INVESTIGACIÓN & DESARROLLO, No. **3**, Vol. **1**: 54 – 60 (2003)
ISSN 1814-6333

MODELADO Y SIMULACIÓN DE SISTEMAS MECATRÓNICOS

Mauricio Castillo Effen
Laboratorio de Mecatrónica y Automatización Industrial
Universidad Privada Boliviana
mcastillo@upb.edu
(Recibido el 10 febrero 2003, aceptado para publicación el 13 de junio 2003)

RESUMEN

En este artículo se introduce el campo de la mecatrónica, su importancia y relación con los sistemas embebidos, procediendo luego a mostrar algunos retos que representa la tarea de modelar y simular sistemas mecatrónicos. Se presentan también algunos paradigmas de modelación y simulación de sistemas de éste tipo así como alternativas prácticas de cómo encarar los retos mencionados.

Palabras Clave: Software de Simulación, Mecatrónica, Sistemas de Control.

1. INTRODUCCIÓN

La mecatrónica se puede considerar como una rama de la ingeniería relativamente nueva y que adquiere cada vez mayor relevancia, ya que muchos productos, tanto de consumo, como las máquinas y procesos necesarios para fabricarlos presentan un grado mayor de integración de diversos aspectos tecnológicos como son el eléctrico, electrónico, mecánico y computacional.

Lo más importante del enfoque "mecatrónico" consiste en lograr que los resultados de combinar estas tecnologías sea mayor que la simple suma de resultados individuales, en lo que respecta a calidad del producto bajo diversos criterios de desempeño. Como resultado de este enfoque, se tiene que cada vez mayor funcionalidad de los productos, máquinas y procesos se desplaza del hardware al software como se observa en la Tabla 1[1].

TABLA 1 – PRODUCTOS EN 1960 Y 2000

	1960		2000	
Producto	Mecánica	Electrónica Informática	Mecánica	Electrónica informática
Auto	90%	10%	50%	50%
Calculadora	100%	0%	10%	90%
Cámara	100%	0%	30%	70%

Una definición ampliamente aceptada de mecatrónica dice que mecatrónica es "la aplicación de la toma de decisiones compleja para la operación de sistemas físicos" [2]. Actualmente por razones de costo y alto desempeño de los microprocesadores, la toma de decisiones se realiza esencialmente por medios computacionales en sus más variadas formas, incluyendo la computación distribuida y paralela, de tal forma que se puede decir que la mecatrónica viene a constituir un superconjunto de los conocidos *sistemas embebidos* ("embedded sytems"), más específicamente los sistemas de control embebidos [3]. De este modo la agenda de investigación y desarrollo en el área de sistemas embebidos resulta también de gran interés para la mecatrónica.

Ejemplos típicos de enfoque mecatrónico son aquellos en los que el uso de un algoritmo sofisticado de control puede compensar no linealidades y deficiencias de la parte mecánica [4], [5], lo cual conduce desde todo punto de vista a mayor eficiencia en lo que respecta al costo.

Sin embargo, la formulación de este tipo de soluciones requiere considerar diversos aspectos, como el de contar con herramientas de modelamiento y simulación adecuadas, así como de una formación especial de los diseñadores con un alto contenido interdisciplinario [1].

Este artículo se concentrará sobretodo en el primer aspecto, que a su vez tiene gran impacto en el segundo ya que contando con estas herramientas es posible formar ingenieros con las cualidades que se requieren [6].

DOI: 10.23881/idupbo.003.1-6i

2. CAD/CAE EN MECATRÓNICA

En el presente, dados los cortos tiempos de desarrollo de producto obligados por la dinámica del mercado, casi todos los procesos de diseño en diferentes ramas de la ingeniería tienen alguna forma de asistencia computarizada, lo cual no es muy distinto para sistemas mecatrónicos, donde por lo general, y en la forma como todavía se encara el diseño, la ayuda de la computadora se puede mencionar para los siguientes aspectos [7]:

- Herramientas CAD/CAE ("Computer Aided Design / Computer Aided Engineering") para el diseño mecánico, eléctrico y electrónico.
- Herramientas para la construcción de modelos dinámicos y estáticos y su respectiva simulación.
- Herramientas de traducción automática en lenguajes de bajo nivel para la simulación.
- Herramientas para la codificación e implementación del software final que se usará en el dispositivo mecatrónico.

