMADO.

El control de procesos programado, normalmente se refiere a control en lazo cerrado con una computadora, se clasifica en cuatro tipos o esquemas de control, denominados: secuencial o lógico, digital directo, analógico-digital y descentralizado.

Al tratarse de control en lazo cerrado, el flujo de información será desde y hacia la computadora.

Control secuencial o control lógico. Se trata de sistemas de control en los que el intercambio de información entre la computadora y la planta o proceso, se realiza mediante señales digitales, o señales del tipo todo o nada.

La computadora realiza, mediante un programa (lógica programada), las funciones de un autómata finito o máquina de estados (lógica cableada), al estilo de estos últimos sistemas.

Control digital directo. En este esquema de control, la computadora sustituye a los correspondientes bloques del sistema de control clásico. Figura 70.2. Se utilizan las interfases necesarias entre el entorno analógico de la planta y el digital de la computadora: los convertidores de datos o convertidores Analógico/Digital (CA/D) y Digital/Analógico (CD/A).

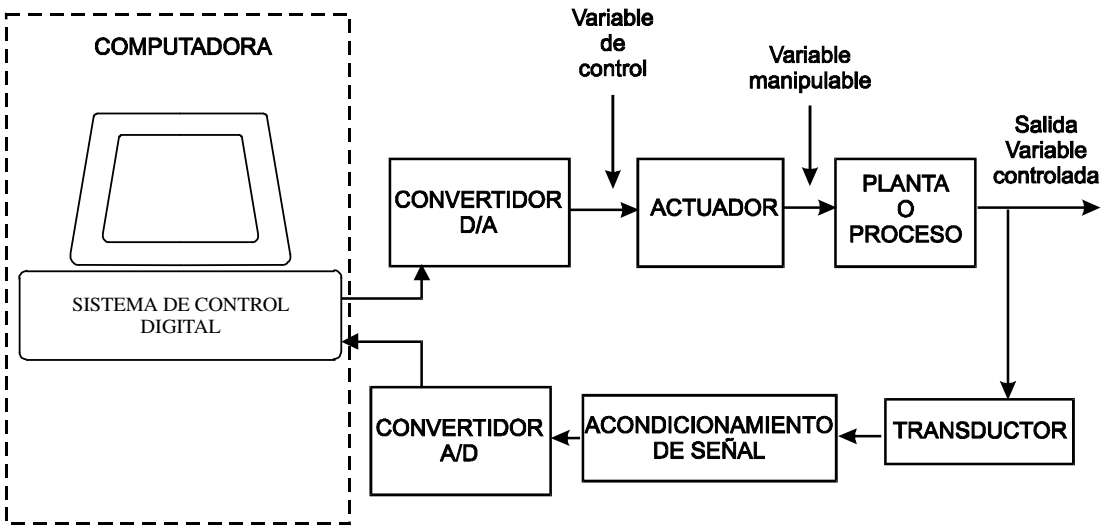

Figura 70.2

Aunque en la figura se representa un solo lazo de control, es normal que una computadora pueda controlar simultáneamente varios de ellos, lo que plantea una de las desventajas de este esquema, ya que un fallo en la computadora dejaría todo el sistema fuera de servicio, mientras que en control clásico, el fallo de uno de los bucles no tiene por qué afectar al resto del sistema.

Control analógico-digital. Permite eliminar el inconveniente del control digital directo. Se trata de un sistema en el que la computadora genera, en lazo cerrado, la señal de consigna de un lazo de control clásico con

regulador independiente. Figura 70.3.

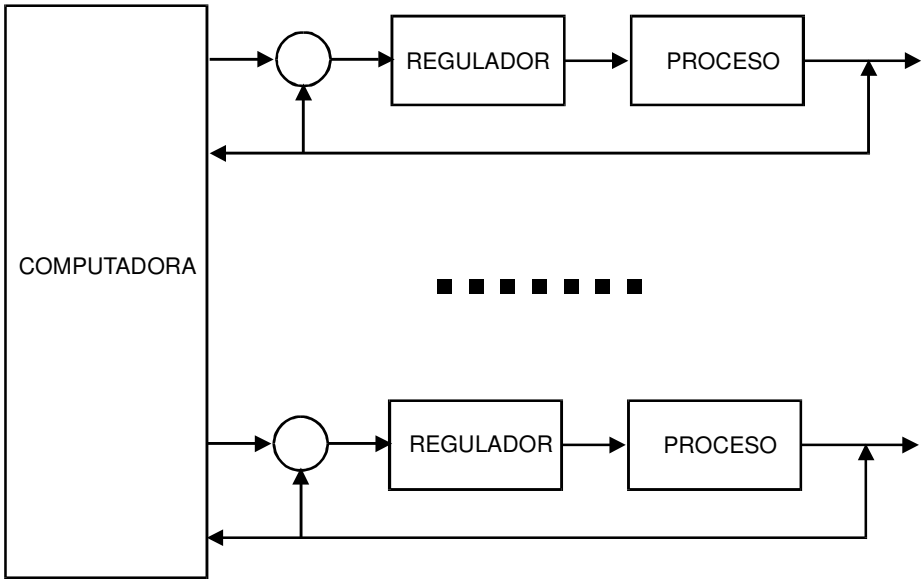

Figura 70.3

El regulador independiente suele ser de tipo analógico, o digital de lógica cableada. Si la computadora dejase de funcionar, este regulador independiente podría hacerse cargo del lazo de control correspondiente, permitiendo que el sistema continúe funcionando.

En la Figura 70.4 se representa uno de los lazos de control en el que se incluye el regulador implementado por la computadora.

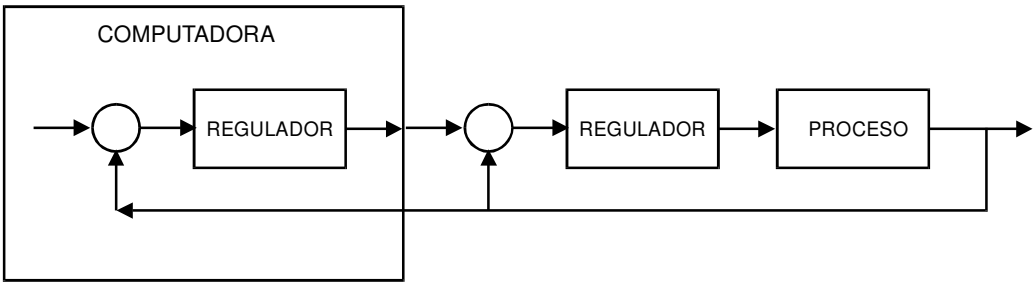

Figura 70.4

Con este esquema, en el que el regulador correspondiente a la computadora puede ser reprogramado fácilmente, se agrega flexibilidad al sistema, y aunque no llegue al nivel de la que tiene el esquema de control digital directo, se agrega un margen de seguridad que, en determinados sistemas puede resultar imprescindible, siempre que el aumento de coste sea permisible.

Control descentralizado. Si en un control analógico-digital los reguladores independientes se sustituyen por reguladores programables, se tiene este tipo de control, en el que todo el conjunto suele estar, a su vez, controlado por una computadora central, tal y

como se representa en la Figura 70.5.

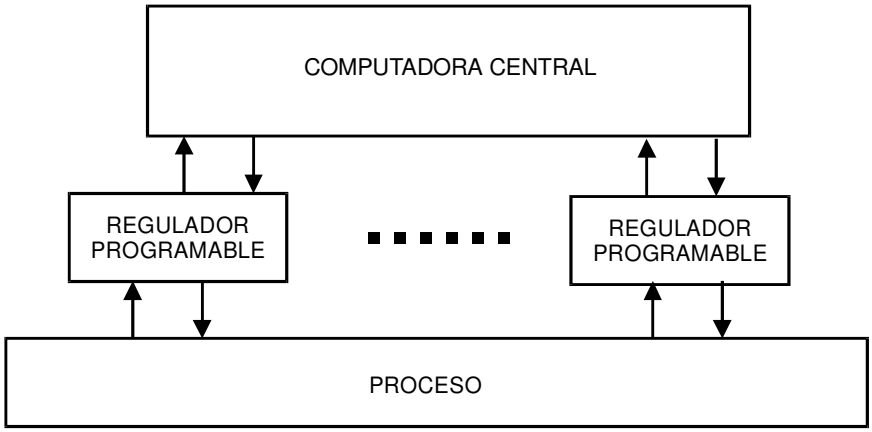

Figura 70.5

La principal característica de este esquema de control, aparte de la gran flexibilidad del mismo, es la seguridad, ya que ante el fallo de uno cualquiera de los lazos individuales, la computadora central puede realizar una reasignación de tareas reprogramando el resto de los reguladores.

