SOBRE LA DOCENCIA DE LA INFORMÁTICA INDUSTRIAL

Rogelio Mazaeda Echevarría, Eusebio de la Fuente López, José L. González, Eduardo J. Moya de la Torre
Depto. ISA, Escuela de Ingenieros Industriales, UVA,
rogelio@cta.uva.es; {eusfue, jossan, edumoy}@eii.uva.es

Resumen

La importancia del impacto de los computadores digitales y de la informática en la industria moderna resulta imposible de exagerar. Las soluciones informáticas resultan imprescindibles a todos los niveles de la fábrica actual: desde la planificación de la producción hasta el control digital directo de los procesos mismos. Incluso, acotando el campo de estudio a su relación con la automatización y el control de los procesos industriales y con la creación de controladores empotrados, la variedad de temas a tratar es enorme, debido, obviamente, al carácter multidisciplinar del objeto de estudio, como corresponde a una tecnología que se ha convertido en el medio técnico integrador de la industria moderna. Unido a lo anterior, se tiene la dificultad que significa la reconocida volatilidad en un campo donde las soluciones técnicas concretas se suceden con enorme rapidez. En esta contribución se comentan algunas ideas y experiencias sobre la docencia de las asignaturas relacionadas con la informática industrial en el marco de la titulación de Grado en Electrónica Industrial y Automática (GEIA) de la Escuela de Ingenieros de la Universidad de Valladolid(UVA).

Palabras Clave: educación en control, sistemas controlados por computador, redes de computadores, arquitecturas concurrentes, sistemas distribuidos de control por computadores, sistemas de tiempo real.

1 INTRODUCCIÓN

Resulta inconcebible la industria contemporánea sin considerar el papel central que juegan los ordenadores y la informática en la misma. Se puede decir que la importancia de la informática en la industria está fundamentalmente ligada al concepto de la automatización en su sentido más amplio, de sustitución, total o parcial de la labor humana, en la realización de los procesos involucrados. Pero si se analiza más en detalle, se advierte las multiplicidad de las tareas concretas en las cuales se utiliza el ordenador y que van desde la administración de los procesos de negocio, de líneas de suministros, de planificación de los recursos y de relaciones con los clientes y proveedores, en el plano de la gerencia de la empresa; hasta su uso para el control directo de los procesos productivos, utilizando dispositivos digitales especializados como los autómatas programables (PLCs) e incluso a nivel inferior, el de la instrumentación y actuadores de campo.

Existe otro ángulo de enfoque de la informática industrial y que se refiere a su empleo en la propia conformación de muchos de los productos fabricados, que pueden contener procesadores empotrados utilizados para el control de su funcionamiento.

De manera que, en resumen, los ordenadores y la informática forman parte integral de muchos productos industriales, que son diseñados utilizando software especializado y cuya producción en serie ocurre en fábricas controladas, a diferentes niveles, por computadores digitales.

La informática industrial constituye entonces el elemento técnico integrador de la industria moderna y ocupa la situación central de privilegio que en su momento ocuparon la mecánica a finales del siglo XVIII en los inicios de la revolución industrial y la electricidad a finales del XIX y durante la primera mitad del siglo XX. Esta función de integración sería, por supuesto, imposible sin la existencia de las redes de comunicaciones informáticas.

Resulta obvio, dado el alcance tan amplio de esta disciplina, que la enseñanza de la informática industrial supone un reto muy importante. La dificultad es agravada por el carácter reconocidamente dinámico de la disciplina, en la que los medios de cómputo, los lenguajes, los protocolos utilizados se modifican constantemente.

En esta contribución se ha procurado, después de describir los temas fundamentales de la disciplina de **Informática Industrial** (en la sección 2), identificar en el marco de la titulación de Grado en Electrónica Industrial y Automática (GEIA) de la Escuela de Ingenieros de la UVA, las dos asignaturas del plan de estudios que cubren los temas centrales relacionados con la disciplina que nos ocupa, y describir el contexto que brindan el

resto de las asignaturas con las que las primeras están más estrechamente vinculadas, para formular una estrategia de abordaje de los contenidos fundamentales (sección 3). Finalmente, en la sección 4, se exponen algunas consideraciones sobre las experiencias de la aplicación de algunas de estas ideas durante el curso 2014-2015.

