Comunicaciones Eléctricas

VOLUMEN 60 N° 3/4 1986

REVISTA TECNICA DE ITT

Volumen 60
Número 3/4 · 1986

Comunicaciones Eléctricas

Edición española de ELECTRICAL COMMUNICATION
Revista técnica publicada trimestralmente por ITT Corporation

Comunicaciones Eléctricas presenta las investigaciones, los desarrollos y las realizaciones conseguidas por ITT y sus compañías asociadas.

Publicada desde 1922 en versión inglesa, se edita actualmente en cuatro idiomas y se distribuye en el mundo entero.

Se invita a los ingenieros de ITT a proponer proyectos de artículos, cuyos resúmenes deben enviarse al editor internacional para su consideración.

Publicado en febrero de 1987

Las direcciones de los editores se dan en la página 322

Sistemas soporte

194 **Presentación**

196 **Metodología y herramientas CAD para el diseño de circuitos VLSI a medida**
J. M. Danneels, P.-P. Guebels y M. C. Rahier

207 **Entorno soporte para desarrollos VLSI parcialmente a medida**
B. Prazic

216 **Galileo: modelo, lenguaje y herramientas**
C. Sánchez Moreno

225 **Sistema de información de componentes de ITT**
J. J. F. Cunningham

232 **Desarrollo de programación para el Sistema 12**
D. R. Illi

239 **Sistema integrado de diseño de placas impresas**
A. Page y S. Grützmann

248 **Soporte de productos basados en microprocesadores durante su vida útil**
T. Haque y J. Montes

256 **Módulo interfaz de medidas para el entorno de desarrollo de programas**
D. Hunter y W. Kobitzsch

259 **Generación de sistemas ejecutivos en tiempo real**
C. S. Baradello y G. Carloni

266 **Entorno distribuido de pruebas de programación**
W. Wellens

270 **Desarrollo y mantenimiento de programas de aplicación**
J. T. Crawford

278 **Herramientas UNIX para el desarrollo de sistemas de terminales**
A. Povelones

286 **Aportación de los factores humanos al desarrollo de productos**
F. R. Brigham

294 **Diseño de interfaces usuario-sistema mediante ayuda cognoscitiva**
P. F. Byerley, R. G. Leiser y R. F. Saffin

303 **Sistema vídeo de análisis asistido por ordenador**
M. de Alberdi

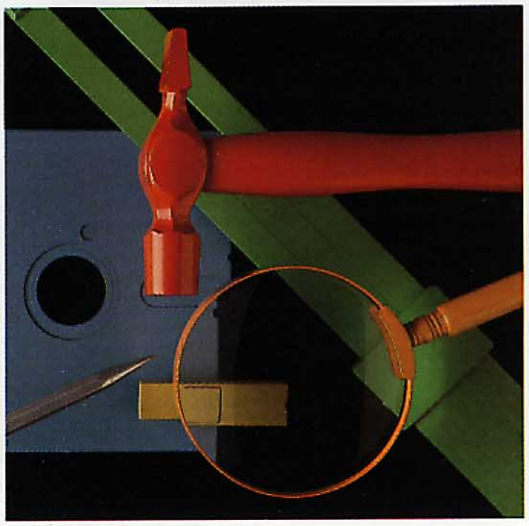
308 **Análisis de planificación, modelado y fijación de precios**
F. Giambalvo y D. Vignazia

314 **Ayudas de ordenador para el diseño mecánico en la industria electrónica**
W. Drtil y G. Klause

320 **En este número**

Comunicaciones Eléctricas, Director en Español: A. Soto, Ramírez de Prado, 5
28045 Madrid (España)

ITT



A lo largo de la historia, el hombre ha elaborado y utilizado herramientas para extender sus facultades a todos los dominios vitales. Sin una colección de refinadas herramientas sería hoy imposible desarrollar equipos de telecomunicación con las técnicas más recientes para la RDSI y la RDSI-BA. ITT ha dedicado considerables recursos a crear una infraestructura multinacional de desarrollo, basada en herramientas que se alinean en la vanguardia tecnológica.

Presentación

El desarrollo de productos es una tarea cada vez más exigente, que requiere la aplicación de tecnologías avanzadas para conseguir equipos de altas prestaciones y elevada calidad, aptos para satisfacer una extensa gama de requisitos de los usuarios. Esto hay que lograrlo en márgenes de tiempo más estrechos que nunca, y dentro de un contexto donde escasea el conocimiento cualificado. Sin medios adecuados es difícil hoy el diseño, desarrollo, producción y prueba de equipos electrónicos de alguna complejidad; con toda certeza, un fructífero desarrollo de productos de telecomunicación para la RDSI y la RDSI de banda ancha de la siguiente generación exige disponer de un completo conjunto de herramientas integradas que cubran todas las etapas, desde la especificación inicial hasta la fabricación y pruebas, e inclusive la futura evolución del producto.

ITT ha reconocido siempre la importancia de disponer de sistemas soporte adecuados, con fuertes inversiones en adquisición de las mejores herramientas y elaboración de unas nuevas cuando aquéllas no existen, integrando unas y otras en una infraestructura viable para el desarrollo de productos en el plano multinacional. La inversión requerida es cuantiosa, pues en los productos de vanguardia tecnológica los costes de I+D+I se aproximan al 20% de las ventas, y hay que dedicar cerca de la décima parte de esto a obtener, instalar y mejorar las herramientas soporte. Compensan, sin embargo, esta inversión los menores costes y mayor brevedad del desarrollo, la mayor calidad del producto, y la facultad de transferir tecnología entre casas ITT, o de éstas a sus concesionarias. La importancia de los sistemas soporte en ITT se refleja en el parque corporativo de más de 500 ordenadores principales, 17.000 terminales y 2.500 ordenadores personales, que en elevada proporción forman parte de un entorno integrado de diseño y fabricación.

Este número doble de *Comunicaciones Eléctricas* describe muchas de las herramientas soporte utilizadas a diario en ITT, que prolongan las facultades visuales, manuales y mentales, y permiten conseguir productos de funcionalidad y calidad siempre en aumento. Así, por ejemplo, el diseño a medida en VLSI – esencial para la producción de equipos de RDSI – sería impracticable sin herramientas avanzadas de diseño asistido por ordenador y de producción. Análogamente, sólo puede concebirse la programación del Sistema 12 si existe un entorno de desarrollo en el ámbito de ITT que preste soporte a todos los aspectos del diseño de programas, desde la especificación inicial hasta la producción del paquete lógico de una central concreta.

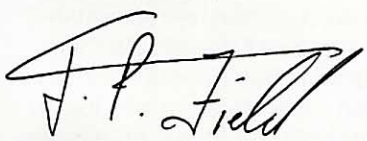
El compromiso de ITT por establecer la infraestructura necesaria fue una poderosa razón para crear en las principales compañías centros paneuropeos como soporte al desarrollo, coordinados por el Engineering Support Centre de Harlow, en Inglaterra. Tales centros se dedican a investigar nuevas herramientas, coordinar los desarrollos de las necesarias para fines concretos, e integrar dichas herramientas en toda la ITT. Esta orientación se refleja en dos de estos programas, el IPDS (diseño integrado de placas impresas) y el ISDS (diseño parcialmente a medida de ITT). El IPDS tiene acceso al sistema de información de componentes (ICIS) de ITT, que aporta información actualizada sobre los componentes autorizados. Tanto el IPDS como el ISDS proporcionan la captación electrónica de diseños y se ejecutan en una misma estación de trabajo, con la ventaja de que no es preciso elegir entre implantar el circuito como LSI parcialmente a medida o como placa impresa hasta no haber captado el diseño. Una vez tomada esta decisión, los ficheros de salida se pueden transferir a un proveedor autorizado de tales LSI, o bien al proceso de trazado de placas impresas, y gracias a ello un circuito realizado en placa puede más tarde implantarse fácilmente como LSI parcialmente a medida, si la superior demanda así lo exige. Esta flexibilidad y la integración de herramientas como los IPDS, ISDS e ICIS se inscribe en la filosofía de los sistemas soporte de ITT. El objetivo final es poder integrar todas las etapas, desde la fase de diseño inicial hasta la fabricación y las pruebas, con una transferencia automática de información de diseño entre dichas etapas, de modo que se suprima toda reinserción y reorganización de los datos.

Las herramientas soporte son tan importantes en el desarrollo de programación como en el de equipo físico, por lo cual ITT ha establecido el SDE (entorno de desarrollo de programación) en Europa y en los Estados Unidos, con el fin de integrar las herramientas de programación existentes. En el núcleo del SDE, un sistema de control de configuración da pleno soporte al desarrollo de productos con microprocesadores a lo largo de su ciclo de vida. Los usuarios con terminales multifunción pueden acceder a los dos principales centros de cálculo de ITT a través de su red Intelnet o de la red pública de datos por conmutación de paquetes.

En la elección y preparación de sistemas soporte, adviértase que una compañía tan diversa como ITT acomete proyectos grandes y pequeños, cada uno con características propias, y que sería claramente rechazable cargar sobre un desarrollo a pequeña escala todos los artificiosos controles que exigen los grandes proyectos, como el del Sistema 12. Inversamente, no puede aceptarse que el desarrollo de sistemas importantes quede limitado por el uso de herramientas concebidas para proyectos de relativa sencillez. En consecuencia, y para los casos que convenga, se han diseñado herramientas que permiten al usuario seleccionar el nivel de control adecuado a la magnitud del desarrollo del producto.

Los sistemas soporte no se limitan a los aspectos primarios del diseño y la fabricación, como expone este número de *Comunicaciones Eléctricas*. El influjo de los factores humanos en el interfaz usuario-sistema, la generación automatizada de programas y el análisis de procesos concurrentes no son sino unos cuantos aspectos donde se utilizan sistemas informatizados de menor alcance en ayuda de los desarrollos.

Gran parte de la fuerza de los sistemas soporte de ITT radica en su capacidad de interacción, creando un todo mayor que la suma de sus partes, tanto a nivel de cada compañía como al internacional. El éxito de los productos de vanguardia depende acusadamente de la tecnología de los sistemas soporte empleados en su desarrollo. ITT se ha esforzado mucho por garantizar que las herramientas de hoy sean enteramente adecuadas para soportar el desarrollo de los productos del mañana.



J. P. Field
Director Técnico General Adjunto y
Director Técnico de Telecomunicaciones
ITT Europe Inc, Bruselas

Metodología y herramientas CAD para el diseño de circuitos VLSI a medida

La creciente complejidad de los circuitos VLSI obliga a mejorar las prestaciones de las herramientas de diseño actuales. Las nuevas herramientas tendrán que basarse en otros modelos de datos y en sistemas jerárquicos de diseño.

J. M. Danneels
P.-P. Guebels
M. C. Rahier
Bell Telephone Manufacturing Company, Amberes, Bélgica

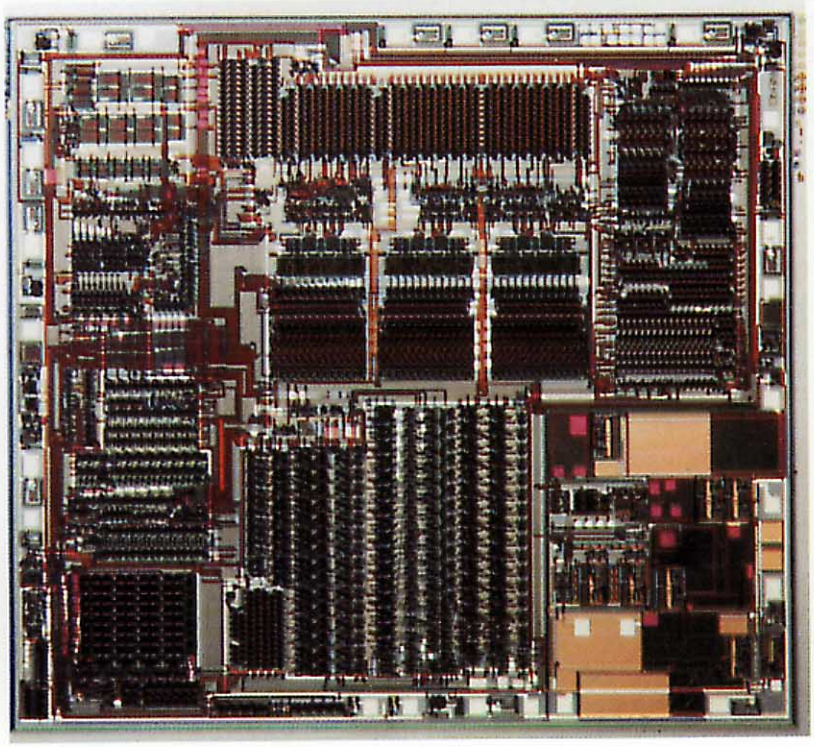
Introducción

Los circuitos LSI y VLSI reducen el tamaño, peso, consumo y número de los componentes de los equipos, a la vez que aumentan la fiabilidad y facilidad de mantenimiento. Como resultado de estas ventajas, se usan microcircuitos en casi todos los equipos electrónicos modernos.

En la actualidad, la mayoría de los diseños VLSI son de tecnología CMOS, adecuada para la producción de circuitos con las funciones y densidad de componentes requeridas en los equipos de la próxima RDSI. Actualmente se producen circuitos de hasta 100.000 transistores por pastilla. Un ejemplo es el interfaz U emisor/receptor del módulo de abonado RDSI de la central digital Sistema 12 de ITT[1]. Próximamente se espera integrar en una sola pastilla un elemento de conmutación Sistema 12 completo, con 16 puertos de conmutación (200.000 transistores), o un interfaz terminal entero.

La experiencia ha demostrado que no pueden diseñarse circuitos VLSI complejos sin herramientas CAD (diseño asistido por ordenador). El éxito alcanzado con el diseño de los circuitos VLSI del Sistema 12[2] demostró la idoneidad de las herramientas actuales para circuitos de unos 25.000 transistores. Sin embargo, la situación está cambiando en una doble vertiente: por una parte, los equipos de telecomunicación del próximo entorno RDSI requerirán circuitos VLSI un orden de magnitud más complejos, y por otra, será necesario acortar el tiempo de los diseños VLSI (aumentando los transistores producidos por día y por diseñador) dada la rapidez del progreso tecnológico.

Para diseñar los circuitos VLSI del próximo futuro, se necesitarán herramientas CAD bastante diferentes de las actuales. Las nuevas herramientas deberán insistir en la eliminación de errores durante los desarrollos y propiciar métodos jerárquicos de diseño y el uso de técnicas de abstracción coherentes. En muchos casos, las nuevas herramientas utilizarán procesadores múltiples para mejorar las prestaciones mediante procesamientos en paralelo, y también técnicas heurísticas de programación, como los sistemas expertos basados en reglas, para mejorar su flexibilidad y simplicidad, y acelerar el perfeccionamiento de las herramientas. Este nuevo entorno de soporte CAD se beneficiará del coste decreciente de la potencia de cálculo. Los



Codec digital con 22.000 transistores. Realizado con tecnología CMOS de 3 µm en una pastilla de 5,5 x 5,3 mm.

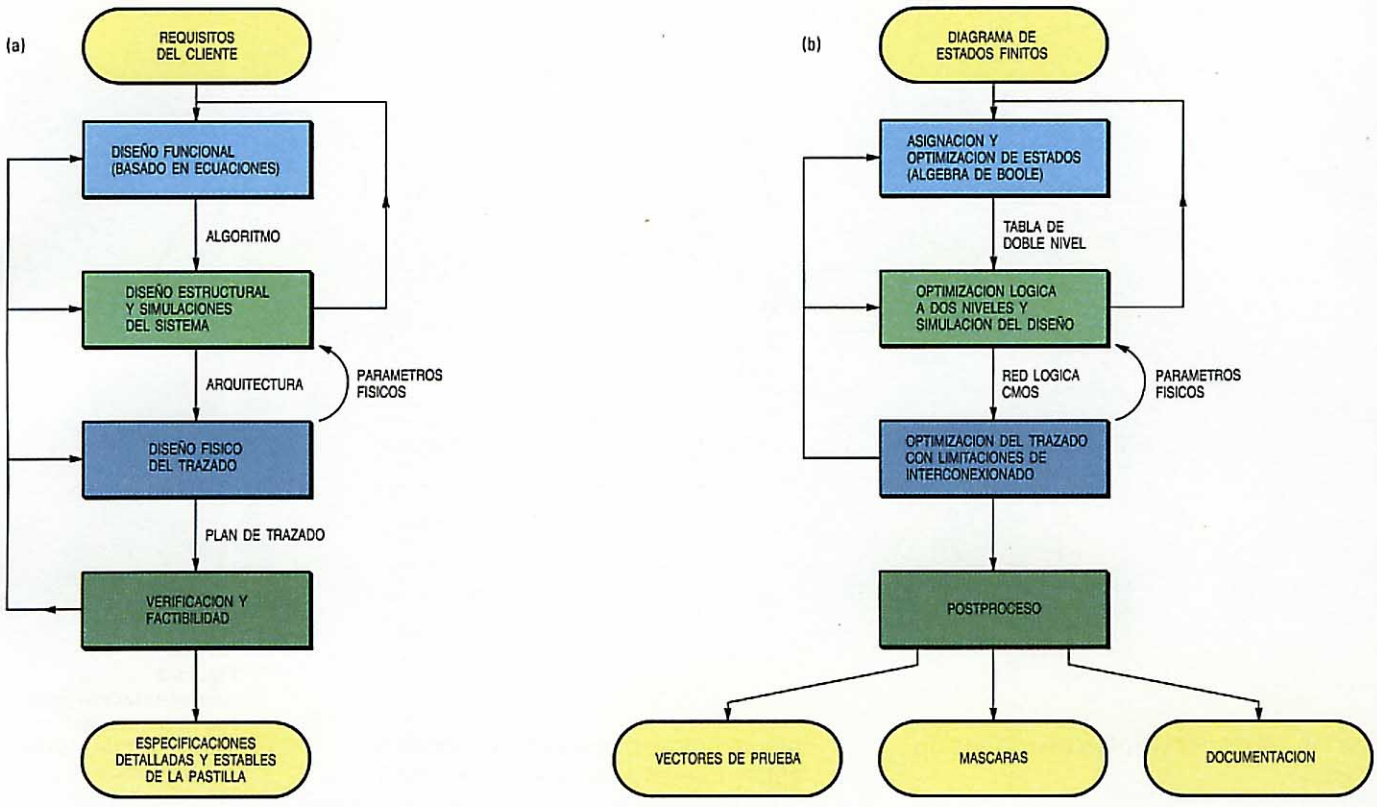


Figura 1
Diagrama ilustrativo de la metodología de diseño VLSI a medida: (a) diseño de un sistema VLSI, y (b) diseño de una pastilla VLSI.

ingenieros de CAD participarán en el diseño de los sistemas desde el principio, con la responsabilidad de garantizar el funcionamiento correcto desde la primera pastilla que se produzca.

Estos son los requisitos que se han tomado como punto de partida para el IVDS, nuevo sistema de diseño VLSI de ITT actualmente en desarrollo.

Metodología para el diseño de circuitos VLSI a medida

La metodología de los diseños VLSI se comprende mejor con la ayuda de un diagrama como el de la figura 1, que considera tres dominios de conocimiento y representación: funcional, estructural y físico, cuyos significados se ilustran a lo largo del artículo.

En la práctica, el proceso de diseño de una pastilla de unos 25.000 transistores, consta de una fase de configuración del sistema y de otra de diseño detallado. En la primera se determinan las decisiones estratégicas, mientras que en la segunda se hace la parte lenta y laboriosa del trabajo. La conversión de los datos lleva el diseño a su forma física final.

Diseño del sistema VLSI

Se comienza por definir en lenguaje normal las funciones de la pastilla, lo cual determina qué ha de hacer y cómo ha de comunicarse con el resto del sistema. En esta etapa todavía no se dice nada acerca del modo de materializar las funciones en circuitos: las señales y los datos de entrada se procesan mediante algoritmos y se transforman en salidas como valores de las funciones.

A continuación, ha de obtenerse la arquitectura del sistema como una realización factible del algoritmo, teniendo en cuenta limitaciones tales como temporización global, cadencia de presentación de datos o velocidad interna. En esta fase se toman ya decisiones importantes, como la adopción de una ordenación sistólica o de arquitecturas de proceso digital de señal en paralelo.

Por último, se concreta un plan de trazado que tenga en cuenta las limitaciones principales de espacio, como la longitud de los buses o de las líneas de reloj y el área total de la pastilla.

En el informe del sistema se incluyen también las especificaciones funcionales y el diagrama de bloques. Mediante un análisis aproximado hay que comprobar si los requisitos de partida pueden alcanzarse de un modo práctico y económico. El estudio de factibilidad tiene que basarse en experiencias anteriores para predecir las prestaciones finales de la pastilla y la duración de la actividad.

El diseño propiamente dicho ha de comenzar una vez aceptados los resultados de la fase de definición del sistema a través

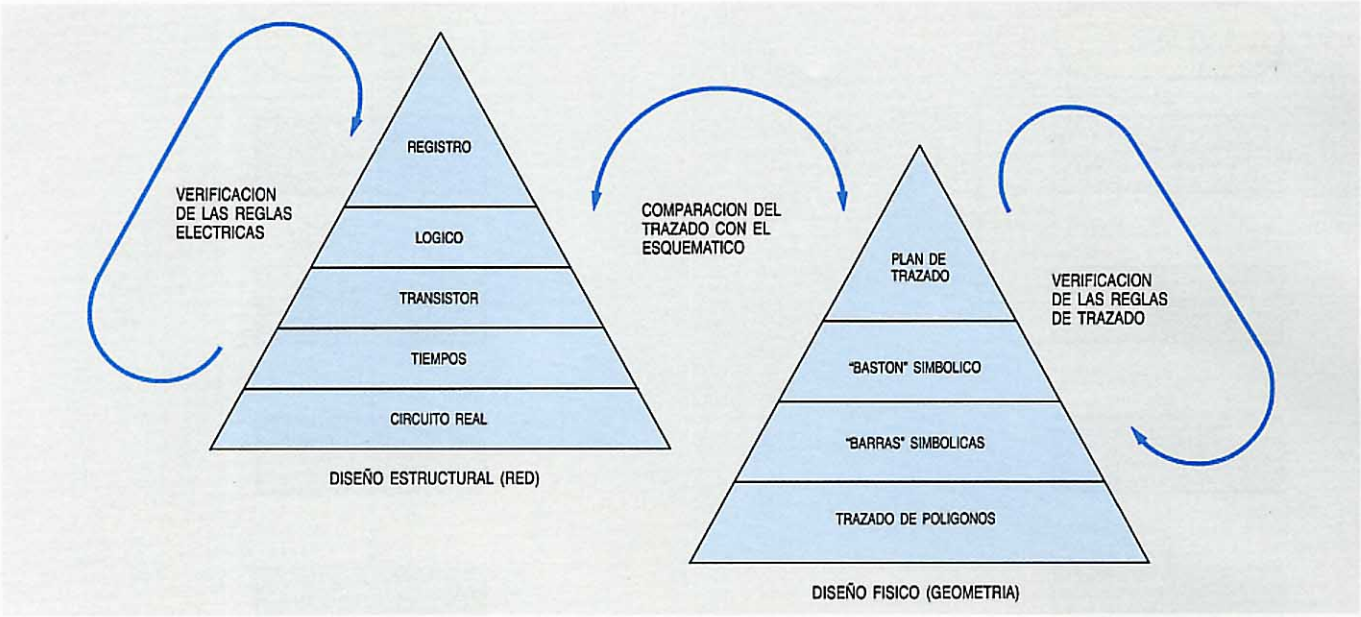


Figura 2
Representación práctica resumida del método ITT de diseño VLSI a medida.

de las correspondientes revisiones de diseño.

Diseño de la pastilla VLSI

El diseño de la pastilla es un proceso complejo con multitud de pasos combinados de síntesis y verificación. La figura 1 (b) muestra el diagrama de flujo del diseño de una PLA (disposición de lógica programable); se usa el simbolismo del álgebra de Boole para resumir las especificaciones en un pequeño número de funciones y secuencias booleanas. La realización se hace con redes lógicas CMOS que pueden tener una estructura estática o dinámica. Finalmente el trazado de la PLA se dibuja con una geometría compacta de filas y columnas.

En la actualidad, el diseño de las pastillas VLSI consta del diseño estructural, que construye mediante elementos de circuito e interconexiones una red eléctrica VLSI, y el físico, que traslada dicha red a un juego de máscaras. Estos dominios, estructural y físico, se muestran en la figura 2. Los triángulos se usan para simbolizar el aumento de los datos en las representaciones más detalladas de la pastilla. El desplazamiento vertical de los triángulos indica que el trazado comienza algo después que la simulación del circuito, aunque en teoría ambos procesos pueden ser simultáneos.

Las redes VLSI están formadas por un número finito de resistencias, condensadores, transistores y diodos MOS, y generadores de tensión o de corriente. Estos elementos o primitivas, son idealizaciones de dispositivos físicos reales y los valores de la corriente, tensión o energía han de seguir las leyes de la física de los semiconductores, mientras que sus interconexiones están sujetas a las leyes de Kirchoff.

La complejidad de las redes VLSI hace necesarias estas idealizaciones (de la física a los circuitos) en el diseño estructural, que dan lugar a los diferentes niveles indicados en la figura 2. Las etapas de idealización reducen los datos VLSI de forma que pueda abarcarlos la mente humana. La precisión se ha de conjugar con la rapidez de análisis, para lo cual se han de utilizar nuevos diagramas de representación de la red, como los de tiempos, transistores, lógica, y de registros. La representación de una pastilla incluye todos los datos asociados con un programa específico de simulación. La salida del simulador expone el funcionamiento de la pastilla como un conjunto de formas de onda.

La interpretación de los dispositivos MOS como una estructura resistiva que permite un flujo bidireccional de señales lógicas entre su drenador y fuente (cuando conduce), o un circuito abierto (cuando no

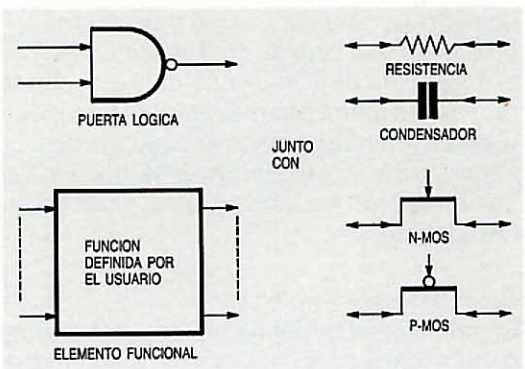


Figura 3
Primitivas básicas VLSI a nivel transistor.

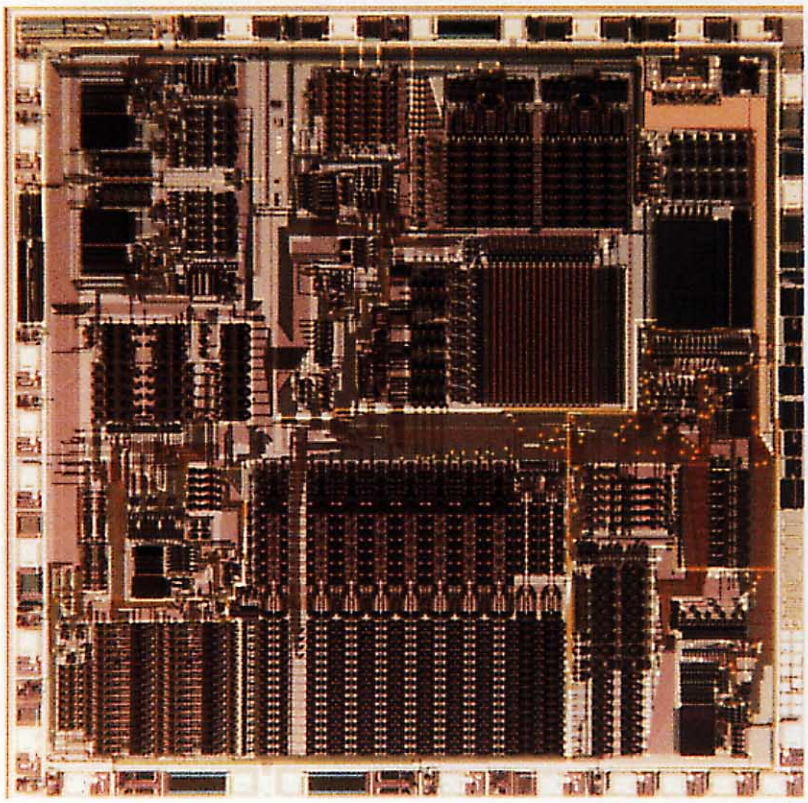
conduce), es fundamental para la metodología de diseño de los circuitos VLSI a medida. Esta interpretación del funcionamiento de un circuito MOS como conmutador es la característica esencial en la representación a nivel transistor de la pastilla. En la figura 3 se indican las primitivas básicas de un simulador al nivel transistor. A ese nivel la red consiste simplemente en un conjunto de nodos y de transistores. Al igual que los condensadores en una red eléctrica, los nodos retienen su estado en ausencia de estímulos y pueden compartir su carga eléctrica con otros nodos.

El trazado del circuito VLSI comienza cuando se ha comprobado el diseño estructural mediante simulaciones en los niveles de circuito, de transistor y de registro. Debe observarse que las simulaciones no son un procedimiento formal de validación del diseño, ni tampoco una comprobación exhaustiva del mismo. Idealmente, los procedimientos de comprobación deberían basarse en las especificaciones funcionales para asegurar la idoneidad de los diseños.

El diseño del trazado físico de una red VLSI en un juego de máscaras se puede dividir en tres partes, no necesariamente secuenciales: definición de la geometría de las células, situación de las mismas en la pastilla y su interconexión. Los datos del trazado se obtienen a partir del conjunto de los dominios poligonales que se utilicen para definir la máscara por superposición o contacto. Los polígonos son líneas que se cortan en los vértices y que delimitan una figura plana cerrada. El trazado es complejo, pero puede simplificarse durante el diseño identificando células repetitivas a nivel de polígono sin interconexión explícita, que luego se puedan colocar e interconexionar para formar el trazado global. El diseño jerárquico (agrupación de células que forman otra de nivel superior ubicable en diferentes puntos de la pastilla) es muy útil para controlar la gran cantidad de datos que se manejan.

Otra forma de controlar la complejidad de los diseños consiste en idealizar la representación del trazado como se indica en la figura 2. El trazado de polígonos presenta los datos geométricos orientados a la tecnología, o sea, los datos del trazado se organizan en capas correspondientes a las diferentes máscaras utilizadas en la fabricación del circuito VLSI. El trazado simbólico "en barras" es una representación orientada al usuario que aísla los polígonos de un dispositivo del resto de los que forman los nodos de la red. El trazado se funde ahora en una sola capa que muestra claramente las conexiones entre redes y dispositivos. El trazado "en bastón" idealiza más aún el anterior y reduce las conexiones a una línea central simbólica. Finalmente, la jerarquía de los bloques se refleja en el planteamiento general del trazado.

Un aspecto importante en el diseño de una pastilla es el procedimiento de comprobación, indicado por flechas en la figura 2. Las comprobaciones se realizan localmente en cada dominio estructural o físico, o bien entre unos y otros, para garantizar la ausencia total de errores.



Controlador terminal de procesador dual para el circuito de línea del Sistema 12 realizado en tecnología CMOS de 3 µm. Contiene 23.000 transistores en una pastilla de 5,96 x 6,02 mm.

Generación actual de herramientas CAD para el diseño de circuitos VLSI a medida

Un factor fundamental en el diseño asistido por ordenador (CAD) es la forma elegida para la comunicación entre el hombre y la máquina. Los lenguajes actuales de diseño para VLSI son en su mayoría gráficos, y combinan la potencia de cálculo con un interfaz cómodo. Los datos del trazado y del esquemático (representación gráfica de la red) se introducen mediante programas de dibujo geométrico.

Los programas de entrada que maneja el usuario suelen reconocer atributos específicos de los datos dibujados, como la conectividad del esquemático, que asocia un subgrupo de datos gráficos a un nudo de la red y que a menudo se comprueba durante la sesión de edición. Esta es una diferencia

esencial entre los programas de captación de esquemáticos y los orientados al trazado.

Actualmente comienzan a utilizarse lenguajes específicos de descripción de esquemáticos, trazados y formas de onda como medio idóneo para introducir los datos en los sistemas CAD, lo que se denomina entrada mediante procedimientos. Los textos se introducen con un editor y se codifican en la base de datos. Las entradas mediante procedimientos no están aún en fase de producción porque su semántica no está orientada al LSI, y todavía son poco útiles para el diseño. Sin embargo, los ingenieros de CAD ya usan procedimientos para generar y comprimir la descripción de los datos. Estos lenguajes son fundamentales en los compiladores de silicio, los cuales traducen directamente descripciones funcionales en lenguajes de alto nivel al trazado físico de la pastilla.

Los diseños de circuitos VLSI incluyen gran cantidad de datos, de muchos tipos diferentes. Por lo tanto, se precisan programas para mantener la base de datos, asegurar el control de las versiones y comprobar la coherencia de los diseños. Estas herramientas de gestión de datos son fundamentales en los sistemas CAD. Anteriormente, cada herramienta tenía su propia estructura y representación de datos, y había poca integración entre ellas. En consecuencia, se necesitaban conversiones muy costosas entre las diferentes bases de datos. Por ejemplo, si en una base de datos de 25.000 transistores había un transistor incorrecto, aunque el esquemático se pudiera modificar con rapidez se necesitaban por lo menos tres conversiones completas entre bases de datos para evaluar las nuevas características del circuito: de captación de esquemático a listado de conexiones, luego a entrada textual para el simulador, y por último a formato binario de entrada al simulador.

Se utiliza hoy una herramienta informatizada para comprobar las reglas del trazado (esto es, la integridad de la base de datos) y comparar la conectividad obtenida del trazado con el listado del esquemático (controlando la coherencia entre las bases de datos del trazado y del esquemático). En ITT se utilizan además herramientas para posicionamiento y conexionado automático de las células estándar y también para minimizar la lógica de las funciones booleanas. Estas herramientas utilizan sólo parcialmente la organización jerárquica de los datos del usuario. Además, como el trazado depende de las reglas tecnológicas VLSI, no es fácil convertir el trazado de una línea de producción MOS VLSI a otra.

Los sistemas de posicionamiento y conexionado de los diseños VLSI a medida son todavía experimentales. Aunque ya se utiliza eficazmente en producción la extracción automática de resistencias, condensadores y otros dispositivos desde un trazado al correspondiente circuito VLSI, todavía no es posible la plena comprobación de los datos estructurales en los niveles de circuito real y de temporizaciones.

Esta falta de herramientas se debe a la complejidad de los diseños VLSI. Por ejemplo, para calcular la resistencia parásita de las interconexiones en los circuitos VLSI hace falta resolver las ecuaciones de potencial de Laplace, con condiciones de contorno que dependen de la forma del conductor y de la posición de los contactos. Las resistencias parásitas integradas presentan generalmente geometrías complicadas determinadas por el trazado global. El tiempo normal que se necesita para obtener el valor de la resistencia de una forma poligonal típica con el 1% de precisión es de unos cinco segundos de CPU, en una estación de trabajo que pueda procesar un millón de instrucciones por segundo y con una memoria de 2 k-octetos para el almacenamiento de los datos. Recordemos que un circuito VLSI normal contiene más de 25.000 polígonos de este tipo.

Nueva generación de herramientas CAD para diseños VLSI a medida

El campo del CAD para el diseño de pastillas VLSI es muy amplio, por lo que se examinarán solamente los atributos esenciales de los nuevos sistemas CAD que se están desarrollando en ITT.

Integración con sistemas CAD

El diseño MOS VLSI está alcanzando un estado de madurez, en el que los conocimientos de diseño son ya estables. Los diseñadores saben cómo hacer una pastilla que funcione, pero también saben que su trabajo no sería practicable sin las herramientas CAD. La próxima etapa consistirá en transferir los conocimientos del diseño al ordenador como una herramienta más. El nivel actual de madurez permite la codificación clara de los datos de la ingeniería de VLSI en un sistema CAD, y por lo tanto es factible definir modelos genéricos de datos utilizables en todas las actividades de diseño VLSI. En la figura 4 se indica la estructura integrada que tendría este sistema CAD para VLSI.

Un modelo de datos es simplemente un tipo de datos, es decir, un juego de objetos (documentos, esquemáticos, listados, trazados) y de operaciones (edición, comparación de trazados con listados, extrac-

ción de parámetros, correcciones). El modelo de datos viene determinado totalmente por la metodología empleada, y es la clave para la integración del sistema. Responde a los conocimientos del diseñador VLSI, esto es, sólo los diseñadores conocen exactamente los tipos de datos y de preguntas que son esenciales para la interacción futura de los modelos de datos de diseño con el sistema CAD. El diálogo hombre-máquina orientado a la aplicación ha de estar basado en campos de datos específicos y en la organización propia de los datos. La integración de las herramientas se consigue con el uso, interpretación y comprensión de un juego de datos único y completo.

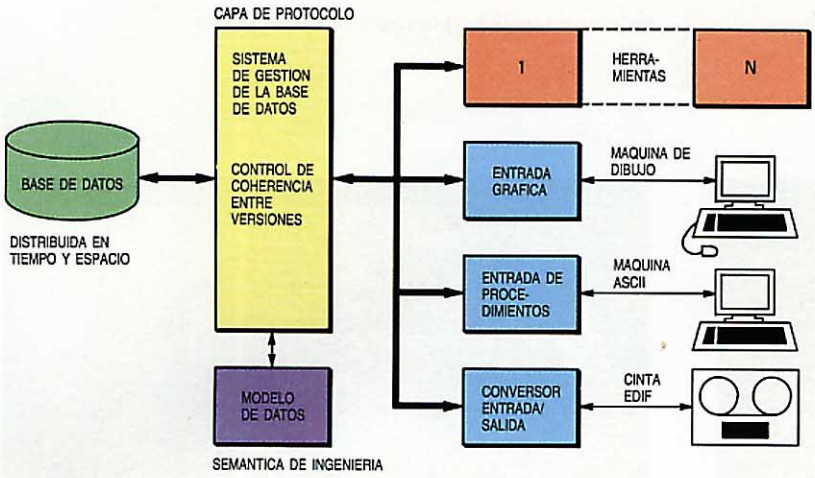


Figura 4
Sistema CAD totalmente integrado para el diseño VLSI.

El protocolo de acceso al modelo de datos está normalizado por el DBMS (sistema de gestión de la base de datos), que controla el almacenamiento de los objetos del modelo de datos y la ejecución de las operaciones del mismo. El DBMS opera en un entorno multiusuario formado por una red de ordenadores personales, estaciones de trabajo, equipos especializados (p.ej., aceleradores de simulación) y ordenadores centrales.

Los diseños VLSI aparecen, pues, como bases de datos, o sea, colecciones de datos administrados por el DBMS. Desde el punto de vista del usuario, los modelos de datos son similares a los lenguajes de programación. El DBMS es como un procesador (interpretador) para los modelos de datos. Por tanto, las bases de datos (diseños VLSI), son como programas específicos realizados en el lenguaje del modelo de datos e interpretados en el DBMS.

Un lenguaje de diseño bien definido y aceptable ha de permitir que los diseñadores puedan comunicarse entre sí o con las herramientas de trabajo de un modo preciso y eficaz. Además, el uso de iguales tipos de datos en todas las herramientas resuelve el problema de la lentitud de conversión de los mismos en los intercambios entre herramientas.

En la práctica, la realización de un DBMS para VLSI es un verdadero reto. Los modelos de datos VLSI necesitan manejar muchas clases de objetos y gran cantidad de datos, lo cual exige utilizar técnicas de estructuración basadas en inteligencia artificial y técnicas de gestión propias de bases de datos comerciales, del tipo de la base de datos relacional Oracle.

Desde el punto de vista de los usuarios, entre las ventajas que ofrece la integración completa de las herramientas están el control de uniformidad de las versiones, la protección de los datos y también la inherente comprobación de la coherencia. Sin embargo, se debe seguir la disciplina impuesta por los protocolos del sistema, aunque sean más difíciles de utilizar.

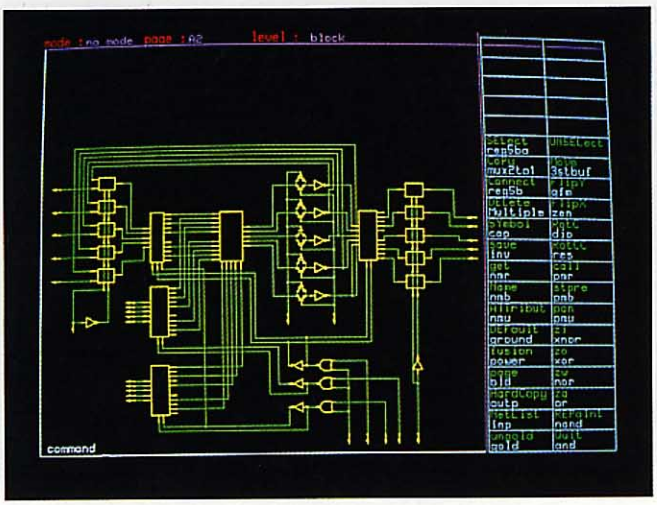
Herramientas de entrada sensibles a la semántica

Los programas para captación de datos de esquemáticos, trazados y formas de ondas, deben comprobar la coherencia de tales datos tan pronto como sea posible, y preferiblemente en la misma sesión interactiva de entrada. El programa procesa los datos del usuario y comprueba que están en el formato apropiado. Asimismo interpreta los datos y reacciona a entradas sin significado, como en el caso de una resistencia negativa.

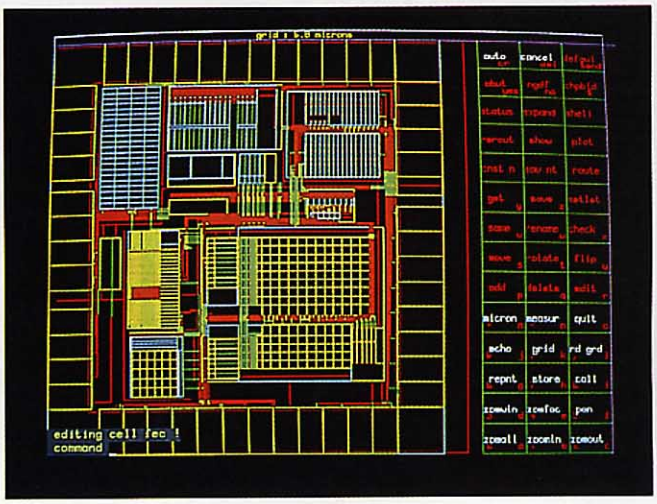
Algunos editores de trazado ofrecen una comprobación casi instantánea de las reglas de diseño, completada después con un entorno multiventana que compara listados (entre trazados y esquemáticos), comprobando y validando. También deberían extraer del trazado las resistencias, condensadores y otros dispositivos.

En el nivel estructural se utilizan procedimientos similares para comprobar que las realizaciones lógicas y eléctricas cumplen con un conjunto de normas de buen estilo de diseño. Por último, los procedimientos de recopilación de la base de datos se completan con la extracción de descripciones funcionales y de comportamiento.

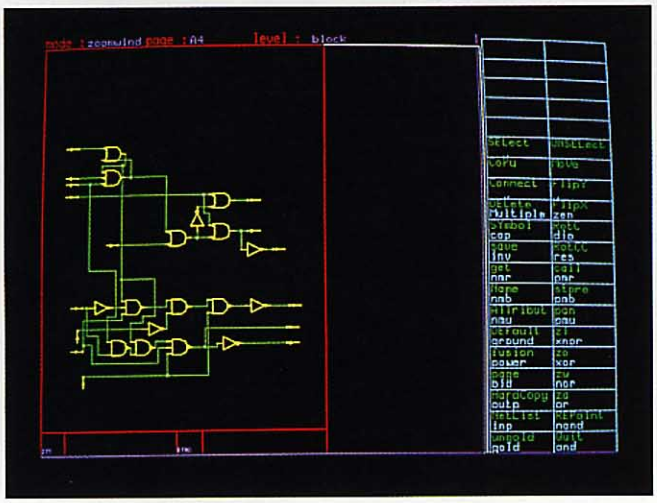
Como la interpretación de datos es una tarea que utiliza mucho la CPU, sólo deberá aplicarse cuando se añadan nuevos datos a las bases de datos existentes (de forma incremental o en proporción limitada), lo cual favorece el uso de la jerarquía para evitar que aumente excesivamente la complejidad de los diseños y la sobrecarga de las estaciones de trabajo en ingeniería.



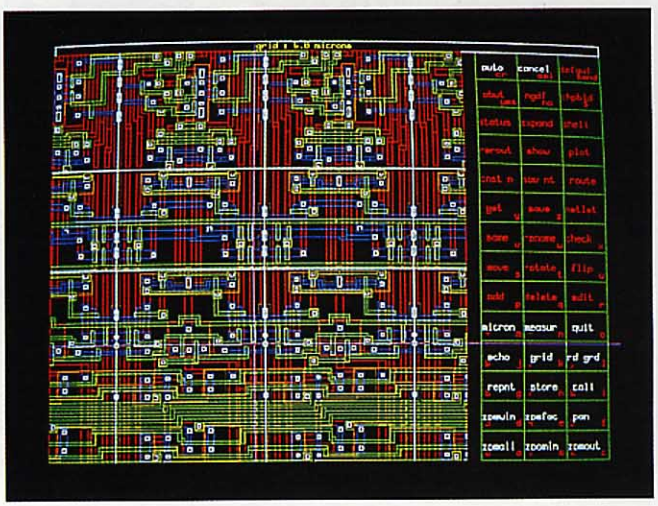
Entrada de esquemático: composición del nivel registros.