3. ELEMENTOS DE LOS SISTEMAS DE CONTROL PROGRAMADO.

El elemento principal en estos sistemas es el regulador programable o, en un caso más general, la computadora. Sin embargo, aparece un elemento nuevo que posibilita la conexión o interfase entre el ambiente analógico inherente al mundo físico de los procesos, y el digital en el que funcionan las computadoras. Este elemento es el convertidor de datos, mencionado anteriormente, en sus dos versiones: CA/D y CD/A.

3.1. Convertidores de datos.

Para la conversión de una señal analógica en otra digital, en primer lugar debemos realizar un muestreo de la señal analógica, esto es debemos tomar el valor de esta señal en instantes de tiempo determinados. Estos valores se denominan muestras, y la señal podrá ser reconstruida a partir de dichas muestras siempre que la frecuencia de muestreo sea superior al doble de la máxima frecuencia de la señal original (teorema de muestreo).

En general, para la frecuencia de muestreo se suele tomar un valor varias veces superior al mínimo especificado, con el fin de facilitar el filtrado que será necesario para reconstruir la señal original, pues en

caso contrario aparecería superpuesto a dicha señal un rizado a la frecuencia de muestreo.
Para realizar el muestreo de una señal analógica, se utilizan unos circuitos denominados etapas de muestreo y retención ("Sample and Hold") S&H, Figura 70.6.

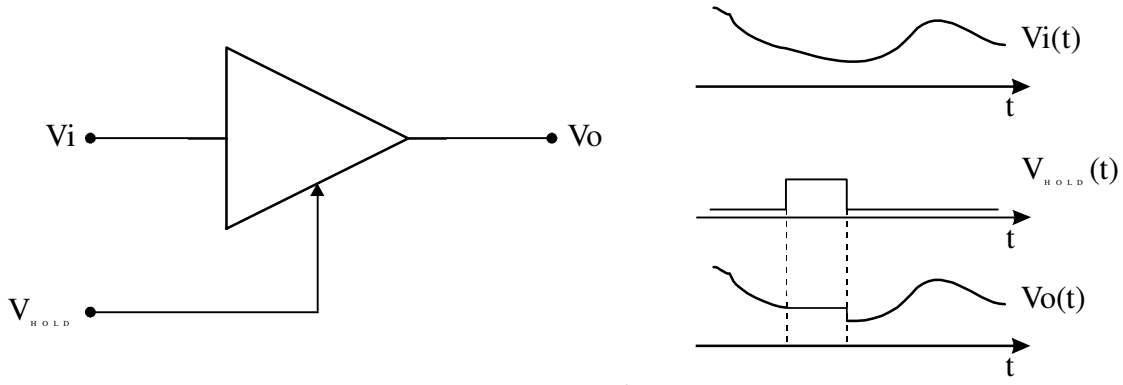

Figura 70.6

Además de tomar la muestra ("Sample"), las etapas de S&H deben mantener ("Hold") el valor de la misma durante el tiempo suficiente para que el convertidor realice la discretización, u obtención del código binario correspondiente a dicha muestra. Otro circuito que se utiliza en los sistemas de adquisición de datos es el multiplexor analógico, Figura 70.7.

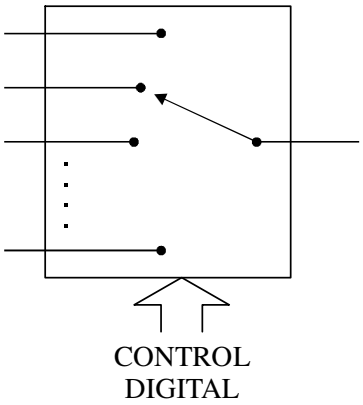

Figura 70.7

La utilidad de este subsistema, es la posibilidad de utilizar un único canal, etapas de S&H y de CA/D, para la cuantificación, siempre y cuando la frecuencia de funcionamiento de este canal sea lo suficientemente alta respecto a las frecuencias máximas de las señales de entrada. Las señales presentes en los sistemas físicos de los procesos industriales y las características de los dispositivos electrónicos que implementan dichos canales actualmente, permiten este ahorro en la mayoría de los casos.

En primer lugar, tradicionalmente, se estudian los convertidores Digital/Analógico, pues estos pueden formar parte de algunos circuitos con los que se implementan los convertidores Analógico/Digital.

Los *Convertidores Digital/Analógico* (CD/A), convierten un código digital de N bits, aplicado a su entrada, en un valor proporcional de tensión o, más habitualmente, de corriente en su salida.

Entre las características de los CD/A que pueden ser de nuestro interés, se tienen:

- Resolución, normalmente relacionada con el número de bits de entrada, o variación en la corriente, o tensión, de salida que se produce para dos códigos de entrada consecutivos. Su valor absoluto es igual al peso del bit menos significativo (1 LSB).
- Tiempo de establecimiento, o de respuesta, o tiempo que transcurre desde que aplicamos el código en la entrada y hasta que la salida alcanza su valor correspondiente, o se encuentra dentro de un margen especificado alrededor de dicho valor.
- Errores, o diferencia entre el verdadero valor que debería tomar la salida y el que realmente toma. Aparte del error inherente a la resolución limitada, se tienen otros debidos a las imperfecciones en los procesos de fabricación, tolerancias y efectos de la temperatura. Los fabricantes agrupan todos estos errores y los expresan normalizados respecto al peso del bit menos significativo.

Los *Convertidores Analógico/Digital* (CA/D), convierten el valor de la tensión aplicada en su entrada, en un código binario de N bits, en su salida, que representa el valor de dicha tensión de entrada.

En general, los circuitos de CA/D se basan en la comparación entre la señal de entrada y una referencia. Dependiendo de cómo se implemente esta comparación se obtienen diferentes circuitos para estos convertidores.

Entre las características de los CA/D que pueden ser de nuestro interés, se tienen:

- Resolución, relacionada con el número de bits del convertidor, es igual al peso del bit menos significativo (1 LSB).
- Tiempo de conversión, o de respuesta, o tiempo que transcurre desde que aplicamos la señal analógica de entrada hasta que aparece el código binario de salida. Este parámetro es fundamental y debe ser tenido en cuenta a la hora de elegir un convertidor, en

función de la máxima frecuencia de la tensión analógica de entrada (teorema de muestreo).

- Errores, o diferencia entre el código binario de salida y el valor verdadero de la tensión analógica de entrada. Aparte del error inherente a la resolución limitada, se tienen otros debidos a las imperfecciones en los procesos de fabricación, tolerancias y efectos de la temperatura. Los fabricantes agrupan todos estos errores y los expresan normalizados respecto al peso del bit menos significativo.

3.2. Instrumentos programables y su interconexión.

Para repasar los distintos elementos que se utilizan para el control vamos a distinguir dos ámbitos.

3.2.1 En el ámbito Industrial.

Actualmente la industria está automatizando la mayor parte de los procesos de producción, con el fin de aumentar la productividad de los mismos y reducir los costes.

Con este propósito, desde hace algún tiempo se comenzaron a diseñar y a construir sistemas que permitieran la monitorización y el control de plantas completas de producción.

Con el fin de flexibilizar los diseños de estos sistemas, surgen normas relativas a los mismos. La primera de ellas data de 1965, IEEE 488 o GPIB ("General Purpose Interface Bus") y define un sistema de instrumentación compuesto por instrumentos programables y cómo se deben interconectar. Mediante este sistema, u otros análogos, se podrían implementar cualquiera de los esquemas vistos para el control programado.

En este apartado vamos a hacer una introducción a estos sistemas de instrumentación, y de control si como tal se les programa.

Se define un instrumento programable como un sistema electrónico de medida o generación de variables, basado en un procesador digital, en cuya memoria se sitúa un programa que automatiza su funcionamiento.

Este tipo de sistemas programables se han implementado en la industria en forma de los conocidos Autómatas Programables (Figura 70.8), los cuales presentan muchas variantes, desde autómatas compactos capaces de gestionar un pequeño número de entradas/salidas, a sistemas más complejos y modulares con los que podremos configurar

el conjunto en función de las necesidades de la aplicación.

Figura 70.8

Estos sistemas de control programado permiten la interconexión entre elementos, permitiendo configuraciones complejas, donde uno de los elementos funciona como Master y el resto de ellos cuelgan del mismo, sincronizando su funcionamiento con el elemento principal.