2 LA INFORMÁTICA INDUSTRIAL EN EL CONTEXTO DE LA INGENIERÍA DE CONTROL Y LA AUTOMATIZACIÓN

En un principio, y como se ha introducido en la sección previa, el ámbito de alcance de la informática en la industria es muy extenso. De manera que lo primero que debemos hacer es acotar de manera más precisa el objeto de estudio específico. Por ejemplo, el uso de sistemas CAD/CAE/CAM durante el ciclo de desarrollo de un producto es una actividad donde el computador digital juega un papel importante, pero no lo consideraremos dentro de los temas a tratar. Resulta más apropiado abordar su estudio en las asignaturas específicas que lidien con los temas concretos que estos sistemas informáticos ayudan a diseñar. Por el mismo motivo tampoco consideraremos, al menos no directamente, la programación específica de sistemas robotizados.

El marco unificador que nos servirá para dar coherencia a los temas de la informática industrial es el provisto por la conocida pirámide CIM, en la reinterpretación que hace el estándar ISA-95 [15] (fig. 1). Además, dentro de este marco, nos interesarán específicamente, los niveles 2 e inferiores, que son los directamente relacionados con el control y supervisión de los procesos industriales.

Estos campos de aplicación tienen un rasgo definidor común: las aplicaciones informáticas, en estos estos casos, se caracterizan por ser sistemas de tipo reactivo que deben adquieren información del mundo exterior a través de sensores, procesan esa información según el algoritmo de control y supervisión apropiado y finalmente entregan la señal de control de vuelta al exterior a través de actuadores. La programación reactiva no es exclusiva de la informática industrial. La mayoría de las aplicaciones convencionales actuales se programan en torno a ese paradigma: el programa en ejecución tiene un rol más bien pasivo, a la espera del surgimiento de eventos externos que deben ser atendidos. Las diferencias, que sin embargo son notables, estriban en que las aplicaciones convencionales típicamente responden a eventos provocados por la interacción con un usuario humano. Evidentemente la respuesta ha de darse en un marco temporal razonable, pero estas restricciones están, en todo caso, ligadas a que el mencionado usuario obtenga una experiencia psicológica satisfactoria. Por el contrario en el caso de la informática industrial el tiempo de respuesta está objetivamente determinado por las exigencias de la realidad física con la que el sistema informático interactúa. El tiempo en este segundo caso, es el **tiempo real físico** y el incumplimiento de los plazos que hayan sido diseñados, pueden tener consecuencias más o menos severas en dependencia de cada caso concreto. Otras implicaciones de la realidad física incluyen, por una parte, el hecho de que los eventos externos a los que se reacciona, en general, ocurren o pueden ocurrir **simultáneamente** y por la otra, la posible existencia de la **dispersión geográfica** de los elementos físicos con los que se lidia.

También se deberá considerar como un tema de relevancia el diseño de sistemas informáticos de control empotrados.

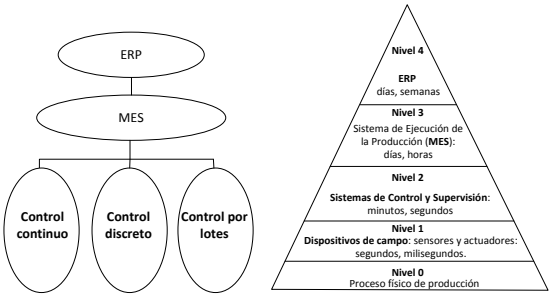

Figura 1. Pirámide propuesta por estándar ISA-95

Resulta también importante comprender los diferentes enfoques que se utilizan a la hora de "modelar" desde el programa informático la realidad física y que dependerá del propósito del sistema controlado resultante. En muchas ocasiones resulta útil considerar el sistema físico como de **tipo continuo**: las variables que se miden y que describen su estado, así como las entradas al sistema que lo modifican, se conciben evolucionando continuamente en el tiempo. En otras muchas ocasiones, sólo interesa la información sobre determinados **eventos discretos** concretos. En la pirámide **ISA-95**, se reconoce, por ejemplo, la existencia de diferentes tipos de procesos productivos: aquellos típicos de la **industria de procesos**, encargados de obtener el producto a partir de la transformación continua de las materias primas que lo conforman, donde la actividad de control y supervisión envuelve fundamentalmente el trato con variables continuas como el nivel, la temperatura, el caudal, la