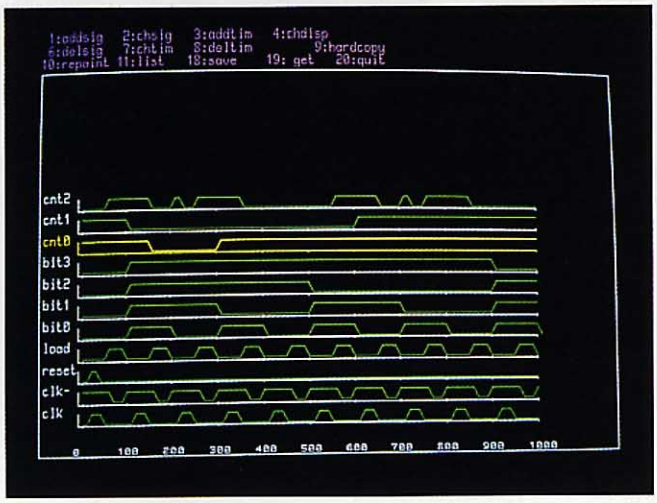
Entrada de trazado: plan general de la pastilla.



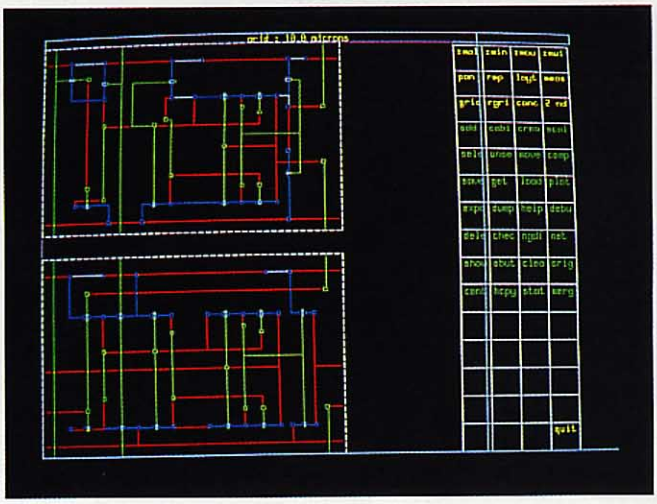
Entrada de esquemático: circuito lógico CMOS.



Entrada de trazado: trazado de células poligonales de un camino de datos.



Salida de simulador: formas de onda que expresan el comportamiento del circuito.



Entrada de trazado: representación en bastón de una célula.

Estas características mejoran muy notablemente la productividad del diseño al permitir la detección de los errores tan pronto como se produzcan.

En la práctica se requiere un cuidadoso compromiso para decidir lo que se debe comprobar y a qué nivel, a la vista de la potencia de proceso disponible y de los tiempos de respuesta. Como las herramientas CAD no pueden comprobarlo todo, deberán limitarse a detectar los errores más corrientes, y, a petición, proporcionar ayudas en forma de menús. Si estos procedimientos se efectuaran verdaderamente en tiempo real, se necesitaría una potencia de proceso excesiva, incluso para las estaciones de trabajo más avanzadas.

Nuevo juego de herramientas VLSI para diseño estructural y físico

Las herramientas del IVDS sustentan en su totalidad la metodología descrita en las secciones anteriores. Las figuras 5 y 6 muestran los juegos de herramientas para los dominios estructural y físico, respectivamente.

Las herramientas para el diseño estructural utilizan básicamente la representación a nivel de transistor, realizándose las abstracciones lógicas y funcionales con el simulador en este nivel. Las realizaciones lógicas y eléctricas se comparan con prácticas y estilos de diseño ya probados, mediante la tecnología de los sistemas expertos. Dichas prácticas y estilos de diseño se codifican como un juego de reglas, y el sistema experto comprueba su cumplimiento en la base de datos de VLSI.

Las herramientas para el diseño físico combinan programas como los editores de trazado global y local con otros medios avanzados, como los compiladores de silicio, posicionadores y programas de conexionado. Los analizadores ayudan al usuario a definir la geometría de las células para obtener un trazado compacto. El trazado completo se genera con algoritmos de posicionamiento y conexión automáticos.

Jerarquía sintáctica y semántica en el método de diseño VLSI

La programación en los lenguajes convencionales se simplifica por medio de las macros y las funciones. Las macros se usan sólo a nivel sintáctico para simplificar las especificaciones, y no proporcionan ninguna abstracción semántica, ya que se expanden durante la compilación y no toman ningún valor de función.

En las herramientas CAD para VLSI, debe distinguirse entre jerarquía sintáctica y semántica, de forma análoga a como se hace entre macros y procedimientos en los

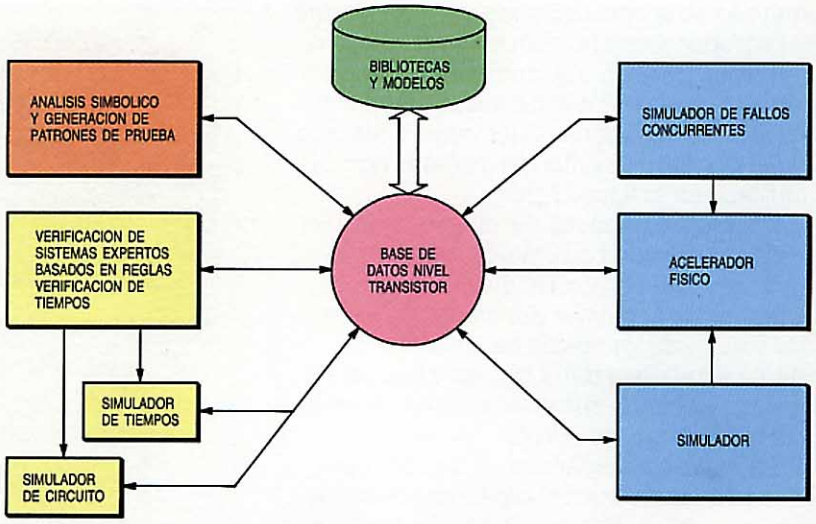


Figura 5
Juego de herramientas de diseño estructural perteneciente al sistema de diseño VLSI de ITT.

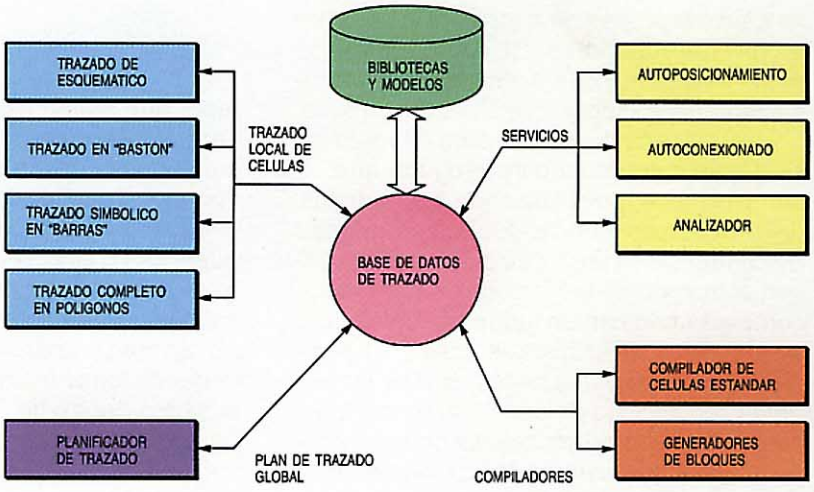


Figura 6
Juego de herramientas de diseño físico del sistema de diseño VLSI de ITT.

lenguajes de programación; esto se relaciona con los procedimientos de eliminación de la teoría clásica de redes[3], en los que una subred se trata como una caja negra cuando se la considera desde el resto de la red exterior a ella. Debe destacarse que así se heredan técnicas ya experimentadas en la programación y en la teoría de redes. La formalización de la jerarquía es de vital importancia al acercarnos a la nueva era VLSI, ya que transforma en ciencia el arte de la comprobación de circuitos implantados en silicio.

La jerarquía sintáctica es un medio para reducir el volumen de los datos, detectando la repetición en varios lugares de un grupo de elementos pertenecientes a la descripción de la pastilla. Los datos repetidos se agrupan bajo un identificador común que define una célula, y que una vez posicionada ésta se puede llamar como si fuera una macro. Hace mucho que se sigue este

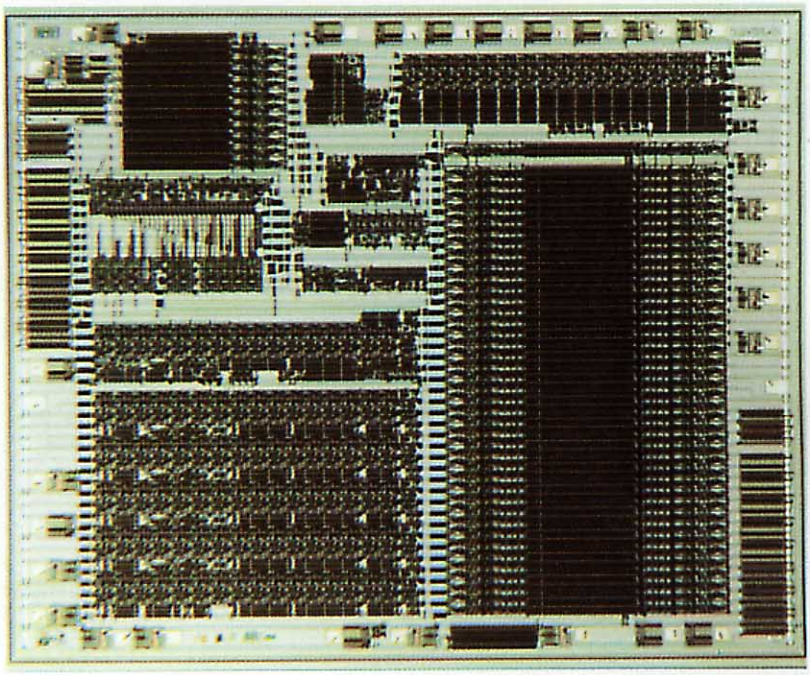
método en las aplicaciones gráficas, donde se agrupan y referencian en ficheros separados los detalles que se repiten, como en los formatos de datos de trazado Apple o GDS II. Formalmente, la jerarquía sintáctica define una representación de datos como la indicada en la figura 7.

El aspecto esencial del diseño jerárquico es la multiplicidad de estructuras jerárquicas que se obtienen con las distintas representaciones de la pastilla (funcional, de transistor y de trazado) típicas de los sistemas deductivos. Para tener una sola estructura jerárquica habría que acudir a un sistema de diseño del tipo inductivo.

La jerarquía semántica es un concepto que va más lejos que la organización sintáctica. Su objetivo no está sólo en organizar los abundantes datos de ingeniería, sino en construir una representación simplificada de un grupo de datos a un nivel sintáctico específico. Esta representación o abstracción puede asumir de modo coherente la función del grupo completo para los análisis en niveles superiores. Se comprenderá mejor este principio al aplicarlo a la comprobación del trazado y al análisis del circuito.

Los trazados de los circuitos integrados se deben comprobar para asegurar que cumplen las especificaciones geométricas del proceso de fabricación. Los programas de control de las reglas de diseño comprueban el trazado de los circuitos integrados, comparándolo con un juego de reglas de diseño físico. El análisis se basa en la organización de los datos de la pastilla y en la definición juiciosa de una abstracción idónea. En el caso de las células compuestas, esta técnica requiere la detección de todos los errores sin conocer la posición de la célula en el circuito, lo cual resuelve el problema de comprobar las reglas de diseño en el interior de la célula, y evita tener que repetir esta comprobación en los análisis posteriores. La información no contrastable mientras no se conozca el contexto de aplicación de la célula, se almacena en la *abstracción* de la célula compuesta, la cual puede especificarse con las mismas órdenes del programa convencional de comprobación de las reglas de diseño. Se crea la abstracción de la célula almacenando todos los datos de las máscaras situados dentro de una distancia dada del contorno de la célula.

Cuando se trabaja con redes VLSI suele ser conveniente separar parte de la red de sus alrededores, como en el caso de que se estudie una célula compuesta sin considerar su estructura interna. La célula se analiza



Circuito ecualizador de decisión realimentada del interfaz U de la RDSI. Usa tecnología CMOS de 3 μm e incorpora 46.500 transistores en una pastilla de 7 x 6 mm.

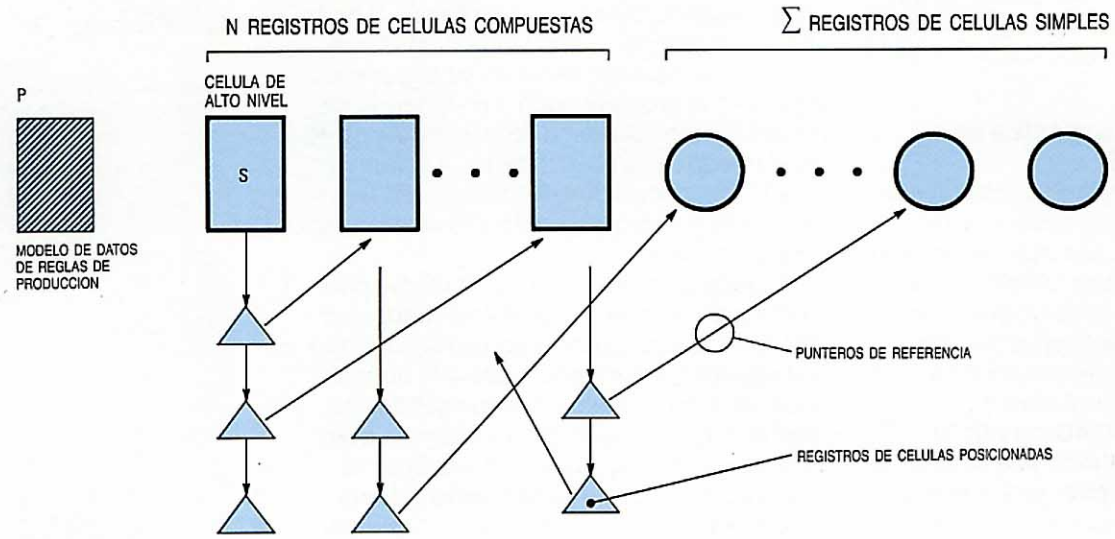


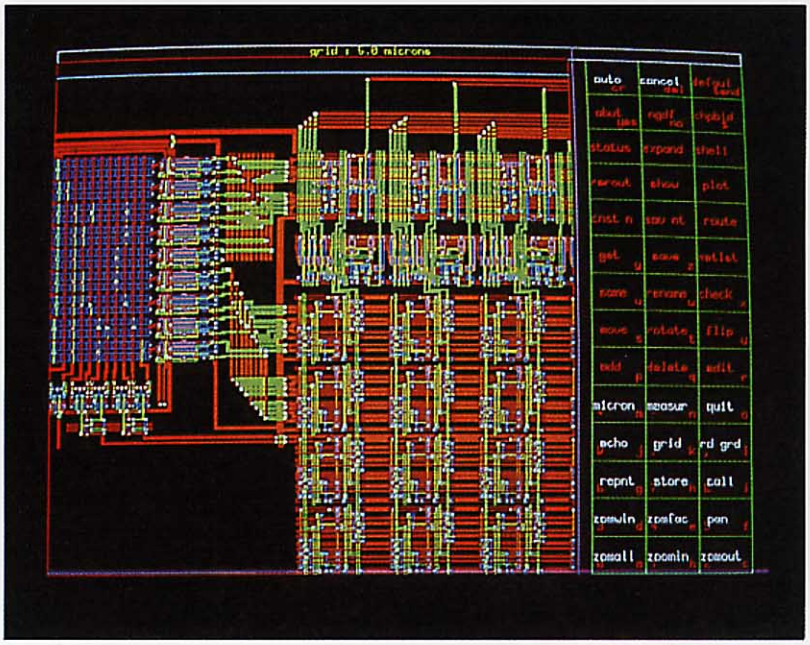
Figura 7
Representación formal de la organización de datos VLSI. Las células compuestas (rectángulos) se forman posicionando células simples (círculos) u otras células compuestas (triángulos). Las referencias (flechas) pueden ser internas al fichero de datos o externas, como en el caso de bibliotecas de usuario.

entonces como una caja negra en el nivel jerárquico siguiente.

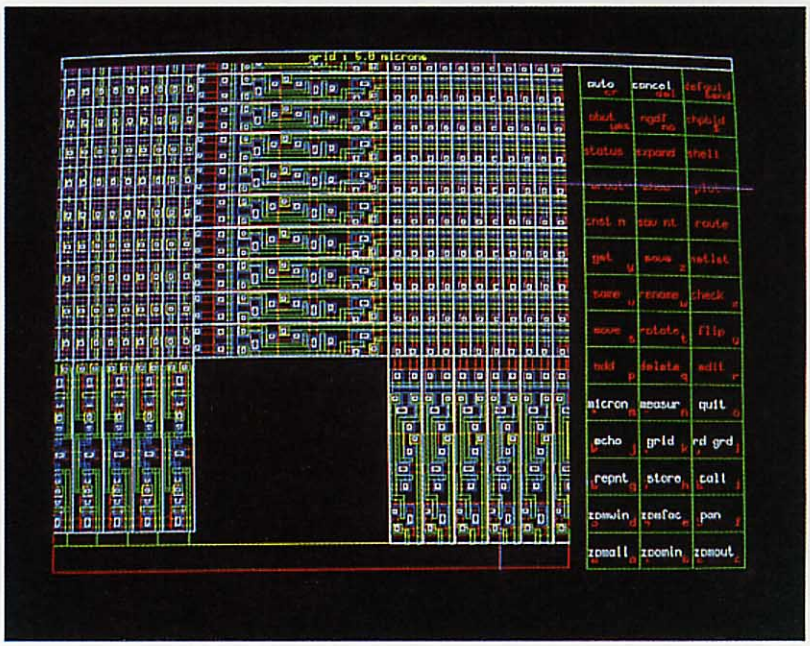
En los análisis de circuitos se da una exacta descripción de la red entera y hay que obtener las señales en diversos puntos de la red, para lo cual se deduce un sistema de ecuaciones de aquella descripción y se resuelve de modo similar al empleado en las ecuaciones matriciales que utilizan los programas de simulación de circuitos.

El problema del análisis de una célula compuesta es distinto, ya que, debido a no estar especificado el contexto, el número de ecuaciones es normalmente inferior al de incógnitas. Sin embargo, éstas se pueden dividir en variables externas e internas, y al eliminar las internas queda un juego de ecuaciones que describe el comportamiento externo de la célula en todos los contextos. El comportamiento externo, junto con las variables externas, forma una abstracción de la célula compatible con la simulación. Es una abstracción porque reduce la cantidad de información, pero es suficientemente completa para permitir el análisis en el nivel jerárquico superior. Esta etapa de eliminación transforma el estado de la célula compuesta y la convierte en célula simple (la célula se representa ahora en el nivel jerárquico siguiente).

La realización automática de abstracciones semánticas en simuladores abrevia mucho los análisis y abre el camino para comprobaciones verdaderamente simbólicas de los circuitos MOS.



Servicios de trazado: el autoconexionado entre células finaliza el trazado de un controlador.



Compilador de silicio: trazado de PLA.

Conclusiones

Al ser los circuitos VLSI cada vez más complejos, se necesita una metodología de diseño adecuada y nuevas herramientas CAD en las que se apoyen diseños VLSI totalmente a medida. Es esencial la integración de las herramientas mediante un modelo de datos común y la eficaz utilización de métodos jerárquicos con abstracciones formales en los dominios estructural y físico. A la vista del gran esfuerzo de desarrollo CAD necesario para alcanzar estos objetivos a medio plazo, es aconsejable la utilización de programas-marco comerciales en los que se puedan integrar las nuevas herramientas a medida que vayan apareciendo. De esta forma se protegerán los actuales conocimientos internos de diseño, al mismo tiempo que se centra la atención en implantar una metodología que se acomode a las futuras necesidades de circuitos VLSI.

Referencias

1 J. Cornu y M. Meinck: Central digital ITT 1240: Tecnología de componentes avanzada: *Comunicaciones Eléctricas*, 1981, volumen 56, nº 2/3, págs. 161–172.

2 J. Danneels y M. Meinck: Metodologías para el diseño de VLSI totalmente a medida: *Comunicaciones Eléctricas*, volumen 58, nº 4, págs. 389–397.

3 V. Belevitch: Classical Network Theory: *Holden Day Series in Information System*, 1968.

Johan M. Danneels nació en Aalter, Bélgica, en 1949. Obtuvo el MS en ingeniería electromecánica y el PhD en electrónica, en la Universidad Católica de Lovaina en 1972 y 1976, respectivamente. En 1983 se graduó MBA en la Universidad de Boston, Bruselas. Ingresó en el grupo de investigación de Bell Telephone Manufacturing Company en 1976, llegando a ser responsable del laboratorio de microeletrónica. En 1984 el Sr. Danneels pasó a encargarse de la división de circuitos de conmutación y transmisión.

Pierre-Paul Guebels nació en Lubumbashi, Zaire, en 1983. Obtuvo el grado en ingeniería electrónica en la Universidad Católica de Lovaina en 1977. De 1977 a 1980 fue ingeniero de investigación en el laboratorio de microelectrónica, trabajando en circuitos MOS analógicos, filtros de capacidades conmutadas y modelado de transistores MOS. En 1980, ingresó en el Centro de Investigación de BTM para dedicarse al diseño de circuitos integrados a medida para equipos de telecomunicación. El Sr. Guebels actualmente dirige el equipo de CAD de diseño VLSI en el grupo de desarrollo de microelectrónica de BTM.

Michel C. Rahier nació en Namur, en 1953. Se graduó en ingeniería electrónica y obtuvo el PhD en microelectrónica en la Universidad Católica de Lovaina, en 1976 y 1979 respectivamente. En 1980 ingresó en el Centro de Investigación de BTM en Amberes, dedicándose al diseño de circuitos integrados y herramientas CAD avanzadas. Actualmente el Dr. Rahier dirige el grupo de desarrollo de microelectrónica de BTM.

Entorno soporte para desarrollos VLSI parcialmente a medida

El diseño parcialmente a medida es un procedimiento rentable para la producción de circuitos VLSI de baja y media complejidad. La metodología ITT de diseños parcialmente a medida hace hincapié en la arquitectura, en el diseño lógico y su comprobación, y, de manera opcional, también en el diseño y comprobación del trazado, mediante el uso de herramientas informatizadas y bibliotecas de células.

B. Prazic
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

La VLSI es una tecnología clave que tiene un impacto creciente en multitud de productos, y en torno a ella se han desarrollado ya equipos como la central digital Sistema 12 y la nueva generación de sistemas ITT de comunicaciones de empresa. Desde hace años ITT reconoció la importancia de esta tecnología y comenzó a crear una completa gama de métodos y herramientas de diseño VLSI, con el objetivo de asegurarse la experiencia y los medios necesarios para desarrollar productos avanzados.

En la metodología ITT, los diseñadores de sistemas participan en el diseño de los circuitos VLSI desde el principio del proceso, o sea, desde la fase inicial de definición de la arquitectura. Esa es la etapa más importante en el diseño de los componentes VLSI, ya que es ahí donde puede ponerse en juego toda la creatividad para aprovechar al máximo las ventajas de la tecnología. Las metodologías de diseño de circuitos integrados parcialmente a medida son más eficaces en este aspecto, ya que los ingenieros de sistemas intervienen también en la fase de diseño lógico.

Otra faceta a considerar es el coste del diseño. Al aumentar la complejidad de los circuitos VLSI, crecen también los costes y la duración del diseño. El método tradicional para abreviar los diseños a base de dedicar a ellos más personas, ha demostrado no ser eficaz, y por lo tanto el único camino restante es el de mejorar la productividad. La experiencia enseña que, con circuitos de baja a media complejidad, los diseños parcialmente a medida ofrecen una productividad bastante mayor que los realizados totalmente a medida (Fig. 1). Los diseños parcialmente a medida, con ordenaciones de puertas o células estándar, son especialmente rentables en aplicaciones de pequeñas series, que requieren cortos ciclos de desarrollo y rápida respuesta sobre el comportamiento de los dispositivos. Sin embargo, los diseños a medida siguen siendo la tecnología óptima para circuitos muy complejos o que vayan a fabricarse en grandes cantidades, en los cuales debe minimizarse tanto el coste unitario como el tamaño de la pastilla.

La metodología ITT para el diseño parcialmente a medida hace hincapié en configurar la arquitectura del sistema, en el diseño lógico y su comprobación, y, de manera opcional, también en el diseño y comprobación del trazado. Todas las fases se apoyan en el uso de herramientas de diseño informatizado y de bibliotecas de células de familias lógicas escogidas para aplicaciones de telecomunicación.

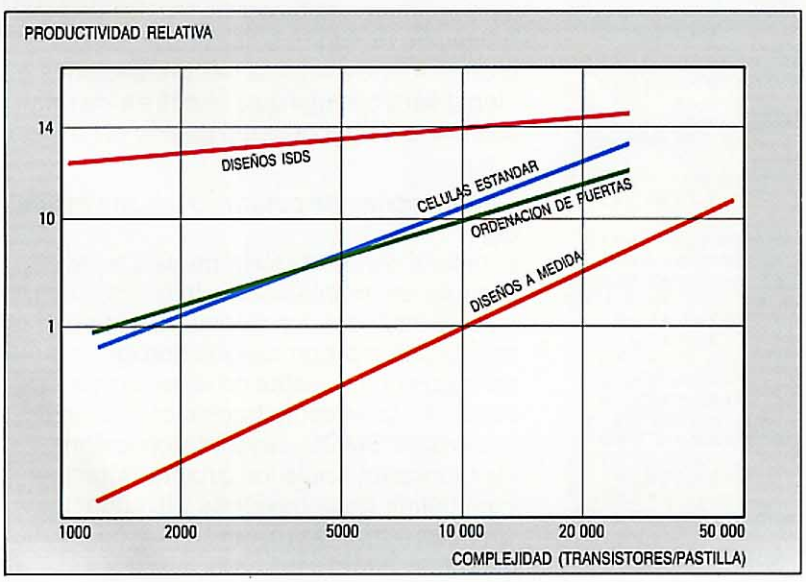
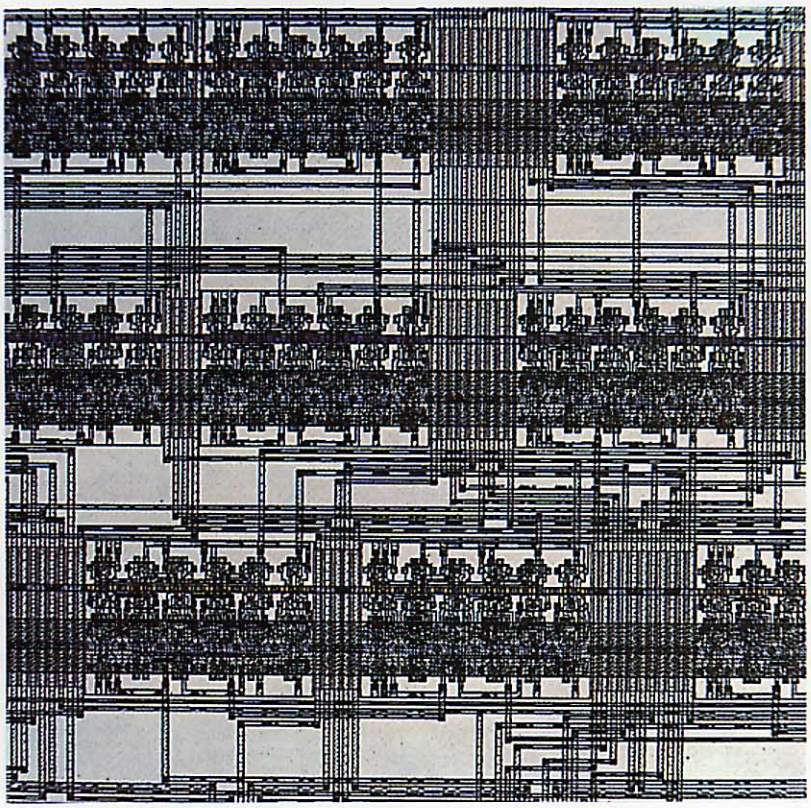
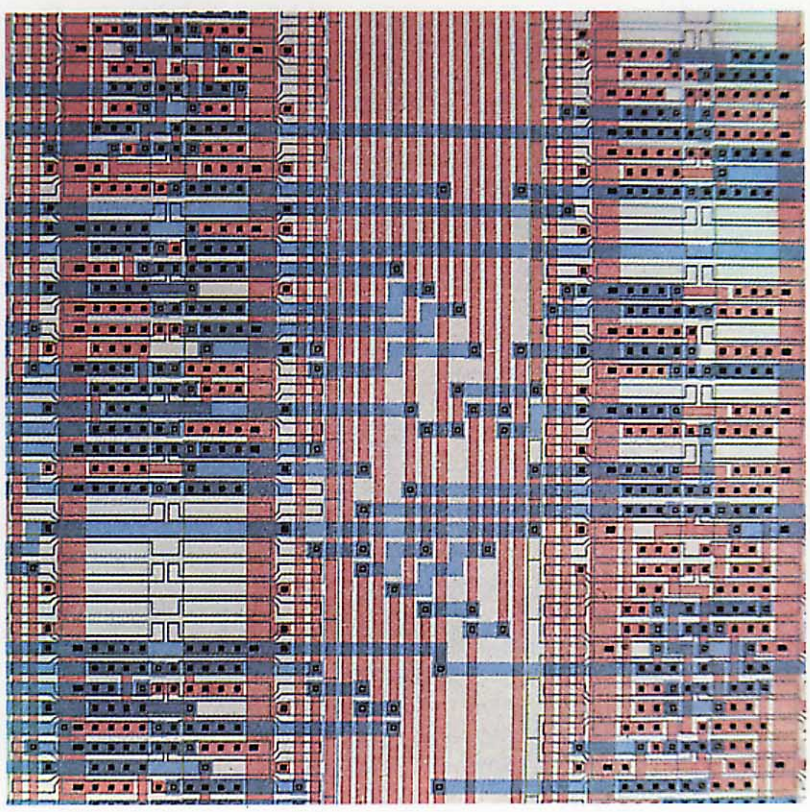


Figura 1
Comparación entre la productividad alcanzable con diferentes métodos de diseño VLSI. La línea superior muestra la mejora sustancial de productividad que resulta de la introducción del ISDS.

Para completar el entorno de diseño parcialmente a medida, los equipos de diseñadores han de contar con la ayuda de especialistas que evalúen las tecnologías de semiconductores, y han de saber apreciar las implicaciones económicas de las diferentes tecnologías de fabricación en las opciones de diseño.



Parte del trazado gráfico de un circuito VLSI parcialmente a medida, con células estándar.



Parte del trazado gráfico de un circuito VLSI parcialmente a medida, con ordenación de puertas.

Elementos del diseño parcialmente a medida

Herramientas CAD

ITT viene usando ya muchos años herramientas de diseño asistido por ordenador. La versión actual de ISDS (sistema de diseño parcialmente a medida de ITT) ofrece un entorno completo de diseño, que contiene diversas aplicaciones. El ISDS es utilizado por los ingenieros de sistemas en los centros de diseño VLSI en toda ITT, y a estos ingenieros se debe en gran medida la introducción de las mejoras y nuevas características que lo distinguen de la versión original[1].

Concepto de célula

El diseño parcialmente a medida se basa en células – elementos lógicos prediseñados que realizan una función lógica concreta –, contenidas en una biblioteca que representa una familia lógica específica (p. ej., 3 µm CMOS). La complejidad de las células puede variar, desde un simple inversor hasta un biestable o célula RAM.

El concepto de célula prediseñada hace posible que los ingenieros de sistemas diseñen componentes VLSI sin un profundo conocimiento de las técnicas de diseño de circuitos al nivel de transistores. En cambio, necesitan conocer y saber manejar las herramientas del diseño asistido por ordenador.

La flexibilidad de las herramientas de comprobación informatizadas y las bibliotecas de células permite que los ingenieros de sistemas ensayen distintas arquitecturas para optimizar las prestaciones del sistema. Asimismo puede usarse simulación por ordenador para analizar las prestaciones e identificar y corregir los errores en los retardos antes de implantar el diseño en silicio.

Ordenaciones de puertas o células estándar

El diseño parcialmente a medida puede basarse en ordenaciones de puertas o en células estándar. La ordenación de puertas es una oblea preprocesada con áreas predefinidas compuestas de dispositivos sin conectar, tales como transistores *p* y *n* en tecnología CMOS, cuya interconexión reproduce en silicio los circuitos lógicos. Para definir las conexiones y los contactos sólo dos o tres máscaras se han de diseñar a medida.

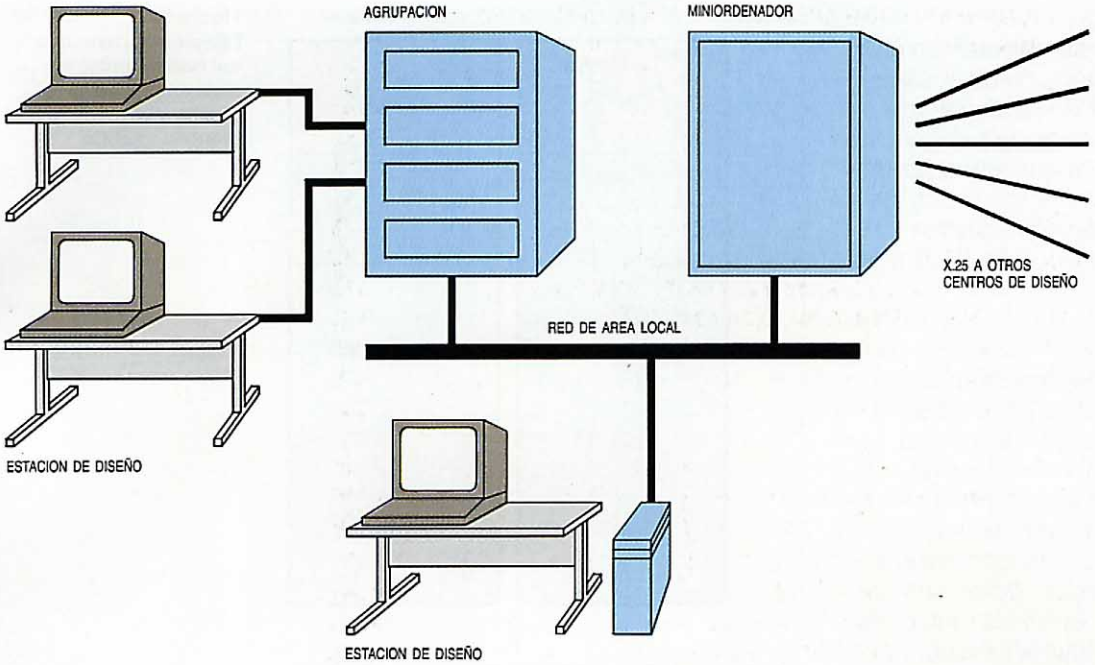


Figura 2
Configuración de equipo para diseños VLSI parcialmente a medida con el ISDS.

Los diseños de células estándar no tienen una posición predefinida para las células, con lo que el trazado puede optimizarse para obtener una pastilla de menor tamaño que con las ordenaciones de puertas. En este caso todas las máscaras se deben hacer a medida, por lo que la inversión inicial habrá de ser mayor que en el diseño primeramente citado.

Desde el punto de vista de diseño del sistema, la elección entre ordenaciones de puertas o células estándar está principalmente condicionada por los requisitos individuales de las funciones del circuito. Así, por ejemplo, las RAM o ROM que por su trazado compacto sobre una pastilla requieren el uso de células estándar. En casi todas las demás aplicaciones se prefieren las ordenaciones de puertas, pues pueden obtenerse en menores tiempos y a costes más económicos que las células estándar.

Sistema ISDS

Evolución de los requisitos

En el sistema ISDS, los requisitos de los programas y de las bibliotecas de células han evolucionado en dos fases. La primera se caracterizó por la integración de herramientas basadas en estaciones de trabajo y de bibliotecas de células desarrolladas y probadas con anterioridad en ITT. Estas bibliotecas se fundamentaban en las tecnologías CMOS de 3 µm y doble metalización, entonces nacientes, que podían obtenerse de fabricantes de semiconductores pertenecientes o no a ITT.

En cuanto a los programas, en el ISDS se adoptó un procedimiento avanzado, basado en la estación de trabajo. Su relativa economía en CPU y la disponibilidad de herramientas CAD interactivas, hizo que las estaciones de trabajo de ingeniería resultasen muy adecuadas para servir como medio de entrada para el diseño parcialmente a medida. Durante el periodo evolutivo se evaluaron e introdujeron programas apropiados, y se añadieron herramientas CAD de diseño interno.

La segunda fase fue de maduración, y coincidió con un mayor uso del sistema. La experiencia demostró que la simulación lógica era la parte más crítica del proceso de diseño, y que se necesitaban prestaciones superiores a las de las herramientas comerciales. El requisito más importante para el simulador lógico se refería a la capacidad de efectuar modelados precisos y que además pudieran controlarse y modificarse en código fuente. En particular, los parámetros de retardo de las células (p. ej., retardo de propagación y capacidad de conducción), que varían apreciablemente de una tecnología de proceso a otra, tuvieron que modelarse en código fuente para garantizar la simulación exacta del diseño.

Durante este periodo, los productos ITT comenzaron a aprovecharse de los progresos en las tecnologías de los semiconductores. En tales diseños, el ISDS se prestó a utilizar familias lógicas propias de escogidos fabricantes de circuitos VLSI parcialmente a medida, aunque el sistema se mantuvo independiente de cualquier suministrador o proceso tecnológico en particular.

Entorno de diseño ISDS
Las herramientas ISDS se ejecutan en un entorno de proceso distribuido disponible en todas las compañías ITT. Dicho entorno está formado por estaciones de trabajo y ordenadores comerciales que actúan como ficheros o periféricos auxiliares. Todos los equipos implicados están conectados mediante una red de área local de 10 Mbit/s (Fig. 2).

Funciones del ISDS
Las tareas del ISDS se dividen en aplicaciones interactivas, ejecutadas por el usuario en estaciones de trabajo, y otras que se ejecutan por lotes como actividades de base en un ordenador más potente. Las funciones de diseño ofrecidas al usuario son la captación de esquemáticos, estimación de retardos, simulación lógica, simulación de fallos y trazado. Entre las funciones de comprobación se incluye la de la conectividad de la red, el análisis de los retardos en el camino crítico entre biestables sincronizados o entre terminales de entrada y salida, y el análisis estadístico de desfases entre vectores de prueba.

Todas las funciones de diseño y de comprobación están estrechamente integradas, y funcionan con una sola base de datos de diseño o sus derivadas. A nivel del sistema, la integración de todas las funciones se obtiene por un programa *monitor de diseño*, que proporciona un interfaz coherente con todas las herramientas. Este monitor guía eficazmente al usuario a través del proceso de diseño, realiza varias tareas de ordenación, y proporciona los medios de adaptación necesarios para transferir al suministrador la base de datos de diseño, con el fin de fabricar las pastillas.

Ciclo de diseño ISDS para circuitos parcialmente a medida

El ciclo de diseño ISDS comprende cuatro grandes fases: diseño de la arquitectura del sistema, diseño lógico y su comprobación, diseño del trazado y su comprobación, y desarrollo de la prueba y verificación final. Cada fase se completa con la redacción del correspondiente documento de ingeniería, y el ciclo termina cuando las pastillas prototipo cumplen las especificaciones del diseño. La figura 3 muestra el reparto aproximado del trabajo entre las distintas fases, y en la figura 4 se indica la secuencia seguida en los diseños ISDS.

Consideraciones iniciales
Antes de comenzar el ciclo de diseño deben especificarse los requisitos físicos de la pastilla, determinados normalmente por la

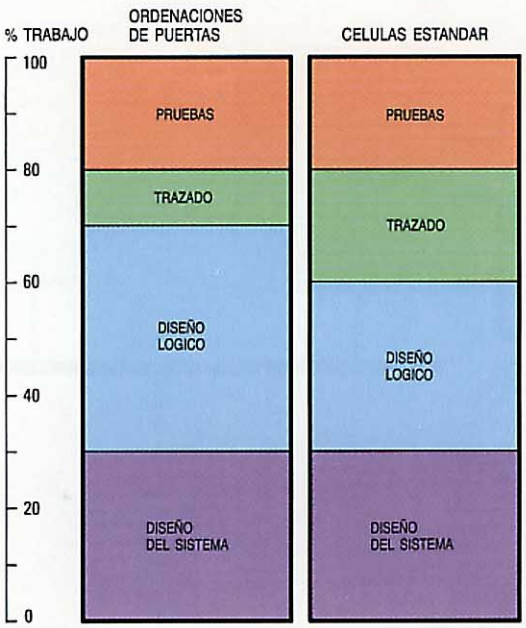


Figura 3
División aproximada del trabajo entre las distintas fases del diseño VLSI parcialmente a medida.

placa de circuito impreso en la que se vaya a equipar. Se han de señalar todas las características típicas del circuito, tales como la temperatura ambiente o la frecuencia de funcionamiento.

Figura 4
Diagrama de flujo ilustrativo de las distintas actividades involucradas en el diseño de un circuito VLSI parcialmente a medida.

Diseño del sistema

Una vez definidos y acordados los requisitos físicos, ya puede iniciarse el ciclo de diseño de la pastilla. En la primera etapa se prepara la especificación del diseño y el diagrama funcional de bloques a nivel de sistema. Una especificación típica de ITT para diseño VLSI incluye la definición del sistema, una especificación del interfaz y de los retardos para todos los terminales de entrada y salida, y un plan de las pruebas en el que se definan los vectores de prueba.

El diagrama de bloques divide el circuito en partes funcionales más pequeñas susceptibles de análisis individual. La ventaja principal de efectuar la partición en esta etapa tan temprana es la de facilitar el diseño jerárquico del circuito, con lo que las funciones más complejas podrán definirse mediante bloques funcionales más sencillos especificados por sus entradas y salidas. De esta forma, los diseñadores pueden trabajar con un reducido número de bloques funcionales en todos los niveles del diseño.

La distribución del sistema en la pastilla es de gran importancia. Los terminales de entrada y salida de la pastilla son un recurso valioso que se debe administrar juiciosamente, pues existe un innegable compromiso entre los mayores niveles de integración y el limitado número de terminales disponibles en la cápsula. La elección de una cápsula determinada afecta significativamente al coste final del dispositivo.

En los diseños VLSI es muy importante la clara definición de los retardos de las señales internas y externas. Como es difícil garantizar unos retardos relativos entre las señales sobre la implantación en silicio, es preferible el diseño síncrono para aquellas señales que deban propagarse por diferentes niveles de circuitos entre biestables; esto significa que todos los biestables que realicen una misma función deberán bascular con la misma señal de reloj.

Otro requisito importante en esta etapa es la definición de un plan global de pruebas que asegure que la pastilla se podrá comprobar fácilmente durante la fabricación y también en la propia placa de circuito impreso. El modo más sencillo es adoptar una estrategia donde se obligue a las funciones que requieran biestables a situarse en un estado conocido al comienzo de las pruebas, tras lo cual ya puede aplicarse la secuencia completa de las pruebas funcionales.

Diseño lógico y verificación

Captación de esquemáticos

Una vez definido el diagrama funcional de bloques del circuito, los diseñadores pueden comenzar a captar el diseño en la base de datos ISDS. El acceso a esta base de datos se hace desde una estación de trabajo, utilizando un editor gráfico interactivo para construir el diagrama lógico en un terminal de pantalla. Por medio del editor gráfico los diseñadores traen a la pantalla los símbolos de las células que vayan a usarse, y establecen luego las conexiones alámbricas necesarias. Asimismo crean símbolos para los bloques de niveles superiores que facilitan la construcción jerárquica del circuito; dichos símbolos consisten en cajas con terminales de entrada y salida, que representan el comportamiento externo de las funciones.

Captación de esquemáticos.

Simulación lógica

Después de haber captado suficientes circuitos, se debe proceder a la creación de un fichero con un listado que describa las conexiones de las células que constituyen el circuito, lo cual es una de las primeras entradas que necesita el simulador lógico. La herramienta ISDS de simulación es el Simlog, desarrollado por ITT especialmente para simular el comportamiento de los dispositivos VLSI. Así, el Simlog simula retardos en el circuito a dos niveles de precisión: nivel funcional y nivel de puertas. Al nivel funcional, los diseñadores pueden

usar el FML (lenguaje de modelado funcional), ampliación del Simlog para describir el funcionamiento de los bloques complejos. Los modelos FML relacionan los terminales de entrada y salida mediante sentencias de comportamiento y declaraciones de datos.