Además, permite la monitorización del sistema, bien mediante el uso de ordenadores, bien mediante el uso de sistema de captación y representación de datos SCADA (Supervisory Control And Data Acquisition (Supervisión, Control y Adquisición de Datos) Figura 70.9) donde empleamos un software para ordenadores que permite controlar y supervisar procesos industriales a distancia, en la mayoría de los casos, interactuando junto con los autómatas programables.

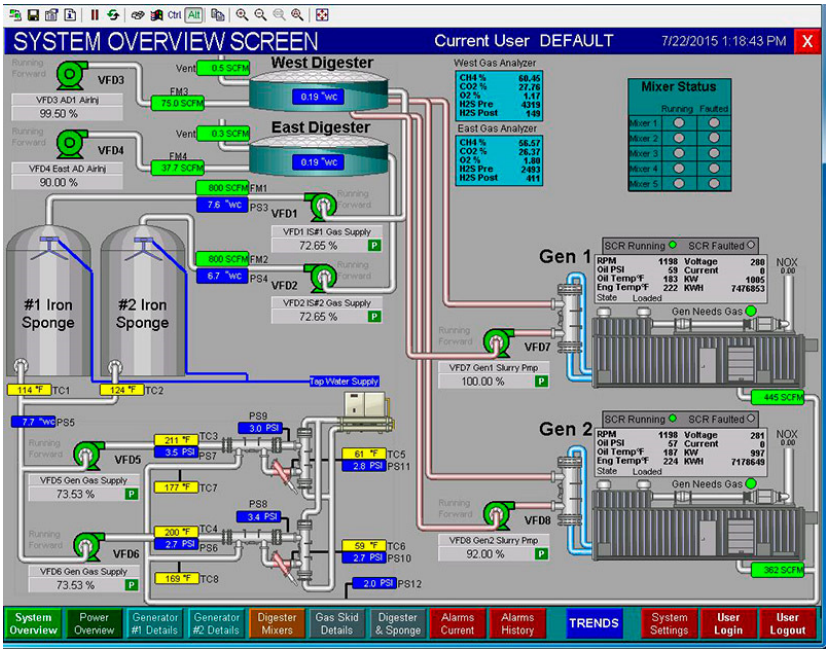

Figura 70.9

3.2.2 En el ámbito escolar.

El desarrollo de este tipo de sistemas no ha permanecido ajeno al ámbito escolar, apareciendo distintos tipos de soluciones que permitían implementar elementos programables para el control de los componentes integrantes de los proyectos tecnológicos.

Poco a poco fueron llegando a los centros educativos dotaciones de este tipo, primero con tarjetas controladoras programables que en realidad funcionaban como pequeños autómatas programables.

En este sentido las primeras en llegar fueron las Tarjetas ENCONOR (Figura 70.10), las cuales contaban con 8 entradas digitales, 5 entradas analógicas, 8 salidas digitales y 4 salidas analógicas.

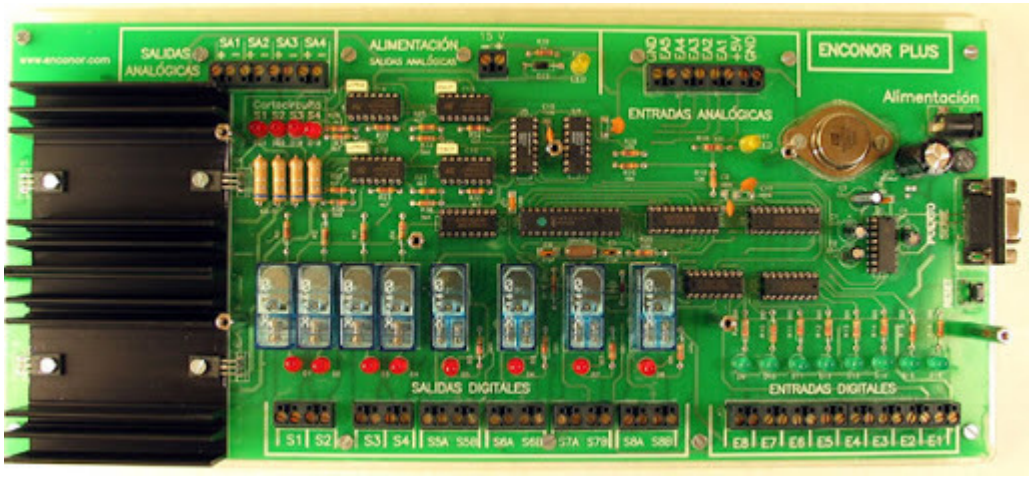

Figura 70.10

Aunque fueron las primeras, no han sido las únicas. Posteriormente en el ámbito docente han ido apareciendo tarjetas o sistemas programables que han permitido ampliar los aspectos del constructivismo que se aplican en los proyectos tecnológicos, incluyendo en ellos sistemas de control programado, añadiendo los elementos de programación necesarios.

Un ejemplo de ello es la tarjeta controladora FISCHERTECHNIK (Figura 70.11) la cual además de permitir controlar los elementos construidos, incluye un kit de piezas fácil y rápidamente ensamblables con las que poder construir proyectos complejos de forma rápida.

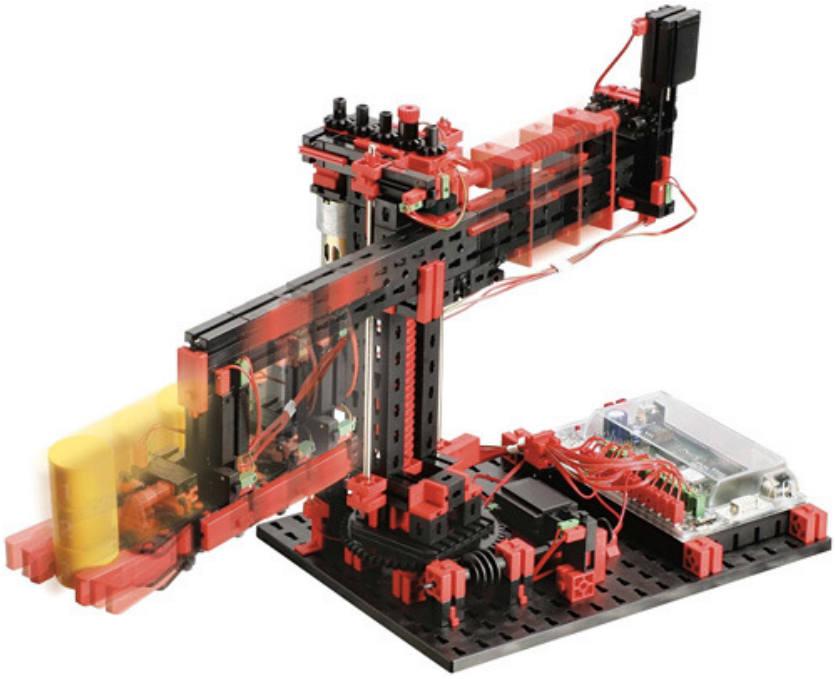

Figura 70.11

Una característica especial que tiene la robótica educativa es la capacidad de mantener la atención del estudiante. El hecho de que pueda manipular y experimentar favorece que pueda centrar sus percepciones y observaciones en la actividad que está realizando. El uso de herramientas programables promueve el proceso de enseñanza-aprendizaje, pues facilita la integración de lo teórico con lo práctico, el desarrollo de un pensamiento sistemático y la adquisición de nociones científicas.

Este tipo de elementos permite la integración del término STEM (Science, Technology, Engineering and Mathematics) dentro del aula. La doctrina de la educación STEM se lleva impulsando en Estados Unidos desde hace dos décadas, siendo en estos momentos el pilar de su reforma educativa.

También permite el desarrollo de alguna de las competencias básicas, como es el **Tratamiento de la información y competencia digital**. Son dos los aspectos de la tecnología digital aplicada al ámbito educativo, como un medio de información y como medio de construcción, ambos con la misma importancia, aunque en muchos casos se le da más importancia al lado de la información que al lado constructivo. Esto es consecuencia del uso masivo de la denominación TIC (Tecnologías de la información y la comunicación) para referirse a la "Tecnología Digital", creando un efecto desfavorable sobre la cultura popular y también sobre el sistema educativo, dando más importancia a la información y su disponibilidad en internet, dejando de lado la tecnología del control

programado. El uso de este tipo de sistemas de control nos permite equilibrar esta deficiencia.