concentración, entre otras. Por otra parte, se describen también, ejemplos donde la información interesa sobre todo en su variante discreta, como es el caso de las industrias de **fabricación discreta** donde el producto a obtener puede ser descrito en unidades contables: la fabricación de automóviles o de dispositivos semiconductores, por ejemplo. Aunque cada uno de estos tipos de industria privilegia uno de los dos tipos de enfoque antes descritos, la situación real es más compleja. En la industria de fabricación habrá también que controlar procesos continuos (velocidades de motores, control de posición, temperatura de algún horno de tratamiento térmico, etc) mientras que en la industria de procesos, el tratamiento de los enclavamientos de seguridad son modelados como eventos discretos: una presión o temperatura que traspase algún límite pre-establecido deberá activar una válvula de seguridad de tipo abierto/cerrado. Aunque conceptualmente pertenece a la industria de procesos, en **ISA-95** se distingue el **control *batch*** o por lotes, con características de proceso continuo pero que debe implementar una receta con cambios discretos decididos por el tiempo o por el estado del proceso.

Esta realidad diversa ha sido sistematizada para su estudio utilizando diferentes descripciones formales: los **sistemas continuos** son descritos utilizando ecuaciones diferenciales en general no lineales, y tienen a su disposición los resultados de la teoría de sistemas y de la ingeniería de control. Para el estudio de los **sistemas discretos** se tiene los formalismos desarrollados en la disciplina de la ciencia de la computación: **máquinas de estado finitas**, ***statecharts***, **redes de Petri** [16], entre otras. Los sistemas reales de cierta complejidad, como se ha adelantado, suelen ser **híbridos** que combinan la evolución continua de variables y la presencia de eventos discretos.

Por otra parte, el medio técnico de supervisión y control, el **computador digital**, como se sabe es de naturaleza discreta: la evolución de su programa avanza en instantes discretos de tiempo, definidos por un **reloj interno** que en principio no tiene ninguna relación explícita con el tiempo real físico. En este sistema dinámico, además, el estado se conserva cuantificado en su memoria interna con palabras de tamaño finito.

El control digital de sistemas continuos, implica la necesidad de efectuar el muestreo y la cuantificación de las señales continuas. Existe aquí una teoría bien desarrollada, para los sistemas muestreados, paralela a la de los sistemas dinámicos continuos, que utiliza ecuaciones en diferencias y el formalismo matemático de la transformada z, y que se apoya en resultados como el brindado por el teorema de Shannon [2]. Esta teoría bien asentada, permite ir sobre terreno seguro a la hora de diseñar controladores digitales basados en computadores, que garanticen la estabilidad y determinadas prestaciones dinámicas. Las características de este proceso de discretización impone determinadas exigencias sobre la frecuencia (medida en tiempo físico) a la que debe realizarse la interacción (y por tanto sobre la velocidad de respuesta necesaria) y también sobre la regularidad o predictibilidad del funcionamiento, para evitar fenómenos como la presencia de desviaciones sobre el verdadero carácter periódico de la interacción (*jitter*). El diseño de un sistema de control muestreado implica otras consideraciones relativas a la necesidad de utilizar filtros anti-alias y de conversores A/D y D/A. El error de cuantificación suele no ser una desventaja a considerar en los ordenadores actuales debido al tamaño de la palabra, típicamente codificada en punto flotante. En algunos sistemas empotrados, que utilizan *hardware* necesariamente más limitado y palabras de menos bits y codificadas en punto fijo, estos problemas todavía pueden requerir una consideración cuidadosa.

El diseño prevalente de los sistemas de cómputo es, por razones históricas, el especificado por la **arquitectura Von Neumann**, y este hecho decide las especificidades de la relación con los sistemas físicos con los que interactúa. La mencionada arquitectura determina a su vez el modelo de ejecución de los programas: un hilo de ejecución secuencial en la que la CPU extrae desde la memoria común tanto la próxima instrucción a ejecutar como los datos sobre los que se actúa. El problema de lidiar con un mundo físico que produce eventos *simultáneos* implica la necesidad de serializar u ordenar, de alguna manera, esa interacción. Esta serialización, puede ser acometida, sin más, por el programador, ordenando en un solo programa monolítico secuencial las tareas a realizar. Esta solución, aunque brinda al programador un grado alto de control sobre lo que sucede, resulta inflexible. La solución preferida es la de realizar aplicaciones concurrentes [4, 12], en las que diferentes **hilos de ejecución secuencial** se ocupan de diferentes aspectos a controlar del mundo físico. Este tipo de programación imita la concurrencia natural existente en la realidad y brinda por tanto un paradigma particularmente adaptado a las necesidades de estos sistemas reactivos. La desventaja de este enfoque reside en un aumento importante de la complejidad. Las tareas concurrentes no son, en general, completamente independientes, de manera que se da la necesidad de **sincronizar** su comportamiento y de un