El FML es idóneo para caracterizar grandes estructuras regulares, como RAM, ROM y PLA (ordenaciones de lógica programable), que por su complejidad no pueden representarse al nivel de puertas. En realidad, para este tipo de bloques la única solución práctica es el modelado funcional.

El nivel de puertas proporciona la precisión necesaria para hacer una simulación detallada. En la biblioteca de modelos Simlog cada puerta tiene un tiempo finito de subida y de bajada, que se utiliza para calcular los retardos de propagación de una señal a través del circuito. Basándose en esta técnica de modelado, el Simlog puede utilizarse de dos maneras para calcular los retardos de los circuitos: simulación nominal y de tiempos.

La simulación nominal establece que el circuito funciona correctamente. No es indispensable la precisión en los retardos mientras no se haya obtenido un esquemático funcionalmente correcto y los vectores de prueba (o estímulos de entrada). La simulación nominal utiliza un modelo de retardos Simlog que impone un retardo mínimo de subida y de bajada en todos los tipos individuales de puertas.

La simulación de tiempos tiene en cuenta el efecto de los retardos causados por las interconexiones en la pastilla. Se utiliza un modelo Simlog que asigna un factor de carga a las entradas de cada puerta para indicar la capacitancia asociada a cada una de dichas entradas. Primeramente, se calculan los retardos en el circuito sobre la base de la carga de puertas (factor de carga de salida) asociada con cada señal. A continuación, dichos retardos han de ser modificados por las capacidades de alambrado y resistencias de las interconexiones, cuyos valores se extraen del trazado físico.

Para que los diseñadores puedan analizar los retardos debidos a las conexiones y determinar cómo afectan a los retardos globales antes de realizar el trazado, el Simlog hace una estimación de la longitud de las conexiones que suele basarse en la relación estadística entre el factor de carga admisible para cada salida y dicha longitud de conexiones. Estos cálculos de retardos son forzosamente aproximados, aunque se basan en datos facilitados por los suministradores. Con esta técnica se logra acercarse mucho a los retardos reales, y se evita que aparezcan "sorpresas" en la temporización una vez terminado el trazado.

Simulación final

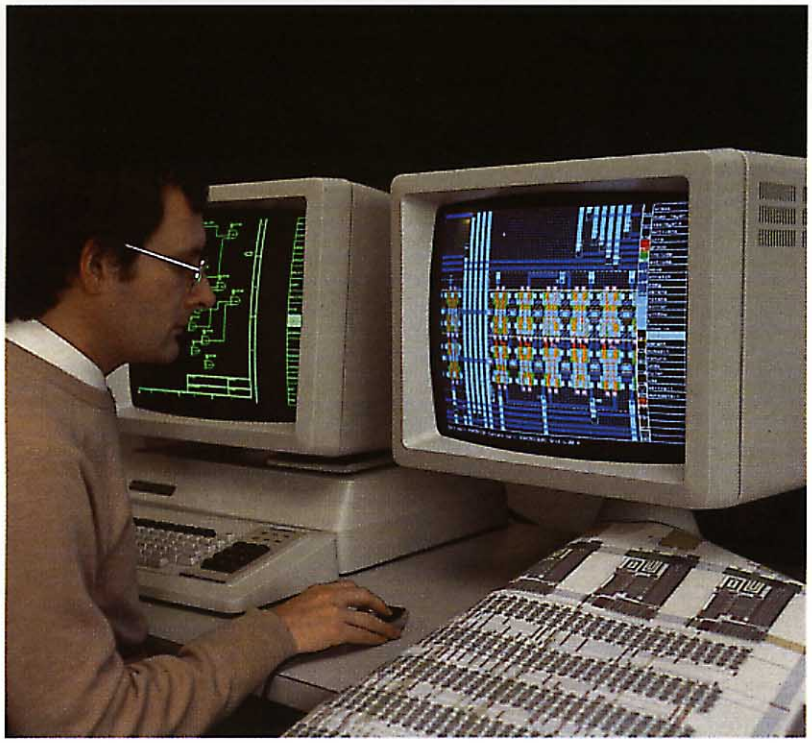
Finalmente los circuitos se simulan en condiciones nominales y extremas del margen de funcionamiento especificado. Dependiendo de la aplicación, el circuito puede destinarse a operar en condiciones "comerciales", "industriales", o "militares". Para cada una de ellas, se deberá simular la operación en el mejor y peor de los casos. Las fórmulas de cálculo de retardos en el caso más adverso implican variaciones en el proceso, la temperatura y la tensión. El ISDS modifica de forma automática los retardos con los factores apropiados que selecciona el diseñador. Las simulaciones de los casos mejor, peor y nominal se realizan automáticamente y se comparan los resultados, debiendo asegurarse el diseñador de que no existen diferencias de temporización entre esas simulaciones.

Desarrollo de las pruebas

Preparación de los patrones de pruebas

Al crear los patrones de prueba para simulación, deberán seguirse las normas que se indican a continuación. Los primeros vectores de prueba de la secuencia han de llevar el circuito a un estado conocido, y dejar estabilizar los efectos sobre los retardos de la lógica combinacional, antes de iniciar la secuencia de pruebas funcionales. Si hay un gran número de vectores de prueba, convendrá dividir la secuencia total de pruebas en varias independientes de menor tamaño. Una buena partición del circuito

Simulación lógica mediante ISDS.



Trazado de un circuito VLSI parcialmente a medida.

permitirá la ejecución separada de las secuencias de prueba de los bloques individuales, comprobando funciones diferentes y grandes secciones del circuito, con lo cual se abrevian la simulación y depuración durante el diseño y se mejora la capacidad de diagnóstico en las pruebas.

Simulación de fallos

En principio, las pruebas de los circuitos lógicos son sencillas: se aplican vectores de prueba a los terminales de entrada y se observan las salidas para identificar los patrones no concordantes con el funcionamiento correcto del dispositivo. Para determinar los vectores de prueba a utilizar, se usa un modelo de fallos en el que las entradas y salidas de las puertas lógicas se fijan a *0* ó a *1*. Este modelo de fallos "preestablecido" tiene un fácil tratamiento por ordenador y además se le puede relacionar con defectos reales de fabricación. El Simlog aporta un análisis de fallos que ayuda a los diseñadores a seleccionar los vectores de prueba que detecten la mayor cantidad posible de faltas. Mediante el citado análisis, se puede probar una parte de los vectores seleccionados aleatoriamente, o bien la totalidad de ellos.

Adicionalmente, el Simlog proporciona un método sencillo para medir la calidad de los vectores de prueba, detectando los nodos actuados, o "movidos", por los vectores aplicados. Aunque este método no sea tan preciso como el de análisis de fallos, ha resultado ser bastante efectivo en la práctica, e incluso más apto para el proceso por ordenador.

Trazado e interfaz con los suministradores

Las facilidades de trazado del ISDS se basan en un sistema comercial, y se utilizan sólo en los diseños de células estándar, donde hay bloques funcionales especiales que pueden necesitar gran cantidad de operaciones manuales. Sin embargo, lo anterior es una excepción más que una norma. Como el trazado forma parte del proceso de fabricación, suele dejarse esta tarea al propio suministrador.

En el caso de ordenaciones de puertas, el diseño se transfiere a los suministradores mediante un lista de conexiones comprobada y una lista de vectores de prueba. El suministrador completa entonces el diseño, incluyendo el trazado y la simulación posterior, fabrica los prototipos, prepara la cinta con el programa de pruebas, y prueba los dispositivos.

En el caso de células estándar en las que el trazado lo hagan los propios diseñadores, el diseño se transfiere al suministrador mediante una base de datos de trazado comprobada y la cinta con el programa de pruebas. El suministrador fabrica entonces los prototipos y los prueba con los datos que se le han entregado.

Selección de la tecnología

Familias lógicas para aplicaciones de telecomunicación

Para diseñar los circuitos parcialmente a medida se requiere un conocimiento profundo de diferentes familias lógicas y capacidad para comparar las prestaciones que pueden conseguirse, tanto en aplicaciones presentes como futuras, con distintas tecnologías de proceso. En la actualidad, cualquier lista de posibles procesos de semiconductores ha de incluir las tecnologías MOSFET, bipolar y de arseniuro de galio. La existencia de varias tecnologías viables indica que no hay solución "universal". La extensa documentación de que dispone el ISDS ayuda a los diseñadores a escoger una tecnología que cumpla los requisitos del circuito. Son factores clave para la elección la disipación (de pico, local y total), la velocidad máxima requerida en las condiciones de carga especificadas, la densidad de encapsulado y el número total de puertas.

En el ISDS se han incluido las siguientes familias lógicas, que se consideran las más importantes para aplicaciones de telecomunicación:

Familias lógicas MOSFET: ordenaciones de puertas y bibliotecas de células estándar CMOS en tecnologías de 2 y 1,5 µm en doble metalización. CMOS es la tecnología normal "de faena" para aplicaciones de baja y media velocidad en los sistemas de conmutación y productos de usuario final, ya que ofrece elevadas densidades de encapsulado y baja disipación.

Familias lógicas bipolares de silicio: ordenaciones de puertas ECL de 2 µm. La tecnología bipolar es la idónea para funciones analógicas de precisión, y es particularmente útil cuando se requiere una transmisión de alta velocidad (de 70 a 140 Mbit/s), como en los circuitos RDSI de banda ancha.

Bibliotecas de células

Las bibliotecas de células de las familias lógicas seleccionadas, se preparan conjuntamente con los suministradores. Los ingenieros del ITT Engineering Support Centre realizan bibliotecas de células en el ISDS que luego aprueban los suministradores antes de ser distribuidas por la red X.25 a todos los centros de diseño ITT. Esta aprobación se basa en circuitos de prueba seleccionados que comprueban que los parámetros de tiempos del ISDS son los especificados en las bibliotecas de células del suministrador, con un error máximo de 0,1 ns.

Como parte del acuerdo con el fabricante, se integran en el ISDS los siguientes datos:

- definiciones de símbolos para esquemáticos
- tablas de verdad
- información completa de retardos para cada componente de la biblioteca en función de la carga, tensión, temperatura, y variaciones del proceso
- tiempos de establecimiento y retención para células específicas (memorias) y restricciones de tiempos para células determinadas, dependiendo de la anchura mínima de impulsos de entrada y salida.
- estimación del retardo imputable al conexionado, así como de la longitud real del mismo después del trazado
- definición sintáctica de los interfaces del listado de conexiones y de los vectores de prueba.

Tabla 1 – Relaciones de costes entre tecnologías de circuito impreso y VLSI

	VLSI	Placa de cto. impreso
Tamaño	810 mm²	36800 mm²
Tiempo de diseño	1,04	1
Coste de producción	72	1
Costes de fabricación (1000 unidades/año durante 4 años)	0,102	1
Total	0,28	1

Suministros para el ISDS

Los aspectos primordiales para la elección de suministradores de circuitos VLSI parcialmente a medida son la calidad de fabricación y los plazos de entrega de los componentes. Tales características sólo pueden alcanzarse cuando se formaliza una relación a largo plazo entre ITT y el suministrador, seleccionado a nivel de toda la Compañía. Antes de llegar a ningún acuerdo, los especialistas del ITTE ESC llevan a cabo un proceso completo de calificación de las instalaciones de los suministradores, es decir, un proceso continuo que incluye la observación y aprobación de los medios utilizados para fabricación de las obleas, metalización, ensamble y pruebas. Además, se discuten y acuerdan con los fabricantes las normas para el diseño y las especificaciones de pruebas.

Los requisitos adicionales de ITT para el suministro de componentes eléctricos estipulan los procedimientos para aprobación de los suministradores, conformidad de la calidad e inspección de entrada.

Análisis de costes

Coste de los componentes de diseño parcialmente a medida

La división del ciclo de diseño en las fases apropiadas permite controlar mejor el proceso de diseño y observar los costes y la productividad del mismo. Sin embargo, el coste del diseño es sólo un elemento del coste total del dispositivo, que debe incluir también el de los prototipos y el coste unitario de cada pieza fabricada.

El coste de diseño comprende la mano de obra de ingeniería y los costes fijos de arranque; en los costes de prototipos se incluyen los de ingeniería no repetitivos y los relativos a herramientas. Los costes de fabricación son suma de los de la pastilla, el encapsulado y la propia cápsula. Normalmente el coste del diseño es el más difícil de determinar, pero ha de calcularse cuidadosamente por ser el factor que más contribuye al coste total.

El coste de mano de obra de ingeniería está en función principalmente de la productividad. Las medidas han demostrado que la productividad alcanzable con el ISDS es notablemente mayor que la obtenida con los métodos anteriores (Fig. 1).

Comparación de costes

La tecnología VLSI es esencialmente una nueva manera de realizar componentes

electrónicos. La tabla 1 muestra la comparación de costes entre el método tradicional a base de componentes estándar en circuito impreso y el mismo circuito diseñado como un componente VLSI. Las relaciones indican claramente dónde radican las ventajas económicas de los diseños parcialmente a medida. Aunque el diseño y el ensamble de los circuitos digitales sea más económico con componentes discretos (aproximadamente un tercio del coste), el coste de fabricación de los circuitos VLSI parcialmente a medida es casi la décima parte del de la placa de circuito impreso, lo cual hace que el coste total de la realización parcialmente a medida sea solamente un tercio del que correspondería al diseño en circuito impreso.

Conclusiones

La versión actual del ISDS es el resultado de la evolución en ITT del diseño de circuitos VLSI parcialmente a medida. Destacan en ella tres características principales: un juego completo de herramientas CAD para captar y verificar el diseño, desde el esquemático hasta el trazado; bibliotecas de células de familias lógicas seleccionadas, preparadas conjuntamente con los suministradores; acceso a avanzados procesos tecnológicos de selectos fabricantes de semiconductores, cuyos medios de fabricación, ensamble y pruebas han sido calificados por ITT. La flexibilidad del ISDS queda demostrada por su uso en el desarrollo de circuitos VLSI en tecnologías CMOS y ECL para aplicaciones tan variadas como centrales digitales del Sistema 12, conmutación RDSI de banda ancha, sistemas de comunicaciones de empresa y televisión por cable.

Este sistema se está utilizando por los centros de diseño de ITT en todo el mundo, para diseñar circuitos VLSI de bajo coste y mediana complejidad, componentes clave en los productos actuales y futuros de ITT.

Referencias

1 A. D. Close, L. Fisher, R. M. McDermott, T. A. Nix, D. M. Perrine y J. M. Schoen: Sistema de diseño para circuitos VLSI parcialmente a medida: *Comunicaciones Eléctricas*, 1984, volumen 58, nº 4, págs. 372–379.

Bora Prazic se graduó por la Universidad de Belgrado, Yugoslavia, en 1974, y obtuvo el MSc en ciencias informáticas dos años después. Trabajó seis años en el Hirst Research Centre de GEC en el desarrollo de herramientas CAD para el diseño de VLSI, y en 1983 ingresó en el ITTE Engineering Support Centre, donde dirige el departamento de investigaciones CAD con responsabilidad especial en el diseño VLSI.

Galileo: modelo, lenguaje y herramientas

La metodología Galileo es un entorno que ayuda a los diseñadores de sistemas y equipos para modelar, analizar y simular fenómenos concurrentes mediante el concepto de red Galileo. En la actualidad se utiliza para facilitar el desarrollo de productos muy diversos.

C. Sánchez Moreno
Standard Eléctrica, S.A., Madrid, España

Introducción

El diseño de los sistemas concurrentes es propenso a errores por varias razones. En particular, estos sistemas son complejos y difíciles de entender debido al gran número de interacciones sutiles y de secuencias inesperadas de funcionamiento que pueden aparecer entre las partes secuenciales de un sistema paralelo. Además, aun siendo relativamente fácil predecir el tiempo empleado en un proceso secuencial, es difícil prever el comportamiento de un sistema concurrente. En realidad, es en la fase de prueba de los sistemas cuando los diseñadores tienen que enfrentarse con problemas tales como lentitud de operación, bloqueos y malas secuencias de funcionamiento, que pueden obligar a rediseños de sistema para lograr un comportamiento aceptable.

Los problemas relacionados con el desarrollo de sistemas concurrentes pueden evitarse en gran parte mediante el entorno Galileo, que facilita el desarrollo de aplicaciones industriales con un alto grado de paralelismo.

La metodología Galileo ayuda a los diseñadores de sistemas de varias formas. Primero, proporciona un conjunto de conceptos y herramientas que les permiten expresar la solución escogida con claridad y precisión (en parte, gráficamente) y, por tanto, documentarla y comunicarla a otras personas. Segundo, incorpora un analizador de redes de Petri que comprueba formalmente los aspectos de coordinación del sistema. Tercero, permite al usuario simular distintas secuencias de funcionamiento y observar cómo se comportaría el modelo completo (con datos y tiempos). De esta forma, los diseñadores reciben impresiones acerca del sistema en las primeras fases del diseño, pudiendo detectar errores y problemas cuando el coste de las modificaciones es todavía relativamente bajo.

La metodología Galileo se ha desarrollado en el centro de investigación de Standard Eléctrica, sobre ideas formuladas por primera vez en 1978[1]. Se construyó y evaluó un prototipo en 1985, y tras varias revisiones se obtuvo una versión producto en 1986, que se está ahora utilizando en varios proyectos.

El modelo Galileo

El modelo Galileo está basado en las redes de Petri. Sin embargo, aunque dichas redes proporcionen una especificación clara y sencilla del flujo de control de actividades, no poseen la suficiente potencia expresiva para describir operaciones que incluyan también datos y temporizaciones. La metodología Galileo viene, pues, a extender el concepto de redes de Petri y hacer posible el especificar cualquier sistema que contenga operaciones con tipos de datos definidos por el usuario, coordinadas secuencialmente o en paralelo. Uno de los requisitos esenciales fue el de mantener todas las posibilidades de análisis formal del modelo original de redes de Petri al añadir los medios necesarios para describir las operaciones con datos y tiempos.

Redes de Petri

Las redes de Petri se han venido utilizando en aplicaciones industriales para modelar sistemas paralelos y concurrentes. Estas redes[2] son grafos finitos con dos clases de nodos, llamados *lugares* y *transiciones*. Los lugares se conectan con las transiciones o viceversa mediante arcos de flechas. Cada lugar contiene un determinado número de *marcas*, cuya distribución en los lugares se denomina *estado* o *marcado* de la red.

Se dice que una transición está *cargada* — esto es, que puede ser *ejecutada* o *disparada* en terminología de las redes de Petri — si todos los lugares hacia los que se dirigen los arcos (*lugares de entrada*) tienen al menos una marca. El disparo de una transición elimina una marca de cada lugar de entrada y la añade en cada lugar de salida.

A partir de un estado inicial, se puede hacer evolucionar una red eligiendo una transición cargada, disparándola y repitiendo el proceso con las marcas resultantes. Si no existen transiciones cargadas, se dice que la red está bloqueada.

Normalmente, un lugar representa un evento, y el número de marcas indica las veces que se repite dicho evento. Las transiciones son acciones que pueden ocurrir tan sólo después de suceder ciertos eventos previos. Una red de Petri representa así un conjunto de evoluciones posibles (secuencias de disparo) de un sistema paralelo.

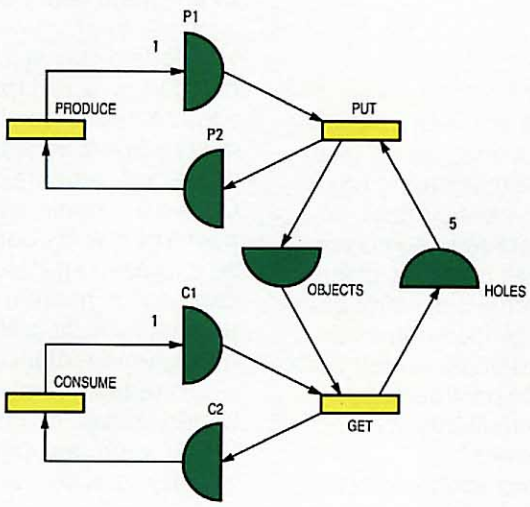


Figura 1
Red de Petri pura de un sistema productor-consumidor (con sintaxis gráfica Galileo). Los componentes gráficos son: cajas para las transiciones, semilunetas para los lugares de control y flechas de línea continua para los arcos de control.

La figura 1 muestra un sistema de dos procesos – *productor* y *consumidor* – que se comunican entre sí a través de un almacén con capacidad para cinco mensajes. Todas las secuencias posibles en la ejecución de las acciones implicadas están representadas por esta sencilla red. La figura ilustra también la forma de expresar redes de Petri en Galileo: los lugares son semilunetas, las transiciones, cajas, y las marcas están indicadas por números enteros. Todo lugar y transición tiene su identificador asociado.

Las redes de Petri son un mecanismo sencillo e intuitivo para describir comportamientos concurrentes y sincronizados, y permiten estudiar propiedades tales como la *acotación* (límite del número de estados), *viveza* (ausencia de bloqueos), y la presencia de conflictos. Sin embargo, tienen algunas limitaciones, siendo la más importante la de no poder describir sistemas cuyo comportamiento dependa de la información generada por un proceso y recibida por otro. El flujo dependiente de datos se representa como un comportamiento no determinista del sistema modelado.

Las redes de Petri puras o clásicas son útiles para describir y estudiar muchos sistemas. El entorno Galileo permite un fácil trazado gráfico, análisis estático y ejecución dinámica de estas redes.

Extensión con datos

Como las redes de Petri no pueden manejar datos y en algunos casos son inadecuadas, se han propuesto numerosas modificaciones, siendo Galileo una de las más sencillas.

La metodología Galileo introduce dos nuevos conceptos para tratar datos. Además de los lugares que contienen marcas (*lugares de control*), Galileo permite que otros tipos de lugar contengan cualquier clase de datos (*lugares de datos*). La carga de una transición puede depender de una condición sobre los datos (*guarda*), y al dispararse (*acción*), modificar los lugares de datos. Una guarda de datos permite que los valores de datos influencien el flujo de control, escogiendo una entre varias posibilidades de dicho flujo. Mediante una acción se asocia el disparo de una transición con la ejecución de una función de datos. Las asociaciones de lugares de datos como argumentos reales de guardas y acciones se denominan *arcos de datos*. A una transición que no tenga guarda ni acción se le llama transición de tipo Petri.

La ejecución de una red Galileo con datos se define como una secuencia de eventos (disparos de transiciones cargadas). Una transición está cargada si las marcas de los lugares de control la cargan, en el sentido de Petri, y su guarda se evalúa como cierta. El disparo de una transición Galileo mueve las marcas según las reglas de Petri y ejecuta la acción. Si hay más de una transición cargada, la elección de la transición a disparar no será determinista. Para una red sin datos esta definición es equivalente a las reglas de evolución de las redes de Petri.

Para describir los tipos y operaciones de datos se necesita un lenguaje. En la metodología Galileo, en vez de definir un nuevo lenguaje específico se utiliza el Pascal estándar. Los lugares pueden ser cualquier tipo de datos Pascal, mientras que las guardas de las transiciones son funciones Pascal booleanas y las acciones de las transiciones, procedimientos Pascal de tipo general.

Utilizando esta extensión, es posible definir con precisión los tipos de datos transferidos en el ejemplo anterior de red de Petri (enteros, caracteres, estructuras, etc.). La figura 2 muestra la red gráfica correspondiente a la figura 1, suponiendo que los datos son caracteres y que el almacén funciona como una pila.

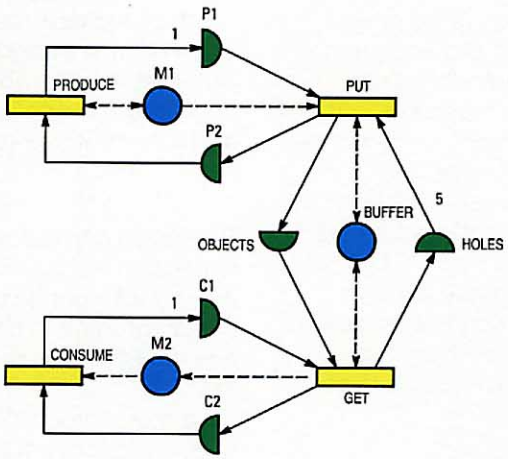


Figura 2
Red Galileo para el sistema productor-consumidor (con datos). Los círculos representan lugares de datos y las flechas con línea discontinua, los arcos de datos.

Extensión con tiempos

Las temporizaciones se incluyen en Galileo para estudiar las prestaciones de los diseños. Con poco esfuerzo (asignando una duración estimada a las transiciones), se pueden obtener datos sobre el comportamiento del sistema en las primeras etapas del desarrollo. Esta información es de gran ayuda para elegir una solución entre varias alternativas posibles. Además, las temporizaciones son necesarias para detectar problemas de acceso simultáneo a datos desde distintas transiciones.

Se introducen temporizaciones en Galileo asignando duración al disparo de las transiciones y dejando que los disparos se solapen en el tiempo (Fig. 3). Se consideran tres clases de duraciones:

- Duración nula: para modelos ajenos al tiempo, o en modelos temporizados que representen actividades de sincronización instantánea.
- Duración fija: para modelar actividades sencillas de duración conocida. Estas duraciones se representan por un entero positivo.
- Duraciones probabilísticas: para modelar actividades complejas o de duración desconocida. Por el momento, sólo se consideran en Galileo distribuciones exponenciales de duración.

Cuando al disparo de una transición se le asigna una duración, es interesante distinguir los instantes inicial y final, con lo que se introduce el concepto de evento. Un evento Galileo puede ser:

- el disparo atómico de una transición no temporizada
- el comienzo del disparo de una transición temporizada
- la terminación del disparo de una transición temporizada.

El modelo Galileo sin tiempos es equivalente a uno con duración cero para todas las transiciones (disparos atómicos).

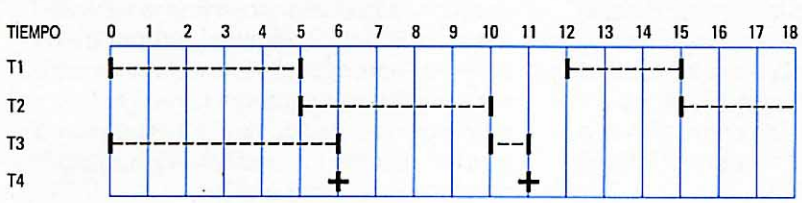


Figura 3
Disparo de transiciones en una red Galileo temporizada.

El lenguaje Galileo

Un modelo Galileo consta en general de dos partes: la red (parte de sincronización) y los componentes funcionales (tipos de datos y operaciones). La red puede expresarse gráficamente con lenguaje gráfico Galileo o textualmente con lenguaje textual, mientras que los componentes funcionales se expresan en Pascal. Aunque es posible expresar un modelo Galileo en forma textual pura (sin utilizar gráficos), en general es mejor representar la red gráficamente.

Las redes de Petri sin datos, incluso las temporizadas, no necesitan descripción Pascal y pueden expresarse totalmente en lenguaje gráfico o textual Galileo.

El lenguaje gráfico Galileo

Las figuras 1 y 2 ilustran la sintaxis gráfica del lenguaje Galileo. Cada nodo (transición o lugar) tiene un identificador. Los lugares de control pueden marcarse con un número entero y positivo. El color, tamaño y posición del identificador de cada elemento gráfico son parámetros a definir por el usuario, aunque en su defecto el sistema proporciona valores sustitutivos. Esta posibilidad permite destacar algunas partes de la red (por ejemplo, los estados u operaciones más importantes). Se puede añadir información textual a cada elemento gráfico (tipo, comentario, etc.) y presentarla a solicitud del usuario.

Una red puede ocupar varias páginas gráficas relacionadas por los lugares de control y datos de igual denominación, como se verá más adelante con el ejemplo del protocolo del bit alternante.

El lenguaje textual Galileo

Las redes Galileo tienen también una expresión textual que puede obtenerse automáticamente a partir de la descripción gráfica mediante el editor gráfico, o bien manualmente con un editor de textos.

La figura 4 muestra la expresión textual para la red de Petri pura de la figura 1 que consta de lugares, transiciones, arcos e

Figura 4
Red de Petri pura del sistema productor-consumidor (con sintaxis textual Galileo).

```
/ * Producer-consumer pure Petri NET * /
GALILEO_NET:pr_co;

PLACES:
        p1: CONTROL;
        p2: CONTROL;
        holes: CONTROL;
        objects: CONTROL;
        c1: CONTROL;
        c2: CONTROL;

TRANSITIONS:
        produce: PETRI;
        put: PETRI;
        get: PETRI;
        consume: PETRI;

ARCS:
        produce[p2/p1];
        put[p1,holes/p2,objects];
        get[objects,c1/holes,c2];
        consume[c2/c1];

MARKING:
        p1 = 1;
        holes = 5;
        c1 = 1;

END_GALILEO_NET;
```

Figura 5
Red Galileo para el sistema productor-consumidor (sintaxis textual).

```
/ * Producer-consumer with data * /
GALILEO_NET:pr_co2;

PLACE_TYPES:
        char;stack;

TRANSITION_TYPES:
        char_next (char INOUT);
        push ($ stack INOUT,char IN);
        pop ($ stack INOUT,char OUT);
        char_none (char IN);

PLACES:
        p1: CONTROL;
        p2: CONTROL;
        holes: CONTROL;
        objects: CONTROL;
        c1: CONTROL;
        c2: CONTROL;
        m1: char;
        m2: char;
        buffer: stack;

TRANSITIONS:
        produce: char_next;
        put: push;
        get: pop;
        consumer: char_none;

ARCS:
        produce (m1) [p2/p1];
        put (buffer, m1) [p1,holes/p2,objects];
        get (buffer, m2) [objects,c1/holes,c2];
        consume (m2) [c2/c1];

MARKING:
        p1 = 1;
        holes = 5;
        c1 = 1;

END_GALILEO_NET;
```

información de marcado. Todos los lugares pertenecen al tipo predefinido *control*, todas las transiciones son de tipo Petri, los arcos se agrupan por transiciones, los *arcos de control* unen las transiciones con los lugares de control y van entre corchetes [[,]]. Los arcos que anteceden a la barra (/) son de entrada, y los que la siguen, de salida.

La figura 5 muestra una red textual con datos (correspondiente a la red gráfica de la figura 2), tal como es generada por el editor gráfico. Se manejan dos tipos de datos, caracteres (char) y pilas (stack), que deben declararse en la sección de *tipos de lugar (place types)*. La sección *tipos de transición (transition types)* incluye el interfaz con todas las funciones Pascal invocadas en la red. La guarda se evalúa sobre los datos precedidos de un signo dólar ($), y luego se declaran los lugares con su tipo correspondiente (control, char ó stack). Las transiciones se declaran también con su tipo: Petri si no manejan datos, o el que se haya declarado previamente en los demás casos. En la sección *arcos (arcs)*, los arcos de control vienen entre corchetes [[,]], utilizando la misma sintaxis que en las redes de Petri. Los arcos de datos representan la asignación de lugares de datos como parámetros reales de las funciones invocadas en la transición (guarda o acción), y figuran entre paréntesis, separados por comas [(,)].

El marcado de datos (valor inicial para los lugares de datos), es una hilera de caracteres entre comillas " " que se interpreta mediante una función definida por el usuario para el tipo de lugar.

Herramientas Galileo

El entorno Galileo consta de tres herramientas principales (Fig. 6) a las que se accede a través de un interfaz guiado por menús. Estas herramientas — editor gráfico, analizador y simulador — se conectan mediante otras herramientas auxiliares.

El editor gráfico permite la edición de redes Galileo, totalmente nuevas o derivadas de otras que ya existan. Produce dos salidas: la red gráfica y la textual. En el resto del entorno sólo se utiliza la representación textual, y si no se dispusiera de terminales gráficos, la red textual se podría obtener directamente con un editor de textos.

A partir de la red textual Galileo se obtiene una red Petri mediante la herramienta auxiliar *Genpetri*, que elimina los lugares de datos de la red Galileo. Se la introduce luego en el analizador de redes de Petri, el cual produce varias clases de resultados: propiedades de la red (viveza, acotación, viveza parcial, repetitividad, etc.), conjunto

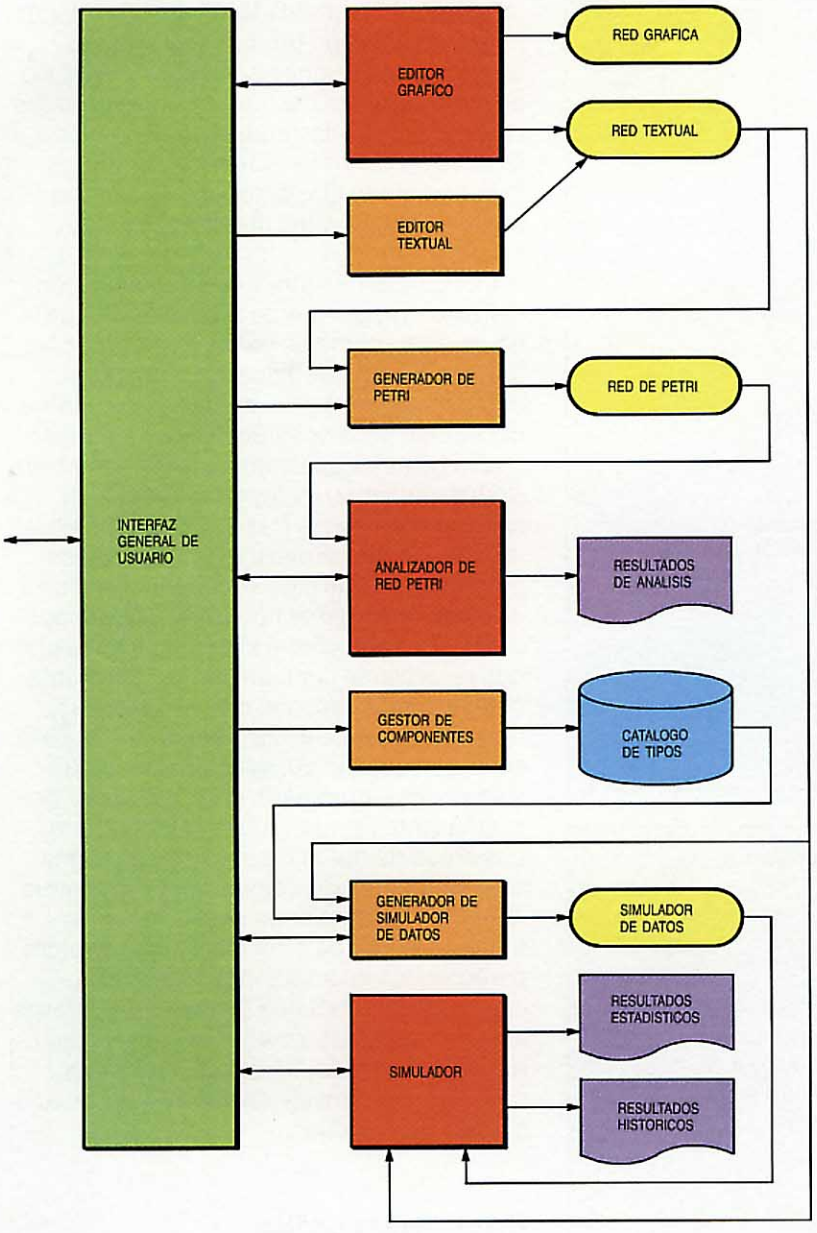


Figura 6
Esquema general del entorno Galileo.

de estados e invariantes (aserciones acerca de los lugares y secuencias de disparo). El analizador sólo considera los aspectos de control o sincronización de la red independientes del tiempo, mientras que para estudiar el comportamiento con los datos y tiempos se utiliza el simulador Galileo.

Para redes sin datos se incluye en el entorno Galileo un simulador de redes de Petri (que incluye el comportamiento variable con el tiempo). Sin embargo, como en las redes Galileo que incluyen datos hay que describir en lenguaje Pascal el modo en que éstos se manejan, se habrá de construir un simulador específico con las necesarias funciones Pascal en cada red Galileo.

La herramienta *gestor de componentes* permite describir nuevos componentes Pascal. El entorno Galileo proporciona un conjunto de componentes predefinidos para manejar los tipos entero, carácter, booleano y real. Es fácil construir nuevos componentes a partir de los ya definidos (tales como registros, matrices). La herramienta *generador del simulador* construye un simulador Galileo a partir de la red y de los componentes Pascal invocados.

Según las reglas establecidas por el usuario, el simulador ejecuta una secuencia de eventos que comprueba la operatividad del modelo y que aporta resultados históricos y estadísticos sobre el comportamiento evaluado. El simulador puede trabajar en modo interactivo o en lotes.

Editor gráfico

El editor gráfico Galileo obedece a sintaxis, es decir, conoce que se está editando una red Galileo y comprueba cada operación. No se permiten operaciones del usuario que contradigan las reglas de la sintaxis gráfica o la semántica del Galileo. El interfaz del usuario está basado en menús en los que puede seleccionarse una de las opciones mediante controles manuales.

Las redes Galileo pueden expresarse en varias páginas conectadas por lugares con un mismo nombre, lo cual facilita el manejo de redes de gran tamaño. La posibilidad de copiar y modificar páginas facilita la edición de redes que tengan partes similares.

El usuario puede cambiar algunas características gráficas de los elementos, como su tamaño y color, o la posición relativa y color de los identificadores. Como los arcos siguen las traslaciones y los cambios de tamaño de los nodos, el usuario no tiene que volver a dibujarlos.

Como ya se ha indicado, la red textual se obtiene automáticamente de la red gráfica por medio del editor gráfico, que está en concordancia con el nivel 2b del estándar gráfico GKS[3] para asegurar la portabilidad entre sistemas diferentes.

Analizador de redes de Petri

El analizador de redes de Petri determina si el modelo tiene alguna de las siguientes propiedades:

- acotado: el número de estados diferentes (marcados) es limitado
- vivo: no se puede alcanzar un estado de bloqueo
- cíclico: la red tiene un comportamiento repetitivo
- conflictivo: el disparo de una transición cargada implica la desactivación de otra
- pseudovivo: la red no llega a bloquearse pero algunas transiciones no pueden ejecutarse
- exclusión mútua, conservación, seguridad, etc.

La red de Petri, que representa la parte de control del modelo, se obtiene de la red Galileo eliminando los datos y los tiempos. Si el flujo de control no depende de los valores de los datos (no existen guardas) y la sincronización de actividades es independiente del tiempo, los resultados del análisis de la red de Petri son enteramente aplicables a la red Galileo original. En caso contrario, algunas de las propiedades de la red (ej., la acotación del número de estados) son muy claras, pero otras, como la viveza, han de estudiarse detenidamente.

El analizador emplea tres técnicas para estudiar estáticamente los aspectos de control de una red Galileo: alcanzabilidad, análisis estructural y reducción. Cada uno de estos métodos es capaz de detectar un conjunto de propiedades diferente.

El análisis de alcanzabilidad se basa en el grafo formado por el conjunto de todos los marcados alcanzables, y en gran medida depende de si la red es acotada o no.

El análisis estructural se lleva a cabo calculando primero los invariantes de estado (S) y y de transición (T), que dan información sobre la conservación y repetitividad de la red. Este análisis depende mucho de la estructura de la red, y no de su estado inicial. Los invariantes proporcionan una visión global muy valiosa para comprobar si el modelo construido se ajusta realmente a la especificación pretendida.

La comprobación de propiedades por el método de reducción se basa en transformar la red en otra con menos lugares o transiciones, que sea más fácil de analizar pero que mantenga las propiedades originales de la red. Este método puede servir también para encontrar modelos más sencillos que eliminen los detalles innecesarios.

El analizador puede operar en modo automático o interactivo. La figura 7 resume los resultados del análisis del ejemplo productor-consumidor dado en la figura 2. La tabla de propiedades de la red ha sido determinada mediante cada uno de los tres métodos de análisis (de reducción, alcanzabilidad, estructural). Con el análisis de alcanzabilidad se han encontrado 24 estados sin bloqueos ni conflictos en la red, mientras que con el estructural se han hallado tres invariantes S y uno T.

Los invariantes S muestran las relaciones entre las marcas de los lugares de la red. En el ejemplo, el primer invariante S (holes + objects = 5) establece que el número de huecos (*holes*) y objetos (*objects*) en el almacén suma siempre cinco. Los invariantes S segundo y tercero indican que las transiciones *get* y *consume* se ejecutan secuencialmente, y lo mismo sucede con las transiciones *produce* y *put*.

ANALYSIS STATISTICS

Number of elemental S-invariants : 3
Number of elemental T-invariants : 1
Bound of the net : 5
Number of mutual exclusions : 4

ELEMENTAL NON-NEGATIVE S-INVARIANTS

1) holes + objects = 5
2) c1 + c2 = 1
3) p1 + p2 = 1

TABLE OF BOUNDS OF THE NET PLACES

Place	Bound	Place	Bound	Place	Bound
p1	1	p2	1	holes	5
objects	5	c1	1	c2	1

ELEMENTAL NON-NEGATIVE T-INVARIANTS

1) produce + put + get + consume

NET PROPERTY TABLE

PROPERTY	REDUCTION	REACHABILITY	STRUCTURAL
Bounded	YES	YES	YES
BINARY		NO	
STR. BOUNDED	YES	YES	YES
CONSERVATIVE	YES	YES	YES
LIVE	YES	YES	
PSEUDOLIVE			
STR. LIVE	YES	YES	
BLOCKED	NO	NO	
S-INVARIANT			YES
EXCLUSIONS		YES	YES
STR. CONFLICT			NO
EFFECTIVE CONFLICTS		NO	
CYCLIC		YES	
REPETITIVE	YES	YES	YES
T-INVARIANT			YES

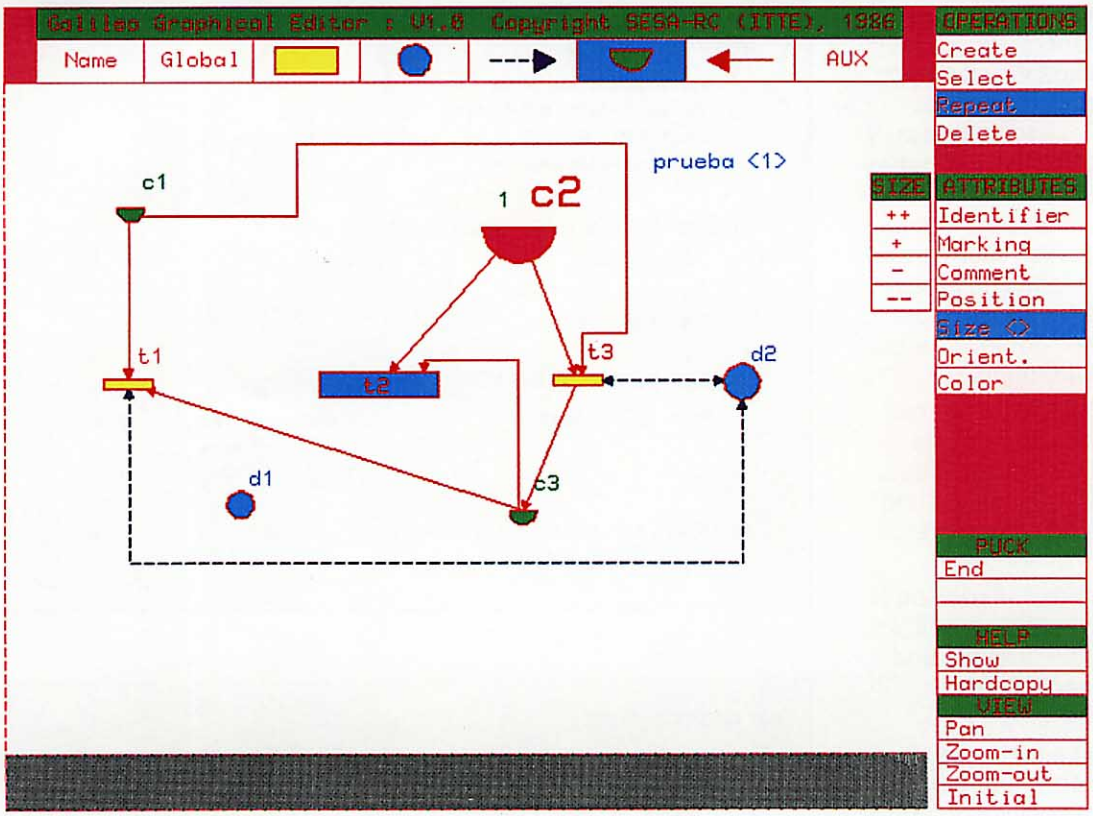
ANALYSIS STATISTICS

Number of reachable markings : 24
Number of effective conflicts : 0
Number of blocked markings : 0

NET without BLOCKED MARKINGS
NET without EFFECTIVE CONFLICTS

Figura 7
Resultados del análisis.

Los invariantes T muestran las secuencias de disparo de transiciones que no alteran el estado de la red. En el ejemplo, la secuencia: *produce*, *put*, *get* y *consume* deja la red en el mismo estado en que estaba.



Pantalla de trabajo durante la edición de una red Galileo.