Esta problemática ya la planteó Seymor Papert al desarrollar la teoría del construccionismo, una teoría educativa que fundamenta el uso de la tecnología digital en educación, que se resume de la siguiente manera:

Constructivismo + Tecnología = Construccionismo.

Papert además de formular dicha teoría, creó objetos de construcción que permitieran cambiar el modo en el que aprenden los niños. Junto a un equipo de investigadores del MIT (Instituto Tecnológico de Massachusetts) en 1967 creó el lenguaje de programación LOGO.

Posteriormente sus ideas construccionistas sobre el aprendizaje interesaron a la Compañía LEGO, naciendo de esta colaboración los "ladrillos programables de LEGO" en el año 1998, los cuales permiten relacionar la construcción con la programación, denominando estos sistemas programables como LEGO MINDSTORMS[1], los cuales permiten crear de forma sencilla elementos fácilmente programables, mediante el ensamble de piezas de lego technics y el control de uno de esos ladrillos programables. La rapidez en el montaje y desmontaje de piezas permite un desarrollo del tipo prueba/error, logrando que el alumno evolucione rápidamente y su aprendizaje sea un proceso constructivo no repetitivo (Learning by doing)

En algunos de los centros educativos, y en concreto en la Comunidad de Madrid, estos "ladrillos programables" Mindstorms RCX 1.5 (Figura 70.12) llegaron como dotación de centro, primero en unos kit (4 en total) que incluían piezas de lego, y posteriormente sólo los ladrillos programables (12 en total).

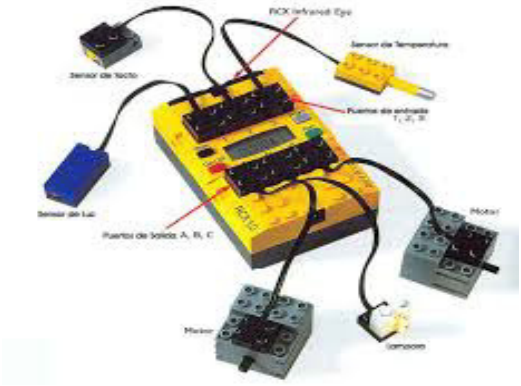

Figura 70.12

[1] De hecho el nombre de Mindstorms proviene del título de un libro suyo llamado "Mindstorms : Children, Computers, and Powerful Ideas"

Actualmente, son varias las versiones de estos “ladrillos programables” de LEGO. Partiendo de la primera versión del RCX la 1.0 , posteriormente se actualizaron en el RCX 1.5 y RCX 2.0, aunque la interacción del sistema era muy limitada por el tipo de motores que utilizaba los cuales permitían muy poco control.

La aparición del NXT (Figura 70.13) en las versiones 1.0 , 1.1 , 2.0 y 2.1 de LEGO, supuso un salto cualitativo al aparecer en el sistema servomotores, y un mayor número de sensores, como el sensor ultrasónico que permite medir distancias, el sensor de sonido, un acelerómetro o un sensor de color mucho más preciso que los anteriores.

Figura 70.13

El último eslabón en la evolución de este tipo de sistemas programables de LEGO es el **EV3** (Figura 70.14), un sistema mucho más potente, con un microprocesador de última generación, y pese a tener una estructura igual a la del NXT, y hacer compatibles todos los sensores del NXT, añade el control de un motor más, y sobre todo añade un girosensor que nos permite medir el ángulo de giro en un plano y con ello interactuar de forma más precisa con el mundo que le rodea.

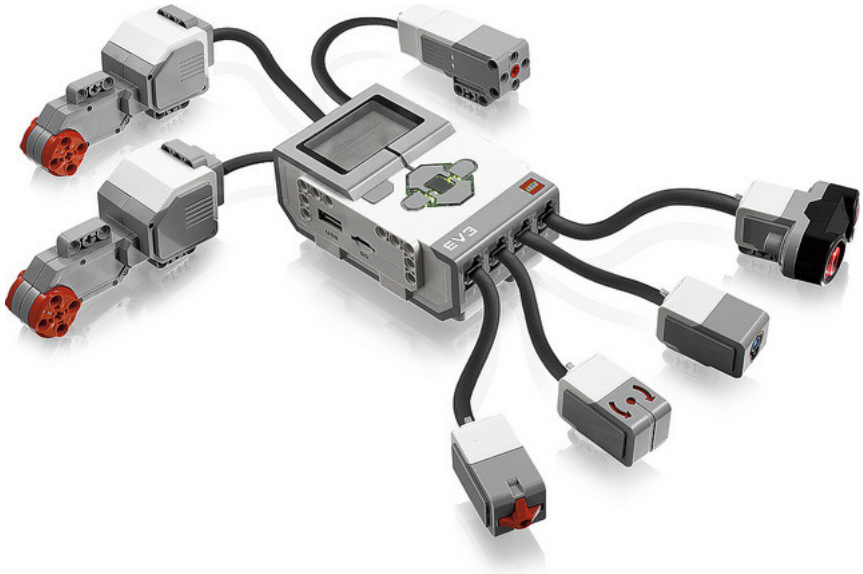

Figura 70.14

El talón de Aquiles de este tipo de sistemas es sobre todo el alto coste de los sistemas actualmente (aunque en el momento que aparecieron era uno de los sistemas más económicos para la iniciación a la robótica, actualmente ha sido superado por otros sistemas más baratos) pues como todos sabemos LEGO es bastante caro.

Por último, conviene destacar elementos programables que permite implementar sistemas de control con costes muy bajos, nos referimos al sistema de control ARDUINO.

El proyecto "ARDUINO" se crea en el año 2005 como un proyecto para estudiantes del Interaction Design Institute (Ivrea – Italia). En aquel momento los estudiantes utilizaban el microcontrolados Basic Stamp con un coste de 100$, un coste más que considerable para el estudiante. El objetivo del proyecto era crear un elemento de bajo coste que permitiera su aplicación en proyectos digitales, sin que fuera necesario grandes conocimientos técnicos.

El proyecto Wiring desarrolla una placa de control cuyo hardware se basa en una placa de circuito impreso (PCB), con un microcontrolador ATmega 168, sumado a un Ambiente de Desarrollo Integrado (IDE)basado en funciones de procesamiento y una biblioteca de funciones que permite programar fácilmente el microcontrolador. En ese mismo año 2005, se sustituye el ATmega168 por el microcontrolador ATmega8, mucho más barato que el anterior, y se produce la separación del proyecto inicial Wiring, apareciendo un nuevo proyecto al que renombraron como ARDUINO (Figura 70.15)

Figura 70.15

El nombre Arduino viene de un bar en Ivrea, Italia; en donde algunos de los fundadores del proyecto Arduino solían reunirse. El bar tiene el nombre de " Bar di Re Arduino", y fue nombrado en honor a Arduino de Ivrea, quien fue el Marqués de la Marcha de Ivrea y Rey de Italia desde el año 1002 hasta el año 1014. El equipo inicial de Arduino estaba conformado por Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino y David Mellis.

Las nuevas placas de control, además de su bajo coste pues rondan los 20 euros (precio de una placa ARDUINO UNO), tienen una versatilidad total, ya que soportan múltiples lenguajes de programación. En la figura XX podemos ver dos placas ARDUINO, la primera un ARDUINO UNO y la segunda una ARDUINO MEGA (Figura 70.16).

Existen gran cantidad de placas con sensores y accesorios para poder implementar sistemas complejos con este tipo de placas, todos ellos con unos costes más que interesantes, permitiendo realizar prototipos a bajo coste, y sin límite en su complejidad.

Como inconveniente podemos indicar que la implementación de este tipo de tarjetas en proyectos resulta sencilla, pero no así el desarrollo de los distintos elementos que forman dicho proyecto, necesitando en ocasiones realizar piezas y mecanismos algunos de ellos con la ayuda de impresoras 3D, que en el caso de no resultar idóneas, producen demoras y sobrecostes en el desarrollo de los proyectos.