eventual **intercambio de información** entre ellas. La programación de aplicaciones concurrentes requiere el concurso de los sistemas operativos en combinación con los lenguajes utilizados. La ejecución de las mismas es además costosa en términos de tiempo, con respecto a la solución puramente secuencial, puesto que incluye la necesidad de realizar cambios de contexto para conmutar de una tarea a otra. Conceptos relacionados con las aplicaciones concurrentes, multi-tareas como los de **proceso** y de **hilo de ejecución** *(threads*) [5], que se diferencian, sobre todo, en el modelo de memoria (individual para cada unidad de ejecución para el primero y común para todos en el caso del segundo) tienen implicaciones sobre los mecanismos de comunicación entre tareas que están disponibles en cada caso y sobre la latencia introducida por el cambio de contexto, que son relevantes para aplicaciones de informática industrial.

Se ha llamado la atención en [3, 1] sobre el hecho de que la relación entre el sistema de control informático y el proceso físico controlado tiene un carácter simétrico y que deben ser vistos como dos sistemas reactivos colocados frente a frente, interaccionando. El sistema global dinámico resultante sería interpretable como un sistema, en general, híbrido. En este sentido, resulta especialmente fructífero, por ejemplo, utilizar el mismo tipo de formalismo (e.g. redes de Petri) en la descripción del proceso físico a controlar y a la hora de dar cuenta del modelo de concurrencia que implementa su sistema de control informático.

La necesidad de cumplir los plazos relacionados con el tiempo físico constituye un reto considerable. En el caso de las aplicaciones concurrentes está tarea es responsabilidad de los llamados sistemas operativos de **tiempo real** [6, 7] y en específico de los algoritmos que utilice como **planificador** (*scheduler*) que deciden el orden y la ranura de tiempo que se debe otorgar a cada una de las tareas que compiten por el tiempo de la CPU. El estudio de los **Sistemas Operativos de Tiempo Real** (**SOTR**): su clasificación, sus características, los algoritmos de planificación y las implicaciones de su uso, son elementos fundamentales a tratar.

Se ha mencionado previamente la dispersión espacial como uno de esos elementos realmente existentes en la realidad industrial que no pueden ser eludido. Este hecho, unido a la necesidad de brindar un control coordinado a nivel de toda la planta, determina la necesidad de la utilización de **redes de computadores** [8] y la apelación a **sistemas operativos distribuidos** [10]. Ahora, a todas las complejidades ya discutidas para el caso de un solo computador, se añaden las dificultades que introduce el uso de las redes informáticas, con implicaciones claras sobre el nivel de determinismo esperable, el incremento de la latencia, y la posible pérdida de datos.

Figura 2. Informática Industrial: grandes temas

De particular importancia para los programas informáticos industriales son los temas relacionados con la **tolerancia a fallos**, la **fiabilidad** y la **seguridad** [13]. La ausencia de las garantías suficientes en estos ámbitos puede dar lugar a situaciones no deseadas con un gran coste material y humano. Los sistemas programados deben ser robustos, y estar diseñados siguiendo determinados patrones [6] y buenas prácticas que minimicen los riesgos asociados a un comportamiento defectuoso. En este sentido las ideas estandarizadas en torno al concepto de **Sistemas Instrumentados de Seguridad** (**SIL**) son de gran ayuda. El término de seguridad también se usa, en ocasiones, en el sentido informático de autenticación y control de accesos y defensa ante accesos no autorizados.

Resulta esclarecedor el considerar los temas fundamentales mencionados en la descripción previa siempre en relación con la posición que ocupa el sistema programado en la pirámide de la fig.1. La urgencia de los plazos a los que responder, por ejemplo, son órdenes de magnitud diferentes, de un nivel a otro. También pueden ser distintas las consecuencias en que se incurra al infringirlos. Para poner sólo un ejemplo, en el caso de las redes informáticas industriales, los modelos y protocolos específicos utilizados en el nivel de control, no presentan las mismas exigencias, en cuanto a volumen de datos a intercambiar, a la latencia exigible y su predictibilidad, que para aquellas redes desplegadas en el nivel de supervisión o, incluso, en una implementación del nivel **MES** encargado, pongamos por caso, de la planificación, en cada turno de trabajo, de los lotes a ser producidos en determinada instalación.