El simulador Galileo

El simulador Galileo tiene dos objetivos: la validación del diseño (ejecutando la red en circunstancias diferentes y contrastando los resultados), y ayudar a la depuración de la red en el caso de que los resultados no sean los esperados. El simulador, trabajando en modo interactivo o en lotes, ejecuta una secuencia de eventos particular para una red determinada y en unas circunstancias específicas (política de resolución no determinista, marcado inicial, duración de las transiciones), y ofrece dos clases de resultados:

- Resultados históricos, que exponen una secuencia de ciertos eventos, pudiendo el usuario seleccionar la información asociada con cada evento. Se puede observar la ejecución de la red paso a paso, aunque el usuario sólo tenga acceso a aspectos concretos del funcionamiento de la red. La historia se puede registrar en un fichero o presentar en la pantalla del terminal.
- Resultados estadísticos, que recogen parámetros globales de la simulación, tales como el número de disparos de una transición, tiempo consumido, tiempo entre dos activaciones, conflictos, etc. Para algunos de estos parámetros se dan los valores máximo, mínimo y medio.

Con el fin de resolver el no determinismo de la red se ofrecen dos estrategias: una manual, en la que el usuario debe escoger uno entre varios eventos que estén dispuestos para ejecutarse, y otra aleatoria (opción sustitutiva). En el futuro se añadirán otras directrices, tales como la fijación de prioridades y el menor tiempo.

El simulador obedece a un lenguaje de órdenes, por el cual el usuario podrá elegir una determinada política que resuelva el no determinismo, cambie el marcado de los lugares o la duración de las transiciones, revise el estado de la red, seleccione los resultados a obtener, etcétera.

Ejemplo de utilización

El protocolo del bit alternante[4] sirve para ilustrar cómo se utiliza la metodología Galileo. Dos entidades se interconectan a través de un medio de transmisión no fiable para enviar mensajes desde un emisor hacia un receptor; dichos mensajes así como los reconocimientos asociados pueden recibirse correctamente, perderse o dañarse. Los mensajes se envían con un número de secuencia de un bit, y antes de enviar el siguiente mensaje se ha de recibir un reconocimiento con el mismo número de secuencia.

El receptor reconoce todos los mensajes que recibe, transmitiendo un reconocimiento con el mismo número de secuencia que lleva el mensaje. Si el reconocimiento

llega dañado o con un número de secuencia distinto, o si no se recibe en un tiempo determinado, el emisor supone que ha habido un fallo del medio de transmisión y vuelve a transmitir el último mensaje. Cuando se recibe un mensaje dañado, se supone un fallo del medio y se transmite un reconocimiento con el número de secuencia opuesto, de forma que el mensaje pueda retransmitirse sin esperar a que finalice la temporización correspondiente.

La figura 8 muestra una red gráfica Galileo que modela el protocolo del bit alternante. La red se ha dividido en tres páginas que representan el emisor, el medio y el receptor.

El emisor (Fig. 8a) tiene cuatro estados principales representados por los lugares *sendm0* (listo para enviar el mensaje 0), *wack0* (a la espera del reconocimiento 0), *sendm1* (listo para enviar el mensaje 1) y *wack1* (a la espera de reconocimiento 1). Los lugares *m0*, *ack0*, *m1* y *ack1* representan los mensajes y reconocimientos con números de secuencia 0 y 1. También aparecen en las páginas del medio de transmisión y del receptor.

Con un medio fiable, el emisor sigue la secuencia de disparos *sm0*, *rack0*, *sm1* y *rack1*. Las pérdidas y degradaciones de mensajes ocasionadas por el medio de transmisión se representan en la página gráfica del medio (Fig. 8b). Cuando el emisor está a la espera de *ack0*, se puede presentar una de las siguientes situaciones:

- *ack0* se recibe correctamente (transición *rack0*)
- se ha perdido el mensaje o su reconocimiento y expira la temporización (transición *t_out0*)
- se recibe un reconocimiento dañado (*wrong_ack*), según transición *ack_w0*
- se recibe *ack1* en vez de *ack0* por haber llegado al receptor un mensaje dañado (transición *r0ack1*).

Lo mismo puede suceder cuando se espera recibir *ack1*. El receptor (Fig. 8c) tiene dos estados representados por los lugares *recm0* (a la espera del mensaje 0) y *recm1* (a la espera del mensaje 1). Cuando el receptor se encuentra en el estado *recm0* puede ocurrir una de estas tres cosas:

- se recibe correctamente el mensaje con número de secuencia 0 (transición *rm0*)
- se recibe un mensaje con número de secuencia 1 debido a la pérdida de un reconocimiento anterior (transición *opp1*)
- se recibe un mensaje dañado (transición *m0_wrong*).

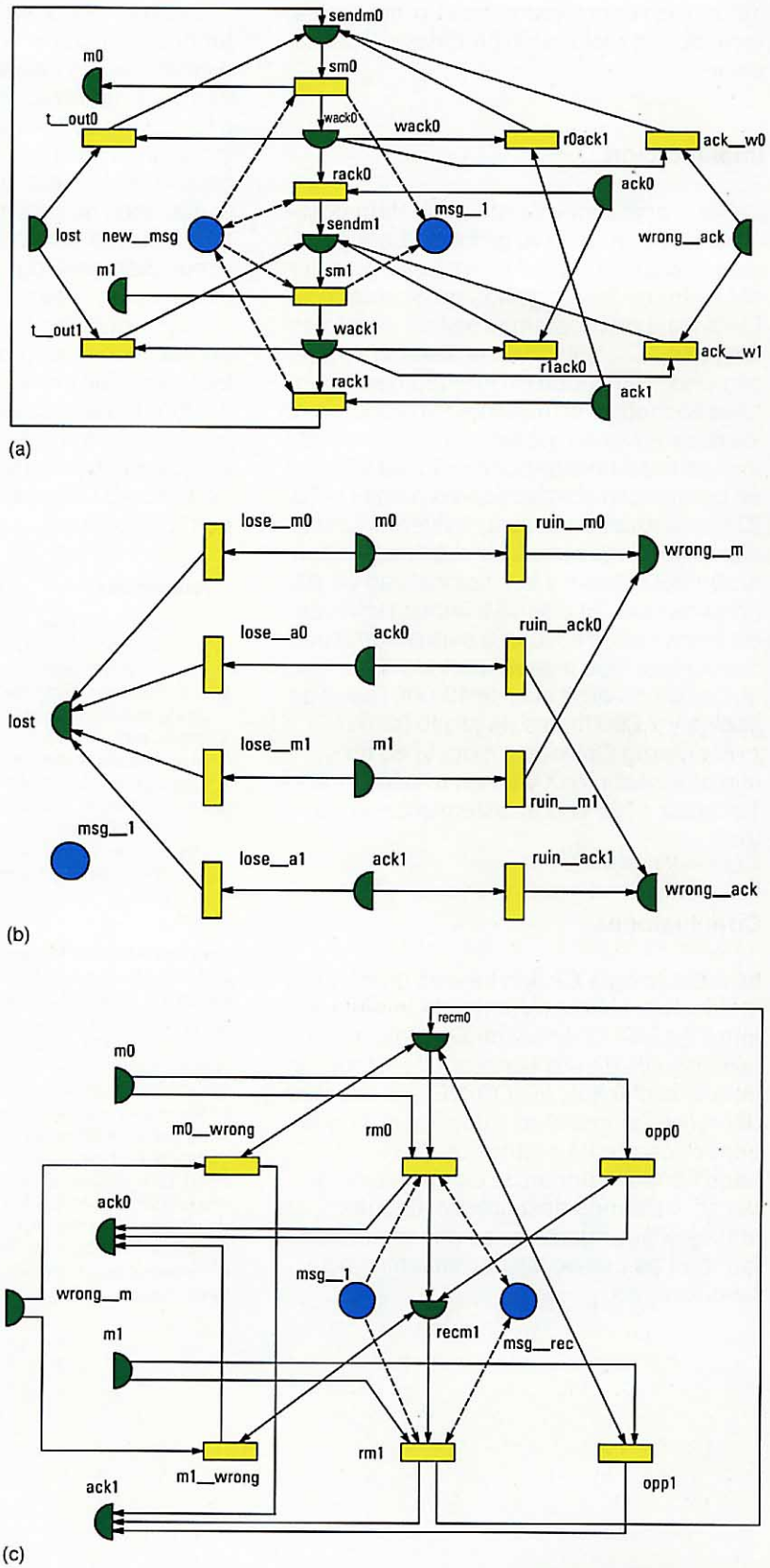


Figura 8
Protocolo del bit alternante (a) emisor, (b) medio de transmisión y (c) receptor.

Lo mismo podría suceder cuando se espera el mensaje con número de secuencia 1.

Todo lo anterior está representado en la red, la cual puede analizarse para ver si tiene todas las propiedades requeridas (ausencia de bloqueos, acotación de estados, repetitividad). Los lugares de datos se

introducen para mostrar que los mensajes enviados se reciben sin pérdidas ni duplicaciones.

Implantación

Salvo el analizador de redes de Petri desarrollado por la Universidad de Zaragoza, todo el entorno Galileo ha sido elaborado en el Centro de Investigación de Standard Eléctrica. Los programas se han escrito en lenguaje "C", utilizando el sistema operativo Unix. Siguiendo un método basado en tipos abstractos de datos se han subdividido los programas en módulos, probados cada uno separadamente por medio del sistema de prueba con abstracción de datos DATS (Data Abstraction Testing System). Las especificaciones iniciales cubrieron la definición del modelo y la funcionalidad de las herramientas. El diseño e implantación de las herramientas ocupó a seis programadores durante seis meses, periodo durante el que se produjeron más de 48.000 líneas de código y 7.000 líneas de pruebas.

El entorno Galileo se ejecuta en un miniordenador VAX y en un terminal gráfico Tektronix 4109 con el sistema operativo VMS.

Conclusiones

La metodología Galileo se está utilizando en muchos tipos diferentes de proyectos, entre los que se encuentra la radio móvil celular (para ajustar las especificaciones y derivar el diseño), el LORINE (para evaluar diferentes arquitecturas de circuitos), y la especificación de protocolos. Se está adquiriendo experiencia en aplicaciones reales, habiendo descubierto algunas ampliaciones interesantes que mejorarán la facilidad de uso de las herramientas o su funcionalidad.

La experiencia también ha indicado varias formas de mejorar el modelo Galileo. Por ejemplo, es posible añadir nuevas funciones de temporización (otras distribuciones o funciones definidas por el usuario) que se incluirán en las prioridades del simulador, así como otras estrategias para resolver las situaciones no deterministas. También se podría proporcionar una salida gráfica de la simulación para lograr una presentación dinámica de la red.

La jerarquía en la parte de datos se obtiene con el lenguaje Pascal, pero también sería útil incluir una jerarquía de redes y subredes para la parte de coordinación. Finalmente, sería posible utilizar otros lenguajes de programación (CHILL, C, Fortran, ADTS) para expresar las operaciones funcionales.

Referencias

1 F. Vidondo, I. López y J. J. Girod: Galileo: método para el diseño de sistemas: *Comunicaciones Eléctricas*, 1980, volumen 55, nº 4, págs. 364–371.
2 J. L. Peterson: Petri Net Theory and the Modeling of Systems: *Prentice-Hall*, 1981.
3 Information Processing Systems–Computer Graphics–Graphical Kernel System (GKS) Functional Description: *ISO-7942-1985 (E)*.
4 K. A. Barlet, R. A. Scantleburry y P. Y. Wilkinson: A note on Reliable Full-Duplex Transmission Over Half-Duplex Links: *Communications of the Association of Computing Machinery*, mayo 1969, volumen 12, nº 5.

Cristina Sánchez Moreno nació en 1950. Se graduó en ciencias físicas en 1972 y ese mismo año entró en el Centro de Investigación y Desarrollo de Standard Eléctrica, donde trabajó hasta 1982. Durante ese tiempo estuvo implicada en una amplia variedad de proyectos, que incluían un equipo para prueba automática de componentes de centrales telefónicas, un simulador de entorno para prueba de programas concurrentes complejos, un generador de llamadas telefónicas y una central telefónica digital distribuida. Después de cortos periodos en Icuatro S.A. dirigiendo el departamento de programación y en ERIA S.A. como promotora del grupo de inteligencia artificial, Cristina Sánchez volvió al Centro de Investigación de SESA donde es responsable de la definición e implantación del lenguaje Galileo y sus herramientas.

Sistema de información de componentes de ITT

La complejidad y cantidad de los datos de componentes necesarios para asegurar el desarrollo eficaz de los productos ha provocado la creación de una base de datos corporativa de ITT. El ICIS es una base de datos potente que proporciona completa información para ayudar a la aplicación y aprovisionamiento de componentes.

J. J. F. Cunningham
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

Los componentes eléctricos duplican su complejidad cada 18 meses, y su vida útil desde las primeras muestras hasta su desaparición puede quedar reducida a tres o cuatro años. El seguimiento de esta situación siempre cambiante es cada vez más difícil y fatigoso. A esto se añade la necesidad de ITT de coordinar las actividades de ingeniería entre casas de diferentes países. Todo ello obliga a recopilar, evaluar y publicar con rapidez gran cantidad de datos sobre componentes para asegurar la disponibilidad inmediata de la información crítica, y para aprovechar al máximo los recursos de ingeniería y los resultados posibles. Para lograr estos objetivos, el ITTE ESC (ITT Europe Engineering Support Centre) ha sustituido la distribución de más de 50 informes diferentes en papel a más de 200 destinatarios, por una base de datos activa informatizada.

El ICIS, sistema de información de componentes de ITT, es la base de datos de la información de componentes eléctricos de ITT centralizada para todo el mundo. El programa de dicha base y el contenido de datos sobre componentes ocupan 400 M-octetos aproximadamente, para unos 30.000 códigos ITT y 60.000 tipos de componentes de diversos fabricantes. Se tiene acceso a ella en todo el ámbito de ITT.

El ICIS presenta al usuario un interfaz sencillo, que necesita muy poco adiestramiento.

Aplicaciones

En el desarrollo de un producto, cuanto antes se haga una selección adecuada de los componentes, menos problemas surgirán después. Uno de los objetivos clave de una base de datos de componentes es, pues, el de mirar hacia el futuro y orientar a los diseñadores hacia los componentes, familias, tecnologías y suministradores autorizados o preferidos. Esta es sin duda una función del ICIS.

Además, el ICIS proporciona información para ayudar en la aplicación de componentes. Por ejemplo, se van pronto a ampliar de un modo apreciable las referencias a la disponibilidad de modelos CAD, se incluyen advertencias de aplicación para evitar posibles problemas, y se identifica la información relevante (junto con su estado). La facilidad de pregunta permite buscar cualquier combinación de parámetros, posibilitando la rápida selección de componentes para una aplicación determinada con base en los parámetros y funciones requeridos (ver ejemplo de la figura 1), así como hallar los códigos equivalentes.

En el ejemplo, el usuario pide al ICIS la lista de suministradores aprobados para memorias EPROM de 32K x 8 y 300 ns. El

programa busca en la primera tabla ese tipo de memorias y, cuando lo localiza, lee el código ITT asociado. Después el programa explora la segunda tabla buscando dicho código ITT, y si le corresponde un "aprobado". Finalmente se obtiene la información relativa al suministrador.

El siguiente paso del ciclo de desarrollo es la producción de la lista de códigos. Generalmente hace falta añadir el código ITT, la descripción normalizada ITT, y otros datos a la lista de códigos proporcionada por el diseñador del circuito. El ICIS es una herramienta ideal para este propósito; suprime el trabajo engorroso, asociando estos datos con los números suministrados por el diseñador, y produce un informe de salida en formato normal o especial.

El departamento de compras puede luego extraer el informe que precisa añadiendo nuevas columnas de códigos de suministradores equivalentes, su estado de aprobación, el número de referencia de compra, etc.

Iniciada la fabricación del producto, los directores de prueba y calidad se pueden dirigir al ICIS para decidir si se requiere inspección de entrada o no para los componentes. No sólo se dispone de todos los resultados de inspección de entrada de todas las compañías ITT, sino también de todas las decisiones tomadas por otras casas sobre hacer una inspección al 100%, un muestreo, o bien un "salto de lote" (probar, por ejemplo, uno de cada cinco lotes), siguiendo el "programa de mejora de la calidad de suministradores". El responsable devuelve su decisión al ICIS para construir la base de datos centralizada y ayudar a otra compañía que afronte una decisión similar.

Se llega así a la parte más estable del ciclo de vida del producto, pero cualquier problema que se presente con un componente, a menudo imputable a una reducción de costes por el suministrador, se puede difundir con rapidez dentro de ITT mediante el sistema de *alerta roja* del ICIS. En la mayoría de los casos, el departamento de ingeniería de componentes del ESC está ya prevenido del cambio por su estrecha relación con los principales suministradores, y los datos que obtiene en forma confidencial. Por ejemplo, para todos los circuitos integrados clave, ESC consigue de los suministradores una declaración sobre el estado en que se encuentra el diseño, fabricación y prueba del componente. Tal *declaración de diseño y prueba* proporciona un perfil del componente, mientras que un fichero lateral describe los planes del suministrador respecto a cualquier cambio dentro de los seis meses siguientes. Esta suele ser la primera advertencia de que va a desecharse ese componente, con lo cual el ICIS podrá identificar un posible sustituto y cuantificar los riesgos asociados.

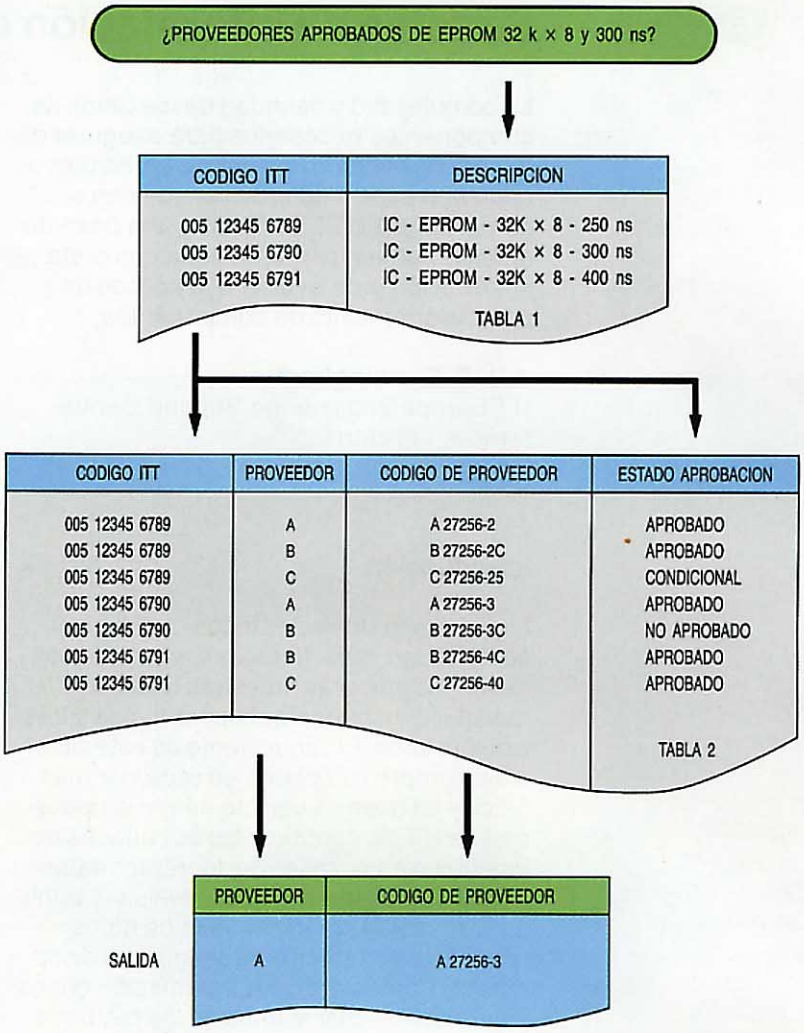


Figura 1
Ejemplo de consulta mostrando el uso del ICIS para obtener datos de un componente. Los datos contenidos en esta figura son ficticios.

Objetivos de desarrollo

Desde la idea inicial en la que una base de datos electrónica pretendía esencialmente sustituir al papel, hasta el presente desarrollo del ICIS como sistema elaborado, se han perseguido una serie de objetivos y requisitos:

- ofrecer a las casas ITT de todo el mundo acceso a la información de componentes más reciente
- eliminar la duplicación y posibles contradicciones en los datos
- combinar todas las fuentes de datos de componentes en un solo lugar
- permitir a los usuarios publicar y actualizar sus propios datos, impidiendo que otros los cambien
- asegurar la comodidad de uso del ICIS

- facilitar la asignación y verificación de códigos ITT
- ofrecer un medio de pregunta/respuesta
- generar informes adaptados al usuario
- validar los datos de entrada
- proporcionar servicio económico y respuesta en corto tiempo
- mantener un registro de los cambios
- utilizar monitor
- evitar abreviaturas y "jergas".

Aunque ya es un sistema de producción en uso, el ICIS será continuamente mejorado tomando estos objetivos como guía de futuros desarrollos.

Estructura de la base de datos

Programa

El programa consta de procedimientos para el interfaz con el usuario y para la gestión de la información (básicamente rutinas de entrada y salida). El interfaz de usuario es un sistema interactivo basado en unos 300 paneles de diálogo del sistema de gestión, desarrollados e implantados por el ESC. Los datos introducidos en estos paneles (menús) se pasan a los esquemas de los lenguajes de control de tareas, que ejecutan los programas de entrada/salida por lotes. Estos programas de entrada/salida son modificaciones o mejoras, realizadas sobre once módulos de programa SPITBOL, originariamente escritos por el International Telecommunications Center de ITT y todavía utilizados para planificación e información sobre el equipo del Sistema 12. Los recientes módulos de programa adicionales (uno para la generación de informes y otro para enlace con el sistema soporte de control de configuración que ahora está en desarrollo) se han codificado en PL/1 por los programadores del ESC.

Tablas de datos

El programa es la implantación de una base de datos relacional que conecta más de 120 tablas de datos, cada una de las cuales es un conjunto de registros y campos (filas y columnas). Las tablas quedan enlazadas, en el momento de la entrada y salida de datos, por la concordancia (igualdad) de los valores de los datos en determinados campos de las diferentes tablas. Por sencillez, este enlace se ha hecho transparente para el usuario e intencionadamente limitado a determinados campos, como el código ITT y el código de fabricante. De esta forma, la información relativa a un código ITT (p. ej., la descripción del componente) se almacena una sola vez en la base de datos, en una tabla en la que ese código ITT también figura sólo una vez. Si son varios los proveedores que suministran componentes conformes a los requisitos de dicho código ITT (especificación de detalle), entonces sus respectivos códigos y datos asociados (p. ej., el estado de aprobación) se almacenan una sola vez en una tabla distinta. Cuando se pregunta qué fabricantes están aprobados para una función determinada, se enlazan los ficheros – de modo transparente al usuario – y se ofrece la respuesta combinada.

Respuesta a una pregunta típica.

```
BROWSE - ESCICIS.REPRTLIB.ESC016(JEZDEMO) - 01.01 -- LINE 000000 COL  001 080
COMMAND ===> _                                          SCROLL ===> CSR
******************************** TOP OF DATA ********************************
ICIS SINGLE FILE REPORT GENERATOR

PRODUCED BY: ESC016 ON 86/10/13 AT 14:22
PART NUMBER DATA
I/C LSI
EXAMPLE
-------------------------------------------------------------------------
 I    DESCRIPTION    I  CONFIGURATION  I VEN TYPE NUMBER IITT PART NUMBERI
-------------------------------------------------------------------------
$ I/C-LSI-EPROM            32KX8          27256          005 62624 0022
-------------------------------------------------------------------------

                END OF REPORT
******************************** BOTTOM OF DATA ********************************
                                                                  02,15
```

Alerta roja.

```
BROWSE - ESCCIS.CIS.REDFLAGS(EXAMPLE) - 01.00 ------ LINE 000001 COL  001 071
COMMAND ===> _                                          SCROLL ===> CSR
-------------------------------------------------------------------------
|       I T T   S Y S T E M   C O N F I D E N T I A L                    |
|       ---------------------------------------                          |
|                                                                         |
| R E D   F L A G                    ITT EUROPE                           |
| D E V I C E     A L E R T          ENGINEERING SUPPORT CENTRE           |
| N U M B E R:    EXAMPLE                                                 |
| *************************                                           |
|                                                                         |
| Date of Alert  : 280586            Valid Until : Withdrawn              |
| Source of Alert: A.N.Other - SRT Stockholm                              |
| Nature of Alert: Quality Problem                                        |
|                                                                         |
| Manufacturer   : EXAMPLE           Type Number :ABC 1234                |
| Die Revision   : B                 Batch Code  :8603                    |
| Package        : CERAMIC CHIP CARRIER                                   |
|                                                                         |
| This Red Flag Device Alert Also affects the following                   |
| variants/other devices: CERDIP DUAL IN LINE                             |
|                                                                         |
|                                                                         |
| ITT Part Numbers Affected:  005 12345 XXXX                              |
|                                                                         |
|                                                                         |
| Details of Alert:                                                       |
| ----------------                                                        |
|                                                                         |
|                                                                         |
                                                                  02,15
```

Interfaz con el usuario
El ICIS lo pueden utilizar usuarios de todo tipo, que en su mayoría no son expertos en informática ni están familiarizados con los sistemas operativos de ordenadores. Por consiguiente, aunque se requiera algún conocimiento para enlazarse al sistema y acceder al ICIS, el manejo debe ser fácil. Para lograrlo, se necesitaba hacer una base de datos de uso cómodo, pero no tanto que pronto llegara a ser aburrida.

Gobierno por menús
Los usuarios van de una parte a otra del sistema seleccionando opciones de los distintos menús. Cuando el usuario está familiarizado con las opciones, puede ir directamente al menú o función deseada sin más que introducir una cadena de números de opciones. Los menús tienen un formato uniforme que se ha desarrollado con ayuda del Centro de Investigación de Factores Humanos del ESC. Las principales características son:

- La parte central superior de la pantalla identifica la situación actual dentro de la base de datos: *dónde estamos*.
- El área superior izquierda recuerda al usuario las oportunas teclas de función programables, y el modo de regresar al menú anterior: *cómo salirse*.
- En el área principal de la pantalla se relacionan las opciones que pueden seguirse (*cómo continuar*), bien sea sobreiluminadas o en otro color; las que no se destacan así, son de acceso restringido o aún no están disponibles.

Cuando deba introducirse un dato, se emplea siempre el signo ◊. Si la pregunta está sobreiluminada, el campo será obligatorio, y en caso contrario, opcional.

Menú de selección típico.

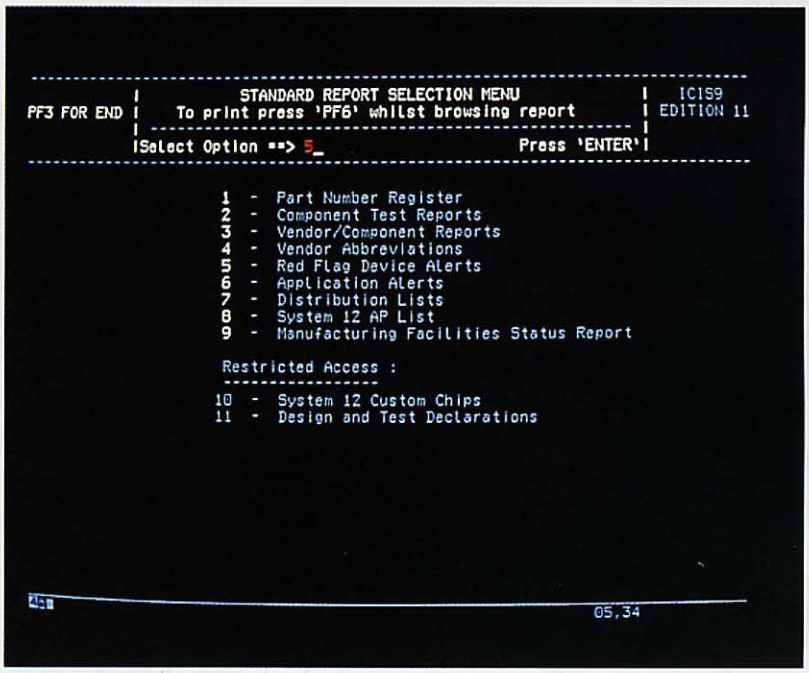


Tabla 1 – Preguntas típicas que puede resolver ICIS

¿Qué es el 005 ITT 62624 0021?
****
¿Cuál es el tipo preferido de condensadores de tántalo para montaje superficial?
****
¿Ha evaluado alguien el Intel 80386?
****
¿Existe especificación para el 80C51?
****
¿Debo emplear resistencias del 1% o del 2%?
****
¿Quién fabrica una RAM estática de 2K x 8, con respuesta de menos de 50 ns?
****
¿Quién más emplea ordenación de puertas Motorola?
****
¿Ha desarrollado alguien un conmutador MIC LSI con posibilidad de conferencias incorporada?
****
¿Cuáles son los suministradores preferidos para el codec/filtro empleado en el BCS ITT 5200?
****
¿Cuál es el mejor marcador multifrecuencia para un nuevo aparato de abonado?

Medios de consulta/generador de informes
La característica más complicada, pero sin duda más eficiente del ICIS, es la de consulta, que permite al usuario buscar en la base de datos respuesta a una pregunta, o bien compilar y preparar un informe hecho a la medida para examinar, archivar o imprimir. Un sistema de base de datos suele proporcionar estas facilidades mediante un interfaz especial de lenguaje, aunque ello requiere capacitación y experiencia. Las bases de datos recientes para ordenadores personales ofrecen interfaces guiados por menús, y los programas de ordenadores centrales empiezan a seguir ese camino. El interfaz dirigido por menús que se ha desarrollado para el ICIS es especialmente elegante, y de uso muy rápido para una pregunta sencilla, utilizando al máximo las opciones sustitutivas. Cuando se requieran informes más complejos, se presentarán nuevos menús que permitan una mayor gama de elección del control de formato. Los informes normales que utilicen los formatos establecidos

por la administración de ICIS, pueden crearse mediante simple selección en el menú.

El interfaz que actualmente se emplea es el resultado de muchas pruebas de usuario, impresiones recogidas, modificaciones y desarrollos en el ESC durante un periodo próximo a un año.

Integridad de los datos

Condición previa del uso de un ordenador para buscar en una base de datos es la de asegurar que los datos se introduzcan y almacenen de modo coherente. Esto tiene especial importancia cuando se introducen datos textuales, ya que una diferencia mínima en los caracteres, espaciado, puntuación, mayúsculas, etc. imposibilita la correcta identificación de un dato buscado.

El ICIS evita este problema empleando para la entrada de datos un interfaz guiado por menús capaz de resistir casi cualquier disparate. Por ejemplo, la descripción de componentes (24 caracteres de texto) se construye pieza a pieza a través de una serie de menús jerárquicos. Análogamente, los parámetros técnicos se formatan en la pantalla de entrada para asegurar que las abreviaturas empleadas son coherentes (p. ej., MHz, no Mhz, ni Mz), y que los datos introducidos son del tipo correcto (numéricos, por ejemplo).

En una base de datos relacional es igualmente importante guardar una relación coherente entre datos de las diferentes tablas. La *integridad referencial*, así denominada, se asegura también mediante el programa de entrada. Aunque los datos relacionados se introduzcan a través del mismo menú, se colocan en diferentes tablas; todas las referencias a otras tablas se comprueban luego para asegurar su aceptabilidad. Así, pues, un código de proveedor no se puede añadir a una tabla si no existe ese componente en la tabla de códigos ITT. El programa se puede configurar, bien para que añada ese elemento a la tabla, o bien para que rechace la transacción con el mensaje de error apropiado.

Estas consideraciones son cada vez más importantes para bases de datos comunales, como el ICIS, en la que usuarios remotos añaden nuevos datos o hacen consultas. La carga de los datos y la generación de preguntas/informes pueden realizarse de modo seguro con el mínimo adiestramiento. Todas las transacciones y errores se registran enteramente para asegurar la integridad de los datos, y permitir la localización de los errores.

Acceso

La base de datos reside en un ordenador central IBM en el ESC de Harlow. Cualquier lugar conectado con la red Datanet de ITT puede acceder al ICIS, y obtener una identificación de usuario. Los usuarios en lugares no conectados a Datanet pueden acceder al ICIS a través de un sistema internacional de conmutación de paquetes. Hay un paquete de emulación de terminal merced al cual un usuario provisto de ordenador personal ITT XTRA* y de módem puede conectarse al ICIS por la red internacional de conmutación de paquetes. Es posible también hacerlo utilizando otros terminales o una estación de trabajo del tipo de ingeniería.

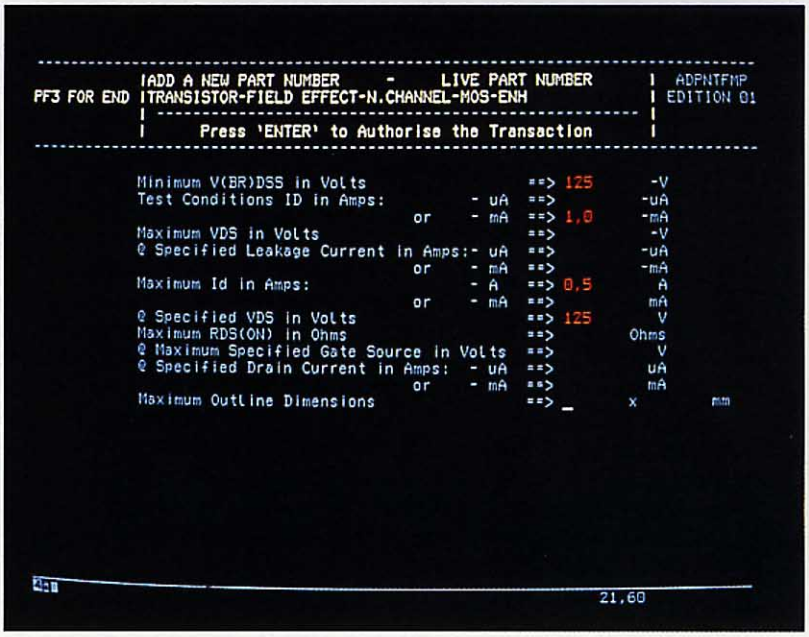
Seguridad

Como la mayoría de los datos incluidos en el ICIS están clasificados como "Confidenciales del Sistema", es preciso que todos los usuarios sufran un examen de seguridad para que se les asigne una identificación individual y una contraseña. También deben ser autorizados por sus superiores para emplear un código contable.

Algunos de los datos del ICIS los proporcionan los fabricantes de componentes bajo el compromiso de que no sean revelados. Tales datos sólo se entregan a una o dos personas autorizadas por cada lugar, o a quienes realmente necesiten saberlos. Análogamente, cierta información de aplicación específica del producto se restringe a las compañías participantes.

Ninguna compañía ITT que fabrique componentes tiene acceso al ICIS.

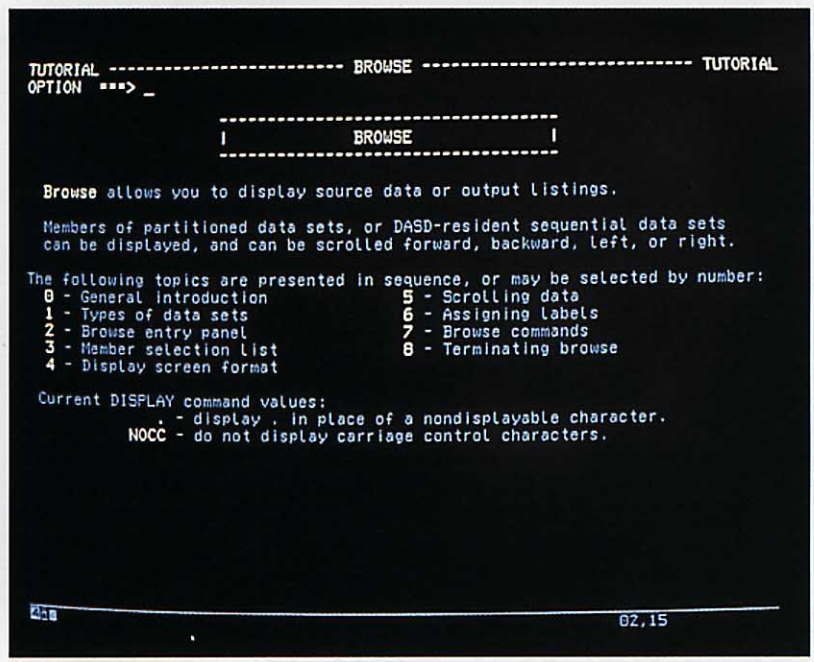
Entrada de datos técnicos al ICIS.



* Marca registrada del Sistema ITT

Control del programa
Para mantenerse en línea con la rápida evolución del entorno en que se mueve ITT, se precisan cambios en la estructura, los programas y el interfaz con el usuario de la base de datos. Según los usuarios descubren las posibilidades del ICIS, van solicitando nuevas aplicaciones. Para garantizar la mayor estabilidad posible del ICIS y no alterar los procedimientos familiares al usuario, sin dejar de ofrecer las facilidades que ITT necesita, toda modificación del ICIS ha de ser aprobada por una Junta Revisora de Cambios. Esta Junta, que está formada por representantes de cada compañía e incluye un número apreciable de usuarios, se reúne cada seis meses para fijar el calendario y prioridades del programa de mejora del siguiente año, y revisar los progresos.

Los cambios en el programa ICIS se gestionan por una autoridad central editora, gobernada por el control de configuración. Los representantes de la Junta Revisora coordinan los cambios que solicitan las compañías, y reciben la documentación publicada sobre nuevas versiones del sistema.



Ayuda instructiva adicional.

Interconexión del ICIS con redes locales de datos
Se pretende que los usuarios que ya dispongan de base de datos local puedan aprovecharse del ICIS, y recíprocamente que los usuarios del ICIS tengan acceso a bases de datos locales. Se ha desarrollado recientemente un programa para leer nuevos datos del sistema soporte de control de configuración del Sistema 12 (versión 3) y cargarlos en el ICIS.

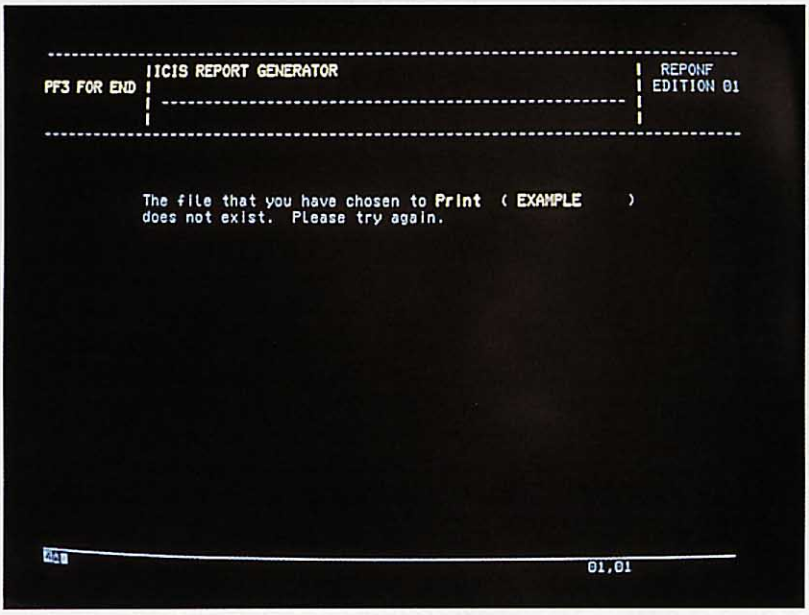
Las tablas de datos del ICIS están implantadas de forma similar a los ficheros comunes secuenciales o particionados. Sin embargo, se incluyen en las cabeceras unos datos especiales de formato/control. Si a esto se añade que los datos de componentes se extienden a varias tablas, la introducción y extracción de datos, y por tanto la comunicación con bases de datos locales, exigirá una programación especial. Se emplea un fichero sin formatar como formato neutro de intercambio. Próximamente se implantará un segundo módulo de programa para introducir cambios en el ICIS procedentes del sistema soporte de control de la configuración. Asociado con el generador de informes para la extracción de datos, y con el interfaz lógico apropiado en el otro extremo, el ICIS se conectará con las bases de datos de Standard Elektrik Lorenz (Stuttgart), Bell Telephone Manufacturing (Amberes), FACE (Milán), y otras. Ello permitirá extraer datos del ICIS con mayor velocidad para las bases locales, y que los datos de éstas puedan cargarse en el ICIS.

De este modo, el ICIS será más que una base de datos de ESC, abarcando todas las casas de ITT. Los usuarios podrán moverse libremente entre sistemas locales y corporativos, en la seguridad de que tienen acceso a los datos de componentes más recientes y adecuados.

Documentación
Los usuarios reciben automáticamente un manual con su identificador, y las actualizaciones que sean necesarias. Este manual contiene casi toda la información que requiere el usuario más exigente, y explica la mayoría de los detalles de la estructura de tablas en la base de datos, expande las abreviaturas, etc.

Pantallas de ayuda
Se han implantado ya en parte medios de ayuda al usuario interactivos, y se ampliarán todavía más. Presionando la tecla de función PF1 aparece una pantalla de ayuda autoexplicativa, que es posible obtener también en idiomas diferentes al inglés mediante una selección automática basada en las necesidades propias del usuario. Análogamente pueden traducirse las abreviaturas de datos a texto completo en cualquier idioma al producirse un informe especializado.

Adiestramiento
Pocas veces se precisa un adiestramiento minucioso del usuario, dada la sencillez del interfaz con éste. Sin embargo, hace falta



Ejemplo de un mensaje de error.

cierta instrucción práctica para los que deban introducir datos en el ICIS, pues han de utilizar alguna facilidad especial. Sobre todo, esto se refiere al uso de ficheros que registran cada cambio (transacción) y confirman al usuario que el dato se ha añadido, cambiado, o borrado acertadamente. Al revés, si ha habido un error (p. ej., un código ITT tecleado erróneamente y no reconocido), se introduce en el fichero de registro de transacciones el mensaje de error apropiado, y el usuario deberá hacer un nuevo intento.

El ESC por consiguiente organiza cursos de adiestramiento en el uso de la facilidad de registro, y también con el fin de que los usuarios puedan recibir asistencia de expertos para optimizar la eficacia del ICIS.

Conclusiones

Mediante el repetido uso, modificaciones, y mejoras de los módulos de programa existentes, y el desarrollo de un interfaz por menú, se ha diseñado, probado, populado y editado en menos de 18 meses una base de datos ingeniosa, compleja, adaptable y de fácil manejo. El ICIS es ahora el foco del intercambio de información de componentes eléctricos, tanto dentro de ITT como para los suministradores de la Compañía. Se emplea a diario en las casas de ITT por todo el mundo, contribuyendo a la toma de decisiones en todas las fases del ciclo de vida de un producto, reduciendo al mínimo los riesgos del desarrollo y optimizando los costes.

Agradecimiento

El autor desea agradecer a H. Janssens del ITC su participación en el desarrollo del ICIS, así como a Carol Wright, administradora del sistema ICIS en el ESC.

Jez Cunningham nació en Londres en 1952. En 1974 se graduó en ingeniería electrónica por la Universidad de Birmingham. Después de trabajar en Marconi Communication Systems en el diseño de equipos de supervisión y control para sistemas de radio naval, pasó a Consumer Microcircuits, donde diseñó varios circuitos LSI a medida. El Sr. Cunningham entró en ESC en 1983 como responsable de ingeniería de componentes para circuitos LSI e híbridos. Actualmente dirige el departamento de ingeniería de componentes dentro de la división de diseño de productos de ESC.

Desarrollo de programación para el Sistema 12

Las herramientas y procedimientos usados para el desarrollo de la programación del Sistema 12 aseguran la producción eficiente de programas de alta calidad. Se ha hecho hincapié en los procesos de diseño formales combinados con el tratamiento informatizado de los datos de diseño.

D. R. Illi
Standard Elektrik Lorenz AG, Stuttgart,
República Federal de Alemania

Introducción

La arquitectura distribuida de la central digital Sistema 12 de ITT se diseñó de modo que cubriera toda la gama de tipos y tamaños de centrales aplicables en distintos países del mundo. La variedad resultante de configuraciones de centrales, y los numerosos sistemas de señalización y procedimientos de administraciones con los que hubo que trabajar, necesitaron de una intensa labor en ingeniería de diseño para clientes. Otro requisito importante fue que el Sistema 12 admitiera una evolución sin grandes rediseños, en particular para facilitar la introducción de redes digitales de servicios integrados, tanto nacionales como internacionales.

Para alcanzar estos ambiciosos objetivos hubo que combinar la experiencia y conocimientos de los ingenieros de ITT en los diversos centros de diseño de Europa. Sin embargo, aunque mucho podía ganarse distribuyendo así el desarrollo de la programación, había también inconvenientes potenciales. Para asegurar su eliminación desde el principio, ITT estableció un entorno globalizador de documentación y de diseño de la programación que manejara este desarrollo multicentro y asegurara la gran calidad de todos los programas producidos, así como su cumplimiento de las normas de compatibilidad establecidas. El diseño formal y los procedimientos de documentación, unidos a los sistemas de distribución de información asistidos por ordenador y controlados por gestión de configuración, permiten desarrollar y mejorar a la vez toda una variedad de subsistemas.