El uso de las herramientas mencionadas todavía muestra el enfoque tradicional de diseño mecatrónico donde existe la visión de subsistemas que se desarrollan de manera concurrente. Por el contrario, si se quiere llegar a aplicar la filosofía de la mecatrónica en diseño, en el sentido estricto de la palabra, resulta de gran interés tener una herramienta que unifique estos cuatro aspectos, lo cual no se ha logrado a cabalidad y en su totalidad. La falencia presentada es evidente cuando se analizan los diversos retos y obstáculos que se interponen a la concreción de una herramienta con las características especificadas.

3. RETOS PLANTEADOS

Si se analizan con detenimiento los cuatro aspectos presentados en la sección anterior, se puede concluir que el centro de gravedad radica en tener una herramienta para el modelado y simulación que incorpore la heterogeneidad y el alto nivel de integración e interacciones que presentan los sistemas mecatrónicos.

En primer lugar, para poder simular un sistema es necesario tener un modelo y los métodos adecuados para poder "correr" el modelo en un sistema computacional. Teniendo el entorno de simulación, se puede usar para poder verificar los distintos pasos de diseño [3], incluso combinando con hardware real en configuraciones conocidas como "hardware-in-the-loop". Como se dijo anteriormente, existen algunos desafíos a considerar.

a. Naturaleza tipo Multidominio

Si se quieren modelar sistemas mecatrónicos, se tienen sistemas de diversa naturaleza interactuando unos con otros, como por ejemplo partes mecánicas, eléctricas, electrónicas, fluídicas, algoritmos de control, etc.
La forma tradicional de representar estos modelos multidominio, es a través de ecuaciones diferenciales, ecuaciones diferenciales algebraicas ("DAEs") o ecuaciones en diferencias que pueden procesarse en forma textual, o de manera gráfica. Dos alternativas gráficas de representación de las ecuaciones diferenciales constituyen los diagramas de bloques y los "bond-graphs".

Los bond-graphs presentan muchas ventajas en relación a los diagramas de bloques especialmente para poder modelar la parte física del sistema (conocida en control como "planta"), ya que éste tipo de diagramas muestra los intercambios de energía entre distintas partes del sistema, permitiendo la modularización y posibilitando así la creación de librerías de componentes reusables en diversos diseños.

Por otro lado, los bond-graphs no se han difundido plenamente y aceptado en el área de la ingeniería de control donde prevalecen los diagramas de bloques, por la facilidad que presentan en su interpretación y flexibilidad para poder modelar tanto controladores como plantas. Los modelos representados en diagramas de bloques sin embargo, tienen la grave falencia de no poder ser modularizados para su reutilización en otros diseños. Posteriormente, en este artículo se mostrará una alternativa de cómo salvar ésta dificultad.

Un punto más que se puede resaltar en ésta parte es que es muy deseable que los modelos que se requieren para poder realizar optimización, análisis de sensibilidad, o aplicación de métodos estadísticos sean de tipo "transparente", y no del tipo "caja negra" [8].

b. Incorporación de dinámica de eventos discretos bajo diversos formalismos

Dado que la complejidad de los sistemas mecatrónicos es creciente, se han creado formalismos que permiten el análisis de su dinámica en otros aspectos, como es el de los eventos discretos.