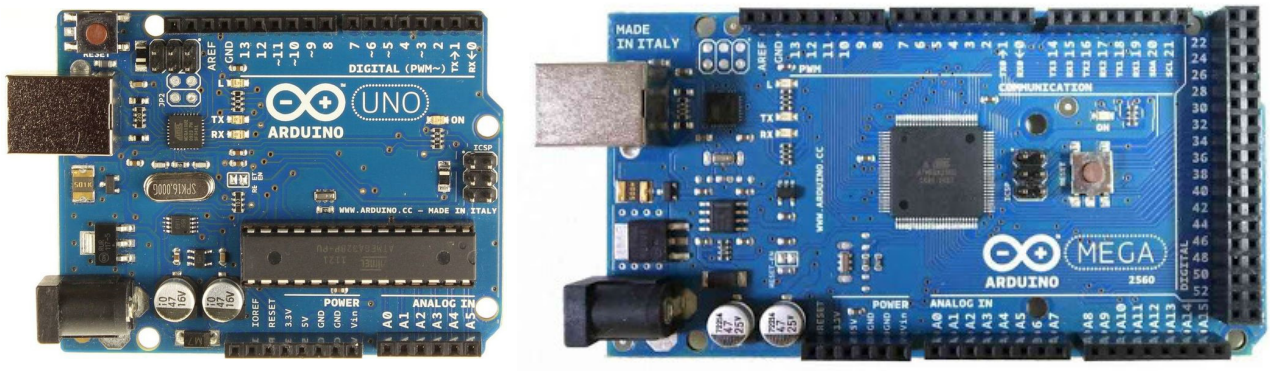

Figura 70.16

Por último, no podemos dejar de incluir en este apartado la placa controladora Raspberry Pi (Figura 70.17), aunque realmente se trata de un ordenador de placa reducida o de placa simple, de bajo coste que ronda desde los 20€ las primeras versiones a unos 50€ en la versión más moderna Raspberry Pi 4. Este proyecto se desarrollo en 2012 en Reino Unido por la Fundación Raspberry Pi con el objetivo de estimular la enseñanza de la informática en las escuelas.

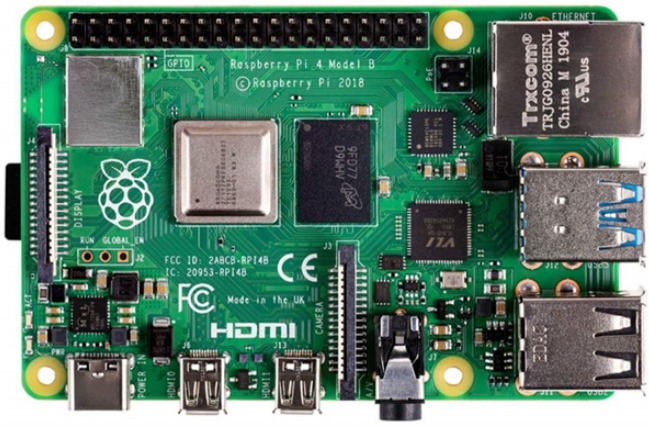

Figura 70.17

3.3. Programas para aplicaciones de control.

Las computadoras necesitan un programa, o relación de instrucciones, para poder realizar las tareas que se les encomiendan. Para crear los programas se utilizan otros programas que se denominan lenguajes de programación. Dependiendo del nivel de concreción que se utilice en stos lenguajes, se pueden clasificar en:

- Lenguajes de bajo nivel.
- Lenguajes de alto nivel.
- Lenguajes orientados a objetos.

❑ Lenguajes de bajo nivel.

Lenguaje máquina: es el único que permite le programación directa de una computadora. Utiliza el alfabeto binario (‘0’ y ‘1’). Fue el primero que se utilizó en la programación de computadoras, pero tuvo que descartarse por la facilidad con que se cometían errores y la dificultad que suponía realizar las correcciones.

El lenguaje máquina sólo puede utilizarse con la computadora para la que fue escrito, por lo que se dice que no es "portable".

Lenguaje ensamblador: supuso el primer paso para eliminar los problemas que entrañaba el lenguaje máquina. Las instrucciones de ensamblador utilizan códigos alfanuméricos para los códigos de operación y numéricos para los operandos, con lo que se disminuye la probabilidad de cometer errores en la escritura de los programas.

Para que una computadora realice un programa es necesario que esté escrito en lenguaje máquina, por lo que los programas escritos en ensamblador tienen que traducirse mediante otro programa que se denomina compilador.

❑ Lenguajes de alto nivel.

Estos lenguajes se crearon con el fin de facilitar la tarea de los programadores, en particular:

- Lograr la independencia entre el programa y computadora que lo ejecuta.
- Aproximar las instrucciones al leguaje humano, utilizando palabras como códigos de operación.
- Obtener librerías de rutinas, con las funciones de uso más frecuente, con lo que evitaría el tener que crearlas cada vez que se necesitaran.

Para que la computadora pueda ejecutar un programa escrito en un lenguaje de alto nivel, deben ser traducidos a lenguaje máquina.
Esta operación se puede realizar por dos métodos, dependiendo del lenguaje:

- Con un *intérprete*: este programa va traduciendo y ejecutando las instrucciones una por una, siguiendo la secuencia del programa. Cuando se produce un error el programa se detiene.
- Con un *compilador*: en este caso, se traduce el programa completo, creándose un nuevo programa denominado programa objeto. Posteriormente, el programa objeto tiene que ser montado ("linkado"), o enlazado con las librerías de rutinas que sean necesarias, dando lugar al programa ejecutable. En el caso de producirse un error en el proceso anterior, el compilador lo indicará para su corrección.

Entre los programas de alto nivel más conocidos, se tienen: Pascal, Fortran, Cobol, C, Basic, etc. Muchos de estos programas se basan en la programación estructurada, que consiste en la creación de procedimientos (programas) que pueden ser "llamados" dentro de otros procedimientos como si se tratara de una instrucción más del lenguaje de programación.

Esta técnica permite la creación de librerías especializadas, que en muchos casos se suministran por los fabricantes de "software", pero que pueden ser modificadas y/o ampliadas por el usuario.

Uno de estos lenguajes estructurados, con aplicación en el control de sistemas, es el lenguaje LOGO, que tiene sus instrucciones escritas en castellano, por lo que es muy fácil de utilizar.

- Lenguajes orientados a objetos.

Estos lenguajes se centran en los datos, objetos, indicando cómo van a ser y definiendo las operaciones a las que se les va a someter.

Muchos lenguajes de alto nivel tienen versiones que soportan la programación orientada a objetos, como por ejemplo: Object Pascal, C++, etc. También existen lenguajes de programación que han sido creados para trabajar con esta técnica, como LabWindws.

Las interfases con el usuario de estos lenguajes, suelen ser bastante amables, basadas en ventanas y en la utilización de bloques que pueden ser programados de forma intuitiva.

Estos lenguajes son muy utilizados en el control programado.

3.3.1 Programación en autómatas programables.

En los sistemas industriales implementados por Autómatas Programables (también llamados PLCs), estos admiten distintos modos de programación, a través de los programas de ordenador que realizan el volcado de dichos programas a los autómatas.

Cada tipo de lenguaje de programación cuenta con sus propias ventajas y desventajas. Todos los sistemas de programación para PLCs pertenecen al estándar IEC 1131-3. Se pueden catalogar, como regla general, entre lenguajes de tipo gráfico o de tipo textual**.**

Uno de los primeros que se implementó fue el de **diagrama de contactos** también llamado **Ladder**, el cual permitía de forma intuitiva programar autómatas con los conocimientos previos que podían tener los operarios en los sistemas de control de lógica cableada. En la figura 70.18 tenemos un ejemplo de ello.