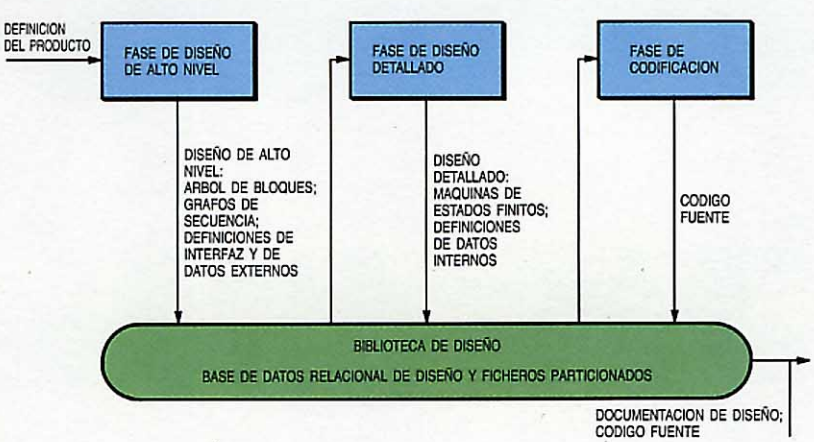


Figura 1
Diseño en tres fases de la programación del Sistema 12, indicando la secuencia y el almacenamiento de resultados en las bibliotecas de diseño.

Panorámica del diseño y documentación de la programación

El método de diseño y documentación de la programación del Sistema 12 contiene tres fases de diseño lógico: diseño de alto nivel, diseño detallado y codificación (Fig. 1). Este método, basado en sucesivos refinamientos de la definición del producto, sustenta los conceptos de máquina de estados finitos y de mensajes finitos usados en el Sistema 12, así como el modelado de datos. El perfeccionamiento gradual del diseño va acompañado por la división de funciones en subfunciones, que a su vez permite el paralelo desarrollo de subsistemas de programación y reduce la complejidad de las siguientes etapas de diseño.

Durante el diseño de alto nivel, se utiliza el LED (lenguaje de especificación y descripción del CCITT) para definir una estructura jerárquica en árbol de bloques (Fig. 2) que indica la ordenación de funciones y características en subsistemas y sus bloques de nivel inferior. La interrelación de los bloques del nivel más bajo del árbol se documenta por medio de grafos de secuencia de mensajes, definiciones de mensajes y datos externos usados por grupos de bloques.

Las descripciones textuales de los subsistemas y bloques individuales se escriben de modo normalizado. El sistema integrado de soporte de documentación ofrece carac-

terísticas tales como verificación ortográfica, formatado de texto y generación de documentos de texto normalizados según patrones predefinidos. Los diseñadores producen textos descriptivos informales a la vez que el diseño formal, sintácticamente correcto, representado por un árbol de bloques LED, grafos de secuencia, definiciones de mensajes y definiciones de datos.

La documentación completa del diseño de alto nivel es la base para el diseño detallado subsiguiente. De nuevo se usa el LED para especificar las FMM (máquinas de mensajes finitos) y las SSM (máquinas soporte del sistema), que son los bloques constitutivos de la programación del Sistema 12. Las FMM y SSM se definen como los bloques del nivel más bajo del árbol, e internamente se componen de máquinas de estados finitos.

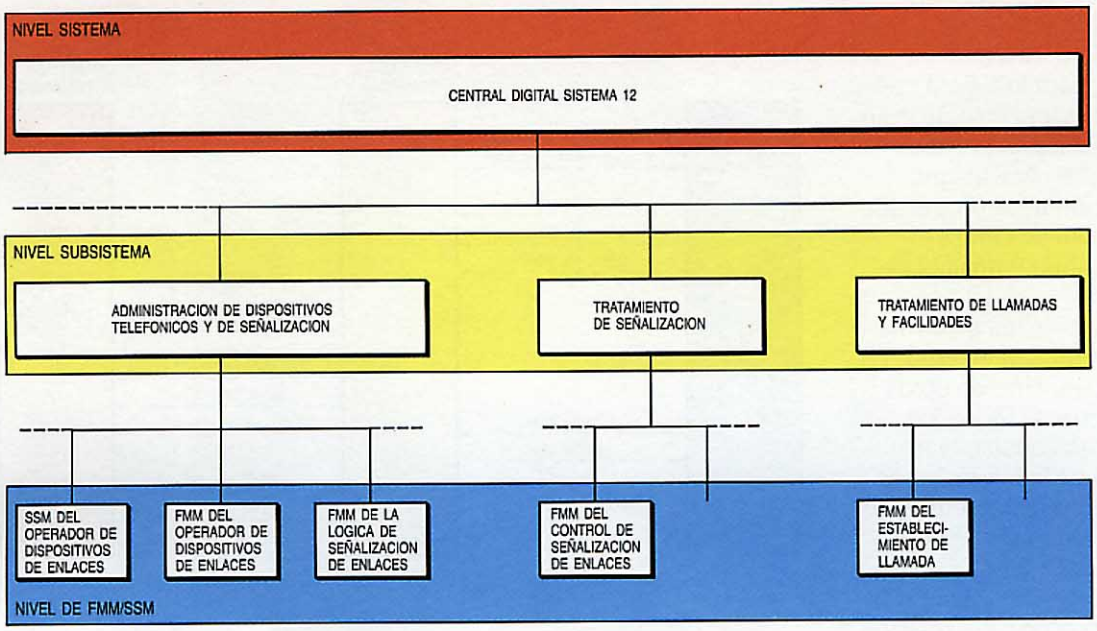


Figura 2
Arbol de bloques LED, mostrando partes de los subsistemas de tratamiento de llamadas. En el nivel inferior están los bloques constitutivos de la programación Sistema 12, que no admiten subdivisión.

La codificación es continuación natural y refinamiento ulterior del proceso de diseño, completando los armazones de módulo establecidos durante las fases de diseño detallado y de alto nivel. La mayoría de los programas del Sistema 12 se escriben en Multipol, lenguaje orientado al multiproblema que ITT ha concebido como superconjunto del CHILL; contiene órdenes específicas que simplifican el acceso a la base de datos distribuida y a las funciones de soporte del sistema operativo. El lenguaje ensamblador se emplea sólo en escasas y pequeñas partes del código, donde la necesidad de trabajar en tiempo real obliga a utilizar un lenguaje de bajo nivel.

Al terminar cada fase del diseño se revisa la documentación producida y, sólo si ésta se califica como buena, podrá iniciarse la fase siguiente. La documentación de diseño se somete a la gestión de configuración, que únicamente acepta documentación conforme a las normas generales de programación de ITT. Una vez aceptada dicha documentación – en código fuente u objeto – por la gestión de configuración en algún centro de diseño, queda a disposición de los demás centros a través de la red de ordenadores de ITT. Los enlaces de alta velocidad entre los centros de cálculo locales permiten un rápido intercambio de los resultados del desarrollo entre centros de diseño del Sistema 12. La citada red de ITT sirve especialmente para reutilizar los programas del Sistema 12 en todos los centros de diseño implicados.

Las herramientas que ayudan al diseño de la programación se ejecutan en ordenadores IBM. Toda la documentación de diseño se almacena en bibliotecas de datos de desarrollo que constan de una base de datos relacional y de ficheros particionados; de ellas puede extraerse información que se distribuya sin ningún proceso manual suplementario, de acuerdo con las necesidades de los clientes y los diseñadores que trabajan en la evolución del Sistema 12.

El método de documentación del Sistema 12 formaliza la representación de resultados de las distintas fases de diseño, siguiendo normas que se hacen cumplir y garantizan por medio de la cadena de herramientas. Así, en efecto, el paquete de manejo de gráficos, los comprobadores de sintaxis y de coherencia para el LED y los grafos de secuencia, las herramientas de

definición de mensajes y datos, y el preprocesador Multipol, así como el compilador CHILL, se han elaborado específicamente para las diferentes fases de diseño. Sin embargo, todos ellos utilizan como interfaz general las bibliotecas de diseño integradas para programación del Sistema 12, facilitando así la transición entre fases sucesivas.

Diseño de alto nivel

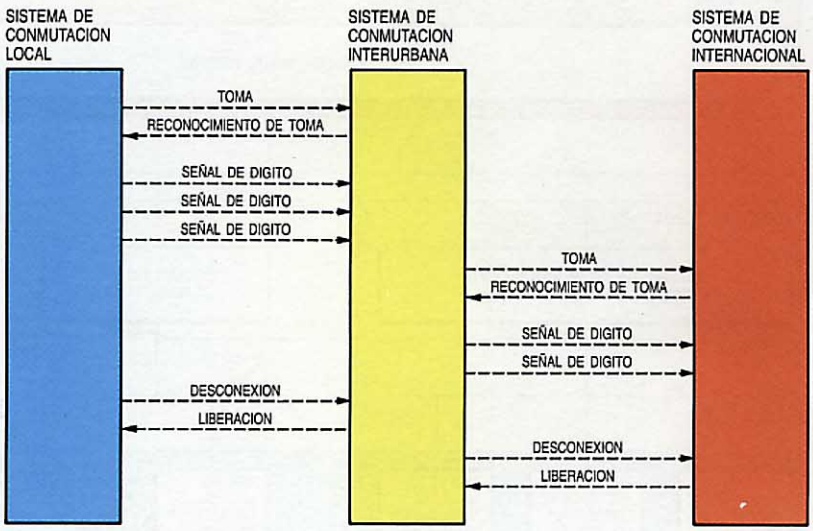
El diseño de alto nivel esencialmente descompone la definición del producto en especificaciones de módulos de programación y especificación de la interrelación dinámica entre dichos módulos. Esta interrelación viene definida por el flujo de mensajes y la utilización de datos externos. Además, el diseño de alto nivel asigna módulos de programación a los procesadores[1] y establece compromisos entre el tamaño de la memoria y los requisitos de capacidad de proceso.

La partición funcional estética se formaliza merced al LED, con su aptitud para definir árboles de bloques. Uno de estos árboles completo documenta todos los subsistemas y sus partes constitutivas, como indica el ejemplo de la figura 2. Los bloques del nivel más bajo representan los módulos de programación que han de realizarse como FMM y SSM[2].

El árbol de bloques definido por instrucciones LED se almacena en la base relacional de datos de diseño. Hay unas herramientas basadas en normas esenciales relativas a gráficos que manejan la definición textual del árbol de bloques LED y producen la distribución gráfica. El interfaz de usuario es de tipo normal, con un conjunto de menús a través de los cuales los diseñadores pueden introducir nuevas informaciones que luego se asocian a bloques individuales. Tales informaciones consisten en descripciones textuales de subsistemas y bloques, junto con listas de características implantadas en los distintos subsistemas y sus bloques de nivel inferior. Además, en las distintas descripciones de subsistemas LED se explican y referencian las estrategias de implantación elegidas.

Los bloques del nivel más bajo del árbol representan los módulos de programación que se han de realizar, y por consiguiente contienen todas las características especificadas del sistema. Es corriente que una característica esté incluida en varios bloques de ese nivel con el fin de acomodarse a restricciones concretas, tales como la limitada capacidad de memoria del módulo, la normalización de subfunciones, las memorias o las funciones de los procesadores. En consecuencia, el árbol de bloques se quedaría incompleto como modelo del sistema si no se diseña la interacción dinámica de los bloques para cada característica implantada. Los bloques del nivel inferior se comunican por medio de mensajes; la recepción de uno de ellos desencadena una cierta actividad que transforma los datos de entrada en datos de salida y actúa sobre la base de datos distribuida. Los datos de salida se formatan en mensajes que se envían a otras FMM o SSM.

Figura 3
Modelo de entorno de un sistema de conmutación interurbana en la facilidad de "llamada a central manual internacional".



El modelado dinámico de características comienza definiendo los sucesos externos, la secuencia en que éstos ocurren, y las respuestas esperadas del sistema. Para asegurar que se capta toda la información relevante del conjunto completo de tales sucesos, es esencial producir modelos del entorno que, previamente al diseño de alto nivel, describan en detalle la secuencia de sucesos externos de la señalización y de todas las características a implantar, así como las facilidades de comunicación hombre-máquina que ha de controlar la programación de la central. Estos modelos del entorno son indispensables para la definición del producto en cada tipo de central Sistema 12 (local, interurbana, internacional), y se unen al conjunto de documentos de diseño a través de la base relacional de datos de diseño. Como ejemplo se muestra en la figura 3 el modelo de entorno para una cierta característica: la conmutación interurbana.

El siguiente paso en la modelación dinámica de características consiste en detallar los flujos de mensajes internos y de datos entre los sucesos exteriores, y las reaccio-

nes que se esperan del sistema, proporcionando así una completa descripción de la interacción entre todos los bloques del nivel inferior del árbol estático, activados por sucesos externos.

El modelado dinámico no sólo considera *lo que el sistema debe hacer,* sino también, concediéndole idéntica importancia, analiza y define *lo que no debe hacer*. Por ejemplo: no pueden bloquearse dispositivos externos que están en servicio, las faltas de los procesadores individuales no deben originar faltas graves en el sistema, no pueden modificarse los datos dependientes por separado, y las discordancias entre estados del equipo y programas no deben acarrear mal funcionamiento del sistema. Aunque estas características no suelen figurar expresamente en los requisitos de las administraciones, deben tratarse con la misma atención que las exigidas de un modo explícito.

Ambas características del sistema, explícitas e implícitas, se analizan mediante los grafos de secuencia que especifican los mensajes implicados en las transacciones entre los bloques del nivel más bajo y describen su comportamiento dinámico. Diversas funciones complejas tienen partes comunes en sus grafos secuenciales, siendo necesario identificarlas y subdividir todas las características en sus elementos funcionales comunes. La figura 4 muestra dos modalidades de tratamiento de llamadas y sus elementos comunes, mientras que la figura 5 es el grafo para la toma de un enlace entrante, que es la primera función común de esos dos procesos.

Es indispensable subdividir las modalidades funcionales que requiere un cierto tipo de central en grafos de secuencia elementales comunes, ya que estos elementos se han de transformar en programas del Sistema 12 en posteriores etapas de diseño. No sólo deben analizarse los casos claros, sino también todos aquéllos en que sea excepcional el comportamiento del abonado o del sistema. El análisis funcional es una tarea compleja: en una central interurbana y solamente para el tratamiento de llamadas, hay que sintetizar unas 100 funciones básicas a partir de unos 100 grafos de secuencia elementales. Esta tarea esencial de diseño de alto nivel se apoya en el generador de grafos de secuencia, que normaliza la documentación de análisis de manera que pueda mantenerse eficazmente.

Mientras que la definición estática del árbol de bloques es el resultado de un desglose verticalmente jerarquizado en bloques funcionales, el análisis dinámico apunta al modelado desde fuera hacia adentro, viendo el sistema como mecanismo de estímulo-respuesta. Ambos análisis estático y dinámico deben ir de acuerdo; solamente los bloques del nivel inferior definidos en el árbol deben aparecer como FMM y SSM en los grafos de secuencia. En las correspondientes bibliotecas de diseño se lleva a cabo una comprobación automática para detectar posibles contradicciones entre el análisis dinámico y el árbol de bloques estático.

Una vez conocidos los mensajes requeri-

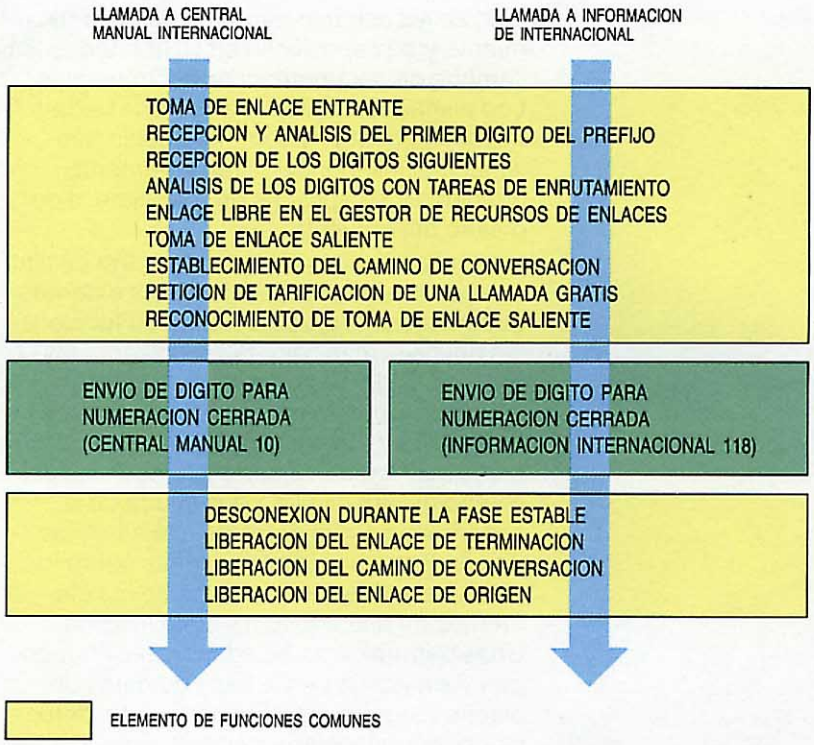
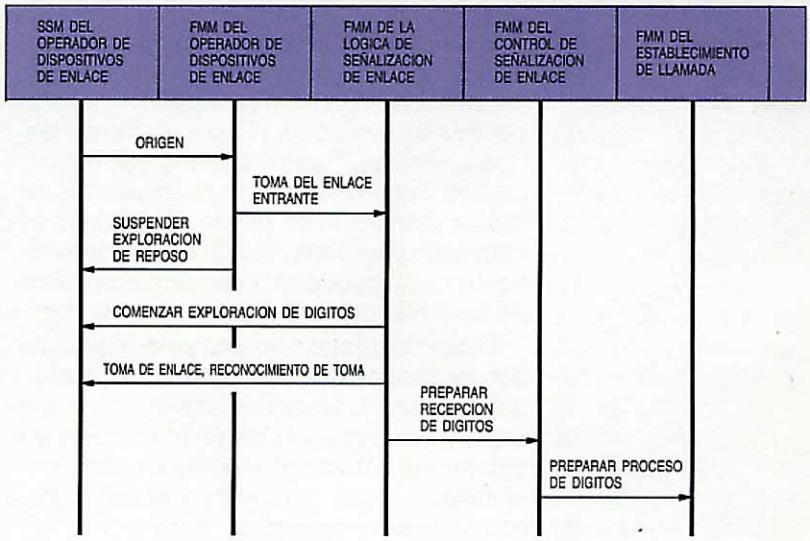


Figura 4
Síntesis de dos modalidades operativas de central interurbana que usan elementos funcionales comunes.

Figura 5
Grafo de secuencia del elemento de función común "toma de enlace entrante", especificando las interacciones dinámicas entre los bloques del nivel inferior de la figura 2. No se detallan aquí números de encaminamiento de mensajes ni nombres de mensajes normalizados, aunque los grafos de secuencia reales contienen esta información para comprobar la coherencia entre los documentos de diseño.



dos, se les asignan números de encaminamiento y se especifican en CHILL todos los campos de parámetros correspondientes. Los diseñadores disponen de una herramienta de gestión de configuración que almacena los mensajes recientemente definidos y compilados en la biblioteca de diseño del proyecto.

Un aspecto importante del diseño de alto nivel consiste en definir los datos externos que necesita el grupo de bloques funcionales del nivel más bajo. Por su diseño, los programas del Sistema 12 son activados por datos y controlan muchas y diferentes facilidades y configuraciones de equipo, por lo cual los datos son parte esencial del diseño de alto nivel. Con la ayuda de la cadena de herramientas de alto nivel, los diseñadores enuncian sus ideas sobre los datos describiendo formalmente los elementos de que constan y su utilización. Unas herramientas de administración recogen y almacenan en la base de datos de diseño la documentación pertinente, detectando redundancias y dependencias ocultas. Antes de construir los modelos de datos ajustados al proyecto, los diseñadores de la base de datos tienen que eliminar estas redundancias y dependencias para asegurar que puede controlarse la actualización de los datos con las centrales en servicio. Como el Sistema 12 utiliza una base de datos distribuida que conserva todos los datos de control dependientes de facilidades y configuraciones, la formalización del diseño de los datos descansa en una estrecha cooperación entre los equipos de diseño de alto nivel y los de diseño de la base de datos.

Otro resultado esencial de la fase de diseño de alto nivel son las descripciones de los interfaces hombre-máquina para gestión activa de los contenidos de la base de datos distribuida. También hay que diseñar y documentar los informes de alarmas y errores para detección y reparación de faltas.

Toda la información del diseño de alto nivel se acumula en bibliotecas de diseño informatizadas, a las que acceden todos los centros de diseño de ITT por medio de las funciones de gestión de configuración locales y centralizadas. Parte sustancial de la documentación de diseño de alto nivel es necesaria para capacitación en las Administraciones, y puede obtenerse directamente de las bibliotecas de diseño.

El diseño de alto nivel acaba formalmente con revisiones que se concentran en la distribución de funciones, diseño de mensajes y datos, y las subsiguientes actualizaciones a la documentación de diseño para eliminar errores, omisiones y ambigüedades.

Diseño detallado

El diseño detallado describe las FMM y SSM que constituyen los bloques del nivel más bajo del árbol, y también especifica los mensajes y datos internos necesarios para completar la definición del interfaz y de los datos.

Las FMM y SSM se activan por mensajes de entrada que disparan programas de transición para transformar los datos de entrada en datos de salida y actuar sobre los datos de la base de datos distribuida. A su vez, los datos de salida se formatan en mensajes que se envían a otras FMM o SSM. De este modo resulta natural construir internamente las FMM y SSM como máquinas de estados finitos.

Una vez establecida la sucesión de estados de FMM y SSM provocada por los mensajes, se pueden diseñar los programas de transición, los cuales especifican en detalle las acciones a realizar en los parámetros entregados por los mensajes de entrada del estado correspondiente. Se detallan todos los accesos a datos externos, así como sus modificaciones, detectados durante el diseño de alto nivel. El diseño de los programas de transición tiene también que mostrar cómo se calculan los parámetros de todos los mensajes de salida de un estado.

El LED, ideado para la especificación de máquinas de estados finitos con sus entradas, salidas y programas de transición, fue elegido para especificar la construcción interna de las FMM y SSM del Sistema 12. Posee formatos de textos y de gráficos, ambos equivalentes y utilizados para el diseño del Sistema 12.

Como soporte de la aplicación del LED hay que proporcionar diversas herramientas de diseño. Primeramente se debe asegurar la coherencia del diseño detallado con el de alto nivel, comprobando todos los mensajes de FMM y SSM con respecto a las especificaciones de diseño de alto nivel y señalando toda discordancia por resolver. En segundo lugar, la representación gráfica del diseño detallado se ha de apoyar en herramientas que faciliten el mantenimiento de la documentación de diseño. Finalmente, debe normalizarse la estructura de diseño de las FMM y SSM para facilitar su lectura, comprensión y reutilización. En apoyo de estos requisitos, los diseñadores del Sistema 12 visualizan en sus terminales formatos de documentación de diseño normalizados, que se rellenan con LED textual y se almacenan en las bibliotecas de diseño. Una herramienta de gráficos traduce dicho LED del formato textual a la representación gráfica equivalente y proporciona la documentación de diseño detallado que suele

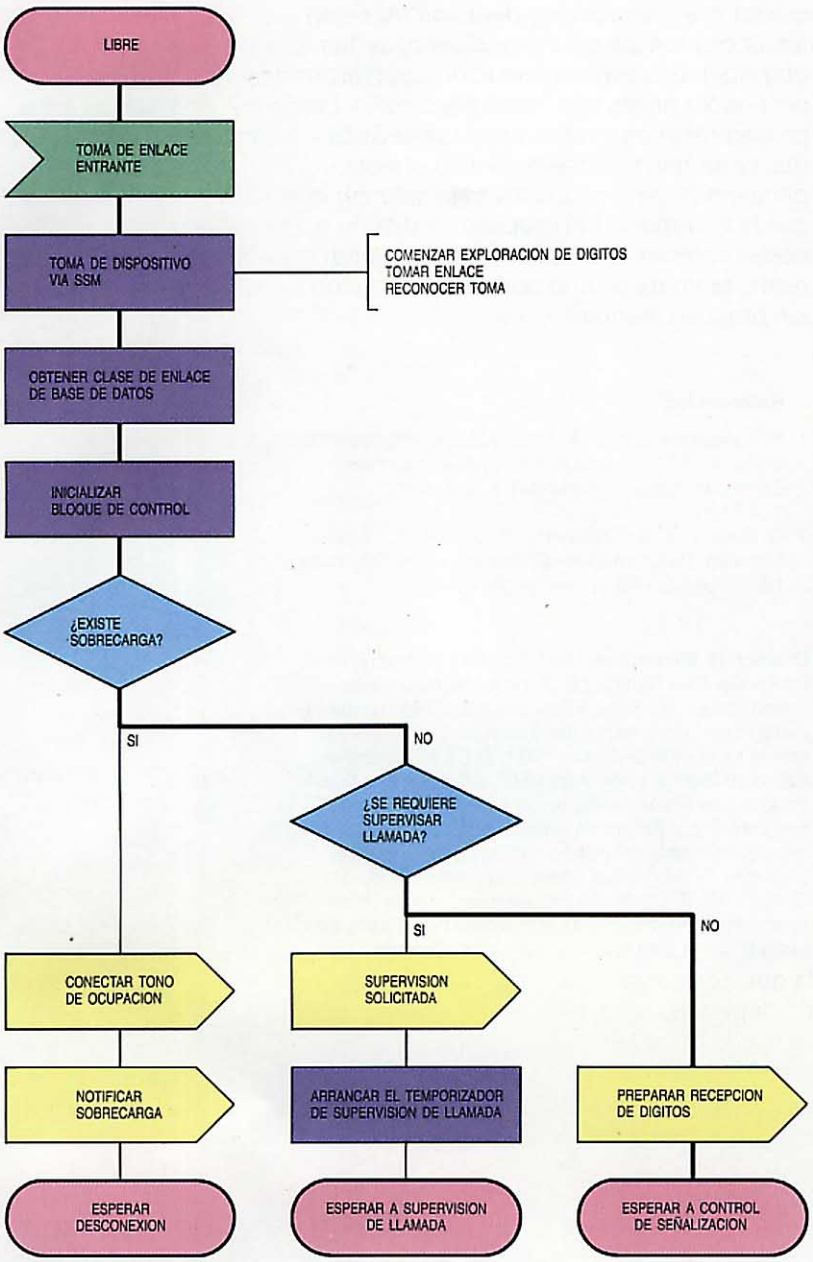


Figura 6
Diagrama de transición LED que muestra el estado libre de la FMM de lógica de señalización de enlace. El camino marcado corresponde al grafo de la figura 5. Aquí también se omiten por claridad, detalles como los números de estado y números de encaminamiento de mensajes, aunque aparecen en los diagramas reales para ayudar a las pruebas de coherencia.

utilizarse en revisiones o como base para mantenimiento del módulo. Como ejemplo, la figura 6 muestra una pequeña parte de un diseño de FMM.

Todas las herramientas de diseño del Sistema 12 (comprobadores de coherencia, comprobadores de sintaxis, y salida gráfica del LED) están concebidas para aceptar solamente formatos estándar de FMM y SSM, con sus estructuras internas normalizadas, ayudando así a la consecución del diseño deseado, conforme a las pautas de la programación.

Algunas FMM han de ser diseñadas para multiproceso, necesitando mensajes internos no especificados durante el diseño de alto nivel. Además, durante el diseño detallado se especifican aquellos datos de control que sólo se requieren para FMM y SSM individuales. Estas especificaciones adicionales para mensajes internos y datos completan el diseño del contexto para cada tipo de central del Sistema 12.

El diseño detallado de una FMM o SSM concluye con una revisión de la documentación de diseño. Es importante que los revisores conozcan todos los interfaces externos y su interacción con las FMM y SSM adyacentes, ya que deben comprobar en particular la compatibilidad de interfaces. Una vez corregido y aceptado, el diseño queda a disposición de todos los centros especializados de ITT, a través de las funciones de gestión de configuración local y centralizada, pudiendo iniciarse la codificación.

Codificación

La codificación es la última fase del proceso de diseño. Por ahora, la información de diseño detallado se traduce manualmente a instrucciones de Multipol. En el futuro próximo se prevé automatizar el proceso de obtener el código Multipol a partir de dicho diseño detallado.

El código fuente no se almacena por separado sino que se escribe en formato estándar en el fichero que contiene la especificación de diseño detallado correspondiente. En consecuencia, las instrucciones textuales de la misma sirven como líneas de comentarios para el bloque de las siguientes líneas de código fuente, terminada la fase de codificación. Cuando se ha completado el código fuente, se recopilan automáticamente las descripciones de interfaces internos y se escriben en los procedimientos descriptivos y encabezamientos de procesos de FMM y SSM. La coherencia entre el código fuente y su documentación de interfaz interno está, pues, garantizada.

Durante la fase de codificación, se generan los modelos de datos físico y lógico a partir de los requisitos de datos reunidos a lo largo del diseño, detallado y de alto nivel. El modelo lógico describe la construcción de todas las relaciones de datos y sus atributos, y en cambio el modelo físico define la distribución de las relaciones entre elementos de control del Sistema 12. Aparte de los requisitos de datos antes mencionados, otras entradas esenciales para el modelado de datos son la asignación de FMM y SSM a elementos de control, la disponibilidad de espacio en memoria de los elementos de control, la seguridad y los aspectos de tiempo de respuesta. En el modelado de datos se utilizan el DDL (lenguaje de definición de modelo de datos concebido por ITT) y su cadena de herramientas soporte para modelar los datos de programación del Sistema 12.

En cuanto se disponga de compilaciones sin errores, se concluye esta fase leyendo el código de los módulos nuevos o cambiados y de los modelos de datos, y tras las correcciones pertinentes queda abierto el camino para las pruebas modulares y de integración. Toda la documentación de diseño aplicable está contenida en el propio módulo fuente o en la base de datos de diseño; las herramientas de comprobación disponibles facilitan la eliminación de faltas y refuerzan la coherencia de la documentación. Como antes, el código fuente del módulo completado queda disponible para todos los centros de diseño de ITT interesados, por medio de las funciones de gestión de configuración local y centralizada.

Conclusiones

El método de documentación y de diseño de la programación del Sistema 12 consta de tres fases: diseño de alto nivel, diseño detallado y codificación. Se basa en refinamientos graduales de la definición del producto mediante el LED, en los conceptos de máquina de estados finitos y de FMM, y en el modelado de datos. La información de diseño se mantiene en una base de datos relacional que se encadena con los ficheros que contienen el código fuente de los módulos y los modelos de datos. Al estar almacenados los datos de diseño, se tiene una ingeniería de programación asistida por ordenador en las tres fases del diseño. Los procedimientos y herramientas del Sistema 12 se han mejorado continuamente para aumentar la productividad, reforzar la calidad, formalizar el proceso de diseño y poder obtener una documentación coherente, tanto de diseño como de aplicación, sin proceso manual.

Referencias

1 M. Langenbach-Belz, A. Melis y H. Verhille: Central digital ITT 1240: introducción: *Comunicaciones Eléctricas*, 1981, volumen 56, n° 2/3, págs. 114–125.

2 G. Becker, R. S. Chiapparoli, R. Schaaf y C. Vander Straeten: Programación: *Comunicaciones Eléctricas*, 1985, volumen 59, n° 1/2, págs. 60–67.

Dietrich R. Illi nació en 1942. En 1967 se graduó en la Universidad de Stuttgart Dipl-Ing en tecnología de comunicación eléctrica, y después trabajó en el departamento dedicado a esa materia en aquella Universidad, donde se graduó Dr-Ing en 1971. El Dr. Illi ingresó en Standard Elektrik Lorenz en 1972, y después participó en el desarrollo de los primeros sistemas PABX de ITT con control por programa almacenado. Desde 1980 ha sido responsable del diseño e integración de varios paquetes de programas genéricos y específicos del Sistema 12. El Dr. Illi es también director técnico de productos, responsable de la ingeniería de diseño de programación y de documentación del Sistema 12.

Sistema integrado de diseño de placas impresas

El sistema integrado de diseño de placas impresas de ITT incluye herramientas informáticas para el diseño, verificación de circuitos y trazado de placas. Los datos generados por el sistema pueden luego transferirse a fabricación para producir un prototipo.

A. Page
ITT Europe Engineering Support Centre, Harlow, Inglaterra
S. Grützmann
Standard Elektrik Lorenz AG, Stuttgart, República Federal de Alemania

Introducción

Todas las compañías que hoy día desarrollan equipos electrónicos se enfrentan con el problema del aumento de complejidad y el consiguiente mayor coste del desarrollo de nuevos productos. La situación se agrava por la rapidez del cambio tecnológico, que implica una pronta sustitución de los productos por otros más avanzados. En consecuencia, los fabricantes se ven obligados a desarrollar los productos más aprisa y a un coste más bajo, para incorporar las tecnologías más recientes y asegurar la calidad óptima.

Una de las claves para conseguir estos objetivos es la fabricación integrada por ordenador (CIM), que integra las funciones de CAE, CAD y CAM, es decir, de la ingeniería, el diseño y la fabricación con asistencia de ordenador. Esto permite transferir datos desde un proceso a otro sin conversión manual, así como la comunicación de herramientas de diseño informatizado en diferentes ubicaciones. Para una multinacional como ITT que tiene compañías de diseño y fabricación en todo el mundo, ello tiene especiales ventajas, pues confiere la flexibilidad necesaria para realizar el diseño, ingeniería y fabricación en los lugares más adecuados.

Ejemplo de este planteamiento es el IPDS (sistema integrado de diseño de placas impresas) de ITT, cuyo desarrollo pretende integrar las diversas etapas que van desde captación del esquemático hasta la fabricación y pruebas. La importancia del IPDS se comprende fácilmente sabiendo que el grupo de telecomunicaciones en ITT Europe diseña alrededor de 2.000 nuevas placas impresas por año.

Tradicionalmente, el diseño de circuitos ha sido en gran parte un proceso manual, aunque se han elaborado herramientas que ayudan en algunas fases del mismo. Por el contrario, el IPDS se ha concebido como un entorno basado en ordenador para el proceso completo. En esencia, comprende tres etapas: diseño del circuito, verificación del diseño y diseño físico. La verificación es particularmente importante porque permite simular y probar un diseño como modelo lógico, sin el coste de construir un modelo de laboratorio. Así, pues, con el IPDS se consiguen placas sin defectos en la fase del primer prototipo, acortando considerablemente los tiempos de desarrollo. Finalizado el proceso de diseño, el IPDS envía los datos compilados a los medios de fabricación y pruebas.

Además de aumentar enormemente la productividad de los diseñadores, el IPDS aporta una ventaja menos tangible: mejorar

El sistema integrado de diseño de placas impresas de ITT proporciona herramientas informáticas para todas las fases del diseño.

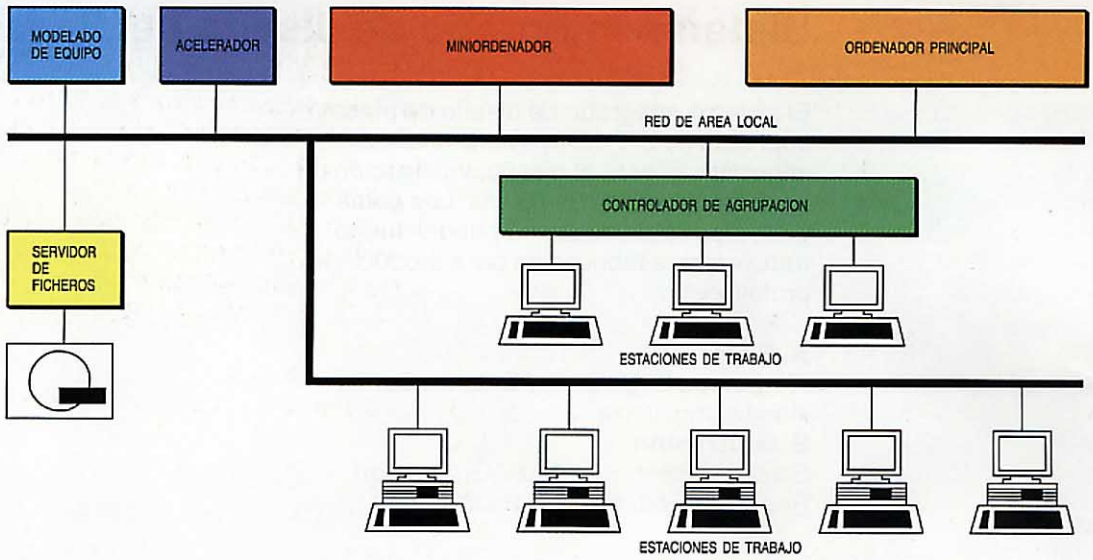


Figura 1
Arquitectura física del IPDS al final de la fase de introducción.

su entorno de trabajo. Nunca más tendrán que preocuparse de las tareas tediosas de documentación y transferencia de datos de diseño a fabricación, pues el sistema genera automáticamente los datos de trazado y de fabricación y los transfiere con exactitud desde un lugar a otro.

Desarrollo del IPDS

Cuando en 1984 ITT comprendió los beneficios que podía reportar un sistema integrado de diseño de placas impresas, acometió una investigación y evaluación en profundidad por grupos interesados en sus compañías de todo el mundo. Lo que se buscaba era definir las facilidades necesarias y cómo podrían conseguirse, teniendo en cuenta que los equipos ya existentes tendrían que integrarse en el nuevo entorno.

Enseguida se identificaron una serie de requisitos básicos para el IPDS:

- tiene que satisfacer las necesidades de todas las compañías ITT involucradas en el diseño de placas impresas
- debe consistir en un juego integrado de herramientas
- ha de proporcionar un entorno integrado, tanto en las compañías locales como en el conjunto de ITT
- debe tener la flexibilidad necesaria para ampliaciones futuras.

Las tres primeras exigencias están claras: la cuarta obedece al "problema generacional" provocado por la rápida evolución tecnológica. La arquitectura del IPDS se ha diseñado de tal forma que puedan introducirse nuevas herramientas a medida que vayan apareciendo, y por otra parte las herramientas existentes permanezcan en el sistema en tanto que sea rentable su utilización. Así, pues, varias generaciones de herramientas pueden estar en uso al mismo tiempo. Esto causa un problema con las bibliotecas, que son la espina dorsal del IPDS. Normalmente cada generación de equipos necesitaría su propia biblioteca, pero no resulta económico construir una biblioteca para un solo sistema, y por ello se ha trabajado mucho en el desarrollo de una biblioteca central para IPDS que puedan utilizar todas las generaciones de herramientas.

ITT aplica la estrategia de comprar herramientas adecuadas siempre que sea posible, y concentrar los esfuerzos en integrar esas herramientas en un sistema común construyendo las bibliotecas necesarias, preparando los interfaces CAD/CAM e introduciendo el sistema y los procedimientos asociados con un enfoque multinacional.

A la vista del importante esfuerzo necesario para conseguir estos objetivos, el desarrollo del IPDS se programó en cinco fases, o versiones, de las cuales se han realizado ya las dos primeras.

Versión 1: fase de consolidación

La primera fase buscaba la consolidación del equipo existente en las diversas compañías ITT, y la introducción de un limitado número de nuevos equipos. La idea esencial del IPDS consiste en disponer de estaciones de trabajo para la captación de esquemáticos, verificación, temporización y simulación lógica. En la primera fase había que escribir programación de interfaz para conectar estas nuevas estaciones de trabajo con las herramientas ya existentes de diseño físico (trazado). También era preciso

desarrollar un interfaz entre la parte de diseño y la de fabricación.

Todas estas herramientas dependen de la existencia de unas extensas bibliotecas de programación (bibliotecas de símbolos, de modelos, físicas). En consecuencia, una gran parte de la primera fase consistió en introducir facilidades para la construcción de bibliotecas y en el desarrollo de las bibliotecas de programas iniciales.

Finalmente, se necesitaban normas de red para conectar las estaciones de trabajo individuales y los entornos de ordenador centralizado, utilizando la red de paquetes X.25, las redes empresariales y redes de área local.

Versión 2: fase de introducción

Durante esta fase se introdujeron una serie de características y facilidades adicionales, y se racionalizaron las normas para transferencia de datos. Dichas facilidades incluían el control de configuración del diseño, la simulación lógica potenciada, simulación analógica, captación de esquemáticos, bibliotecas nuevas y ampliadas, simulación de fallos, mejores herramientas de trazado, interfaz a pruebas y transferencia de datos por una red de área extensa. La figura 1 muestra el sistema IPDS al final de esta fase.

Versiones futuras

Durante los próximos años están proyectadas nuevas ampliaciones merced a las cuales el IPDS ofrecerá un entorno completo para el diseño de placas impresas. En la tabla 1 se resumen las ampliaciones previstas para las fases 3, 4 y 5.

Tabla 1 – Versiones futuras del IPDS

Versión 3: fase de ampliación
Interfaz ampliado con la fabricación
Medios físicos ampliados para construir bibliotecas
Gestión ampliada de bibliotecas
Gestión ampliada de configuración
Medios de documentación
Gestión de reglas de diseño por inteligencia artificial.
Versión 4: fase de integración
Gestión de proceso de datos
Análisis térmico
Análisis de aptitud para prueba
Interfaz con otras tecnologías
Recuento de patillas de alta densidad incorporado en herramientas de diseño físico.
Versión 5: fase de integración final
Integración funcional y física del entorno IPDS
Incorporación de herramientas de gestión de proyectos
Incorporación de herramientas de automatización de oficinas
Realización completa de herramientas de comunicación entre compañías.

Proceso de diseño de placas de circuito impreso

Con la introducción del IPDS, el diseño de placas de circuito impreso en ITT se realiza ahora en un entorno integrado donde la mayoría de las tareas se completa con la ayuda de ordenador. En la figura 2 se muestra el flujo de información a través del proceso de diseño. Las áreas de actividad se dividen en tres etapas: captación del esquemático, verificación del diseño y trazado físico de la placa. Para respetar la integridad de la base de datos, la transferencia de información entre etapas tiene que ser bidireccional, esto es, cualquier cambio efectuado durante el trazado de la placa ha de reflejarse en un cambio en el esquemático y en la documentación. Esto se consi-

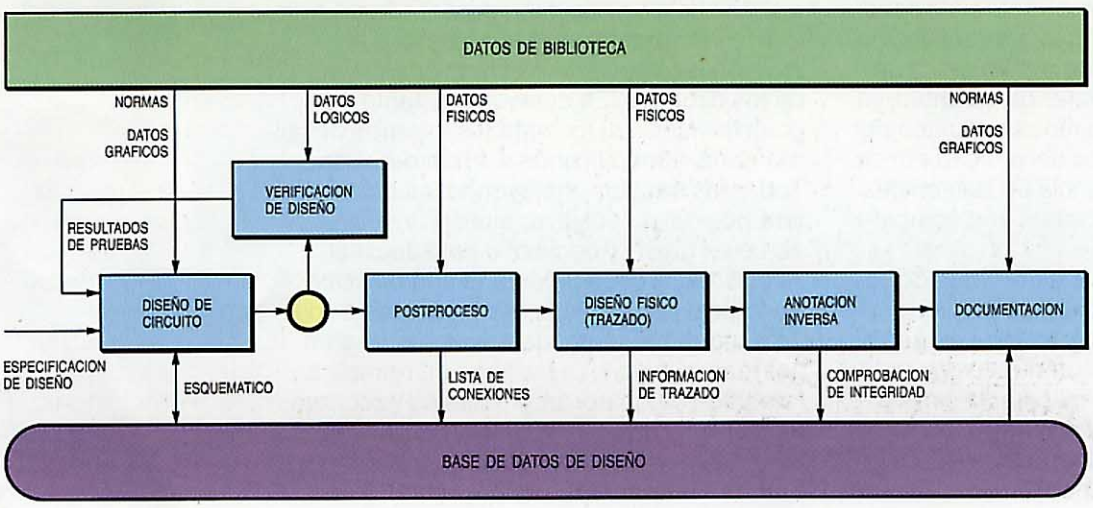


Figura 2
Flujo de información del IPDS.

gue por medio del camino de la anotación inversa indicado en la mencionada figura.

Captación del esquemático

Es la fase primaria del proceso de diseño, donde las ideas del diseñador se convierten a representación lógica. Esto se consigue dibujando un diagrama del circuito por medio de las facilidades gráficas de la estación de trabajo, lo cual, además de ofrecer al diseñador una representación familiar del circuito, presenta otras ventajas evidentes. Así, en efecto, el formato del dibujo es automáticamente el correcto y, cuando se solicita la colocación de un componente en el esquemático, aquél figura con el símbolo normalizado por ITT, consiguiendo sin trabajo adicional del diseñador acomodar el esquemático a las normas de la Compañía y poder utilizarlo en la documentación para el cliente. Asimismo, la base de datos compilada a partir del dibujo se utiliza en las etapas siguientes del diseño, desde la verificación hasta el trazado, asegurando así la coherencia.