La dinámica de eventos discretos puede aplicarse a la parte física del sistema mecatrónico (por ejemplo a fenómenos de fricción), pero resulta especialmente interesante en el modelado del comportamiento del dispositivo de control, que hoy en día trabaja en gran manera de forma jerárquica [3], teniendo en el nivel más bajo controladores continuos ("controlador de lazo"), un poco más arriba controladores de secuencias que contienen estrategias de tipo lógico o

diagnósticos de fallas y, finalmente, en la capa más alta un control supervisor, juntamente con un interfaz con el usuario y muy probablemente módulos de comunicaciones. Los elementos de las capas superiores son los típicos generadores de eventos discretos controlados y pueden diseñarse bajo diversos formalismos como las máquinas de estados finitos, redes de Petri, "statecharts", GRAFCET, etc.

c. Integración del software

Poder integrar los sistemas de software en la modelación y simulación, sin duda constituye un gran salto en el desarrollo de herramientas asistidas por computador para la ingeniería de sistemas mecatrónicos. Sin embargo, si se quiere enfocar este reto desde el punto de vista de las ciencias de computación como procesamiento de datos, se debe lidiar con muchas limitaciones [9], como por ejemplo el factor tiempo, el cual prácticamente no se toma en cuenta, mientras que en sistemas de control embebido es crucial, ya que se deben analizar restricciones de tipo temporal que garanticen el funcionamiento correcto y seguro del sistema. Si se trata de usar lenguajes de modelamiento que capturen las propiedades más importantes del software (por ejemplo UML[10]), éstos no son muy adecuados para modelar la parte del "hardware" o el propio entorno donde trabajará el sistema embebido. Por otro lado, es interesante observar que uno de los diagramas definidos en UML es muy apropiado para modelar sistemas reactivos, nos referimos a los "statecharts". Statecharts (diagrama de estados) pueden ser entonces muy útiles para incorporar al software en el modelo total del sistema mecatrónico.

d. Incorporación de detalles del hardware computacional

Finalmente, un gran reto representa el poder incluir en el proceso de modelación y simulación, detalles de las restricciones de recursos computacionales, es decir de la implementación del control. En sistemas de control embebidos, los recursos computacionales se consideran heterogéneos y distribuidos con limitaciones en velocidad de procesamiento, en la exactitud de representación de los datos, memoria, etc.. Si se pudiera verificar el efecto de éstos aspectos a través de simulación, se lograría una gran reducción en el tiempo de desarrollo que es el objetivo del diseño asistido por computadora.

4. PLATAFORMA DE SIMULACIÓN DE SISTEMAS MECATRÓNICOS

A continuación se presenta una plataforma específica que sirve como herramienta para la asistencia computarizada del diseño de sistemas mecatrónicos. El entorno mencionado es ampliamente usado en la institución que patrocina este trabajo, donde se lo aplica tanto en investigación como en educación.

Evidentemente, la herramienta no llega a cubrir todos los retos y especificaciones deseadas pero constituye una alternativa de costo relativamente bajo considerando que se la utiliza de forma múltiple en diversos campos.
Matlab, con la caja de herramientas de control, y Simulink constituyen uno de los ambientes más versátiles y completos para el desarrollo de modelos, control y simulación en mecatrónica.

Por un lado, Matlab tiene grandes habilidades numéricas y de visualización, y la caja de herramientas de control incorpora una gran cantidad de cálculos rutinarios en esta área.

Para la simulación de sistemas dinámicos continuos, Matlab tiene la herramienta llamada Simulink que es un simulador de sistemas dinámicos basado esencialmente en diagramas de bloques, con la habilidad de intercambiar datos con Matlab en forma transparente. Con Simulink y Matlab ya se cubre una buena parte de las necesidades de apoyo en el diseño mecatrónico pues se pueden modelar sistemas dinámicos continuos lineales, no lineales, y de tiempo discreto, se puede realizar diseño de los sistemas de control, probar leyes de control, realizar optimización de parámetros y otras tareas necesarias en el diseño.