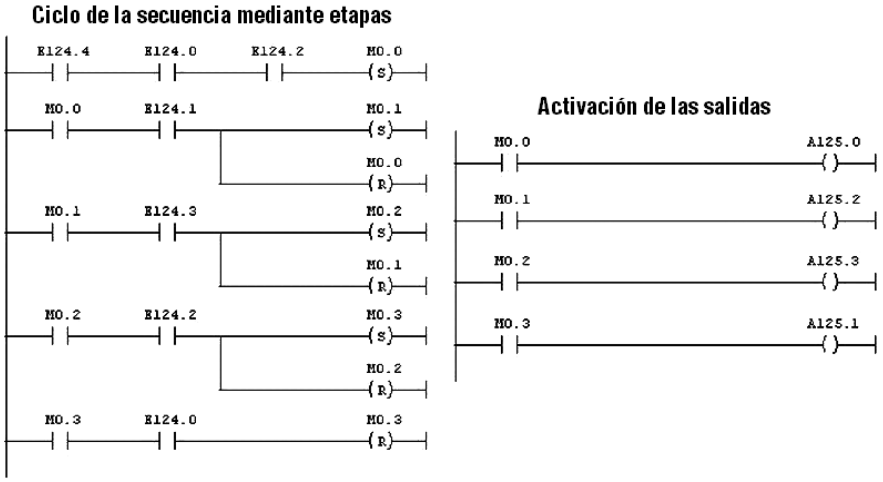

Figura 70.18

Otro modo de programar, mucho más eficiente y potente es el de lista de instrucciones (STL), el cual se parece más a los sistemas de programación utilizados en informática, y el cual utiliza una serie de instrucciones básicas y una estructura de programa concreta en función de la marca del autómata que utilicemos. Vemos un ejemplo en la figura 70.19

```
000   LD     %I0.1   Bp. inicio ciclo
      AND    %I0.0   Dp. presencia vehículo
      AND    %M3     Bit autorización reloj calendario
      AND    %I0.5   Fc. alto rodillo
      AND    %I0.4   Fc. detrás pórtico
005   S      %M0     Memo inicio ciclo
      LD     %M2
      AND    %I0.5
      OR     %I0.2   Bp. parada ciclo
      R      %M0
010   LD     %M0
      ST     %Q0.0   Piloto ciclo
```

Figura 70.19

También se pueden desarrollar estos programas utilizando el Lenguaje de Diagramas Básicos de Funciones (FBD) también llamado diagrama de bloques, el cual utiliza bloques de función para la realización de dichos programas, como se muestra en la figura 70.20.

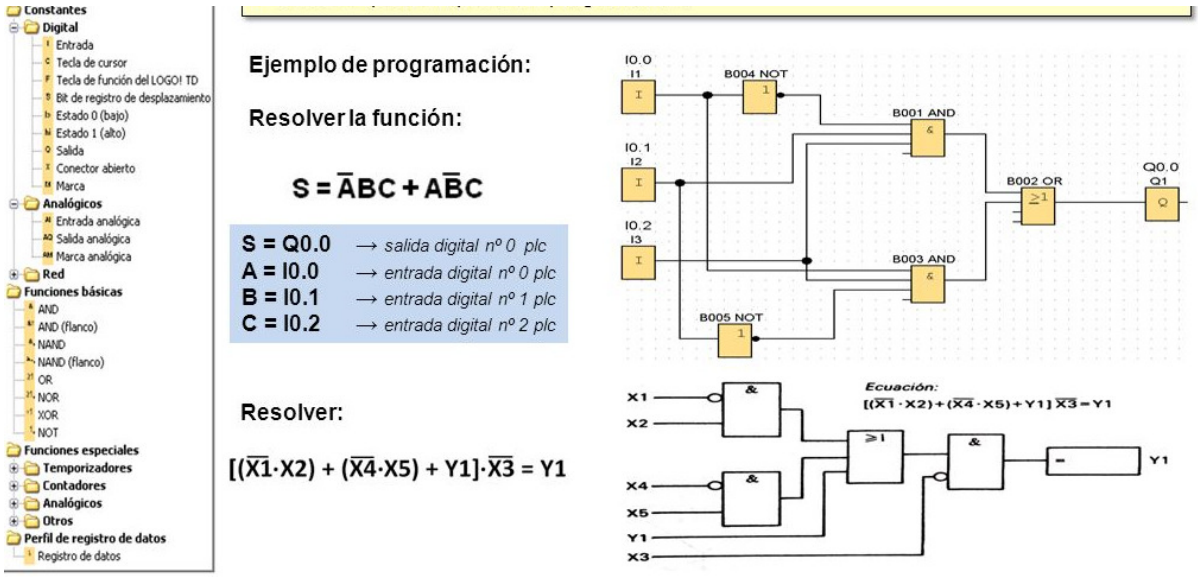

Figura 70.20

Por último podemos nombrar el GRAFCET (Graphe Fonctionnel de Commande Etape Transition), que significa **diagrama de control con etapas y transiciones**, nació en 1977, fuente del trabajo de la Asociación Francesa para la Cibernética Económica y Técnica (AFCET), en principio como síntesis teórica de las diferentes herramientas existentes por aquel entonces (organigrama, organifase, diagramas de Girard, red de Petri, etc...). Le otorgó su actual forma en 1979 la Agencia Nacional para el Desarrollo de la Producción Automatizada (ADESA) francesa. Normalizada en Europa como (EN61131) e internacionalmente en 1992 como (norma CEI 1131).

Básicamente, el GRAFCET es un modelo de representación gráfica, de los sucesivos comportamientos de un sistema lógico, predefinido por sus entradas y salidas. También es un grafo, o diagrama funcional normalizado, que permite hacer un modelo del proceso a automatizar, contemplando entradas, acciones a realizar, y los procesos intermedios que provocan estas acciones (Figura 70.21). En la actualidad no tiene una amplia difusión como lenguaje, puesto que la mayoría de los autómatas no pueden programarse directamente en este lenguaje, a diferencia del lenguaje Ladder. Pero se ha universalizado como herramienta de modelado que permite el paso directo a programación, también con Ladder.

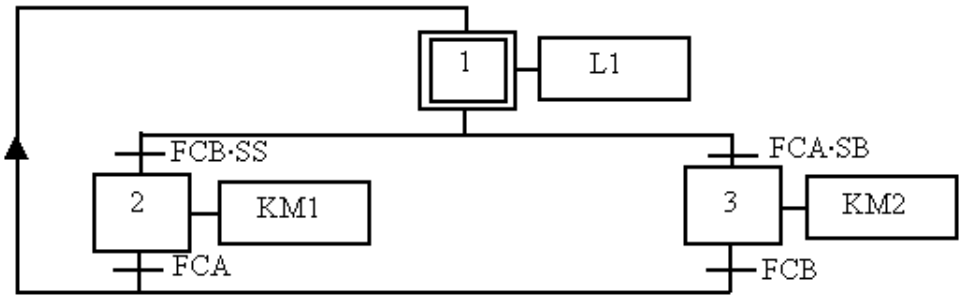

Figura 70.21

3.3.2 Programación de los elementos en el entorno escolar.

Ya hemos visto en los apartados anteriores algunos de los elementos de control que podemos utilizar en el entorno escolar. Ahora trataremos de enumerar alguno de los entornos de programación que podemos utilizar en estos y otros elementos.

Enumeremos algunos de estos lenguajes de programación.

a. SCRATCH

El entorno Scratch (Figura 70.22) es un entorno de programación gráfico por bloques desarrollado por Grupo Lifelong Kindergarten del MIT (Instituto Tecnológico de Massachussets) y uno de los más conocidos para la programación por bloques e iniciación de niños a la programación (por bloques)(Figura 70.23). Su principal ventaja es que permite el desarrollo de habilidades mentales mediante el aprendizaje de la programación, sin necesidad de un conocimiento profundo de un lenguaje de programación.

Figura 70.22

Se puede trabajar de dos formas: online o local, con Scratch instalado en los ordenadores. En el modo online es necesaria la creación de una cuenta online es necesario el uso de una cuenta de correo electrónico (ojo a la creación de cuentas de correo electrónico por parte de los alumnos, ya que se trata de menores y debemos también cuidar la Ley

de Protección de Datos). El entorno online es interesante para el trabajo en grupo si los alumnos comparten usuario y contraseña, pero se ve limitado por la calidad del acceso a Internet en los centros, ya que al conectarse muchos usuarios en ocasiones el trabajo se ralentiza en demasía.