El sistema de captación de esquemáticos da soporte al diseño jerárquico, tal como indica la figura 3. Una parte del circuito se representa por un símbolo gráfico (llamado "cuerpo") con patillas que corresponden a sus conexiones externas. Así, pues, una parte duplicada de un circuito sólo ha de captarse una vez y luego puede utilizarse muchas veces en un diseño, fomentando una mayor estructuración del mismo y permitiendo que diferentes ingenieros contribuyan a producir la base de datos que contiene el diseño completo. Una ventaja más del diseño jerárquico es que la documentación muestra la división del sistema en módulos. Esta información puede también incluirse en manuales.

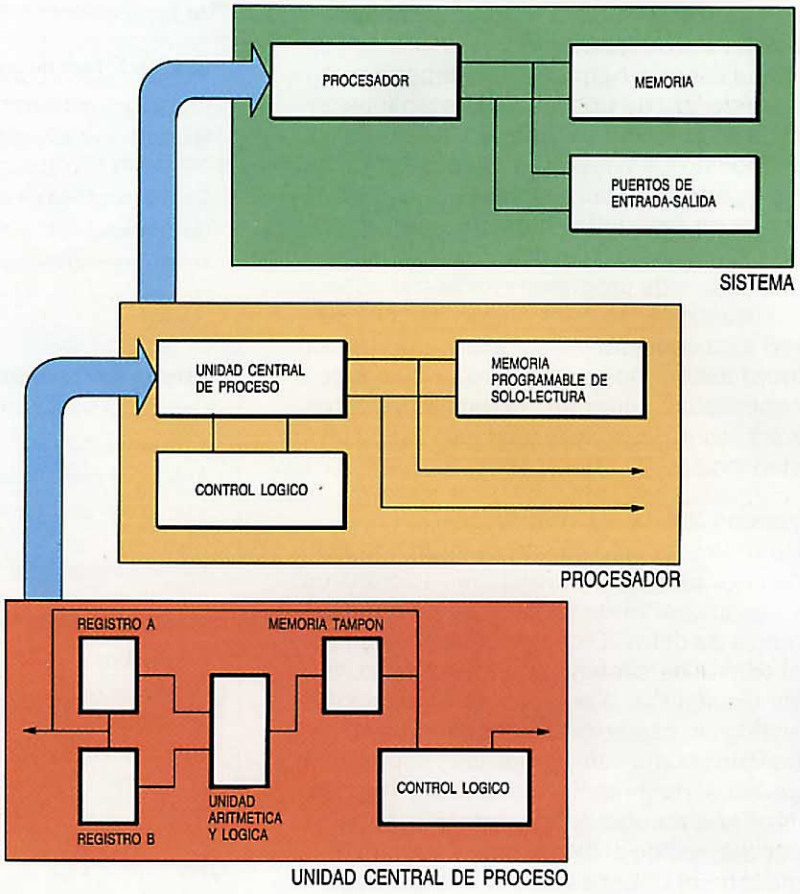


Figura 3
Ejemplo de diseño jerárquico.

Verificación del diseño

El IPDS sustenta a la vez la simulación y la verificación de tiempos. La simulación, como implica su nombre, modela la actuación de un circuito en respuesta a las señales de estímulo que le llegan. La verificación de tiempos, por otra parte, no necesita un conjunto preciso de señales de estímulo, ya que el análisis de los caminos seguidos por la temporización a través del circuito puede hacerse con independencia de las señales de entrada. Las dos opciones son complementarias.

El IPDS presta soporte a un simulador lógico interactivo. En contraste con los simuladores convencionales, que exigen al usuario definir cada vez un nuevo conjunto de estímulos y volver a ejecutar la simulación, el simulador interactivo inicializa el circuito y pregunta al usuario los valores de la señal y para qué tiempos tienen que simularse. Estos valores pueden introducirse por el teclado, o bien leerse de un fichero de estímulos. De este modo el simulador puede servir para experimentar el comportamiento de un circuito; puede imponerse a los nodos un estado que sustituya al estado existente, y realizarse "parches" lógicos limitados para investigar el efecto de pequeños cambios de diseño. El resultado final es un simulador interactivo que funciona más como ayuda al diseño que como simple herramienta de verificación.

El simulador es también utilizable como instrumento de verificación lógica para diseños basados en microprocesadores, ya que los programas pueden cargarse en memorias y modelarse el funcionamiento de los circuitos. En cualquier instante es posible visualizar los estados internos de la memoria, y modificarlos si fuera necesario. Todas las señales, incluyendo los estados internos del dispositivo, pueden visualizarse en binario, decimal o hexadecimal. Asimismo, si el usuario tiene una estación de trabajo con gráficos, las formas de onda se pueden presentar de modo similar a un analizador lógico. La facilidad de realizar la integración equipo/programas es exclusiva para el entorno de simulación, pues a la vez puede actuar como emulador en-circuito y entorno de sistema.

El verificador de tiempos produce una salida sin necesidad de definir los vectores de entrada. Se genera un listado describiendo las violaciones de tiempo en el establecimiento, retención y anchura de impulsos, y en qué lugar y momento ocurren. En modo sustitutivo, el verificador de tiempos modela el circuito durante un periodo de reloj, sean cuales fueren los estados lógicos, pero el usuario puede solicitar más de un periodo de reloj y restringir los estados lógicos cubiertos. El uso más eficaz de dicho verificador es descubrir errores de tiempos en partes pequeñas del circuito, a medida que se diseña, cuando tal vez no estén totalmente definidos los estímulos que recibirá. La salida del verificador de tiempos puede procesarse para generar los diagramas de tiempo del funcionamiento del circuito.

Verificación analógica

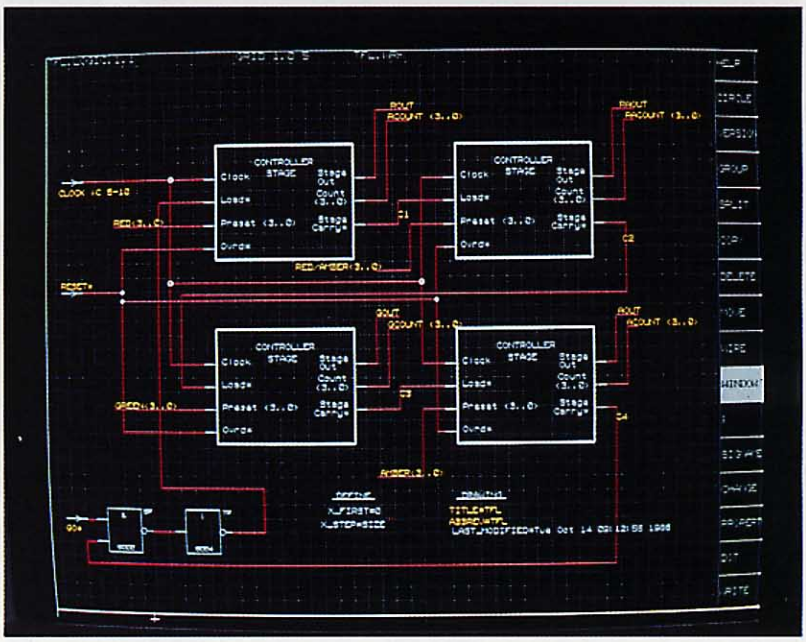
El simulador y el verificador de tiempos son únicamente válidos para circuitos digitales. La simulación analógica hasta hace poco se ha limitado a unos cuantos paquetes especializados, difíciles de usar y necesitando máquinas cuya potencia de cálculo superaba la de una estación de trabajo de ingeniería. Ahora, no obstante, se ha desarrollado un soporte lógico más eficiente y fácil de usar, y las estaciones de trabajo son más potentes, haciendo posible incluir herramientas de simulación analógica en el IPDS. Se ha completado la evaluación de los programas disponibles y comenzado la integración, esperando que concluya en la primera parte de 1987.

Técnicas de modelación

La integridad de cualquier paquete de verificación depende de los datos disponibles, en este caso de los modelos de los dispositivos. Para la verificación de tiempo se necesita un modelo simplificado, que describa únicamente los tiempos requeridos de establecimiento, retención y ancho mínimo de impulsos, así como los retardos de propagación. No obstante, el simulador necesita una descripción completa del dispositivo a modelar, incluyendo la información de tiempos, que puede ser compleja. Por esta razón los modelos de temporización se basan en lógica y se crean a partir de bloques denominados "primitivas". Los modelos de simulación presentan tres opciones. La más básica es construir un circuito que represente al dispositivo partiendo de un conjunto limitado de primitivas: típicamente biestables, elementos de registro, sumadores y bloques similares, que modelan un pequeño número de puertas. El modelo se construye como un diagrama de circuito, utilizando las facilidades de captación de esquemáticos.

En algunos casos es más fácil describir un dispositivo por medio de un modelo de comportamiento, que se escribe en Pascal y utiliza sentencias de alto nivel, permitiendo así la modelación de operaciones complejas. Estos modelos son especialmente útiles a la hora de simular bloques de circuitos, especificados pero no descrito su diseño en detalle, puesto que permiten las pruebas de integración en las primeras fases de un proyecto.

La tercera técnica es el uso de un sistema de modelación física: dos de ellos tienen soporte en el IPDS. El dispositivo a modelar se coloca en una placa pequeña o "módulo de personalidad", y se simula utilizando las señales que recibe el modelo en-circuito. El modelo físico responde con salidas que el simulador lee y realimenta al circuito simulado.

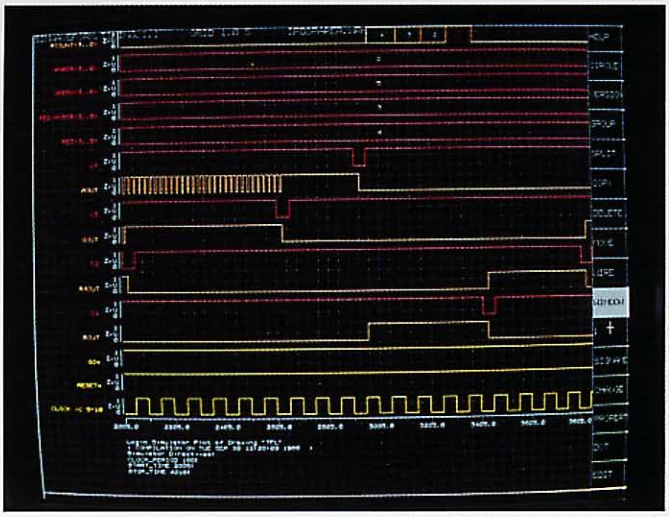


El IPDS aporta un enfoque jerárquico del diseño de placas impresas.

Puede utilizarse modelación física para simular elementos complejos, tales como microprocesadores y dispositivos soporte, para los cuales no pueden construirse modelos lógicos o sería muy lenta la simulación resultante. Una ventaja suplementaria es que el sistema sirve para investigar el comportamiento de un dispositivo físico, asumiendo así un rol de pruebas.

Soporte de modelación

El uso eficaz de procesos de diseño con ayuda de ordenador depende de utilizar componentes normalizados, siempre que sea posible. También es importante que los usuarios puedan confiar en la funcionalidad



Formas de onda de simulación.

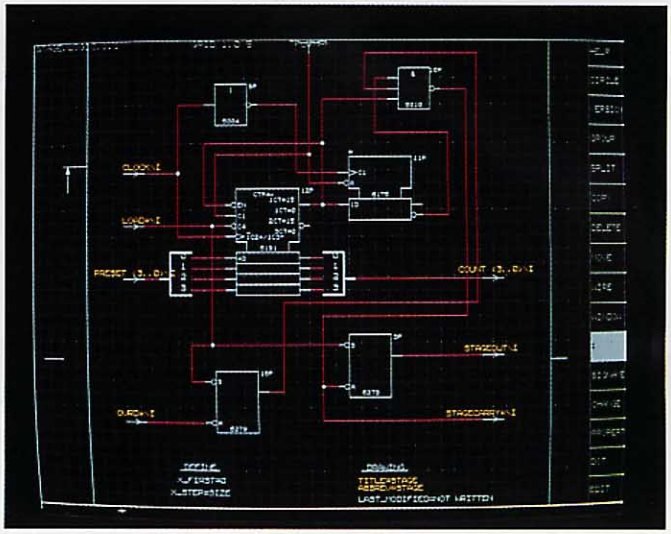
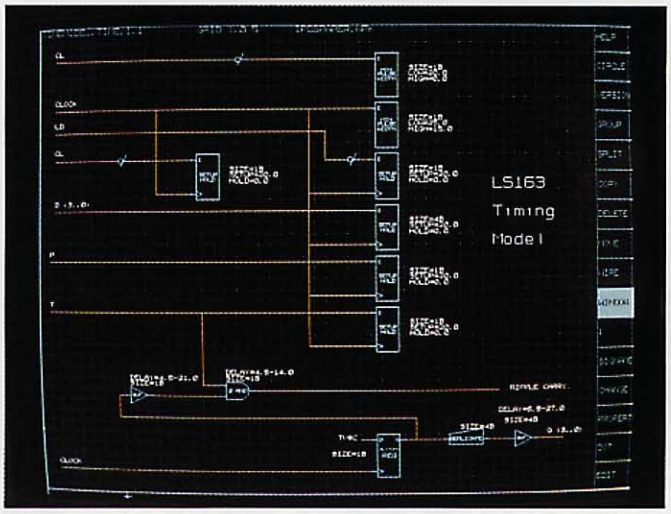
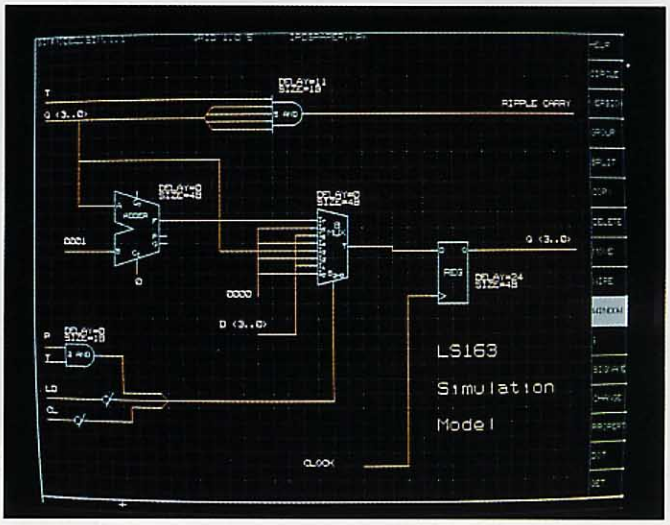


Diagrama del circuito.



Modelo de tiempos.



Modelo de simulación.

de esos componentes. Al mismo tiempo, la creciente complejidad de los circuitos modernos se refleja en los componentes normalizados; como ejemplo, un algoritmo para el posicionado automático de componentes en una placa impresa necesita tener acceso a los datos de intercambiabilidad de patillas para encontrar las conexiones más cortas entre componentes.

Por todo ello es característica esencial del IPDS el soporte de bibliotecas de modelos.

Bibliotecas

Reconociendo su importancia, se ha trabajado con intensidad para asegurar que las bibliotecas del IPDS sean completas, sin errores y probadas adecuadamente antes de utilizarse para el diseño de placas. Esto se ha conseguido instalando herramientas informatizadas para la generación y gestión de bibliotecas de tipo estándar. Las características esenciales de estas herramientas son:

- los elementos de la biblioteca se crean sólo una vez en un formato neutral
- los elementos creados en la biblioteca son correctos y completos
- las bibliotecas específicas de un sistema se crean siempre a partir de la biblioteca maestra
- se entrega de forma inmediata una documentación completa
- los requisitos futuros pueden incorporarse fácilmente.

Hasta la fecha se han creado dos bibliotecas maestras para símbolos de esquemáticos y símbolos físicos (de trazado), apoyadas por herramientas para la generación, manejo y

postproceso de elementos. La biblioteca de símbolos de esquemáticos contiene datos sobre todas las propiedades del dispositivo junto con información sobre patillas físicas y su intercambiabilidad. Análogamente, la biblioteca de símbolos físicos contiene todos los datos necesarios para posicionamiento de dispositivos y trazado de interconexiones.

Dado que un modelo contiene no sólo información de simulación y de tiempos, sino también información física y una representación gráfica, construir una biblioteca requiere mucho tiempo y personal especializado. Para repartir el trabajo, se ha instruido a personal de las diferentes compañías ITT en generación de bibliotecas locales. La duplicidad de trabajo se evita mediante la existencia de un grupo central de bibliotecas que coordina las entradas desde las diferentes compañías a una base de datos central en el ITT Europe Engineering Support Centre. A esta base de datos pueden acceder los usuarios remotos a través de la red de conmutación de paquetes o de la red ITT para mantener sus bibliotecas locales.

Postproceso

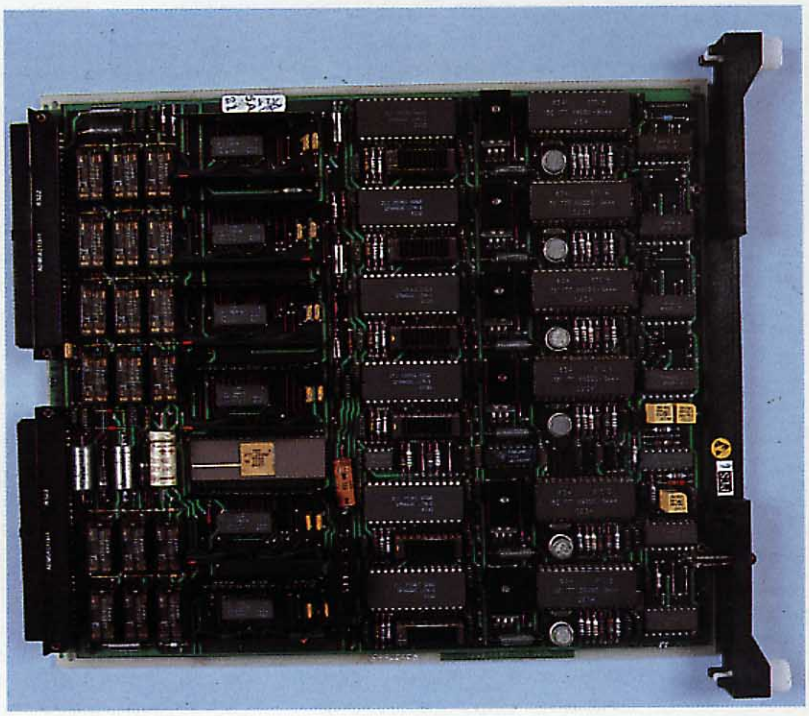
A la captación del esquemático y su verificación sigue la ejecución de un postprocesador que convierte la representación lógica del circuito a versión física. Dicho postprocesador empaqueta el diseño en un número mínimo de dispositivos físicos, y produce una lista de piezas y una lista de puertas sobrantes. También se genera una lista de conexiones que se pasa al sistema de trazado de placas impresas. Durante este proceso, se realizan comprobaciones para asegurar que todas las entradas están conectadas a salidas o conectores, e inversamente que todas las salidas están unidas a entradas o a conectores. Se genera una lista de advertencias para comprobación por el usuario. También se comprueba el no haber sobrepasado la capacidad de despliegue lógico, utilizando datos contenidos en las bibliotecas que describen las corrientes de entrada y salida.

El sistema crea un conjunto de "ficheros de estado" que ofrece una representación completa del estado lógico y físico del circuito en el momento actual. Estos ficheros sirven para comprobar que los datos devueltos por el sistema de trazado de placas impresas no contradicen a los datos que se le envían (véase la anotación inversa). En esta fase se ejecuta un programa de conversión de la lista de conexiones, con el fin de formatar los datos de modo que puedan ser introducidos en uno de los sistemas de trazado de placas.

Trazado de placas de circuito impreso

Posicionado

La primera etapa en el trazado de una placa impresa es colocar los componentes en la posición deseada sobre la placa. Se pueden solicitar las dimensiones normalizadas de la placa (bases) llamando a una biblioteca. Algunos componentes (ej., conectores) tendrán una posición predeterminada y se podrán situar inmediatamente, mientras que otros se colocarán dependiendo de las conexiones con los demás componentes. El posicionado puede hacerse automáticamente, calculando el sistema el emplazamiento óptimo de los componentes, o bien manualmente. En este último caso, las conexiones a otros componentes ya colocados se muestran como líneas entre patillas que dan al diseñador de la placa una idea de las densidades de conexionado. El diseñador de circuitos puede añadir también propiedades al esquemático para orientar la colocación de componentes críticos durante la fase de posicionado.



Placa impresa diseñada por medio del IPDS.

Trazado

Una vez posicionados todos los componentes, hay que trazar todas las interconexiones entre ellos. También aquí se puede actuar manual o automáticamente, pero lo normal es ejecutar el trazado automático en un porcentaje elevado de pistas y completar el resto de forma interactiva. Los algoritmos

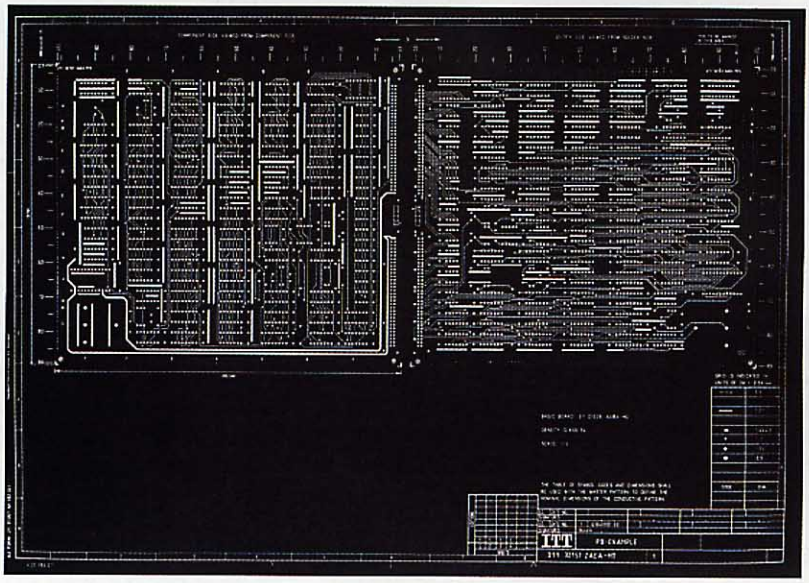
de trazado automático pueden restringirse a utilizar un número limitado de capas de la placa o de *puntos de paso* (orificios que unen una capa de la placa con otra). Esta facilidad sirve para realizar trazados sucesivos que aumenten las posibilidades del sistema a cada nuevo intento. Cuando se ha completado una proporción elevada de pistas, un diseñador de placas experimentado trazará las pistas restantes generalmente con mayor eficacia que el trazado automático. Para ayudar al diseñador, las conexiones no finalizadas se visualizan como líneas de patilla a patilla. Si el diseñador intenta una conexión errónea, el sistema le advierte del error, asegurando así que la intervención manual no perturba la base de datos de diseño.

Intercambio de dispositivos

El postprocesador del sistema captor de esquemáticos empaqueta arbitrariamente dispositivos (funciones) en componentes físicos, lo que puede acarrear dificultades innecesarias en el trazado de la placa impresa. Para aminorar este problema, el sistema de trazado de placas puede intercambiar un dispositivo lógico, como una puerta NAND, de un componente físico a otro. También es posible intercambiar patillas similares. Con tales modificaciones se obtiene un circuito con la misma representación lógica que el original, pero con distinta representación física. En consecuencia, el esquemático captado en un principio tal vez no represente ya al circuito con exactitud, para corregir lo cual se necesita un camino de anotación inversa.

Anotación inversa

A menudo se pasa por alto esta fase crítica entre la captación del diseño y el trazado de la placa, durante la cual los cambios introducidos al trazar la placa se reflejan en la base de datos de diseño. Mediante el sistema IPDS se ejecuta un interfaz inverso cuya entrada es la lista de conexiones del sistema de trazado, para generar una lista revisada en el sistema de captación de esquemáticos. Los "ficheros de estado" generados durante el postproceso se usan para comprobar que la información que devuelve el trazado es lógicamente idéntica a la que se le envió. Si esto es así, se ejecuta un programa de anotación inversa que actualiza el esquemático automáticamente para reflejar el trazado físico, cambiando conforme a ello el empaquetado de dispositivos y los números de patillas.



Realización física de una placa producida mediante el IPDS.

Interfaz CAD/CAM

Gran parte de los datos generados y utilizados para el diseño del circuito y de la placa impresa son también útiles en el proceso de fabricación, necesitándose sin embargo un interfaz CAD/CAM adecuado para poder transferirlos. El IPDS se vale de una red y una base de datos para almacenar y transferir los datos desde diseño a fabricación. La gestión de datos y la de configuración aportan los controles de administración y emisión necesarios.

La emisión de datos durante el desarrollo se controla por "puntos de anuncio", que definen en qué momento pueden utilizarse datos cualificados y autorizados para la siguiente fase. La disponibilidad de los datos emitidos se anuncia a todos los usuarios, que pueden acceder a ellos si tienen la autorización necesaria. Un sistema gestor de configuración supervisa el estado de los datos emitidos y asegura su integridad en cualquier punto de anuncio. Los datos solicitados se envían únicamente a través de la red de datos ITT, y el formato de los mismos viene definido con independencia del equipo físico.

Las reglas anteriores aseguran un flujo de datos coherente desde el comienzo del proceso de diseño hasta la fabricación del producto.

Documentación

Cuando se termina un diseño utilizando el IPDS, se dispone de la documentación que describe el esquemático, diagramas de bloques, lista de partes, trazado de la placa, serigrafías y, si se desea, diagramas de tiempos para las pruebas. Esta información

se genera desde las bases de datos de diseño o de trazado, con un esfuerzo adicional mínimo, fuera del proceso de diseño. Los formatos de dibujo de la compañía pueden incluirse como elementos de la biblioteca, asegurando así que toda la documentación está preparada para su distribución con un trabajo de delineación mínimo. La posibilidad de introducir errores queda, pues, prácticamente anulada.

Conclusiones

Se ha alcanzado ya el objetivo inicial del IPDS, que era consolidar e integrar de manera efectiva los equipos existentes, y además se han instalado varias facilidades adicionales. Con todo ello, el IPDS ofrece ahora un entorno integrado eficaz para el diseño de placas impresas en 12 compañías ITT, en 16 localidades de Europa.

En los próximos años, el sistema continuará su evolución a medida que aparezcan nuevas versiones, se identifiquen nuevos requisitos y se desechen los equipos más antiguos. La capacidad de simulación del IPDS está en curso de desarrollo, incluyendo simulación de faltas y generación de configuraciones de prueba que permitirán realizar un completo análisis de la aptitud para pruebas en la fase de diseño. También se estudian los paquetes de análisis térmico y de fiabilidad con intención de integrarlos en la serie IPDS.

Tales mejoras se han elegido como resultado de impresiones de los usuarios en casas ITT, por lo que la futura orientación del programa IPDS dependerá en gran medida de las exigencias de los usuarios. A lo largo de todo el desarrollo del IPDS se ha mantenido la táctica de apoyar a los usuarios y ayudarles a potenciar los medios disponibles en sus compañías.

Agradecimientos

Los autores quieren agradecer a J. Schoemaker del ITT Advanced Manufacturing Technology Center (Bruselas) sus orientaciones en cuanto responsable del proyecto de desarrollo del IPDS.

Alan Page nació en Inglaterra en 1959. Se graduó en 1980 por la Universidad de Londres en ingeniería electrónica. Durante cinco años trabajó en Standard Telephones and Cables en el diseño de sistemas de telecomunicación, antes de ingresar en ITTE ESC en 1985. El Sr. Page trabaja actualmente en el departamento de ingeniería asistida por ordenador del departamento de diseño de productos, en el cual presta su apoyo para el programa IPDS.

Sigmar Grützmann nació en Alemania en 1934. Se graduó en 1960 por la Escuela Técnica Superior de Munich con un Dipl-Ing en física, incorporándose luego a la ingeniería de Siemens en dicha ciudad. En 1968 obtuvo el grado de Dr-Ing por la Universidad de Stuttgart. El Dr. Grützmann trabajó después como responsable técnico en comunicación de datos, redes y equipos de alimentación de línea, pasando luego a trabajar en Crypto Ag, Zug, Suiza, donde se encargó de la investigación y desarrollo. En 1981 ingresó en SEL, y allí actualmente dirige el departamento de soporte de ordenadores.

Soporte de productos basados en microprocesadores durante su vida útil

ITT ha creado un entorno completo de desarrollo de programas para asegurar una elevada calidad del producto y un desarrollo eficiente. El núcleo de este entorno es un sistema de gestión de la configuración, que aporta un conjunto integrado de programas para el control de los equipos físicos, de la programación y de la documentación.

T. Haque
J. Montes
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

El sistema de gestión de la configuración de productos (PCMS) de ITT proporciona medios de soporte para el desarrollo de productos basados en microprocesadores. Se le concibió para ser el núcleo central del entorno de desarrollo de programas (SDE), con intención de que se pudiera utilizar también para el desarrollo de otro tipo de productos, y no exclusivamente de programas.

El PCMS es un conjunto integrado de programas para la gestión de elementos que pueden ser circuitos, programas o documentaciones. No está sujeto a una serie de reglas preestablecidas: pueden definirse un conjunto de funciones en el proyecto y diferentes modelos de ciclos de vida, ofreciendo al usuario del sistema facilidades ajustadas a su cometido en el proyecto (por ejemplo, ingeniero de desarrollo, responsable de producto). En especial, el sistema PCMS realza la misión del encargado de un proyecto o producto al aumentar la visibilidad de la información, clave del éxito del desarrollo.

Características de la ingeniería de programación

Son muy variadas las características de la ingeniería de programación de los productos de ITT basados en microprocesadores. Por un extremo, vienen marcadas por las características generalmente aceptadas para grandes proyectos de esta índole. En ITT las actividades de desarrollo de programas para tales productos suelen realizarse en varios centros de diferentes países e involucran a especialistas en gran número de disciplinas. Tradicionalmente, la vida útil de un producto abarca varios años e incluye muchas fases, tanto durante el desarrollo como ya puesto en el mercado, requiriendo mejoras y mantenimiento durante largos periodos. Por ello son esenciales los controles exhaustivos del ciclo de vida del producto.

En el otro extremo de la gama, el desarrollo de productos a menor escala puede realizarse en una sola localidad, en cuyo caso un sencillo conjunto de herramientas de control de configuración y de realización bastaría para satisfacer las necesidades de desarrollo y de gestión.

A pesar de la naturaleza tan diferente de muchos de los proyectos, es preciso controlarlos todos, ya que normalmente las compañías de ITT trabajan simultáneamente en varios tipos de desarrollo. Además, cada vez es más necesario acortar el desarrollo de los productos, lo que exige un detallado control de su ciclo de vida.

Con el fin de atender estas nuevas necesidades, ITT utiliza sus compañías en los diferentes países para acomodar los productos a las exigencias locales. En consecuencia, la transferencia de productos entre casas ITT es una actividad clave, que exige mucho de las facilidades de gestión necesarias en el SDE a la hora de modificar un producto para acomodarlo a los requisitos específicos de un país, proceso que en ITT se denomina CDE (ingeniería de diseño para el cliente). Sobre tales facilidades ejercen presiones aún mayores las diferencias de gestión, cultura, y experiencia tecnológica entre compañías de ITT. Es, pues, esencial disponer de un conjunto integrado de medios de gestión del ciclo de vida del

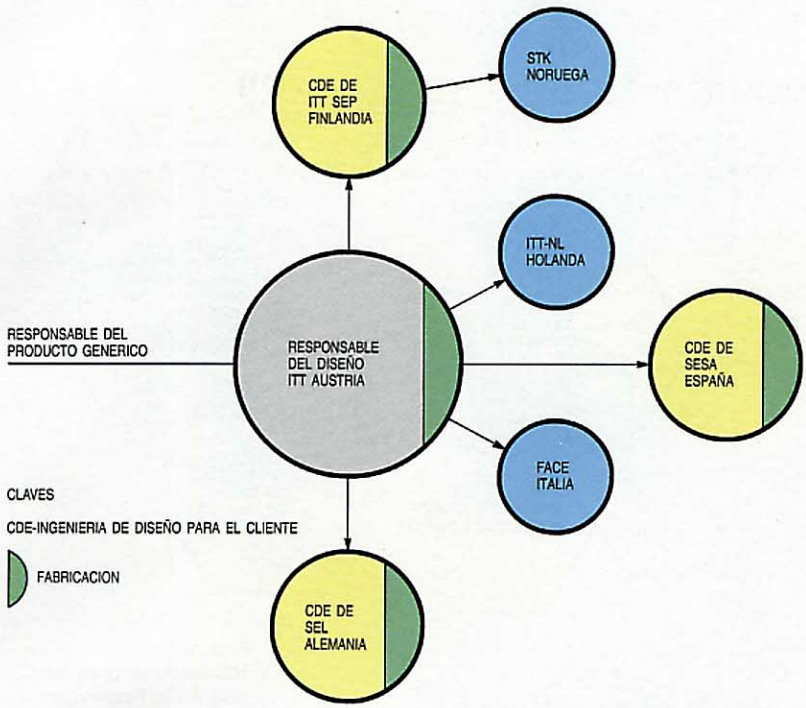


Figura 1
Para una multinacional como ITT es crucial la transferencia de productos, que las distintas compañías nacionales acomodan mediante CDE a sus exigencias locales.

producto para controlar y sincronizar los cambios en CDE que resulten de la evolución tecnológica, mejoras en el diseño y corrección de defectos.

La figura 1 muestra un diagrama de transferencia de un producto típico, en este caso desarrollado por ITT Austria para uso local; en ella se aprecia que el producto se personaliza todavía más y se fabrica en Austria por encargo de FACE e ITT Niederlandsche para uso en Italia y en Países Bajos, respectivamente. Además la figura muestra el trabajo en CDE en otras compañías de ITT.

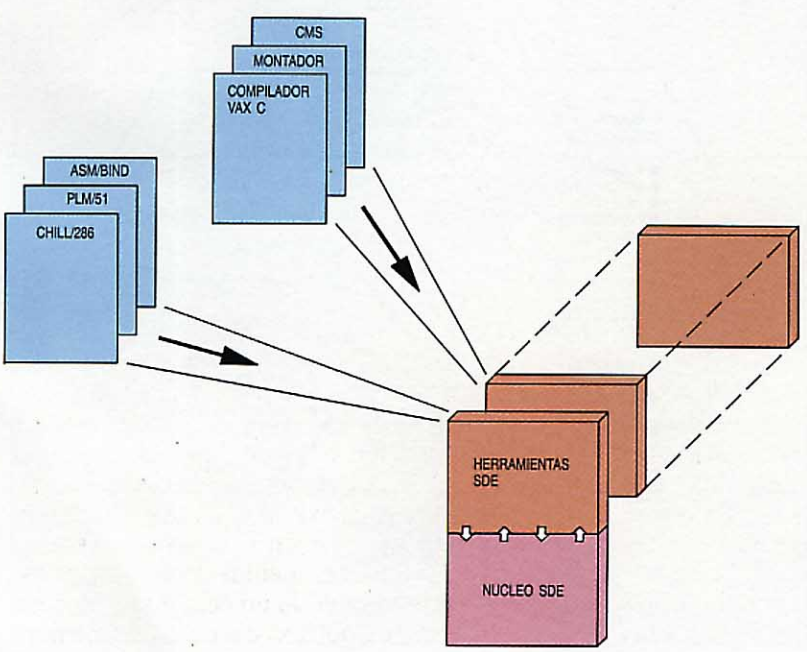


Figura 2
Una vez integradas en el SDE herramientas específicas del producto, éstas tienen acceso a todas las funciones disponibles en dicho SDE.

Entornos de desarrollo de programación

El primer juego de herramientas elaboradas para la industria de desarrollo de programación apoyaba el propio proceso de producción. Realmente, los programas no hubieran podido cargarse, ensamblarse ni ejecutarse sin utilizar cargadores, operadores de dispositivos y otras herramientas rudimentarias. ¡Incluso era normal la introducción de parches por la consola del ordenador!

La evolución a partir de estos sencillos medios condujo a obtener una serie de herramientas básicas que ayudaban a los programadores individuales a escribir y depurar los códigos. Gradualmente las herramientas de codificación se fueron refinando al introducirse compiladores, referenciadores cruzados, medios de depuración, editores de textos y sistemas de biblioteca. En realidad había surgido un entorno básico, aunque subsistía un conjunto de herramientas que ante todo prestaba apoyo a programadores aislados. A medida que iban apareciendo nuevos lenguajes, tales entornos básicos crecieron en número.

Tradicionalmente las herramientas de programación han prestado soporte a las actividades de ingenieros de programación individuales más bien que a las de un grupo, proyecto u organización. Por el contrario, el SDE se basa en que la programación es sólo una de las muchas actividades que necesitan soporte en el desarrollo de un producto. Así, pues, el SDE puede concebirse como un marco dentro del cual se diseñan, construyen, prueban y lanzan productos, y se puede dirigir la gestión del ciclo de vida de los mismos.

El SDE se ha elaborado como ámbito neutral donde integrar herramientas específicas para el producto. Una vez integradas, tales herramientas pueden utilizar todas las funciones del SDE comprendidas dentro de ese ámbito (Fig. 2), el cual aporta además las facilidades necesarias para la gestión de un producto durante toda su vida útil.

Fijar las herramientas para el desarrollo de un producto concreto implica la selección de una cadena de herramientas del SDE (procesador objeto y basado en lenguaje), junto con cualquier otro medio específico que pueda necesitar el producto (p. ej., herramientas para la población de datos o para prueba). Los servicios de gestión del ciclo de vida no dependen ni de la naturaleza del producto al que se aplican, ni, en la medida de lo posible, de las características de la ingeniería.

Es bastante fácil adquirir sistemas para actividades tales como el control de cambios, si bien permanecen separados de

otros sistemas necesarios, como los de control de bibliotecas. Su integración raras veces es posible, lo que origina importantes problemas técnicos y de control (Fig. 3).

El sistema PCMS

El núcleo del SDE se articula en el PCMS, el cual aporta las facilidades integradas que permiten gestionar y controlar eficazmente un producto a lo largo de todo su ciclo vital. El PCMS recopila toda la información relativa a un producto en una base de datos, y ofrece diversos mecanismos para controlar el uso de esta información y los procesos implicados en el desarrollo de ese producto (Fig. 4).

La base de datos del producto se realiza aplicando técnicas de bases de datos relacionales, que facilitan la integración de medios coadyuvantes al control del ciclo de vida del producto. Como todos estos medios utilizan el mismo modelo (base de datos del producto), pueden aportarse diferentes planteamientos para aplicaciones o tipos de usuario final concretos (ejemplo: el técnico de desarrollo, el probador, el responsable del producto).

Otra característica clave del PCMS es su clara distinción entre los mecanismos que apoyan el periodo de vida útil del producto y las reglas que rigen el proceso de su desarrollo. Por ejemplo, hay mecanismos como los informes de fallos que dan soporte a cualquier ciclo de gestión de cambios del proyecto. Las reglas y disciplinas utilizadas para controlar un proyecto varían notablemente desde uno que necesite tres hombres-año para terminar a otro que todavía requiera 30 hombres-año. Por ello se consideró esencial separar los procesos de los mecanismos para satisfacer las diversas exigencias del desarrollo de productos.

Base de datos del producto del PCMS

La base de datos del producto (Fig. 5) controla la información que describe los siguientes aspectos del desarrollo:

- descomposición del diseño del producto en sistemas, subsistemas, áreas funcionales, módulos, etc.
- elementos físicos tales como documentación y módulos de programas
- paquetes de configuración (modelos del producto, variantes específicas para países, etc.)
- documentación de cambio (informes de fallos, peticiones de cambio, etc.)

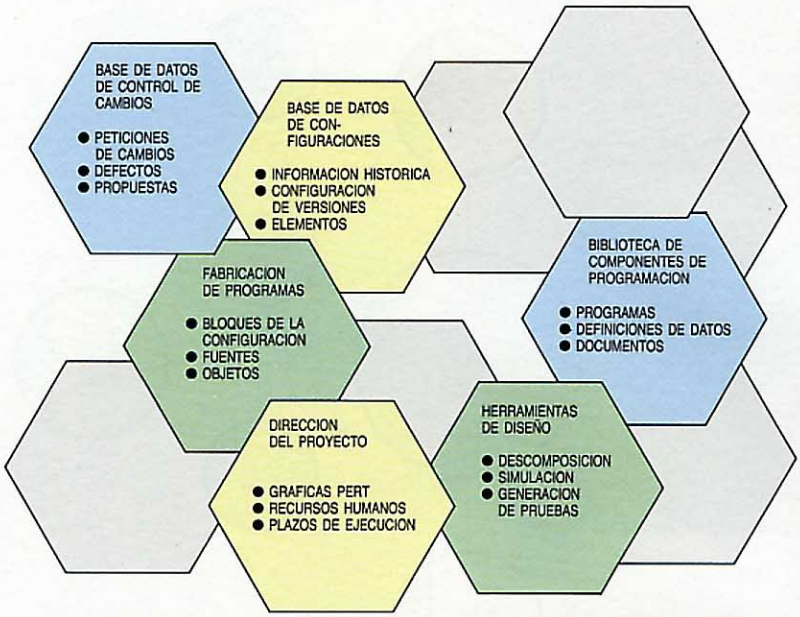
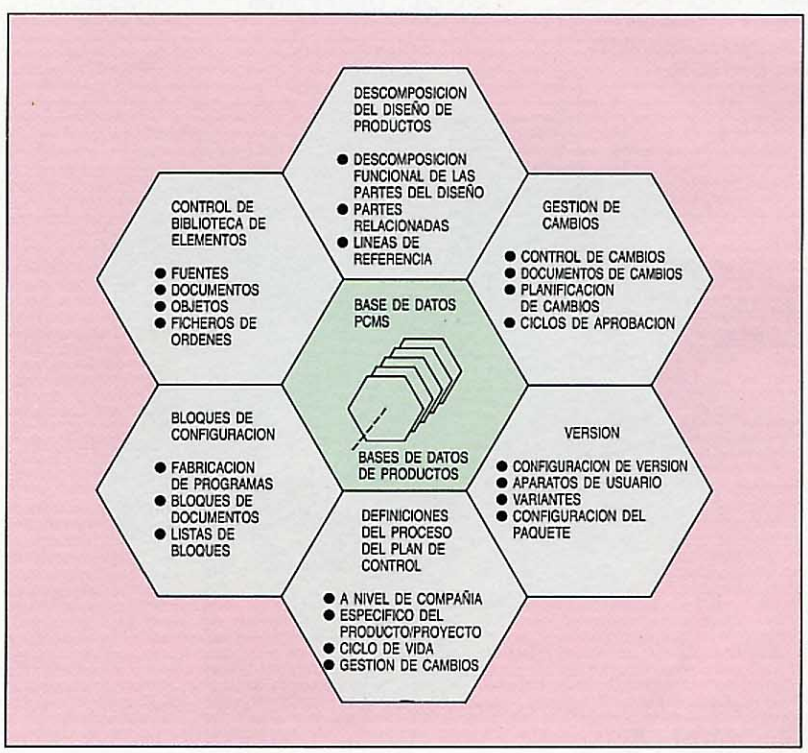


Figura 3
Habitualmente se han adquirido herramientas para atender necesidades específicas, que no se ha podido hacer trabajar juntas.

- roles de usuarios del proyecto
- destinatarios de las versiones del producto (fabricación, otras compañías de ingeniería, usuarios finales, etc.).

Cada aspecto posee atributos que describen su identidad y contenido (p. ej., propietario, fecha de creación, descripción), y además se relaciona con otros para así aportar las perspectivas que requiere un equipo de proyecto, como pueden ser los documentos de cambios que afectan a la documentación editada con el paquete de la configuración para un país concreto.

Figura 4
En el entorno SDE pueden operar a la vez diversos sistemas que utilicen la misma base de datos del producto.



Una base de datos del PCMS puede controlar al mismo tiempo muchos productos. En este contexto, producto puede ser una parte del diseño o un elemento que será utilizado en otros productos. Como ejemplo, un sistema operativo es en sí mismo un producto y, además, parte del conjunto de programas operacionales de una PABX.

Según evoluciona un producto añadiendo nuevas características y eliminando faltas, la base de datos rastrea todos los cambios y guarda un historial de ellos, expresando *qué cosa* se realizó, y *cuándo, quién* y *por qué* lo hizo.