Por otra parte, y también en relación con la aplicación práctica en la industria de las soluciones informáticas, no se puede ignorar la existencia de dispositivos programables especializados como los **autómatas programables** y similares [9]. Estos

dispositivos digitales, además de presentar características constructivas diseñadas para enfrentar el agresivo ambiente industrial, poseen un modo de funcionamiento y un conjunto de primitivas de programación, diseñadas para la tarea específica del control. En su origen, eran dispositivos que se enfocaban al control de sistemas discretos de fabricación pero actualmente pueden ser usados con éxito frente a sistemas continuos e híbridos. En las versiones actuales de estos dispositivos programables (estándar IEC 61131-3), los modelos de programación utilizados y lo lenguajes han experimentado una evolución considerable. En este sentido vale la pena la comparación entre soluciones brindadas utilizando lenguajes de programación convencional (C/C++, Java) y algunos de los lenguajes propios de autómatas [11]. El uso de una u otra alternativa depende de muchos factores, pero puede decirse que lo que se pierde en flexibilidad al utilizar lenguajes menos ricos, como son los de los autómatas, se gana en fiabilidad precisamente por la misma razón. En la capas de supervisión resulta común el uso de sistemas comerciales configurables del tipo de **Sistemas de Control Distribuidos** (DCS de sus siglas en inglés) y sistemas de supervisión de tipo **SCADA** [14].

Tabla 1: Temas.

T	Descripción
1	**Contexto Industrial**: El alcance de la **II**. La pirámide **ISA-95**. Diferentes tipos de plantas industriales. Problemas discretos, continuos e híbridos. El controlador como **sistema reactivo**.
2	**Modelado Formal de Sistemas Discretos**: Redes de Petri, máquinas de estado finitas, *grafcets*…
3	**Sistemas muestreados**: teorema de Shannon, ecuaciones en diferencia, transformada z, filtros anti-alias, *jitter*.
4	**Concurrencia**: Arquitectura de ordenados y soporte del S.O y del lenguaje en relación a la concurrencia. Procesos e hilos. Modelos de concurrencia. Sincronización y comunicación entre tareas.
5	**Sistemas de Tiempo Real**: Discusión de alternativas y algoritmos de planificación de tareas. Determinismo.
6	**Tolerancia a fallos y Seguridad**
7	**Comunicaciones Industriales y Sistemas Distribuidos**
8	**Sistemas de Control y Supervisión Industriales**: Control digital directo, **autómatas programables**, sistemas **DCS** y **SCADA**. Ejemplos de aplicación.

Un resumen de los temas identificados como fundamentales, numerados para su posterior referencia, se muestra en la tabla 1 y se ilustran en la fig. 2.

3 ESTRATEGIA DOCENTE PROPUESTA

En la titulación de Grado en Electrónica Industrial y Automática (GEIA) de las Escuela de Ingenieros de la Universidad de Valladolid, la materia que nos ocupa, la informática industrial, entendida en el marco brindado en la sección anterior, estaría representada básicamente por dos asignaturas obligatorias: la asignatura de tercer año, que se llama precisamente **Informática Industrial** (**II**) y por otra asignatura de cuarto año **Control y Comunicaciones Industriales** (**CCI**), ambas de 6 créditos ECTS.

Se tienen entonces dos asignaturas, programadas convenientemente en años sucesivos, para cubrir los temas básicos identificados en la sección anterior. La estrategia que proponemos es la de considerar la primera, **II**, como introducción de la segunda (**CCI**). Esto es, **II** deberá explicar el marco conceptual general e introducir las herramientas que serán desarrolladas y profundizadas en **CCI**. Este ajuste de los contenidos de ambas asignatura deberá realizarse respetando los objetivos y competencias, tal y como vienen recogidas en las respectivas guías docentes.

3.1 RELACIÓN DE LA INFORMÁTICA INDUSTRIAL CON OTRAS ASIGNATURAS DE LA TITULACIÓN

Ahora bien, este ajuste de las dos asignaturas de **Informática Industrial** debe realizarse no sólo atendiendo a las necesidades propias y a una supuesta lógica interna divorciada del contexto, sino que deben armonizarse con el resto de las asignaturas de la titulación con las que conviven en el plan de estudio y con las que están, de una manera u otra, directamente relacionadas (fig 3). En particular se tiene una vinculación particularmente estrecha con **Fundamentos de Informática (FI)**, asignatura básica de primer año, y con **Fundamentos de Automática (FA)** y **Sistemas de Producción y Fabricación (SPF)**, estas dos últimas de segundo año, que están clasificadas como asignaturas comunes a la rama industrial. **FI** proporciona una introducción a la programación de ordenadores, utilizando como vehículo el lenguaje C, también brinda algunas nociones a los sistemas operativos de propósito general. Esta asignatura precede en el calendario a