Si se quiere incorporar aspectos del análisis de eventos discretos del sistema, puede usarse "stateflow", que es un aditamento a Simulink basado en el formalismo de los "statecharts". Con este complemento pueden analizarse en forma sistémica comportamientos de tipo reactivos en los sistemas de control, y de este modo pueden incorporarse estructuras jerárquicas con control lógico, control supervisor, algoritmos de diagnóstico de fallas y similares. Contando con la teoría adecuada, pueden verificarse también aspectos como vivacidad, concurrencia y situaciones de bloqueo.

Finalmente, disponiendo de las herramientas "stateflow-coder", y "real-time workshop", también se puede generar código de bajo nivel en forma automática, si es que se dan las instrucciones adecuadas a la herramienta que adapta el código generado para distintos compiladores ("target language compiler"). El código generado puede ser utilizado directamente en el hardware objetivo en el cual residirá el sistema de control embebido ("target"), o en la misma máquina donde se realiza la simulación para obtener simulaciones más rápidas, o en conjunción con tarjetas de adquisición y control de alto desempeño para realizar pruebas tipo "hardware-in-the-loop".

5. MÉTODO DE MODULARIZACIÓN

Ya que se está planteando el uso de Simulink para la modelación y simulación de sistemas multidominio, surgen algunos problemas inherentes a los diagramas de bloques, que deben ser solucionados.

El proceso de modelar sistemas mecatrónicos puede tornarse tedioso debido a que los diagramas de bloques no reflejan la estructura física del sistema, en otras palabras los modelos representados no son isomórficos con el sistema real a modelar, y si se quiere tener una herramienta flexible para modelar y simular sistemas mecatrónicos, sería muy interesante poder conectar las distintas partes del sistema mecatrónico en forma modular y así reutilizar partes en diversos diseños o cambiar topologías en forma rápida y efectiva. De este modo, como primer paso fundamental en la realización de una herramienta de modelación y simulación de sistemas mecatrónicos, se plantea la creación de una librería de componentes básicos que sean frecuentemente usados en mecatrónica. Estos componentes tienen que ser fácilmente acoplados unos con otros y sin embargo deben estar diseñados bajo la representación gráfica de diagramas de bloques. Ésta tarea no es tan sencilla como aparenta, por ejemplo, cuando un motor de corriente continua de excitación independiente mueve una carga mecánica (figura 1), la inercia rotacional de la carga y la fricción se deben considerar en el diagrama de bloques del motor (figura 2). Se hace evidente la dificultad del uso de diagramas de bloques, ya que resulta ilógico cambiar un parámetro de la carga en el diagrama de bloques del motor siendo los mismos dos sistemas físicos independientes. En realidad, debería ser posible tener el diagrama de bloques del motor y de la carga en forma independiente, con la posibilidad de cambiar sus parámetros propios, de esta manera se podría estudiar el comportamiento de distintas combinaciones de motor-carga.

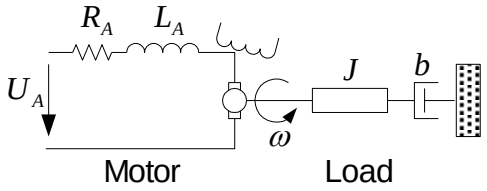

Figura 1 - Ejemplo Motor – Carga.

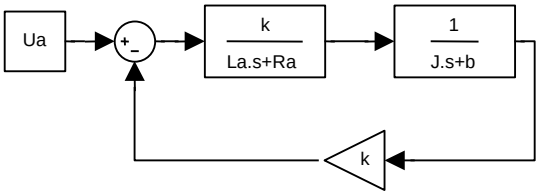

Figura 2 - Diagrama de Bloques.

El método de desacoplamiento de partes que se presentará a continuación puede ser considerado como una aplicación y extensión de la teoría de redes que usualmente se aplica para el análisis y cálculo de circuitos eléctricos lineales.