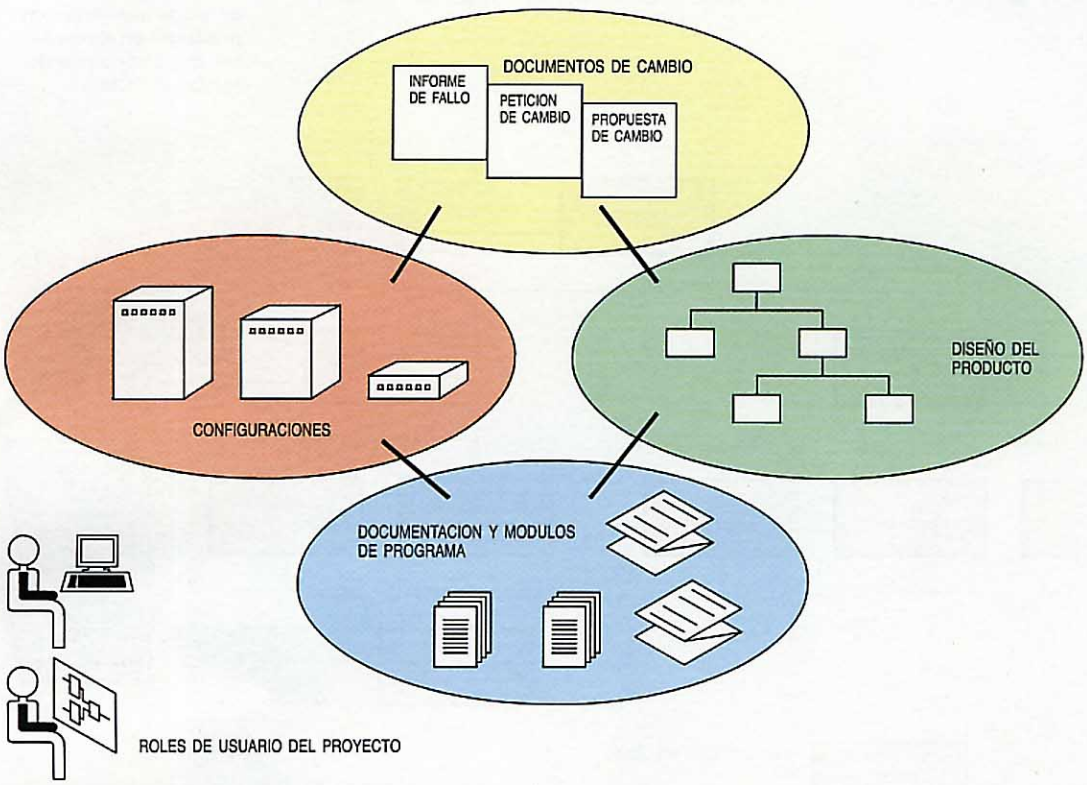


Figura 5
El PCMS proporciona un modelo de producto que describe todos los elementos integrantes de un sistema.

Planes para el control de productos

El PCMS controla un producto dado conforme a un conjunto de reglas fijadas por el responsable del mismo. Cada producto dentro de la base de datos PCMS se rige por sus propias reglas, que pueden diferir de las de otros productos.

Con el fin de servir a proyectos de diferente magnitud, el PCMS admite dos niveles de plan de control:

- plan de control que define las reglas y procedimientos normalizados para toda la Compañía (planes de documentación de equipos físicos, etc.)
- plan de control a nivel de producto, que define las reglas, procesos y configuraciones de vida útil adoptados.

El sistema PCMS da medios para ajustar el plan de la Compañía de modo que apoye las prácticas y métodos de trabajo de la misma, con lo cual todos los productos que después se vayan creando heredarán, por omisión, el referido plan de control. No obstante, el responsable del producto tiene medios para acomodar dichas prácticas y métodos de trabajo a exigencias específicas, o ajustar el nivel de control al volumen del proyecto. El PCMS reconoce así que los proyectos menores en los que participen, por ejemplo, cinco personas, necesitan un grado de control diferente que otros donde tal vez trabajen 100 o más personas.

Un plan de control identifica:

- tipos de elementos físicos a controlar
- procedimientos para gestión de los cambios
- planes de documentación de producto y requisitos en cuanto a números de pieza
- controles de elementos físicos, como aprobaciones de documentos, revisiones y pruebas de módulos de programas
- funciones de los usuarios y reglas de control de acceso.

Roles en un proyecto

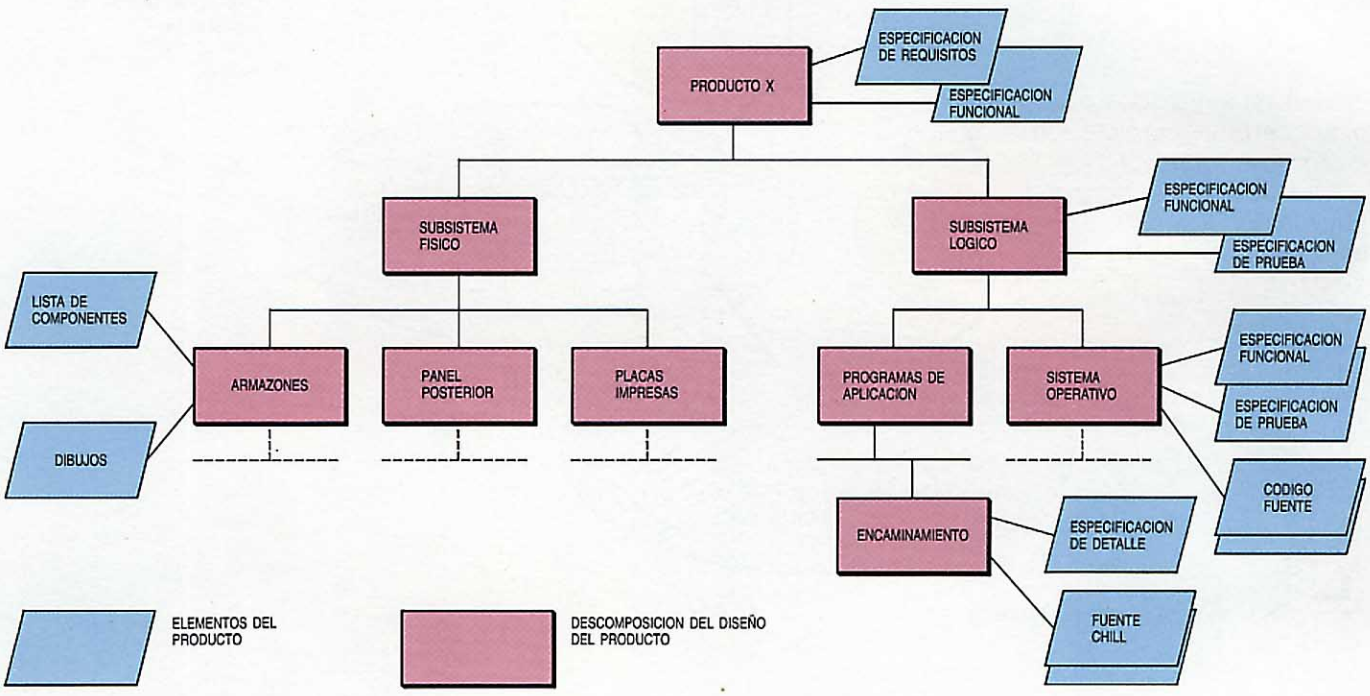
Las personas que trabajan en un proyecto tienen asignados roles dentro de una estructura de producto (el que desarrolla, el que prueba, etc.). Por ejemplo, GARCIA es el que DESARROLLA EL SUBSISTEMA DE PROGRAMACION. El rol de una persona la faculta para ciertas funciones dentro de un área definida del producto, tales como diseñar nuevas piezas, crear y cambiar elementos, o arreglar fallos en ese área. Algunas funciones pueden exigir que intervengan al mismo tiempo personas con roles diferentes, por ejemplo, cuando el diseñador crea unas especificaciones que ha de aprobar el responsable del producto.

El sistema PCMS aporta mecanismos para definir y asignar los roles de los participantes en un proyecto, controlando conforme a ellos el acceso a los datos y las operaciones, y para definir y apoyar las aprobaciones de los ciclos de vida de los elementos en consonancia con los referidos roles (p. ej., los usuarios PROBADORES aprueban cambios en los programas).

Desglose del diseño del producto

Durante el diseño y desarrollo, un producto se divide inicialmente en subsistemas, como el equipo físico y la programación. Seguidamente se subdividirá todavía más en áreas funcionales (p. ej., bastidor, placas de circuito impreso, controlador del sistema), y éstas, a su vez, en otras más detalladas o en elementos del diseño, que se realizarán como módulos de programas o de equipo físico. El PCMS faculta a los que diseñan y a los que desarrollan para describir la estructura funcional y las partes del diseño de un producto. No obstante, dado que la estructura y el diseño iniciales tienden a cambiar a medida que evoluciona el producto, el PCMS da también medios para rastrear cualquier cambio que se realice.

Figura 6
Modelo que muestra cómo se subdivide un producto en elementos cuyo seguimiento realiza el PCMS.



Control de la biblioteca de elementos del producto

El diseño y desarrollo del producto origina elementos físicos que, o bien describen sus características (requisitos, funciones), o bien realizan sus funciones (programas fuente, placas de circuito impreso). Los elementos se refieren a partes del diseño del producto: una especificación para el subsistema del equipo físico, o un programa fuente para el módulo de encaminamiento.

El PCMS controla todos los elementos destinados a un producto y va siguiendo su existencia como parte del modelo de ese producto (Fig. 6). Aquellos elementos contenidos en forma digital en el ordenador principal, como la documentación, se almacenan dentro de bibliotecas controladas por

el sistema. En el caso de documentos que sólo existan en papel o como esquemas, el sistema almacena un identificador que define sus atributos (por ejemplo, nombre, tipo, edición y estado) y la localización de dicho documento.

En el PCMS se guarda un historial de cada elemento en cuanto a sus diversos estados (desarrollado, probado, entregado) y versiones, y además se asegura que mientras se modifica un elemento ningún otro usuario puede cambiarle. El acceso a los elementos se controla de acuerdo con las funciones de los usuarios del producto.

Líneas de referencia del producto

Una "línea de referencia" es un inventario del estado de un producto en un momento concreto de su ciclo de vida. Por lo tanto las "líneas de referencia" captan la estructura actual del producto junto con sus elementos correspondientes (véase la figura 7), y deberían utilizarse como el principal interfaz entre las diferentes actividades de los roles implicados, aportando una clara definición de todos los elementos y partes compatibles.

Un historial de las líneas de referencia muestra la evolución de un determinado producto. El PCMS incorpora mecanismos para comparar tales líneas e informar sobre sus contenidos, pudiendo además establecer líneas de referencia para los subconjuntos de un producto (subsistemas, partes).

Soporte de versiones del producto

Además de describir la estructura del producto y todos sus elementos, el responsable del mismo puede definir los paquetes de configuración que van a generarse para sus diferentes versiones (Fig. 8), referidos

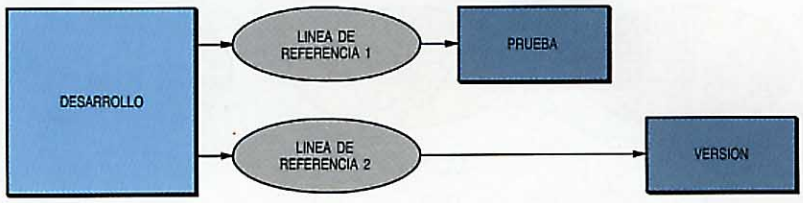


Figura 7
"Líneas de referencia", que indican el estado de un producto en un momento determinado de su ciclo de vida.

a las partes y a los elementos del diseño que realizan las exigencias específicas de esa configuración del producto. A partir de un diseño único pueden definirse muchos paquetes de configuración y variantes diferentes.

Para la prueba y la entrega puede establecerse una "línea de referencia" del paquete de configuración, que solamente capte aquellos elementos necesarios, junto con las aprobaciones oportunas (p. ej., programas ya probados, documentos admitidos).

El PCMS ofrece mecanismos para identificar los elementos a incluir en un paquete que se entrega (p. ej., programas ejecutables mas no programas fuente), así como las personas y funciones que deban recibir esa versión. De acuerdo con ello se produce el paquete de la versión, y se le carga en cinta magnética que lleva adjunta la identificación de la versión destinada a unos determinados usuarios. El seguimiento de las versiones permite analizar la repercusión de un cambio; por ejemplo: ¿a quién afectará el cambio de una parte del documento XYZ?

Construcción de la configuración de programas

Un paquete de configuración de un producto viene normalmente descrito por sendas configuraciones de equipo físico y de programas. El PCMS ofrece una facilidad

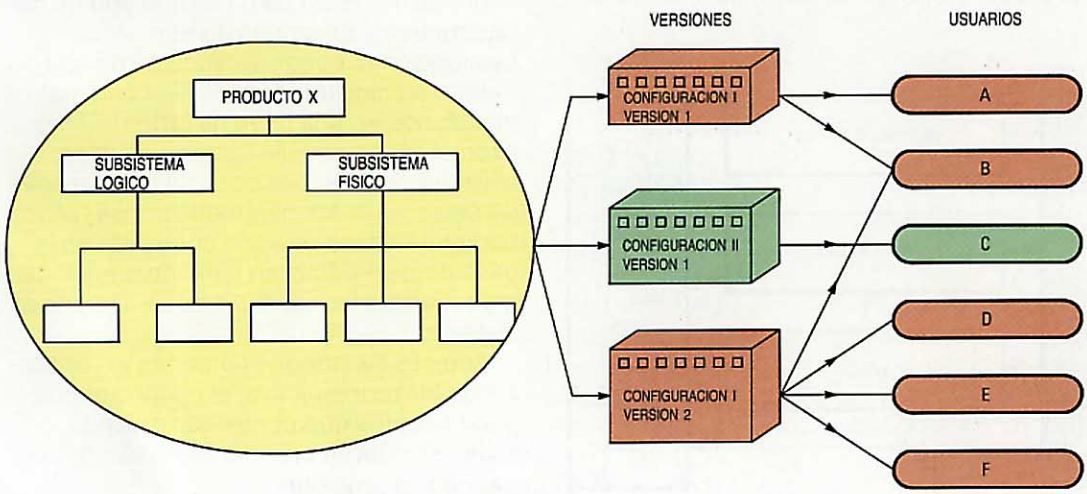


Figura 8
Soporte mediante el PCMS de las nuevas versiones de un producto.

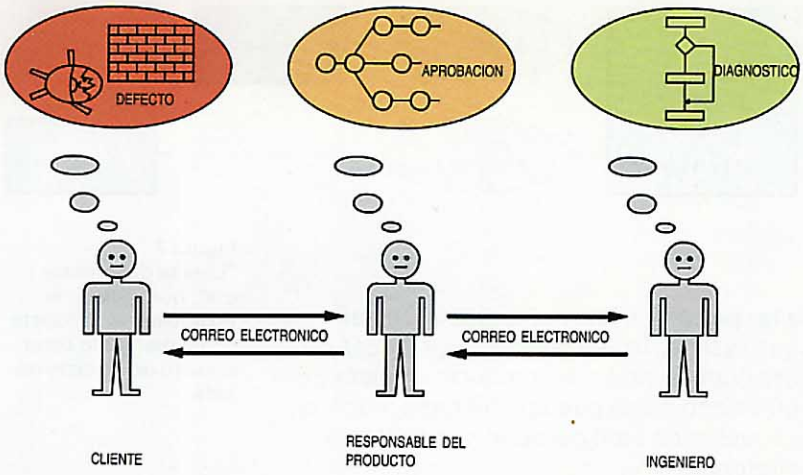


Figura 9
El PCMS ofrece toda clase de medios para el tratamiento de informes de fallos, solicitudes de cambio y propuestas de cambio.

para construir un paquete de programas a partir de módulos existentes; el usuario puede así definir los elementos en lenguaje fuente que constituyen un programa ejecutable, y generar las órdenes primarias que compilan y unen los módulos de programa necesarios.

Al seleccionar elementos para construir un paquete, el usuario puede especificar una versión concreta o bien la más reciente. El proceso de construcción registra información sobre la versión de los elementos procesados, las herramientas requeridas y los resultados obtenidos; tal información permite reconstruir paquetes formados con anterioridad y puede servir para producir documentos típicos, tales como una lista de los módulos de programación que se utilicen.

Esta facilidad puede también aplicarse para construir la documentación del producto, realizando en caso necesario tratamiento de documentos, tal como formatación en página o combinación de texto y gráficos.

Gestión de cambios

El PCMS incorpora medios para captar y procesar informes de fallos, solicitudes de cambios, propuestas de cambios, y la mayoría de los demás tipos de documentación de cambios. La información relevante se procesa de acuerdo con los procedimientos de control de cambios del proyecto, que habrán de identificarse en los datos sobre el plan de control del producto almacenados en la base de datos PCMS. El procesamiento de un documento de cambio conlleva la notificación a las personas adecuadas (al responsable del proyecto, al ingeniero de diseño) de haber recibido nuevos documentos, y la petición a los mismos de su valoración (aprobación, rechazo, etc.). La figura 9 muestra una situación típica en la cual las diversas personas implicadas son informadas automáticamente por medio de correo electrónico.

La información del cambio va asociada a las configuraciones y partes de diseño del producto afectadas. El PCMS aporta medios para consultas e informes que muestran los cambios y su estado (por ejemplo, en revisión, en ejecución, completado) en relación con cualquier producto, subsistema o área de diseño.

Planificación de los cambios

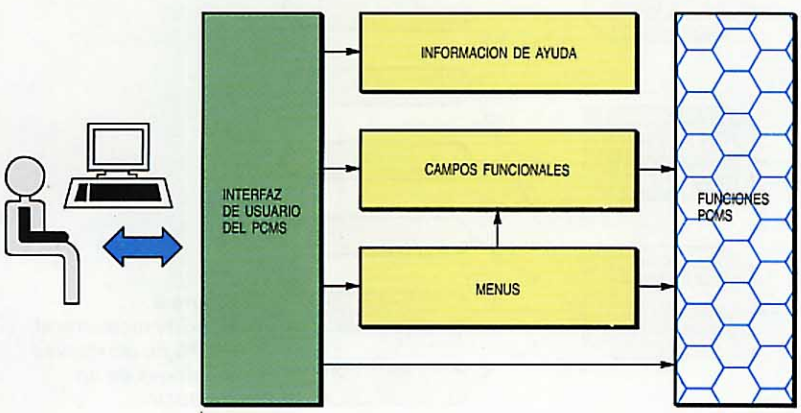
Utilizando el PCMS pueden planificarse futuras versiones de configuraciones del producto. Toda modificación de un producto debe describirse mediante documentos de cambio (propuestas y solicitudes de cambio, informes de fallos). Las facilidades PCMS permiten al responsable del producto planificar un número cualquiera de futuras versiones de dicho producto, e identificar los cambios y correcciones de faltas que han de realizarse en cada versión.

Interfaz de usuario del PCMS

Las funciones PCMS se llaman a través de un interfaz de órdenes (Fig. 10) que aporta información de ayuda para cada uno de los parámetros y funciones de este sistema. Las funciones pueden invocarse directamente, suministrando todos los parámetros necesarios en una línea de orden. Como alternativa, es posible llamar funciones utilizando los campos de la pantalla capaces de seleccionar los parámetros necesarios, partiendo de información contenida en la base de datos. Existen también menús que muestran las facilidades de que dispone el sistema.

Además de introducir órdenes y ejecutarlas de forma interactiva, el usuario puede crear ficheros que contengan muchas órdenes PCMS y ejecutarlos en el modo de tratamiento por lotes.

Figura 10
El PCMS está provisto de un atractivo interfaz mediante órdenes, el cual ofrece al usuario abundante información de ayuda.



Conclusiones

Aunque el sistema PCMS fue diseñado y desarrollado en respuesta a las exigencias internacionales de ITT, ha demostrado igualmente que puede satisfacer las necesidades específicas de las distintas compañías asociadas. Ello se debe a que la gestión del ciclo de vida del producto es esencialmente igual, y no depende de la magnitud del proyecto, ni de que el desarrollo se distribuya entre varias compañías o entre varios grupos de una misma compañía.

El PCMS ha madurado dentro del marco del SDE. Su arquitectura, que separa la base de datos del producto de los medios de gestión y de las reglas del proceso de desarrollo, permite utilizarle con una gran variedad de productos, además de poder añadir funciones gestoras del proyecto y del producto a medida que surja la necesidad. Están en desarrollo nuevos sistemas "incorporables" que harán posible apreciar el "estado de salud" de un producto desde el punto de vista del responsable del proyecto, del especialista en evaluación de programas, y del responsable del desarrollo[1].

El PCMS pasa ahora por la deseada transición, desde ser el eje del SDE hasta aplicarse en otras disciplinas de ingeniería. En particular, se está analizando su inclusión en los entornos de soporte de ITT para el diseño de placas de circuito impreso, y de circuitos VLSI a medida y parcialmente a medida.

Agradecimientos

Los autores desean agradecer a todos los responsables de desarrollo y de diseño de SDE en ITT Europa sus contribuciones a dichos SDE, y en especial a P. Westby y T. Johannessen de Standard Telefon og Kabelfabrik, a quienes, junto con ITTE ESC, cabe atribuir el desarrollo del PCMS. Los autores también agradecen a W. Zwickl y W. Würth de ITT Austria su constante ayuda.

Referencias

1 D. Hunter y W. Kobitzsch: Módulo interfaz de medidas para el entorno de desarrollo de programas: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 3/4, págs. 256–258 (en este número)

Tanl Haque se graduó en 1969 por la Universidad de Londres. Tras un periodo inicial como ingeniero eléctrico en Westinghouse y en ICL, se dedicó a ingeniería de programación. Ingresó en Standard Telecommunications Laboratories, donde se le encomendó la programación de soporte de la central METACONTA* 10C. En 1979, fue transferido al ITTE ESC para acometer las actividades de herramientas de programación para el Sistema 12 local. Ha dirigido numerosos proyectos, incluyendo importantes partes del entorno de desarrollo del Sistema 12 y la programación del depurador concurrente ADA, y encargándose de iniciar el proyecto SDE. En 1985 se le encomendó el programa técnico europeo de lanzamiento del ordenador personal ITT XTRA XP*. Actualmente el Sr. Haque es responsable de la dirección del European Product Support Centre y la dirección global de productos de SDE.

Juan Montes nació en Murcia, España, en 1951. Estudió en la Escuela Técnica Superior de Ingenieros de Telecomunicación de Madrid y continuó sus estudios en el Hatfield Polytechnic de Inglaterra, donde consiguió un HNC en informática y un BSc en ciencias del ordenador. En 1969 ingresó en el Centro de Investigación de Standard Eléctrica, trabajando en ayudas de ordenador para aplicaciones soporte de diseño, ingeniería y fabricación de centrales PENTACONTA*. Posteriormente pasó a Standard Telecommunication Laboratories, Harlow, donde trabajó en sistemas CAD integrados para placas de circuito impreso. Tras un periodo en Rank Xerox, el Sr. Montes entró en ITTE ESC, Harlow, donde actualmente dirige el desarrollo de programación para SDE.

* Marca registrada del Sistema ITT

Módulo interfaz de medidas para el entorno de desarrollo de programas

Se está desarrollando un módulo interfaz de medidas que permitirá a un responsable de proyecto extraer y analizar datos a lo largo del ciclo de vida de los programas. Este módulo trabaja eficazmente como un revisor de proyectos independiente, aportando la información necesaria para evaluar el progreso de un proyecto y ejercer con prontitud eventuales acciones correctivas.

D. Hunter
ITT Europe Engineering Support Centre, Harlow, Inglaterra
W. Kobitzsch
Standard Elektrik Lorenz AG, Stuttgart, República Federal de Alemania

Introducción

A finales de 1984 ITT Europe acometía un proyecto para especificar, desarrollar y llevar a la práctica un entorno genérico de desarrollo de programas (SDE, *software development environment*). Su núcleo es el PCMS, sistema gestor de la configuración de productos en cuya base de datos se describe la situación de muchos proyectos diferentes, así como las actividades que tienen lugar durante su puesta en práctica (por ejemplo, generación de informes sobre defectos y su eliminación)[1]. Esta información es compleja, pero importante cuando se desea terminar los proyectos con éxito y a tiempo. El MIM (módulo interfaz de medidas) está concebido para recoger esos datos, relacionarlos y presentarlos a los responsables de forma fácilmente comprensible.

Los gráficos son mejores que los listados para presentar información de un modo sucinto; una investigación reciente de ITT[2,3,4] muestra el tipo de información que deberían exhibir esos gráficos. En esa investigación se examinaron alrededor de 100 proyectos de ITT, señalando unas doce medidas que atañen a su progreso: por ejemplo, la cantidad de código probada en la realidad frente a la prevista inicialmente. Se asignó el término "indicador de gestión", de procedencia económica, a la combinación de un gráfico con algún informe de incidencias.

El MIM presentará los gráficos de algunos de los indicadores de gestión, pero al principio no informará sobre las incidencias, ya que para ello hay que comparar los resultados del proyecto con los datos históricos.

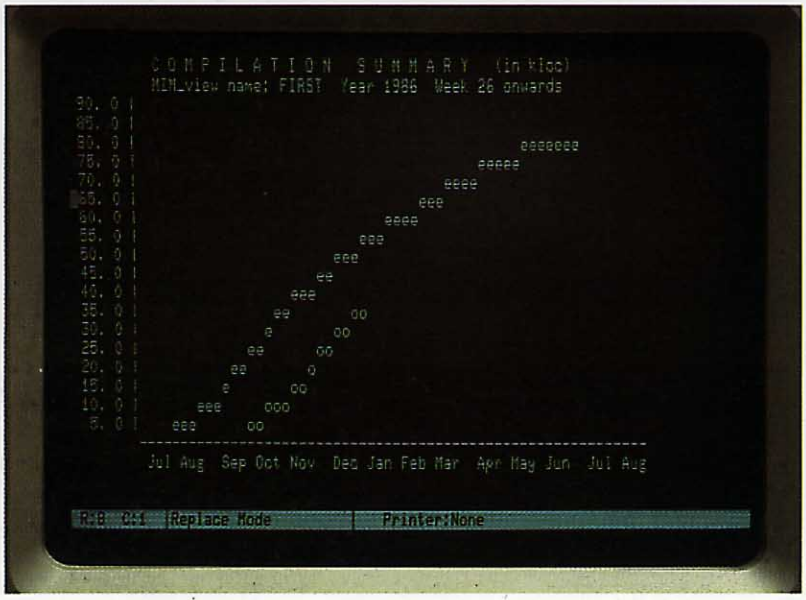


Figura 1
Resumen de compilaciones mostrando el número esperado de líneas de código y el número de las realmente producidas, a lo largo de 14 meses.

Necesidad de medidas

Al comenzar un proyecto, el responsable del mismo aprovecha su conocimiento del riesgo contraído, es decir, la probabilidad de no acabar el proyecto a tiempo y dentro del presupuesto. Cuando estima que el riesgo es muy alto, intenta llegar a un compromiso con el cliente variando, bien la dotación de personal, o bien la duración del proyecto, desde una posición negociadora

más sólida por conocer las ventajas de cada alternativa. Se logra así ajustar el precio de la propuesta de modo más competitivo.

Aunque los programas de estimación de coste son cada vez más asequibles, sus predicciones suelen depender de datos del exterior. Dada esta necesaria relación con el ámbito de cada proyecto, ITT ha reunido gran cantidad de información sobre proyectos terminados, utilizable para calibrar los programas de estimación de coste.

Una vez iniciado el proyecto, se pueden comparar las características observadas con las esperadas, lo que tal vez conduzca a una pronta corrección. Por ejemplo, el MIM indicará si el número de informes de fallo para el volumen de código que se desarrolla es excesivo o no.

Debe advertirse que el MIM requiere ciertos datos del SDE exterior, incluyendo predicciones obtenidas de los paquetes de estimación de coste, e información de horas-hombre a partir de las hojas de trabajo.

Módulo interfaz de medidas

El fin principal del MIM es recopilar estadísticas semanales acerca de los tamaños de los módulos y los informes de fallos, y presentar diversos gráficos de datos históricos a los responsables de proyectos. Esta información esencial sobre la "salud" de un proyecto afianza el control sobre su progreso. Como ejemplo, la figura 1 muestra un gráfico resumen de compilaciones, ilustrando los tamaños esperados y observados de todos los módulos de código fuente, a lo largo de 14 meses. Esta selección particular se llama "MIM view" (visión MIM), y en este ejemplo su nombre es "FIRST" (primera). El MIM puede también aportar gráficos con los informes de fallo (originados y resueltos), tasas de error (por cada 1.000 líneas de código) y productividad (líneas por hombre-año), todos ellos en escala de tiempos similar. Estos son los "indicadores de gestión" antes mencionados.

Aunque una oficina de proyectos bien organizada pudiera conseguir los mismos resultados con los clásicos métodos de "papel y lápiz", ello no sería económico. Por el contrario, pueden recogerse datos de modo automático teniendo una base de datos, y así el responsable del proyecto dispone siempre de la información que necesite.

En primer lugar, dicho responsable adquiere los derechos de uso del MIM, y decide cuál información desea rastrear a lo largo del proyecto. El PCMS le permite escoger un grupo de elementos para formar su visión MIM creando una tabla apropiada en la base de datos y guardando en ella sus nombres. Generalmente, los elementos serán ficheros de código fuente, pero se podría incluir cualquier otro relacionado con un informe de fallo. A continuación, elige libremente la fecha de comienzo y duración de la visión MIM y los factores de escala para los gráficos, de tal modo que éstos puedan modificarse después en caso necesario. Finalmente, con ayuda de un programa de estimación de costes, suministra datos semanales sobre la cantidad de código que se espera escribir. Se requiere, pues, cierto trabajo administrativo para cada visión MIM que el responsable desee ejecutar.

Una vez creada dicha visión, se ha archivado ya información suficiente para que el MIM atribuya datos de medidas a los ficheros históricos que guarda para esa visión, y ello normalmente se ejecutará como tarea semanal por lotes.

Existen varias técnicas para medir el tamaño de los módulos de código fuente, como es la de analizar el resumen de las compilaciones efectuadas. Sin embargo, en el presente caso se decidió contar líneas de código fuente con independencia del compilador, y descartar las líneas de comentarios del modo apropiado a cada lenguaje, siendo idénticos los convenios para el CHILL y el PLM. El módulo contador examina el nombre del fichero fuente (en realidad la primera letra de su nombre completo), y obedece a las reglas de comentario adecuadas.

Este procedimiento, extensible hasta incluir tamaños de documentación, tiene la ventaja de no depender de cambios en el compilador. No obstante, no responde a preguntas sobre la calidad del código fuente, que por suerte se contestan mediante consultas a la base de datos.

Todo elemento del PCMS tiene un estado, como es el de *inicio* al empezar la codificación, que se va actualizando a lo largo de la vida del proyecto. El MIM sólo selecciona los ficheros de código fuente cuyo estado es el apropiado; por ejemplo, en *compilación depurada* o en *unidad probada con éxito*. Así se pueden reunir datos para un gráfico que indique cantidad de código compilado o probado.

El informe de fallos solamente implica el acceder a las tablas de PCMS, cuyas facilidades de base de datos se explotan al máximo pues permiten elaboradas consultas sobre factores comunes a varias tablas. Para cada elemento de la visión MIM se hace el recuento de todos los informes de fallo que le atañen, tanto originados como resueltos. En un gráfico que combina ambos tipos de informes se muestra el ciclo de vida de un fallo típico y el número de

ellos todavía persistentes en cualquier momento.

Otra actividad del MIM es vigilar el uso de las herramientas en el SDE. La figura 2 presenta un histograma de los cómputos acumulados a una cierta fecha. Ello advierte al responsable del proyecto, bien del excesivo uso de las herramientas, o bien de su infrautilización, aunque para utilizar con eficacia estos datos él tendría que conocer las maneras de depurar programas de su gente, y también profundizar un poco en el problema. Asimismo deberá haber posibilidad de poner a cero los cómputos acumulativos.

Pueden recogerse datos sobre el uso de las herramientas interceptando su invocación por los usuarios, y registrando tanto el nombre de la herramienta como el del fichero sobre el que actúa, los cuales a su vez sirven para actualizar el oportuno cómputo en todas las visiones MIM afectadas. De nuevo hay que consultar a la base de datos para relacionar estos dos nombres con las visiones MIM correspondientes. Como asistencia final, el MIM prepara un resumen al terminar el proyecto, donde se señala el volumen de código realmente entregado y el que sólo tenía finalidad de soporte, además del número de fallos relativos a código y a documentos, y la tasa de fallos por cada 1.000 líneas de código. Esta última medida refleja la calidad del código suministrado.

Figura 2
Histograma de la utilización de las herramientas SDE indicando cuántas veces se ha utilizado cada herramienta hasta la semana 43.

Futuros desarrollos

A la hora de determinar la proxima etapa del desarrollo del MIM será esencial recoger impresiones de los usuarios. Dicha etapa podría muy bien incluir ayuda para introducir información de horas-hombre, lo que facilitaría la preparación de gráficos de productividad (líneas de código por hombre-año). No obstante, los sistemas de información financieros son complejos y difieren entre las compañías ITT.

Otra mejora podría ser el incorporar informes de incidencias, en los indicadores de gestión antes mencionados. Con ello el MIM pasaría de ser un mero productor de información gráfica a revisor crítico independiente de proyectos.

Referencias

1 T. Haque y J. Montes: Soporte de productos basados en microprocesadores durante su vida útil: *Comunicaciones Eléctricas*, 1986, volumen 60, nº 3/4, págs. 248–255 (en este número).

2 H. Malec: Automated Data Collection and Reporting System for Programming Productivity, Quality, and Cost Baselines: *Proceedings of the ITT Conference on Programming Productivity and Quality*, Nueva York, junio 1983, págs. 134–145.

3 A. Albert: Programming Leading Indicators: *Proceedings of the ITT Conference on Programming Productivity and Quality*, Nueva York, junio 1983.

4 R. Wolverton, J. Vosburgh, A Albert, S. Hoben e Y. Liu: 1982 Programming Measurement Baseline, Report of ITT Programming Productivity, Quality and Cost, with Analysis of Productivity and Quality Improvement Factors: *ITT Applied Technology*, diciembre 1983, ITT Stratford.

Don Hunter se graduó en físicas en la Universidad de Cambridge. Al comienzo de su carrera en STL participó en comités técnicos de ITT relacionados con las compañías ITT de Bélgica, Francia, Alemania y Estados Unidos. Seguidamente pasó a CAP, llegando a ser Director Técnico y controlando proyectos sobre compiladores, redes de área local y procesos tolerantes de fallos. A su regreso a STL dirigió la planificación PERT de recursos del sistema operativo en tiempo real multimínimo. El Sr. Hunter trabaja en la actualidad en ITTE-ESC, dentro del equipo SDE responsable de las medidas. Durante varios años ha servido en el Comité de Cifrado de Datos del British Standards Institute, y es experto principal del Reino Unido en el grupo de trabajo de la ISO sobre criptografía pública.

Werner Kobitzsch nació en 1947. Se graduó MS en comunicaciones eléctricas por la Universidad de Stuttgart. En 1973 entró en el Centro de Investigación de SEL donde trabajó en evaluación de microprocesadores y diseño de sistemas. En 1981 fue nombrado director técnico del proyecto de un sistema de mecanización de edificio del aeropuerto Changi, de Singapur. Desde 1983, el Sr. Kobitzsch dirige un departamento de programación que desarrolla sistemas de control remoto, terminales de voz y datos, y avanzados sistemas de enclavamiento ferroviario.

Generación de sistemas ejecutivos en tiempo real

GERTEF es una herramienta de generación de programas ejecutivos en tiempo real a partir de módulos ya comprobados. Ofrece importantes ventajas, tales como tiempo de desarrollo reducido, mayor fiabilidad y mejor rendimiento.

C. S. Baradello
G. Carloni
Centro de Investigación de FACE, Pomezia, Italia

Introducción

En la ingeniería de equipo es práctica común utilizar bloques normalizados LSI/VLSI para acelerar el diseño, reduciendo así al mínimo los costes de desarrollo. Parece, pues, razonable esperar parecidas ventajas en el desarrollo de la programación cuando los paquetes de programas se construyen a partir de módulos ya existentes y probados. Esto impulsó al Centro de Investigación de FACE a emprender el desarrollo del GERTEF, herramienta de programación capaz de construir RTE (rutinas ejecutivas en tiempo real) a partir de módulos normalizados.

La RTE es un núcleo lógico que actúa como interfaz entre los programas de aplicación y el equipo de la unidad central de proceso; en general, aporta las funciones comunes (p. ej., programación de tareas, gestión de temporizaciones, gestión de zonas de memoria, tratamiento de interrupciones) que requieren la mayoría de las aplicaciones. El GERTEF es el primer sistema en ITT que genera programas RTE adecuados a una arquitectura de microprocesador prefijada, integrando muchos módulos pequeños en un paquete único de programas capaz de ejecutar fiable y eficientemente las funciones necesarias para la aplicación[1,2]. Se asemeja así al diseño del equipo, en el que un circuito VLSI sustituye a miles de componentes individuales.

Este enfoque presenta las siguientes ventajas:

- Reutiliza módulos ya probados para construir paquetes fiables y de alta calidad.
- Se aplica al ciclo entero de diseño de RTE, desde la descripción de arquitecturas del equipo y requisitos de la programación hasta la producción de código ejecutable.
- Permite a los técnicos experimentados generar una RTE con la funcionalidad requerida, en no más de una hora.
- Reduce de dos a cuatro hombres-año el desarrollo de la programación, lo cual acorta el diseño en unos seis meses.
- Disminuye redundancias en los programas.

El desarrollo del GERTEF se orientó a conseguir tales beneficios, habiéndose demostrado además que las RTE así producidas se comportan mucho mejor que las desarrolladas partiendo de la nada.

Principio del sistema GERTEF

El GERTEF apoya el diseño de RTE para las actuales arquitecturas de microordenador, y podría ampliarse su concepto para cubrir arquitecturas futuras. Ayuda al ingeniero de sistemas a verificar la solución elegida en el momento de definir la arquitectura del sistema, mientras que el programador se vale de él para construir las RTE ejecutables. La figura 1 muestra la estructura lógica de una RTE generada por medio del GERTEF.

La metodología de diseño GERTEF se sirve de un juego de herramientas asociado para desarrollar una RTE, reutilizando módulos de programación donde se recoge la experiencia anterior en diseño y realización de RTE.

Como se ilustra en la figura 2, el GERTEF consta de las siguientes partes[3]:

- interfaz de usuario
- lógica de control
- generador de descripciones
- base de datos

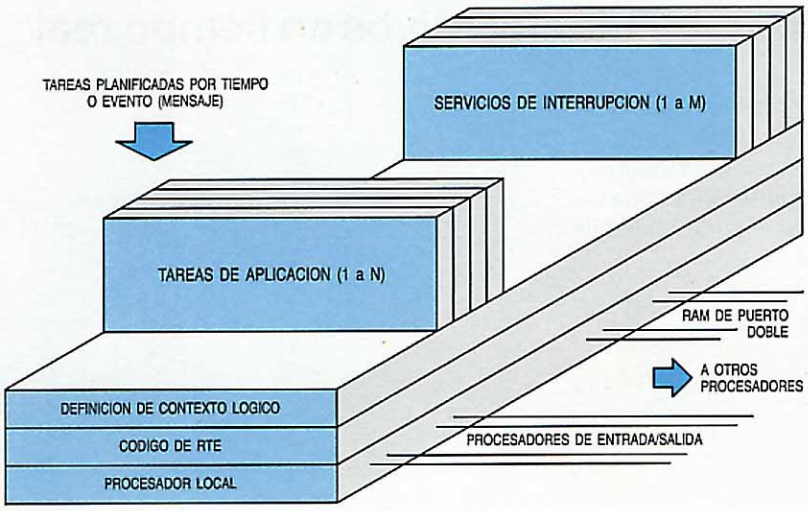


Figura 1
Estructura lógica de una RTE generada por el GERTEF.

- selección y extracción de módulos
- definición del contexto de equipo/programación.

La figura 3 muestra la arquitectura de un entorno de desarrollo de microsistemas, más general, orientado hacia FSM (máquinas de estados finitos) y FMM (máquinas de mensajes finitos). Dentro de dicho entorno, el "nuevo" diseñador captará la arquitectura y requisitos de los sistemas en un lenguaje similar al de las FSM/FMM, asumiendo finalmente los roles de arquitecto y programador del sistema.

Este método incrementará la eficiencia de las compañías ITT durante la fase de desarrollo de programación, creando un entorno integrado de desarrollo para los sistemas basados en microordenadores.

Características de las rutinas ejecutivas en tiempo real

Las RTE producidas por GERTEF están escritas en el lenguaje ensamblador de los sistemas basados en procesadores INTEL iAPX 88/86/188/186. En un entorno de múltiples procesadores, en el que varias CPU comparten una memoria común, la RTE proporciona un mecanismo de comunicación entre procesadores basado en interrupciones y "buzones" electrónicos. También vale un mecanismo similar para comunicar los iaPX 88/86 con los procesadores de entrada/salida. La RTE es responsable de gestionar los siguientes recursos físicos de la placa del procesador:

- tiempo de proceso de CPU
- subsistemas Intel, tales como el controlador de interrupciones, el generador de temporizaciones, y el procesador de entrada/salida
- periféricos internos programables (sólo para iAPX 188/186)
- memoria de datos (tipo RAM)
- comunicación con otras CPU
- comunicación con procesadores de entrada/salida.

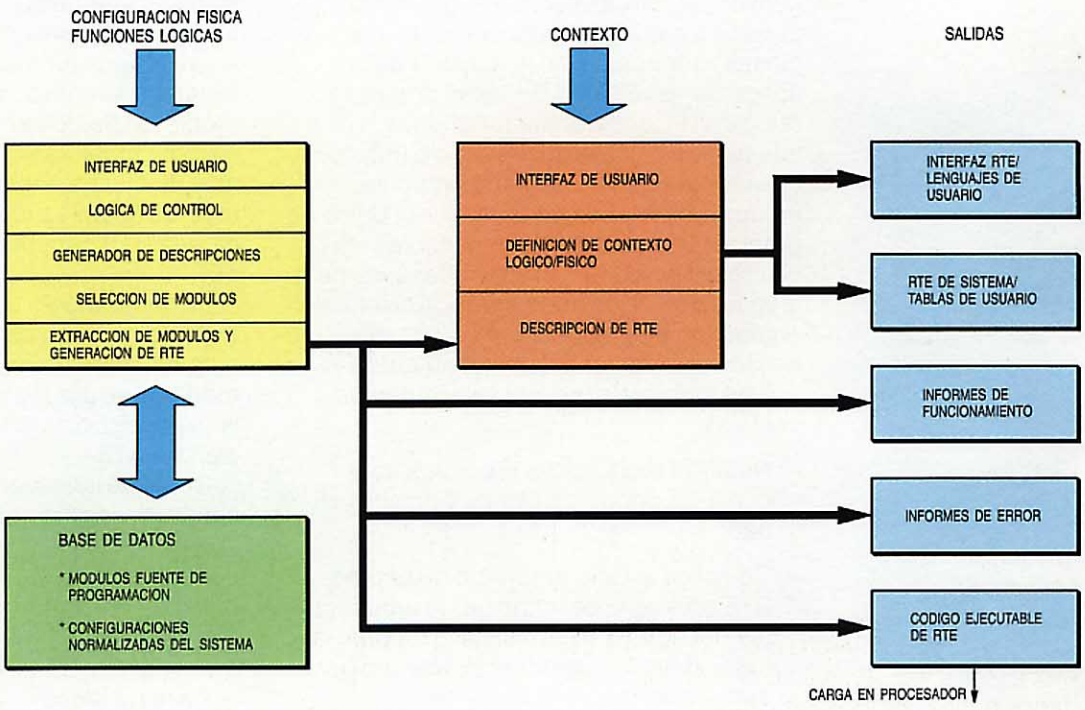


Figura 2
Estructura lógica interna de la herramienta GERTEF.

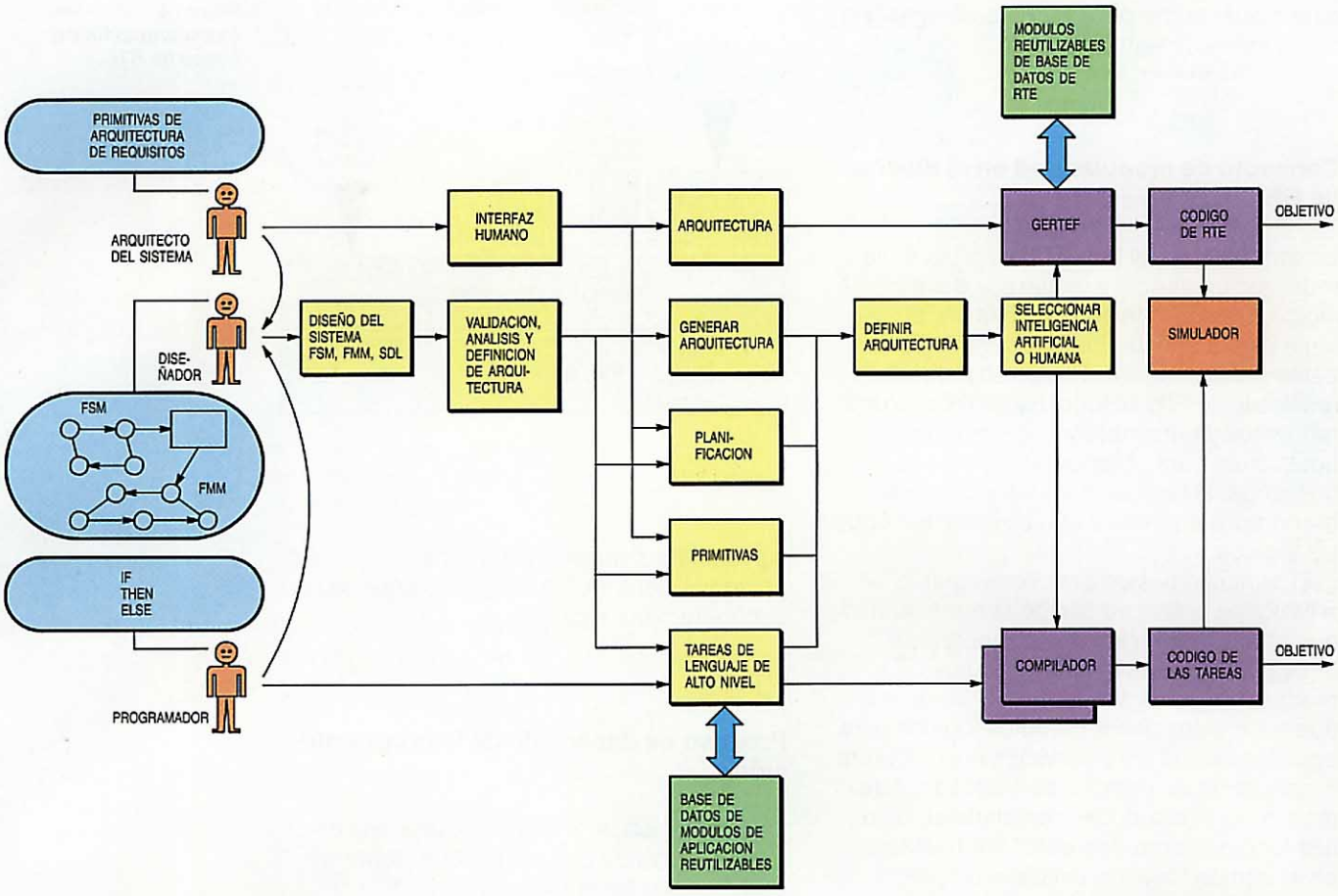


Figura 3
Entorno integrado de desarrollo de microsistema.