II y naturalmente crea, en gran medida, el contexto que la segunda deberá utilizar y ampliar. La asignatura de **Fundamentos de Automática** (**FA**), por su parte, brinda como antecedentes y a nivel introductorio, los importantes conceptos de sistemas dinámicos continuos, funciones de transferencia y de control de lazo cerrado por realimentación. Por su parte, la asignatura de 2º año **Sistemas de Producción y Fabricación** (**SPF**) describe, de forma general, los diferentes elementos que integran la producción y fabricación industrial contemporánea. Debido a su objeto de estudio, es una asignatura relacionada con las que nos ocupan. Tiene un temario muy amplio que incluye la descripción de los sistemas de fabricación, de la automatización industrial y del modelado y la simulación de procesos y por tanto brinda conceptos útiles a nivel introductorio.

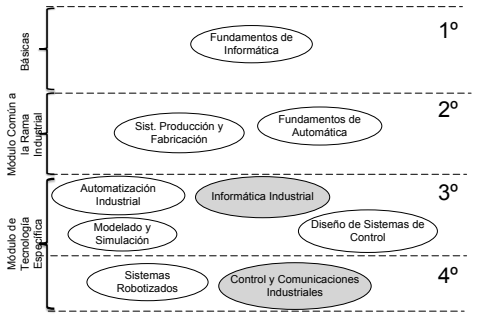

Figura 3. Programación titulación GEIA

La asignatura **Automatización Industrial** (**AI**) tiene una orientación de tipo más práctico hacia la disciplina de la automatización. El énfasis viene dado en el diseño de sistemas de control de tipo secuencial y combinatorio, en la programación de autómatas mediante algunos lenguajes descritos en el estándar IEC 61131, en el diseño de automatismos con componentes neumáticos y eléctricos, entre otros tópicos. Es una asignatura obligatoria que se impartiría de forma simultánea (en el mismo cuatrimestre) que **II** y por tanto, si la interacción resulta bien manejada, se podría dar un enfoque complementario bien balanceado. La asignatura de **Diseño de Sistemas de Control** (**DSC**) se imparte en el segundo cuatrimestre del tercer año, una vez concluida **II** pero en un curso previo al comienzo de **CCI**. En **DCS**, asignatura que pudiera concebirse como la continuación de **FA**, se estudia con profundidad teórica, los sistemas muestreados y el formalismo de la transformada z, además de los sistemas dinámicos en el espacio de estados. La segunda asignatura de **Informática Industrial**, como consideramos a **CCI**, si puede aprovecharse del terreno abonado por **DCS** para estudiar, desde el punto de vista de la implementación informática, los controladores digitales, asumiendo ya en el alumno un conocimiento teórico adecuado de las bases matemáticas subyacentes.

El rol central que tiene la informática en la industria moderna, hace que las asignaturas de este perfil tengan relación, aunque menos directa, con otras muchas otras del grado, como por ejemplo las de **Modelado y Simulación (MS)**, **Sistemas Robotizados (SR)** y de **Visión Artificial (VA)**, entre otras.

3.2 TRATAMIENTO DE LOS CONTENIDOS FUNDAMENTALES

La docencia de la **informática industrial** debe entonces ser capaz de encontrar compromisos razonables a lo largo de, al menos, dos ejes. El primer eje daría cuenta del balance necesario, entre la volatilidad de la tecnología informática y la necesidad de impartir, en el ámbito universitario, conocimiento perdurable. Mientras que el segundo reflejaría la tensión existente entre el alcance tan amplio de la informática industrial y el grado de profundidad y de atención a los detalles que una disciplina de esta índole (programación de ordenadores) requiere.