El principio de analogía dice que sistemas de diversa naturaleza tienen comportamiento análogo al de circuitos eléctricos [11]. Específicamente, el punto de partida para el desarrollo de una librería de componentes mecatrónicos constituye una parte de la teoría de redes conocida como teoría de cuadripolos o redes de dos puertos. Utilizando éste enfoque es posible descomponer un circuito eléctrico en subcircuitos o módulos [12], donde cada uno a su vez, puede estar descrito por una matriz. Existen diversos tipos de matrices que describen una red de dos puertos, la matriz usada para el desarrollo de la librería mecatrónica es la matriz híbrida *g* (la inversa de la matriz híbrida más conocida: *h*).

Figura 3 - Red de dos Puertos.

En la Figura 3 se muestra una red de dos puertos con la magnitudes más importantes: E_1, E_2, I_1, e I_2, en la manera que se usa en el desarrollo de la librería de componentes mecatrónicos. Cabe resaltar la dirección de I_2 que difiere de la usada normalmente en la literatura.

Las matrices en la teoría de redes de dos puertos establecen una relación entre dos pares de variables, uno de las cuales se toma como conocido mientras que el otro par se puede calcular.

La matriz que se usará para describir el comportamiento del cuadripolo mostrado es:

$$\begin{bmatrix} E_2 \\ I_1 \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} E_1 \\ I_2 \end{bmatrix} \tag{1}$$

Los elementos de esta matriz difieren ligeramente de los que se ven en la literatura ya que el vector de salida se lo utiliza normalmente invertido ($\begin{bmatrix} I_1 & E_2 \end{bmatrix}^T$ en lugar de $\begin{bmatrix} E_2 & I_1 \end{bmatrix}^T$).

Cuando se apliquen estas relaciones a otros dominios de energía se debe tomar en cuenta que voltajes y corrientes constituyen magnitudes conjugadas en potencia [13], y que se pueden generalizar a cualquier otro par de magnitudes conjugadas en potencia. Por ejemplo, en el dominio mecánico-rotacional las variables conjugadas en potencia son el torque y la velocidad angular. Otro punto a considerar es que los voltajes deben ser reemplazados por magnitudes tipo "esfuerzo" y las corrientes por magnitudes tipo "flujo". La tabla mostrada en la figura 4 muestra algunas magnitudes conjugadas en potencia de diversos dominios energéticos indicando además cual es la magnitud tipo "esfuerzo" y cual es la magnitud tipo "flujo".

Dominio	**Esfuerzo**	**Flujo**
Mecánico Lineal	Fuerza *F*	Velocidad *v*
Mecánico Rotacional	Torque *T*	Vel. Ang. ω
Electromagnético	Voltaje *V*	Corriente *I*
Fluídico	Presión *P*	Flujo *Q*
Térmico	Temp. *T*	Entropía *E'*

Figura 4 - Magnitudes Conjugadas en Potencia [6].

Considerando las cuatro magnitudes de la red de dos puertos, el flujo de energía queda claramente establecido: $E_1 I_1$ constituye la potencia de entrada y $E_2 I_2$ la potencia de salida.

El uso de la matriz *g* propuesta, posibilita al simulador la solución numérica de la red completa en forma simultánea, entregando hacia delante la magnitud tipo "esfuerzo" y retroalimentando hacia el componente precedente la magnitud tipo "flujo", como se puede observar en la figura 5.

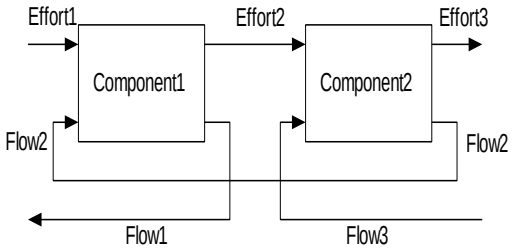

Figura 5 - Señales entre Componentes.