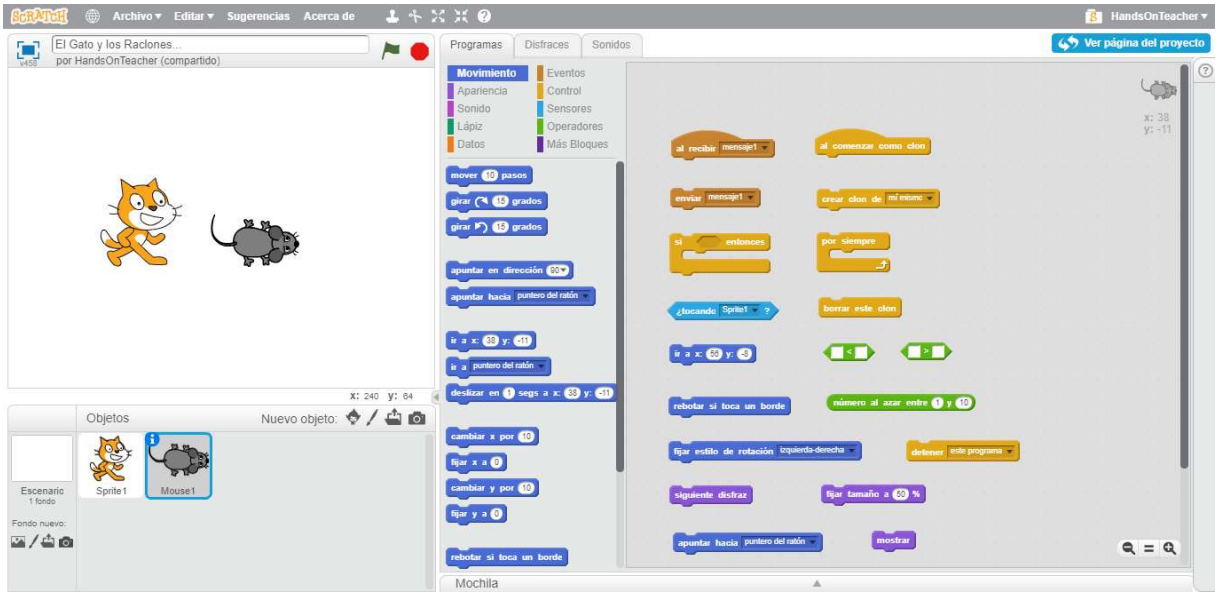

Figura 70.23

La programación se lleva a cabo a través de personajes que desarrollan acciones en un escenario. Los desencadenantes de las acciones que se desarrollan son los llamados Eventos.

De izquierda a derecha, en la zona superior izquierda de la pantalla hay un *escenario*, que muestra los resultados del proyecto actual (animaciones, gráficos tortuga, etc., en tamaño reducido o normal, estando también disponible un modo a pantalla completa) y todas la miniaturas de los *sprites* (llamados *"objetos"* en la versión en castellano de Scratch) listadas en la zona inferior. El escenario usa coordenadas x e y, siendo el punto (0,0) el centro del mismo.

Una vez seleccionado un objeto en la parte inferior izquierda de la pantalla, se le pueden aplicar bloques de instrucciones simplemente arrastrándolos desde la paleta de bloques hacia el área derecha de la pantalla, que contiene todos los pasos asociados con el objeto seleccionado. Bajo la pestaña "Programas" se listan todos los bloques de instrucciones disponibles en las siguientes categorías: Movimiento, Apariencia, Sonido, Lápiz, Datos, Eventos, Control, Sensores, Operadores y Más Bloques, como se ve en la tabla a continuación. Cada instrucción se pueden probar individualmente simplemente haciendo doble clic sobre ella.

Las categorías de los distintos bloques los las que se muestran en la siguiente tabla 1:

Categoría	Notas	Categoría	Notas
Movimiento	Mueve objetos y cambia ángulos.	Eventos	Contiene manejadores de eventos situado al principio de cada grupo de instrucciones.
Apariencia	Controla el aspecto visual del objeto, añade bocadillos de habla o pensamiento, cambia el fondo, ampliar o reducir.	Control	Sentencian condicionales "Si-sino", "Por siempre", "repetir" y "detener programa".
Sonido	Reproduce ficheros de audio y secuencias programables.	Sensores	Los objetos pueden interactuar con el ambiente que ha creado el usuario.
Lápiz	Control del ancho, color e intensidad del lápiz.	Operadores	Operadores matemáticos, generador aleatorio de números, sentencias "y" y "o" que comparan posiciones de los objetos.
Datos	Creación de variables y listas. Hay variables de la nube, pero aún no hay listas de nube. Se podrían implementar en la tercera versión de Scratch.	Más Bloques	Control de bloques y dispositivos externos.

Tabla 1

El sistema de programación es muy intuitivo, y los alumnos son capaces en muy pocas sesiones de realizar programas con cierta complejidad. Además, este lenguaje de programación puede ser usado para programar tanto tarjetas ARDUINO como sistemas LEGO EV3.

b. LEGO® MINDSTORMS®

LEGO Mindstorms (Figura 70.24) es en la actualidad el material de construcción más eficaz para comenzar a experimentar con robots y concentrarnos en el aspecto académico del aprendizaje.

Figura 70.24

Este lenguaje fue desarrollado a partir de LabVIEW (marca comercial de National Instruments). Una de las principales características de este software de programación, es su entorno visual, el cual emula la construcción por bloques, dando la posibilidad a cualquier alumno a acostumbrarse relativamente rápidamente a la programación de bloques.

Este lenguaje permite las instrucciones secuenciales, instrucciones de ciclos e instrucciones de decisiones, éstas últimas, basadas en los datos reportados por los sensores que se pueden añadir al robot.

Todos los bloques de programación que se utilizan para controlar el robot se encuentran en **Paletas de programación** en la parte inferior del Área de documento de programación. Los bloques de programación se dividen en categorías cada una de un color, según su tipo y naturaleza, lo que facilita la búsqueda del bloque que necesita.

BLOQUES DE ACCIÓN
(En orden de izquierda a derecha)
- Motor mediano
- Motor grande
- Mover la dirección
- Mover tanque
- Pantalla
- Sonido
- Luz de estado del Bloque EV3

BLOQUES DE FLUJO
(En orden de izquierda a derecha)
- Iniciar
- Esperar
- Bucle
- Interruptor
- Interrumpir bucle

BLOQUES DE SENSORES
(En orden de izquierda a derecha)
- Botones del Bloque EV3
- Sensor de color
- Giroscopio
- Sensor infrarrojo
- Rotación del motor
- Sensor de temperatura
- Temporizador
- Sensor táctil
- Sensor ultrasónico
- Medidor de energía
- Sensor de sonido NXT

BLOQUES DE DATOS
(En orden de izquierda a derecha)
- Variable
- Constante
- Operaciones secuenciales
- Operaciones lógicas
- Matemática
- Redondear
- Comparar
- Alcance
- Texto
- Aleatorio

BLOQUES AVANZADOS
(En orden de izquierda a derecha)
- Acceso al archivo
- Registro de Datos
- Mandar mensaje
- Conexión Bluetooth
- Mantener activo
- Valor del sensor sin procesar
- Motor sin regular
- Invertir el motor
- Detener programa

Con estos bloques crearemos programas arrastrándolos sobre la zona de trabajo, relacionando e interconectando unos con otros, de modo que programaremos el elemento que hayamos construido con el ladrillo EV3, con programas como el de la figura 70.25.

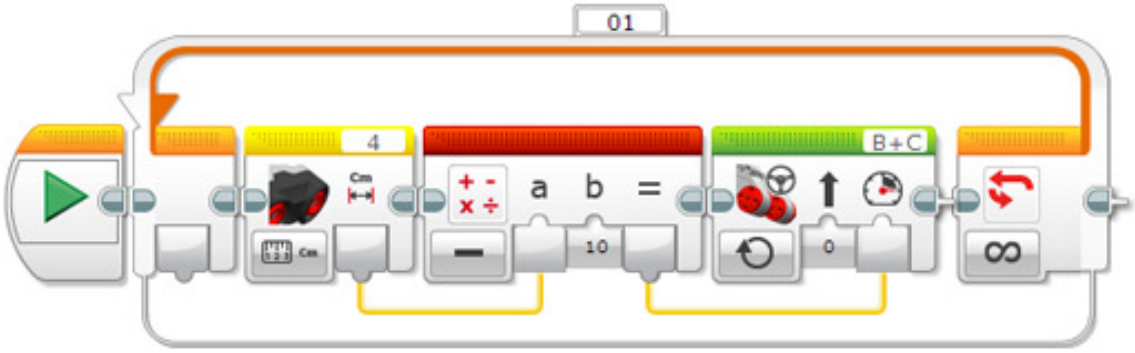

Figura 70.25

c. IDE de ARDUINO.

Se trata de uno de los entornos que podemos utilizar para programar nuestra placa de control ARDUINO (Figura 70.26)

Figura 70.26

Un sketch (o programa) de Arduino consta de dos partes agrupadas dentro de dos funciones: setup y loop.

La función setup es una función que se ejecuta una única vez en el arranque de la placa y al comienzo de la ejecución del programa cargado en la memoria.

La función loop es una función que se ejecuta por primera vez tras la finalización de la función setup y a partir de ahí se repite de forma indefinida.

Además de estas dos funciones, se pueden declarar variables al comienzo del programa que podrán ser utilizadas dentro de cualquiera de ellas. Son las llamadas variables globales.

El lenguaje de programación de Arduino, Wiring, es muy similar al lenguaje de programación utilizado en el lenguaje de programación Processing y similar al C, uno de los lenguajes de programación más utilizados. Cuenta sin embargo con una serie de funciones específicas para el control de las entradas y salidas digitales, tanto para su lectura como su escritura.