Un primer paso, ha sido el identificar, por una parte, los grandes temas a tratar (tabla 1) y por otra, las asignatura relacionadas con estos temas en el marco de la titulación, como se ha hecho en las secciones previas. A partir de este análisis, y respetando la idea de que, aun conservando las independencia de los contenidos respectivos, la asignatura de **II** debe considerarse como introductoria a la **CCI**, se propone el reparto de temas mostrado en la tabla 2. Nótese que la tabla más bien refleja el énfasis que cada uno de los temas recibirá en cada caso, y no el hecho de que los mismos sean patrimonio exclusivo de ninguna de ellas. Se debe en todo momento mantener la unidad conceptual de toda la disciplina de manera que el alumno sea capaz de situarse rápidamente dentro del temario común. En este sentido el anclaje a la realidad industrial que implica la referencia a la pirámide **ISA-95** sería de gran ayuda. Hecha esta salvedad, es cierto que en la primera asignatura (**II**) el énfasis se hará en la programación concurrente con un lenguaje de alto nivel y en el control de procesos discretos, brindando una introducción más ilustrativa que formal, de las herramientas de modelado pertinentes (e.g: máquinas de estados finitas, redes de Petri). Por su parte, la asignatura de **CCI** construiría, sobre la base creada por la anterior, para hacer énfasis en los **sistemas de tiempo real** y las **comunicaciones industriales**.

En sintonía con la discusión previa, las prácticas de laboratorio de **II** se limitarían al trabajo en el aula de informática, mientras que en **CCI**, la interacción de **sistemas en tiempo real** para el control maquetas físicas, sería una prioridad. La programación de **sistemas empotrados** también podría ser abordada en este contexto.

Por otra parte, resultaría interesante que las aplicaciones mostradas como ejemplo, en prácticas de laboratorio y proyectos, no fueran demasiado triviales y si tuvieran un contenido ligado a las actividades de control y supervisión. Para ello, sería conveniente, que el alumno pudiera situarse, en lo que a las habilidades de programación informática se refiere, a un nivel de abstracción mayor que el que provee la formación básica en lenguaje C dada por la asignatura **FI** de primer año. En este sentido, se ha considerado conveniente, durante las primeras semanas de prácticas frente al ordenador, brindar elementos básicos de C++ partiendo del conocimiento previo en C. De esta forma se abriría, por ejemplo, la posibilidad de utilizar clases contenedoras como vectores, conjuntos, mapas, entre otras de la biblioteca estándar del lenguaje (STL), y no perder tiempo de diseño en la implementación de estas estructuras. Por otra parte, el introducir conceptos de orientación a objetos (O.O) puede ser puesto en valor al estudiar otros lenguajes. Por ejemplo, el concepto de **bloque funcional**, definido en los estándares IEC 61131 (autómatas) [9, 11] y IEC 61499 (sistemas distribuidos) [17], puede ser, de alguna manera, asimilable al concepto de clases, tal y como se define en la O.O. tradicional. Claro que las ventajas de trabajar a mayor nivel de abstracción y más alejado del *hardware* tiene un coste en términos de la velocidad de ejecución, que debe ser sopesado en cada caso concreto. Su uso en soluciones empotradas, por ejemplo, podría resultar cuestionable. Por este, motivo, y teniendo en cuenta que la informática industrial está muy ligada, entre otros temas, a la necesidad de brindar reacciones con poca latencia a los estímulos del mundo externo, se ha decidido brindar también una introducción no formal al tema de la **eficiencia computacional** y la notación de *o-grande* (*O(n)*), contenido que se encontraba, por otra parte, completamente ausente del temario de la titulación.

Otra forma de otorgar un mayor valor de permanencia a los temas estudiados, es el apelar a estándares tanto sancionados por organismos internacionales como *de facto*. En este sentido, a parte de los ya mencionados (**ISA-95**, **IEC 61131**, **IEC 61499**) resultan importantes aquellos que estandarizan los propios lenguajes utilizados (C/C++), así como el conjunto de estándares **POSIX** (IEEE) [4] que garantizan la portabilidad de soluciones de **programación concurrente** y de **tiempo real** a través de los diferentes S.O. Los temas de seguridad instrumental vienen tratados en los estándares **IEC 61508** y **61511**.

Pero los estándares oficiales son documentos complejos, que deben prever muchos casos especiales, no todos con la misma relevancia. Es por ese motivo que se ha realizado un esfuerzo especial en el caso de la **API** (interfaz de programa de aplicación) de **POSIX** para aplicaciones concurrentes, con el objetivo de aliviar la curva de aprendizaje, brindado una versión reducida de la misma, suficiente a los propósitos planteados en el curso.

Tabla 2: Reparto de temas por asignatura.

A.	Temas enfatizados
II	**Tema introductorios** (1). El tema 4, (concurrencia) será el tema central de la asignatura. Elementos del tema 6 (tolerancia a fallos), y 2 (**Modelado Formal**) serán tratados. El tema 8 en relación al control de sistemas de fabricación y ejemplos simples del tipo del como control **on-off** de sistemas continuos. Las prácticas de laboratorio en aulas de informática.
CCI	**Temas centrales**: Serán el 5 (**STR**), y el 7 (**Comm. Ind.**). Se profundizará en el Tema 3 (**Sistemas muestreados**) y en el Tema 8 con controladores para procesos continuos. Las prácticas de laboratorio incluirán además el control de maquetas físicas.