Inicialmente, para poder comprender el método de desacoplamiento, se pueden usar métodos de la teoría de cuadripolos para obtener los parámetros que describen a los componentes [12]. Sin embargo, después los diagramas de bloques se pueden desarrollar en forma intuitiva, usando el principio de entregar hacia delante la magnitud tipo "esfuerzo" y retroalimentar la magnitud tipo "flujo". Ésta forma de trabajo es necesaria sobre todo cuando se modelan componentes que contienen no linealidades.

El método de desacoplamiento aplicado a un motor de corriente continua con excitación independiente da la siguiente matriz:

$$\begin{bmatrix} T_L \\ I_A \end{bmatrix} = \begin{bmatrix} \dfrac{k}{R_A + sL_A} & -(b + sJ) \\ \dfrac{1}{R_A + sL_A} & -\dfrac{k}{R_A + sL_A} \end{bmatrix} \begin{bmatrix} U_A \\ \omega \end{bmatrix} \tag{2}$$

donde U_A, I_A, T_L, ω ,R_A, L_A, k, J, b son el voltaje de armadura, la corriente de armadura, el torque en la carga, la velocidad angular del eje, la resistencia de armadura, la inductancia de armadura, la constante electromagnética, la inercia rotacional del eje, y el coeficiente de fricción viscosa respectivamente. La elección de las magnitudes para la red se ha realizado de acuerdo al principio de hallar magnitudes conjugadas en potencia; específicamente, el motor recibe potencia eléctrica ($U_A I_A$) y entrega potencia mecánica ($T_L \omega_L$) a la carga mecánica. El esfuerzo de entrada U_A y el flujo de salida ω_L se toman como entradas del bloque que representa el sistema a modelar. Las señales de salida del bloque son el esfuerzo de salida T_L y el flujo de entrada I_A.

Como se dijo anteriormente puede resultar más sencillo tomar a U_A y ω como entradas para obtener T_L y I_A en las salidas como se observa en la Figura 6.

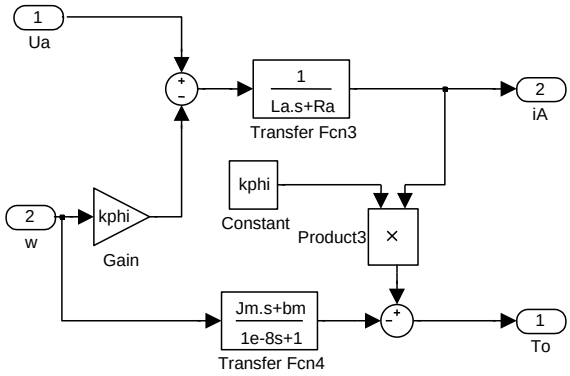

Figura 6 - Diagrama de bloques interno del motor DC.

Se debe hacer notar que en la matriz que describe al motor DC (2), hay un término de naturaleza derivativa, concretamente el término g_{22}. Este término aparece también en el diagrama de bloques. Normalmente, este tipo de expresiones se consideran no muy apropiadas para la realización de simulación numérica, por esta razón se debe agregar un polo que no afecte demasiado a la dinámica de la planta. Por otra parte, el valor del polo no puede ser ajustado a un valor muy grande, ya que el sistema tendría una naturaleza que se conoce como "stiff" en el área de simulación de sistemas dinámicos continuos, lo cual afecta al tiempo de simulación prolongándolo innecesariamente.

En todo caso, se recomienda el uso de algoritmos de resolución del tipo "stiff" para la simulación de sistemas dinámicos con componentes de la librería de componentes mecatrónicos que se pretende desarrollar.

Resumiendo, las ventajas obtenidas del método de desacoplamiento propuesto son:

- Conservación de la estructura del sistema. El diagrama del modelo reflejará la construcción física del sistema.
- Se da la posibilidad de reutilizar partes en distintos diseños o de intercambiarlas libremente, ya que en cada componente se ajustan parámetros relevantes solo para el propio subsistema.