Entre los ejemplos del IDE (Entorno de Desarrollo Integrado) existen varios ejemplos. El más sencillo es el blink, que hace parpadear el led integrado (Figura 70.27) en la placa con un periodo de 2s.

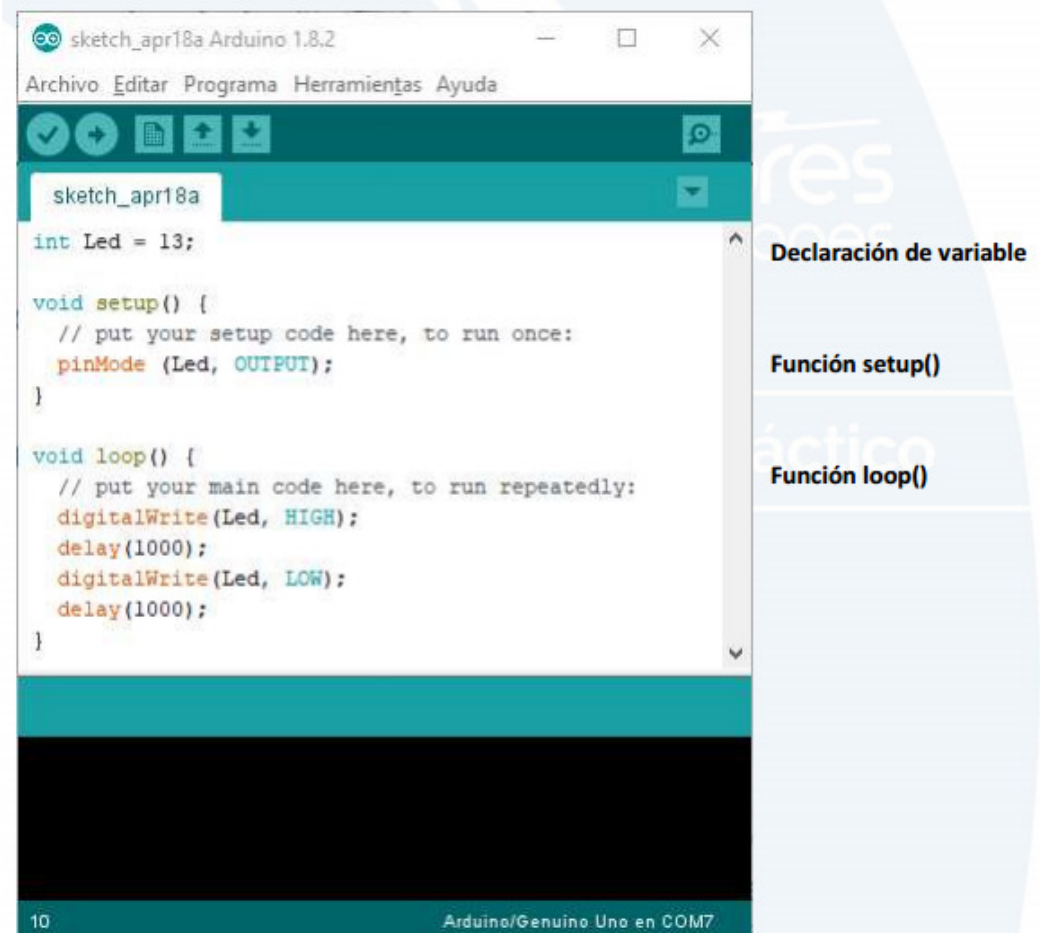

Figura 70.27

Además del propio entorno de desarrollo de Arduino, el conocido como IDE de Arduino, es interesante trabajar con Fritzing (http://fritzing.org) , programa utilizado fundamentalmente para el montaje en placa protoboard y 123d Circuits (https://circuits.io), simulador de Arduino que puedes utilizar para realizar montajes y probarlos con el código.

Fritzing es muy útil para ahorrar tiempo con el montaje en el aula y permitir a los alumnos trabajar de forma práctica sin disponer físicamente de los materiales. Tanto para proyectos con Arduino como para el montaje de componentes de circuitos.

123d Circuits (desarrollado por Autodesk, al igual que Tinkercad, www.tinkercad.com), en el que se puede simular el software en la placa Arduino junto con el correspondiente montaje en una placa protoboard. Permite probar su funcionamiento "virtual" en casa previo al montaje y puesta en marcha "real" en el taller.

El uso de ambos programas es relativamente sencillo. Puedes utilizar los siguientes tutoriales para practicar y realizar los montajes.

Tutorial de Fritzing:
https://www.youtube.com/watch?v=x2qg9nvZ5LI

Tutorial de 123d Circuits:
https://www.youtube.com/watch?v=c0edvZIrjUY

4. CONCLUSIONES.

En este tema hemos presentado los principios del control programado y sus esquemas básicos.

Esta técnica para el control de procesos, aporta las siguientes ventajas:

- Ausencia de limitaciones en cuanto a la complejidad del sistema de control.
- Flexibilidad, el programa de control puede modificarse fácilmente.
- Posibilidad de monitorizar el proceso no solo en tiempo real, sino también en cualquier momento, al ser fácilmente almacenables los datos en la computadora, permitiendo además análisis a largo plazo, etc.

También hemos repasado los distintos sistemas de control que podemos utilizar en el aula, así como sus lenguajes de programación.

Justificamos el uso de estos elementos ya que con frecuencia la educación en ciencias se basa en el aprendizaje abstracto de fórmulas y leyes. En nuestro caso la función del Método de Proyectos o

Aprendizaje Basado en Proyectos (en inglés PBL, Proyect Based Learning) es darle la vuelta a esta situación.

Los fundamentos didácticos del Aprendizaje Basado en Proyectos se pueden resumir en estas orientaciones:

• Que el alumno tenga una situación auténtica de experiencia, es decir, una actividad continua en la que esté interesado por su propia cuenta.
• Que se desarrolle un problema real dentro de esa situación como un estímulo para el pensamiento.
• Que el alumno posea la información y haga las observaciones necesarias para manejarla.
• Que las soluciones sugeridas se le ocurran a él, lo cual le hará responsable para desarrollarlas de un modo ordenado.
• Que tenga la oportunidad para comprobar las ideas por sus aplicaciones, para aclarar su sentido y descubrir por sí mismo su valor

5. REFERENCIAS BIBLIOGRÁFICAS Y DOCUMENTALES.

- ARACIL S., R. y JIMÉNEZ A., A.
  Sistemas discretos de control.
  Ed. ETSII (UPM).
- PUENTE
  Regulación Automática I
  Ed. ETSII (UPM).
- OGATA
  Ingeniería de Control Moderna
  Ed. Prentice Hall.
- LEIGH
  Applied Digital Control,
  Ed. Prentice Hall.
- FRAILE y GARCÍA
  Instrumentación Aplicada a la Ingeniería
  Ed. ETSI C C P (UPM).
- MANDADO, MARIÑO y LAGO
  Instrumentación Electrónica
  Ed. Marcombo.
- PALLÁS
  Sensores y Acondicionadores de Señal
  Ed. Marcombo.
- HUMPHRIES y SHEETS
  ELECTRÓNICA INDUSTRIAL: Dispositivos, Equipos ....
  Ed. Paraninfo.
- HUMPHRIES y SHEETS

ELECTRÓNICA INDUSTRIAL: Dispositivos, Máquinas ....
Ed. Paraninfo.

- MILLMAN&GRABEL
  Microelectrónica
  Ed. Hipanoeuropea
- http://www.educa.madrid.org/web/colegio1/equipamientos/convocatoria_2003/ADL2003/contenido/carpetas/documentos/enconor-plus.pdf
- Empezando con Scratch: (Guía para la realización de nuestro primer programa)
  https://resources.scratch.mit.edu/www/guides/en/Getting-Started-Guide-Scratch2.pdf
- Guía para el currículo de Computación Creativa (¡para una asignatura de 70 horas!):
  http://scratched.gse.harvard.edu/guide/download.html
- Mi página favorita sobre Scratch: Scratch Adventures (en inglés)
  https://sites.google.com/site/scratchadventures/
- Página de tips (Con enlaces a proyectos sencillos, Scratch Cards, etc...) https://scratch.mit.edu/tips
- Guía de Integración Curricular LEGO MINDSTORMS
  Gabriel Ocaña Rebollo – Fundación Scientia

NOTAS