4 RESULTADOS PRELIMINARES

En el curso 2014-2015 se han ido introduciendo progresivamente las ideas comentadas en las secciones previas en el curso de II de 3º año de GIEA. Los resultados, expresados a partir de las calificaciones, se detallan en la tabla 3 en comparación con los resultados del año previo. Sin pretender que los mismos sean estadísticamente significativos, o atribuirlos exclusivamente a la puesta en práctica de las mejoras aquí descritas, si se puede constatar un aumento significativo del porcentaje de aprobados y una mejora sustancial de la calidad de las calificaciones durante el curso que acaba de concluir.

Encuestas realizadas han mostrado la satisfacción de los estudiantes con respecto a algunos de los

temas más polémicos, como la introducción al estudio de elementos de programación orientada a objetos con C++.

Tabla 3: Resultados curso 2014-2015.

Calificaciones Informática Industrial GIEIA Convocatoria Ordinaria				
Calificaciones	**2013-14**		**2014-15**	
	#	**%**	**#**	**%**
NO PRESENT.	3	8,33	4	9,30
SUSPENSO	8	22,22	0	0,00
APROBADO	16	44,44	10	23,26
NOTABLE	6	16,67	25	58,14
SB.	2	5,56	2	4,65
MAT. HONOR	1	2,78	1	2,33
SIN CALIF.	0	0	1	2,33

Agradecimientos

La contribución aquí descrita ha sido desarrollada en el marco del Proyecto de Innovación Docente (PID 14-15 Nº 39) de la Universidad de Valladolid titulado "Armonización de las asignaturas relacionadas con la informática industrial en el Grado de Electrónica Industrial y Automática"

Los autores también desean agradecer la ayuda brindada por el MINECO a través de su programa DPI2012-31859.

Referencias

[1] Åström, K. J., Kumar, P. R. (2014) "Control: a perspective", *Automatica*, pp. 3-43.

[2] Åström, K. J., Wittenmark, B. (1996) Computer Controlled Systems: Theory and Design. Prentice Hall.

[3] Benveniste, A., Åström, K. J., 1993. Meeting the challenge of Computer Science in the industrial application of control: an introductory discussion of the special issue. Automatica, 1169–1175.

[4] Burns, A., Wellings, A., (2001) Real Time Systems and Programming Languages: ADA 95, Real Time Java and C/POSIX. Addison Wesley.

[5] Breshears, C., (2009) The art of concurrency: A thread monkey's guide to writing parallel applications. O'Reilly.

[6] Douglass, B.P. (2002) Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems. Addison Wesley.

[7] Liu, J.W.S. (2000) Real-Time Systems. Prentice Hall.

[8] Neumann, P. (2006) "Communication in Industrial Automation: What is going on?" *Control Engineering Practice*. pp: 1332-1347.

[9] Öhman, M., Johanson, S., Årzén, K.E. (1998) "Implementation Aspects of the PLC standard IEC 61131-3", *Control Engineering Practice*. pp: 547-555.

[10] Pereira, C.E., Carro, L. (2007) "Distributed Real-Time Embedded Systems: Recent Advances, Future Trends and their Impact of Manufacturing Plant Control", *Annual Reviews of Control*. pp: 81-92.

[11] Plaza, I., Medrano, C., Blesa, A. (2006) "Analysis and Implementation of the IEC 61131-3 software model under POSIX Real-Time Operating Systems", *Microprocessors and Microsystems*. pp: 497-508.

[12] Raynal, M., (2013) Concurrent Programming: Algorithms, Principles, and Foundations, Springer.

[13] Radvanovsky, R., Brodsky, J. (2013) Handbook of SCADA/Control systems security, CRC Press.

[14] Rodríguez, A., (2007) Sistemas SCADA. Marcombo.

[15] Scholten, B., (2007) The road to integration: the ISA 95 standard in manufacturing. Marcombo.

[16] Silva, M., (2013), Half a century of C.A. Petri PhD Thesis. A perspective of the field. *Annual Reviews of Control*, pp. 191-219.

[17] Viatkyn, V. (2012) IEEE 61499 functions blocks for embedded and distributed control systems design. ISA.