6. ESTADO ACTUAL DE LA LIBRERÍA

En la actualidad la librería descrita en este artículo ha sido aplicada exitosamente a algunos ejemplos concretos del área mecatrónica, como ser vehículos eléctricos [14] y sistemas de propulsión de motores [8]. También se siguen desarrollando más partes para llegar en algún momento a tener una librería completa.

Por el momento, la incorporación de dinámica de eventos discretos está restringida a los "statecharts", pero se están evaluando alternativas de usar paradigmas de metamodelamiento para ampliar los lenguajes de especificación [15].

7. CONCLUSIONES

En este artículo se ha presentado brevemente la importancia de la mecatrónica y las dificultades que representa el crear herramientas que asistan en el diseño en ingeniería de este tipo de sistemas, también muestra un trabajo que se encuentra en proceso, como es el de la librería de mecatrónica para Simulink.

Como se menciona en [9] se ha abierto un campo amplio de investigación para profesionales del área de ciencias de la computación que puedan contribuir al desarrollo de teorías, creación de paradigmas de modelamiento ("multiparadigm modeling") y desarrollo de herramientas de asistencia computarizada para el desarrollo de sistemas de control embebidos que trabajan esencialmente en tiempo real y en conjunción con elementos físicos de diversa naturaleza.

8. REFERENCIAS

[1] R. Siegwart. "Grasping the interdisciplinarity of mechatronics", *IEEE Robotics and Automation Magazine*, vol. 8, No. 2, pp. 27-34, Jun. 2001.

[2] D. M. Auslander. "What is mechatronics?", *IEEE/ASME Trans. Mechatronics*, vol. 1, No. 1, Mar. 1996.

[3] Jan F. Broenink y Gerald H. Hilderink, "A structured approach to embedded control system implementation" en *Proc.Int. Conf. on Control Applications*, México, 2001.

[4] B. Ericksson. "Optimal force control to improve hydraulic drives", Tésis de licenciatura, DAMEK Research Group, Dept. of Machine Design, The Royal Institute of Technology, Estocolmo, Suecia, Mar. 1996.

[5] C. Canudas de Witt and P. Lischinsky. "Adaptive friction compensation with dynamic friction model" en *Proc. 13th IFAC Triennial World Congress*, San Francisco, CA, 1996.

[6] Takashi Kenjo, Tatsuya Kikuchi and Masatoshi Kubo."Developing educational software for mechatronics simulation", *IEEE Transactions on Education*, vol. 44, No. 2, suplemento electrónico, Mayo, 2001

[7] R. Isermann. "Modeling and design methodology for mechatronics systems", *IEEE/ASME Trans. Mechatronics*, vol. 1, pp. 16-28, Mar. 1996.

[8] M. Castillo. "A mechatronics library for Simulink" en *Proc.Int. Conf. on Control Applications*, México, 2001.

[9] E. A. Lee. "What's ahead for embedded software?", *Computer Magazine*, vol.3, No 9, pp. 18-26, Sept, 2000.

[10] UML Semantics. ver. 1.1, Rational Software Corporation, et al., September, 1997.

[11] N. S. Nisse. "Control Systems Engineering", 2nd Edition. Menlo Park, CA: Addison-Wesley, 1995.

[12] K. L. Su. "Fundamentals of circuits, electronics and signal analysis", Boston: Houghton Mifflin Co., 1978.

[13] S. Stramigioli, "Modern Control of Physical Systems", lecture notes in "Mechatronics", Delft University of Technology, 2000.

[14] M. B. Barron and William F. Powers. "The role of electronic controls for future automotive mechatronic systems", *IEEE/ASME Trans. Mechatronics*, vol. 1, No. 1, Mar. 1996.

[15] A. Ledeczi *et al*. "On metamodel composition" en *Proc.Int. Conf. on Control Applications*, México, 2001.