Gracias al GERTEF, el usuario puede generar una RTE para una arquitectura física particular que posea las funciones y primitivas básicas requeridas. La principal ventaja de tal enfoque es que optimiza el funcionamiento en tiempo real y la asignación de memoria, seleccionando únicamente los módulos necesarios para las funciones implicadas en la configuración objeto del diseño.

Además, el GERTEF apoya la gestión de los procesos coexistentes y cooperantes en un sistema, relacionados por los siguientes mecanismos de planificación[4,5]:

- ordenación por tiempo, para procesos más rápidos y más cortos
- ordenación por mensajes, para procesos más lentos.

Cuando ambos mecanismos coexistan será posible clasificar los procesos en actividades *prioritarias* (programadas por tiempo), para procesos "más rápidos", y actividades *de baja prioridad* (programadas por mensajes) para procesos "más lentos"[6,7]. Así se aprovecha mejor el tiempo de proceso disponible, satisfaciendo las prioridades de las diversas peticiones.

Asignación física de las RTE

La RTE puede ubicarse, como módulo aislado, en la RAM o ROM de cualquier procesador; no necesita enlazarse con los módulos usuarios excepto en el tiempo de encadenamiento a través de las direcciones apropiadas en la tabla de vectores.

Se necesita una gestión de interrupciones para los mecanismos de comunicación con otras CPU y procesadores o eventos de entrada/salida. Por lo tanto, el sistema buscado debe aportar medios para intercambiar señales de interrupción con otras CPU y procesadores de entrada/salida. La RTE puede atender un máximo de nueve controladores de interrupciones: estas señales proceden de eventos generados por la alarma de vigilancia, el temporizador, otras CPU o dispositivos externos.

La RTE gestiona directamente el controlador de interrupciones, tratando los eventos de interrupción mediante la correspondiente tabla de vectores. Por medio de las tablas de definición del sistema, el usuario puede definir el modo de operación para cada periférico físico programable, sin más que aportar la información del contexto físico. Todos los periféricos se pueden

direccionar como posiciones de memoria o direcciones de entrada/salida.

Concepto de modularidad en el diseño de GERTEF

La concepción del GERTEF se basa en la reutilización directa e inalterada del soporte lógico. De acuerdo con tal objetivo, el sistema está diseñado para generar RTE que traten protocolos ejecutados en procesadores tipo Intel. Ello se logra especificando los requisitos y ensamblando los módulos necesarios para obtener las funciones buscadas, al tiempo que se facilita un flexible compromiso entre equipo físico y módulos lógicos.

El principio básico para conseguir lo anterior es el fino detalle de la modularidad, que afecta tanto a los componentes del equipo como a las funciones lógicas. El proceso de generación de RTE permite al operador seleccionar módulos lógicos para aquellas funciones y servicios que requiera la aplicación de tiempo real buscada. Además, gracias a la doble modularidad, algunas funciones pueden optar por realizarse en el equipo (cuando exista) o en programas, decisión que se adoptaría al definirse las arquitecturas física y lógica.

Dependiendo de la configuración de equipo elegida y las funciones lógicas que se necesiten, la RTE se organiza según indica la figura 4. El GERTEF asegura que

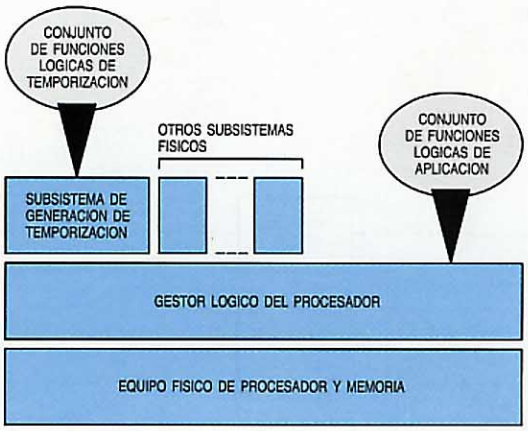


Figura 4
Modularidad física/ lógica de RTE.

las funciones requeridas (primitivas) tienen un apoyo del sistema independiente de la configuración física real.

Proceso de desarrollo de la programación

La generación de RTE es sólo una fase en el proceso total de desarrollo, que debería comprender las etapas siguientes:

- generación de RTE
- definición de contexto físico/lógico
- desarrollo de programa de aplicación en lenguaje de alto nivel

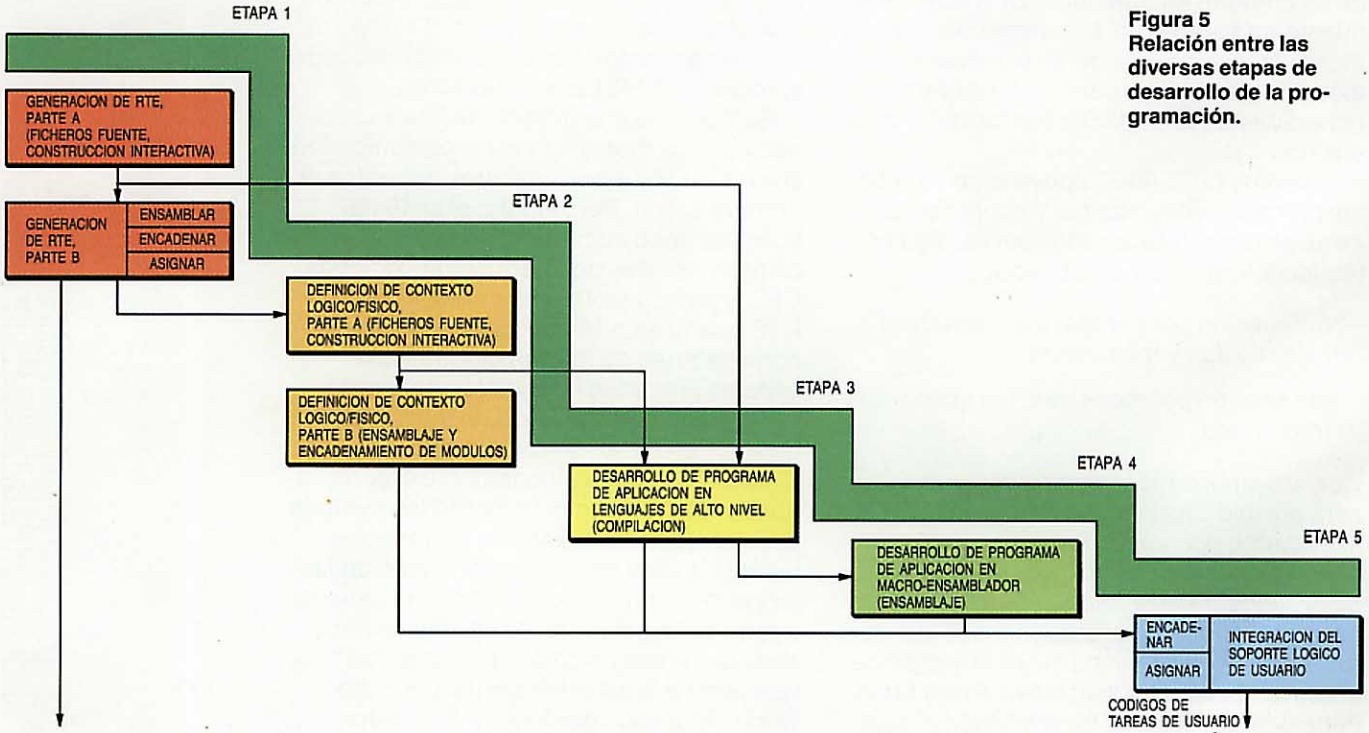


Figura 5
Relación entre las diversas etapas de desarrollo de la programación.

- desarrollo de programa de aplicación en lenguaje macroensamblador
- integración del soporte lógico.

Primera etapa: el ingeniero de sistemas introduce las funciones requeridas y la configuración física utilizando un procedimiento interactivo guiado por menú; el GERTEF selecciona luego los ficheros fuente de RTE apropiados.

Segunda etapa: el ingeniero de sistemas introduce toda la información relevante sobre el contexto físico/lógico (tareas de aplicación, tablas de direcciones, direcciones físicas, tablas de interrupción de vectores, etc.).

Etapas tercera y cuarta: todas las tareas de aplicación de usuario se desarrollan en un lenguaje de alto nivel (p. ej., PLM, C, CHILL) o un lenguaje ensamblador (ASM'86). En la práctica, estas etapas podrían desarrollarse en paralelo con las dos primeras.

Quinta etapa: se integran los programas de aplicación de usuario y el código objeto reubicable de la RTE.

La figura 5 indica la relación entre estas cinco etapas, y la figura 6 muestra, con cierto detalle, el proceso de generación de RTE.

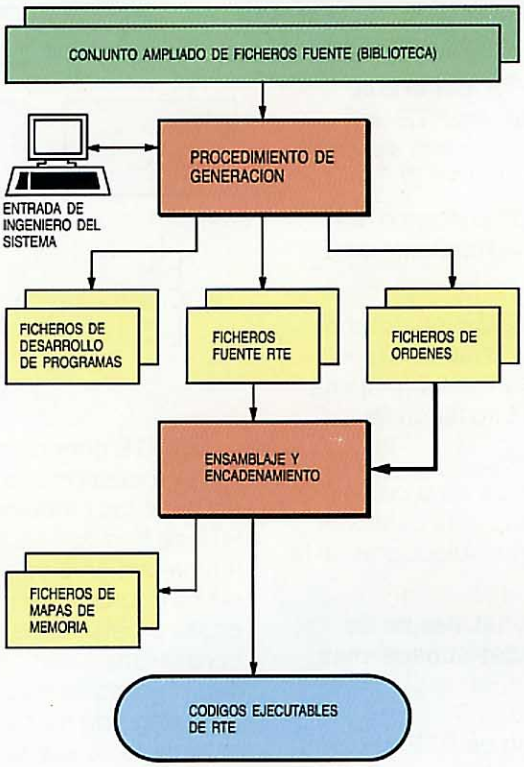


Figura 6
Flujo del proceso para generación de RTE.

Sistema GERTEF

En el GERTEF se ha recogido la experiencia del Centro de Investigación de FACE en el desarrollo de RTE para diversos proyectos. Mediante un menú principal único, que utiliza una terminología familiar a los técnicos de programación, puede accederse a funciones tales como descripción de RTE y su codificación, definición del contexto físico/lógico a través de las tablas RTE-usuario, desarrollo de programas del usuario y enlaces con el código de la RTE. De esta forma, el GERTEF dirige al usuario, paso a paso, a través del proceso de generación de RTE.

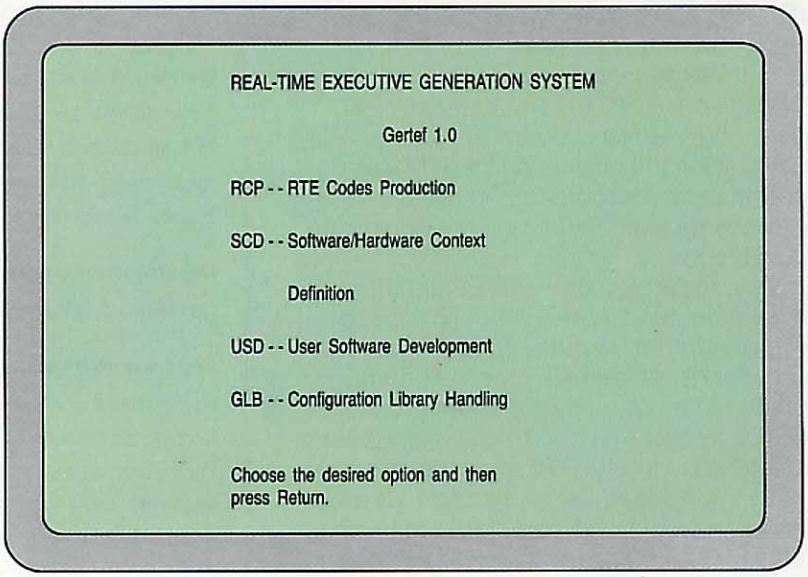
Menú principal

Se elige entre diversas opciones de diseño tomando del menú la característica deseada. Estas opciones permiten al usuario seleccionar un subsistema (menú principal en la figura 7), visualizar un formato en la pantalla o realizar una tarea. En la figura 8 se muestran los dos primeros niveles funcionales del árbol de selección de GERTEF.

Subsistemas GERTEF

Un subsistema es una aplicación capaz de desarrollar las diferentes tareas solicitadas al código RTE, y de relacionar la RTE producida con los programas de aplicación del usuario. Cada subsistema principal contiene una opción del menú que permite ejecutar inmediatamente las tareas específicas o proporcionar acceso a otras funciones relacionadas con ellas. Por ejemplo, el menú de producción de código RTE ofrece diversas opciones al usuario, incluyendo el generar una nueva descripción (DSC) de RTE, o seleccionar una descripción existente a partir de la biblioteca (SEL), tal como indica la figura 8.

Figura 7
Menú principal de GERTEF.



Los cuatro subsistemas GERTEF accesibles a partir del menú principal son:

Producción de código RTE, por el cual puede el usuario generar una RTE, en código ejecutable, a partir de una descripción.

Definición de contexto físico/lógico, que permite al usuario definir el contexto para la RTE seleccionada.

Desarrollo del soporte lógico del usuario, mediante el cual el usuario desarrolla el programa de aplicación en un lenguaje de alto nivel (C, PLM, CHILL) o lenguaje ensamblador 86.

Tratamiento de la biblioteca de la configuración, que faculta al usuario para gestionar ficheros e imprimir los que seleccione de la biblioteca.

En la tabla 1 se dan más detalles de las opciones que aportan estos subsistemas.

Salidas de GERTEF

El proceso de generación de RTE da como resultado las siguientes salidas:

- código fuente RTE
- código reubicable RTE
- código ejecutable RTE con lista de mapas de memoria
- tabla de símbolos de primitivas para incluir en tareas de usuario
- tablas de sistema.

Se están desarrollando además, tres nuevas características: interfaz de lenguaje de usuario RTE, informes de error e informes de funcionamiento.

Conclusiones

El concepto de reutilización, como medio de mejorar la productividad del soporte lógico al tiempo que se incrementa la fiabilidad del sistema, es piedra angular de la filosofía del GERTEF. En la práctica, las principales ventajas obtenidas afectan a la generación del núcleo lógico (RTE) de sistemas de microordenador tales como equipos de telecomunicación y terminales de usuario.

La introducción de GERTEF durante el desarrollo de un nuevo producto puede llegar a ahorrar de dos a cuatro hombres-año, acortando de este modo el tiempo requerido hasta en seis meses de calendario. De hecho, un experto en la herramienta puede generar una RTE en un tiempo que va de 1 a 4 horas, dependiendo de la arquitectura escogida, las características físicas/lógicas y los acuerdos que se establezcan.

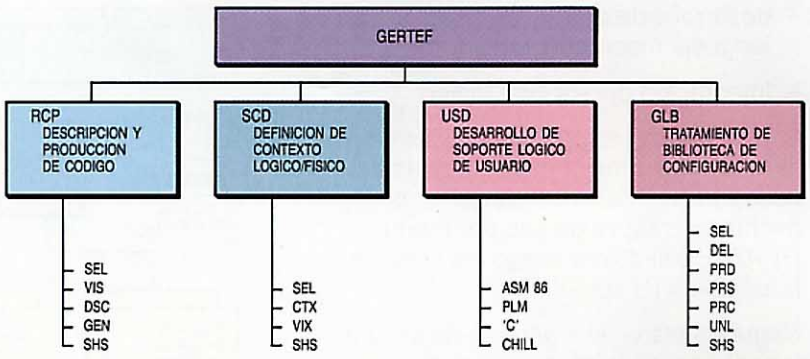


Figura 8
Arbol funcional.

Las RTE generadas por GERTEF requieren una memoria de 1 a 12 k-octetos.

Como los módulos que constituyen las RTE se han optimizado en cuanto a su funcionamiento en tiempo real, la experiencia ha demostrado mejoras del 20 al 50% en las prestaciones con respecto a las RTE construidas a partir de la nada para una determinada aplicación (Tabla 2).

El juego de herramientas GERTEF se ha entregado ya al entorno de desarrollo de programas de ITT, y por lo tanto pueden utilizarlo todas las casas asociadas[8].

Referencias

1 P. Glass: Real-Time Software: *Prentice-Hall*, 1985.
2 G. Goss y M. Paul: Distributed Systems: *Springer*, 1985.
3 G. Kruse: Data Structures and Program Design. *Prentice-Hall*, 1985.

Tabla 1 – Facilidades prestadas por los cuatro subsistemas GERTEF

Producción de código RTE
Selección de una configuración RTE existente de la biblioteca (SEL)
Visualización de la configuración seleccionada (VIS)
Proceso de generación de una nueva descripción (DSC) de configuración RTE
Desarrollo de programación (GEN) para la configuración RTE escogida
Estado de generación de RTE (SHS)
Definición de contexto lógico/físico
Selección de la RTE cuyo contexto lógico se está definiendo (SEL)
Proceso de generación de contexto (CTX) para RTE
Visualización del contexto (VIX)
Estado de generación (SHS) de RTE
Desarrollo de programación de usuario
Desarrollo del código de aplicación en lenguaje de alto nivel o ensamblador
Tratamiento de biblioteca de configuración
Selección de una RTE (SEL)
Borrado de ficheros asociados con la RTE seleccionada (DEL)
Impresion de ficheros asociados con la RTE seleccionada (PRD, PRS, PRC)
Desbloqueo del estado de la RTE seleccionada (UNL)
Estado de generación de RTE (SHS)

Tabla 2 – Comportamiento de las RTE generadas mediante GERTEF

Primitiva	Resultados en microsegundos			
	8086 4 MHz	80186 4 MHz	80186 5 MHz	80186 8 MHz
CELLCLAIM	230	182	150	90
CELLRELEASE	257	197	165	98
CELLENQUEUE	259	199	164	99
SEND	380	298	242	148
SEND I	380	298	242	148
MESSAGE RESCHEDULING (SCHEDULE BG) (tiempo medio)	700	600	480	300
TIME RESCHEDULING (SCHEDULE FG) REAL TIME CLOCK (si está presente) (tiempo medio)	300	280	230	165
TIMESTART con SCHEDULE BG	430	339	295	168
con SCHEDULE BG + FG	480	379	315	188
TIMERRELEASE con SCHEDULE BG	470	380	305	186
con SCHEDULE BG + FG	520	421	338	206
TIMERCLOCK (REAL TIME CLOCK si sólo SCHEDULE BG está presente) con SCHEDULE BG	200	144	125	72
con SCHEDULE BG + FG	155	111	92	55

FG - prioritaria BG - de baja prioridad

4 C. A. R: Hoare: Monitors: An Operating System Structuring Concept: *Communications of the ACM*, octubre 1974, volumen 17, n° 10, págs. 549–557.

5 P. Brinch Hansen: The Architecture of Concurrent Programs: *Prentice-Hall*, 1977.

6 C. Peterson: Petri-Net Theory and the Modelling of Systems: *Prentice-Hall*, 1981.

7 G. Estrin y otros: Concurrent System Software Design: *Symposium on Design Automation and Microprocessors*, febrero 1977, Palo Alto, California.

8 T. Haque y J. Montes: Soporte de productos basados en microprocesadores durante su vida útil: *Comunicaciones Eléctricas*, 1986, volumen 60, n° 2/3, págs. 248–255 (es este número).

Carlos S. Baradello nació en Argentina en 1950. Se graduó en ingeniería eléctrica y electrónica en la Universidad Católica de Córdoba (1972), obtuvo el MS en ingeniería electrónica en el Instituto Internacional Philips de Estudios Tecnológicos, Eindhoven, Holanda (1975), y el PhD en ingeniería eléctrica en la Universidad Carnegie-Mellon, Pittsburgh (1978). El Dr. Baradello ingresó en ITT en 1978, ocupando puestos de responsabilidad en diversas compañías. Desde octubre de 1983 es director adjunto del Centro de Investigación de FACE.

Guido Carloni nació en Italia en 1943, estudió electrónica en la Escuela Técnica Superior de Fermo, y economía en la Universidad Católica de Milán. Entre 1967 y 1975 trabajó como ingeniero jefe en el Laboratorio CSELT, y desde 1975 a 1980, como director de programación en la introducción de tecnología informática en centrales intercontinentales. Ingresó en el Centro de Investigación de FACE en 1980, trabajando al principio en el desarrollo de las pruebas de campo de RDSI con el Sistema 12. En los tres últimos años, el Sr. Carloni ha dirigido el grupo de programación y el centro soporte de ordenadores en dicho Centro.

Entorno distribuido de pruebas de programación

La prueba de programas para sistemas de tiempo real, operando en proceso distribuido, es una tarea compleja. Por su diseño, el GESTOM integra la prueba de módulos, la prueba del sistema y la prueba de verificación, presentando así un entorno uniforme al probador de programas.

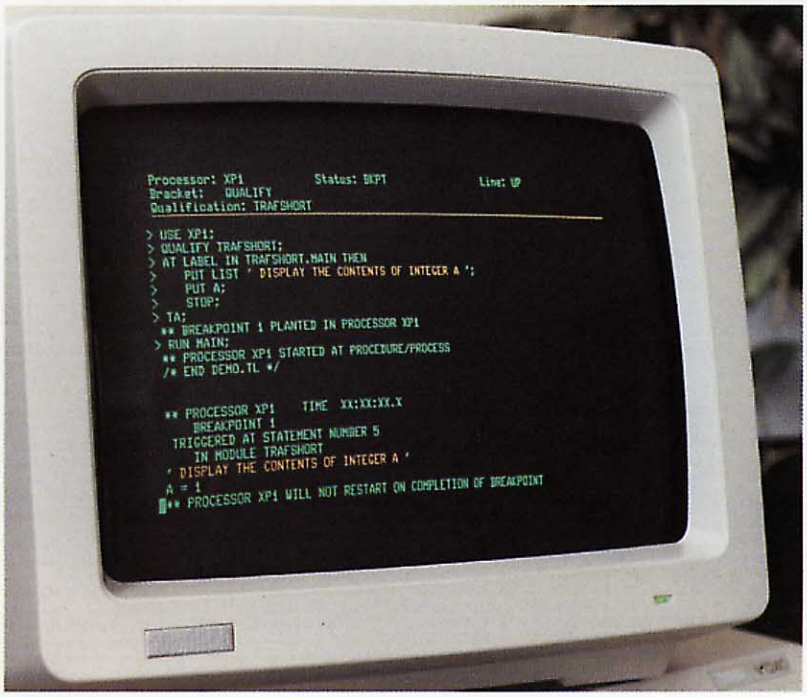
W. Wellens
Bell Telephone Manufacturing Company, Amberes, Bélgica

Introducción

La prueba de programas mediante herramientas de depuración con lenguajes de alto nivel, exige considerables recursos de procesamiento en CPU, y por ello ha padecido, más que otras fases de desarrollo de programas, el irritante problema de la sobrecarga en ordenadores centrales o miniordenadores. Se espera, pues, que esta parte del ciclo de desarrollo mejorará en eficiencia y productividad al introducirse estaciones de trabajo.

Con el nombre de GESTOM, se ha diseñado una herramienta general de prueba de programas para redes de microprocesadores capaz de integrar las pruebas de módulos, de sistema y de verificación en un entorno de estaciones de trabajo. Esto tiene la gran ventaja de que el personal de pruebas sólo tiene que familiarizarse con una herramienta.

Presentación en pantalla típica del GESTOM.



La aplicación inicial de GESTOM es en la central digital Sistema 12, y por consiguiente trabaja con el lenguaje CHILL. En su desarrollo han cooperado tres compañías de ITT: Bell Telephone Manufacturing Company (Amberes), Standard Elektrik Lorenz (Stuttgart) e ITT Europe Engineering Support Centre (Harlow).

Requisitos básicos

Aunque la mayoría de los proyectos actuales de ITT utilizan microprocesadores, éstos ya no son el eje del proyecto sino meros bloques constructivos modificables al compás de la evolución y los requisitos. El GESTOM está concebido para probar sistemas basados en microprocesadores cuyo tipo pueda cambiarse con relativa sencillez. Dado que estos sistemas suelen utilizar programación en alto nivel, y que todavía se necesita el ensamblador para rutinas críticas en el tiempo, el GESTOM tendrá que trabajar con ambos lenguajes.

A menudo proyectos diferentes utilizan un mismo compilador y tipo de microprocesador, por lo que conviene hacer una herramienta de prueba de programas genérica, apta para el mayor número posible de proyectos. Así, la HLDT (herramienta de depuración de alto nivel) presta apoyo a compiladores de lenguaje de alto nivel determinados, mas no contiene inteligencia sobre los sistemas a los que se ha de dar soporte (sistema operativo, base de datos, etc.). Si surge la necesidad de inteligencia propia de un proyecto, ésta se podrá aportar en un nivel adicional realizado en torno a la herramienta de prueba real.

Tanto en proyectos grandes como pequeños, los diseñadores suelen participar en pruebas de sistema y de campo para mejor cumplir los planes generales, mientras que los probadores asisten a las primeras fases del diseño a fin de habituarse al sistema. Esto, sin embargo, será dificultoso

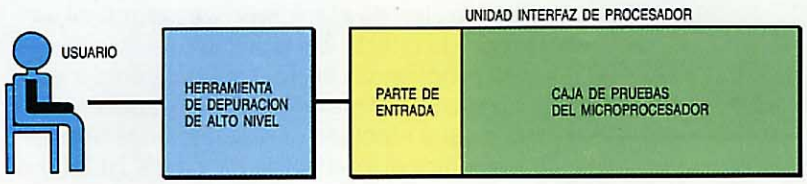


Figura 1
Concepto básico del GESTOM.

si cada herramienta tiene distintas características e interfaz de usuario, ya que hace falta adiestramiento y experiencia para utilizar con eficacia una herramienta. Es, pues, importante que sea mínima la diversidad de herramientas para probar programas y para generar un paquete completo. Por otro lado, la reutilización de datos de entrada en fases posteriores de la prueba sólo es posible si se maneja una herramienta común a lo largo de toda ella.

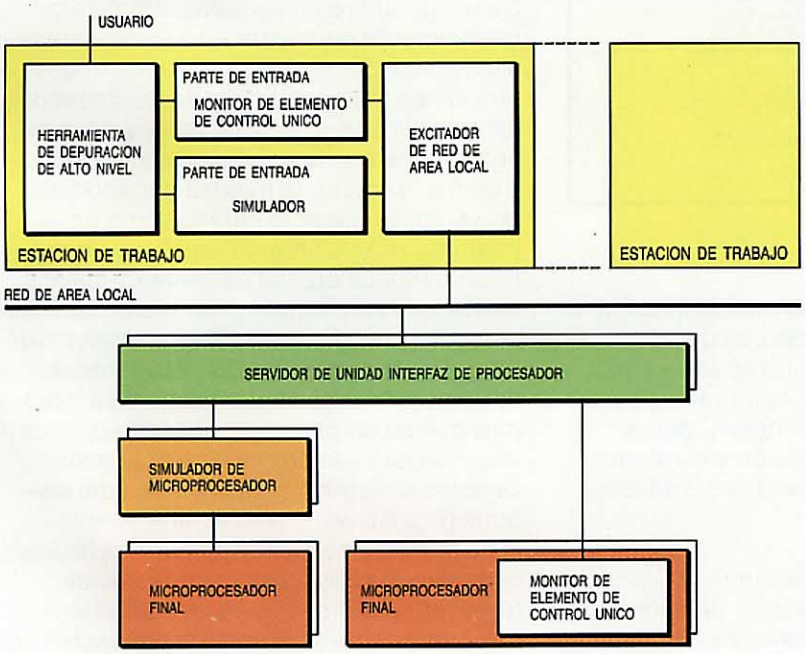
Como la prueba del sistema solamente se puede realizar en los propios circuitos del proyecto, las demás fases de la prueba también deben efectuarse sobre esos circuitos, para evitar la diversificación de herramientas que supondría utilizar la simulación para algunas pruebas. En consecuencia, el GESTOM se debe conectar al equipo final, requisito particularmente importante en proyectos donde pueda variarse el tipo de microprocesador, así como en los que utilizan a la vez diferentes tipos.

Por último, el GESTOM residirá en estaciones de trabajo, dando al usuario libertad de distribuir la capacidad de proceso disponible entre diferentes tareas.

Según se muestra en la figura 1, el GESTOM consta de una herramienta de depuración potente y de alto nivel, independiente del tipo de microprocesador que se adopte y del sistema final, y además de una caja de pruebas del microprocesador final o PIU (unidad interfaz del procesador) provista de un PIU FE (parte de entrada del PIU) que no varía con el lenguaje de alto nivel empleado. Se consigue así poder utilizar diferentes lenguajes de alto nivel y distintos tipos de microprocesadores.

El PIU FE ayuda a optimizar las prestaciones. Por ejemplo, cuando el PIU sólo contiene programas integrados en el microprocesador final, como muestra el SCEM (monitor de elemento de control único) en la figura 2, el tiempo de organización que necesitan los programas de prueba del PIU en el microprocesador final se puede minimizar trasladando tantas funciones como sea posible a la parte de entrada que reside en el ordenador GESTOM.

Figura 2
Configuración completa del GESTOM.



GESTOM

Como indica la figura 2, el usuario se relaciona con la HLDT a través de la estación de trabajo. La HLDT se ajusta al lenguaje de alto nivel utilizado en los programas que se prueban o los programas finales del sistema. Así, traduce las órdenes de usuario a órdenes de bajo nivel que tratará el PIU FE ó PIU, e inversamente descifra los mensajes que devuelve el PIU a respuestas en lenguaje de alto nivel que el usuario visualizará en la pantalla.

La HLDT permite pruebas simbólicas, es decir, a nivel de los símbolos utilizados en el código fuente del lenguaje de alto nivel para el módulo de programas, en modos interactivo y por lotes. La sintaxis de su lenguaje de órdenes debe ser muy similar a la del citado lenguaje de alto nivel, a fin de minimizar el adiestramiento del usuario.

El servidor del PIU tiene dos funciones: la conversión de protocolos entre la LAN (red de área local) y el interfaz PIU (RS 232), y la multiplexación/demultiplexación entre la conexión única de la LAN y los múltiples interfaces del PIU.

SCEM

En su forma más simple, el PIU sólo contiene programas que residen junto con los programas a probar en el microprocesador final. Esta es la solución más económica pues no se requiere un equipo especial, si bien el microprocesador final debe tener un puerto RS 232 para conectarse al servidor del PIU. Los programas del SCEM deben controlar el microprocesador, tomando medidas para evitar que los programas objeto perturben el funcionamiento correcto de los programas del SCEM (sobreescri-

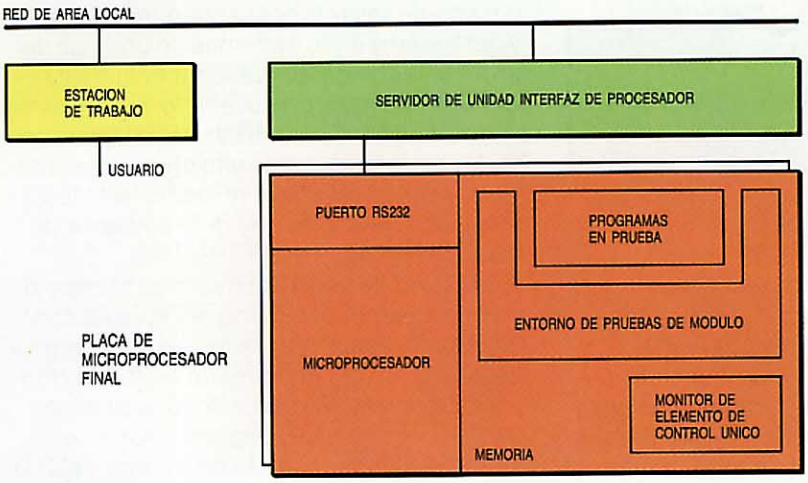


Figura 3
Configuración GESTOM para prueba de módulos, indicando el entorno de pruebas de módulo en el microprocesador final.

biendo datos, inhibiendo interrupciones, etc.). Más aún, los programas del SCEM pueden leer y escribir en memoria, insertar puntos de detección, arrancar y parar los programas objeto, siempre bajo pleno control del usuario a través de la HLDT. Por desgracia, la solución SCEM presenta algunos inconvenientes:

- se requieren circuitos especiales para algunas funciones como la traza
- se utiliza memoria y capacidad de proceso que de otra forma estaría a disposición de los programas objeto.

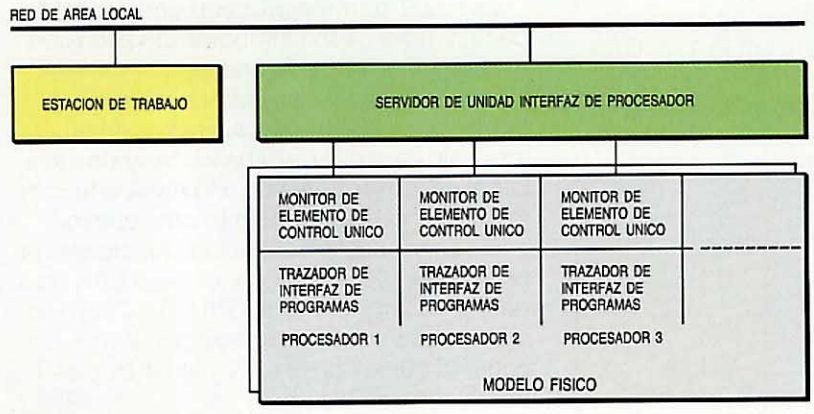


Figura 4
Configuración GESTOM para prueba de sistema.

El PIU FE (SCEM FE) realiza aquellas funciones que no se necesita ejecutar en el SCEM, minimizando así la capacidad de proceso sustraída por los programas de SCEM a la programación objeto, como sucede con la desagregación y el reformatado de la memoria de trazas para adaptación al formato HLDT.

Simulador de microprocesador
Algunos tipos de simuladores de microprocesador no necesitan programas de prueba especiales en el microprocesador final, dejando intacta la capacidad de éste para los programas objeto. Una desventaja es que la parte de entrada del PIU o del emulador debe efectuar las conversiones necesarias entre el interfaz de salida del HLDT y el interfaz de entrada del PIU.

Aunque este modo de proceder ofrezca mucha más funcionalidad para pruebas que el SCEM, también es notablemente más costoso. Sin embargo, dado que estas nuevas funciones sólo se necesitan durante una breve parte del tiempo total de pruebas (para encontrar fallos difíciles), ambas soluciones se pueden utilizar en paralelo (Fig. 2) para obtener un conjunto económico de herramientas de prueba.

Pruebas de programas con el GESTOM

Prueba de módulos
La prueba de módulos afecta a un módulo de programación aislado o a un componente (conjunto de módulos que constituyen una función lógica, como puede ser el tratamiento de llamadas). Como estos módulos no se pueden ejecutar de manera autónoma, se requiere un entorno de prueba para simular, excitar y trazar todos los interfaces con el resto del paquete de programas objeto: servicios del sistema operativo, accesos a bases de datos e interfaces a los otros módulos. Una vez que el usuario forme un paquete ejecutable que incluya el módulo a probar, podrá realizar la prueba utilizando el GESTOM en cualquier microprocesador conectado que esté libre (Fig. 3).

Prueba de sistema
Se realiza sobre un paquete completo y normalmente no necesita un entorno de simulación. Sin embargo, sí requiere una función de traza en los interfaces de módulos, por ser éste el campo primario de investigación durante la prueba de sistema. Como el trazador de interfaz depende del proyecto, es mejor integrarlo como un módulo extra de programas en los programas objeto. La prueba del sistema suele realizarse sobre un grupo de microprocesadores (modelo físico), todos los cuales han de conectarse al GESTOM. Este modelo físico será amoldado al paquete de programas que ha de probarse, de forma que los usuarios se vean obligados a usar esos circuitos concretos para la prueba del sistema (Fig. 4).

La prueba en el campo se puede efectuar con relativa sencillez por la facilidad de transporte de la configuración GESTOM, que consiste en algunas estaciones de

prueba y un servidor del PIU. Si esta solución no es viable, habrá que utilizar otras herramientas de prueba en el campo. Se ha de advertir que, si el trazador de interfaz de módulo está integrado en el paquete a probar, podrá todavía ser utilizado en campo siempre que se le pueda controlar directamente desde una pantalla u ordenador personal.

La configuración mostrada en la figura 5 admite la prueba remota, permitiendo a los expertos probadores de un centro ITT ayudar a las pruebas de una central en la instalación.

Pruebas de verificación

Estas pruebas prestan soporte a las pruebas de regresión automáticas de un paquete completo. Permiten una fácil regresión durante la prueba del sistema para verificar que se han corregido los fallos en un paquete reconstruido, y además pueden servir para comprobar los programas producidos para una configuración de central típica y así alcanzar un primer nivel de confianza en esos programas antes de su envío a la central. Las pruebas de verificación se realizan sobre un modelo físico, logrando un total automatismo mediante la simulación de sucesos externos (p. ej., DESCUELGUE).

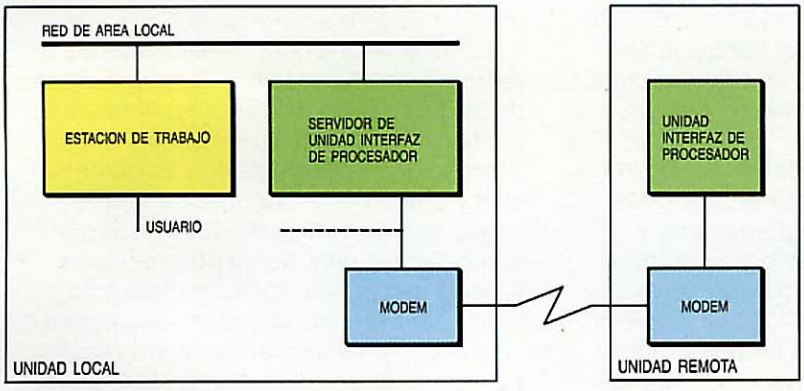


Figura 5
Soporte de prueba remota.

Entorno de pruebas distribuido

La arquitectura del GESTOM mejora la productividad de la ingeniería, al poder estar físicamente separadas las estaciones de trabajo y las conexiones del procesador al PIU. Una vez instalada una LAN en una casa ITT, los modelos de equipo físico y los microprocesadores aislados situados en áreas diferentes se pueden conectar al GESTOM a través de servidores de PIU. Con la estación de pruebas sobre la mesa, el usuario que desee probar un módulo de programas puede llamar al GESTOM y pedir un microprocesador libre sin conocer las direcciones o situaciones físicas de los circuitos. En tales condiciones, el usuario tiene todo el material necesario a mano (documentos, especificaciones, listados) y se puede concentrar totalmente en la sesión de pruebas. Cuando se necesiten circuitos especiales, como en las pruebas de sistema, el usuario deberá conocer el modo de acceso al modelo específico.

El aprovechamiento de la estación de trabajo es óptimo cuando ésta atienda al proceso completo de desarrollo de programas. En tal caso el usuario podrá incorporar un paquete de pruebas de módulo en la estación de trabajo, actualizar el código fuente del módulo para arreglar cualquier fallo, volver a ejecutar el paquete y continuar probando de una forma muy eficaz.

Conclusiones

Aunque bastante sencillo en cuanto a arquitectura, la concepción del GESTOM se apoyó en el actual estado del arte, con plena visión de futuro. Fue construido para satisfacer necesidades concretas de usuarios, recogidas en una larga experiencia de pruebas de la gran variedad de productos ITT para telecomunicación. El servidor de PIU aparecerá a finales de 1986; para 1987 se prevé dar soporte a los simuladores de microprocesadores. El desarrollo de otras HLDT dependerá de las necesidades de futuros productos ITT en las diferentes compañías. La flexibilidad que ofrece el GESTOM es elevada, su mantenimiento es fácil y se podrá modificar a compás de la evolución tecnológica en la industria de los próximos años.

Willy Wellens nació en Genk, Bélgica, en 1947. En 1970 se graduó en ingeniería electrónica en la Universidad Católica de Lovaina, entrando luego en BTM, donde trabajó en programas de pruebas de ordenadores y simulación. Más tarde desarrolló un paquete de simulación para Metaconta 10CN* y actualmente es responsable de las herramientas de prueba para el Sistema 12 en dicha Compañía. El Sr. Wellens es responsable técnico de producto de las herramientas y estrategias en las áreas de pruebas de programas y control de cambios para todo el proyecto del Sistema 12.

* Marca registrada del Sistema ITT

Desarrollo y mantenimiento de programas de aplicación

Como gran usuario de programas de aplicación, el Hartford Insurance Group de ITT, se vió afectado por la productividad de la escritura de nuevos programas. Para solventar este problema la compañía introdujo un nuevo marco para el ciclo de vida basado en una estación de desarrollo. Los resultados han sido impresionantes, mostrando una mejora en la productividad de hasta un 30%.

J. T. Crawford
Hartford Insurance Group, Hartford, Connecticut, Estados Unidos

Introducción

The Hartford, subsidiaria de ITT y una de las compañías de seguros más grandes de los Estados Unidos, depende de los sistemas de programas de aplicación en tal forma que sin ellos no podría librar una póliza, pagar una reclamación o calcular una prima. Por consiguiente es esencial que The Hartford pueda desarrollar programas de aplicación técnicamente avanzados, de alta calidad, a un coste mínimo y mantenerlos una vez puestos en servicio. Para ello, tiene a 1.300 programadores y analistas trabajando en desarrollo y mantenimiento de programas, y a otros 1.000 para soporte técnico en tres centros de datos.

Al terminar la década de los 70, la dirección de The Hartford comprendió que la productividad de los programadores y analistas era un gran problema, común a muchas compañías que desarrollaban sus propios programas de aplicación. El desarrollo costaba demasiado tiempo y dinero por la dificultad en controlar la programación: 10 programadores dejados en libertad seguramente producirían 10 programas diferentes en 10 tiempos distintos partiendo de un mismo encargo. Está claro que los controles y planes son vitales para obtener pleno rendimiento de los gastos en proceso de datos y para comprender el compromiso exigido, en cuanto a entrega de los productos en plazo adecuado a los planes estratégicos de la Compañía.

Los ciclos de desarrollo de programas se iban alargando en un momento en que era preciso producir nuevos programas con más rapidez y eficacia. Además, a la necesidad de mantener bajos los costes se enfrentaba la de competir por los programadores, no sólo en salario, sino también ofreciendo condiciones de trabajo favorables y oportunidades para personas capacitadas y motivadas.

Los grupos de desarrollo de programas tenían demasiada responsabilidad en la planificación del desarrollo de nuevos programas de aplicación, determinando los elementos a entregar, las tareas y las herramientas, caso por caso. Estas decisiones corresponden a la gestión, mientras que el grupo del proyecto debe dedicarse a la creación de sistemas para resolver los problemas de la empresa.

Aunque se disponía comercialmente de herramientas de desarrollo y mantenimiento de los programas, tales como generadores de datos de prueba y de programas, el proceso de diseño, escritura, documentación y prueba de programas seguía ocupando mucho personal. Además, según aumentaba el número de programas de aplicación, crecían las necesidades de mantenerlos, hasta alcanzar un punto en el cual el personal de mantenimiento casi sobrepasaba en número a los diseñadores. En el caso extremo, el personal de programación sería dedicado sólo a mantenimiento, ya que sus compañías no podrían atender a otra cosa.

En un intento de controlar el coste de desarrollo y mantenimiento de los programas de aplicación, The Hartford se esforzó mucho por crear y dirigir un entorno para la productividad de la programación.

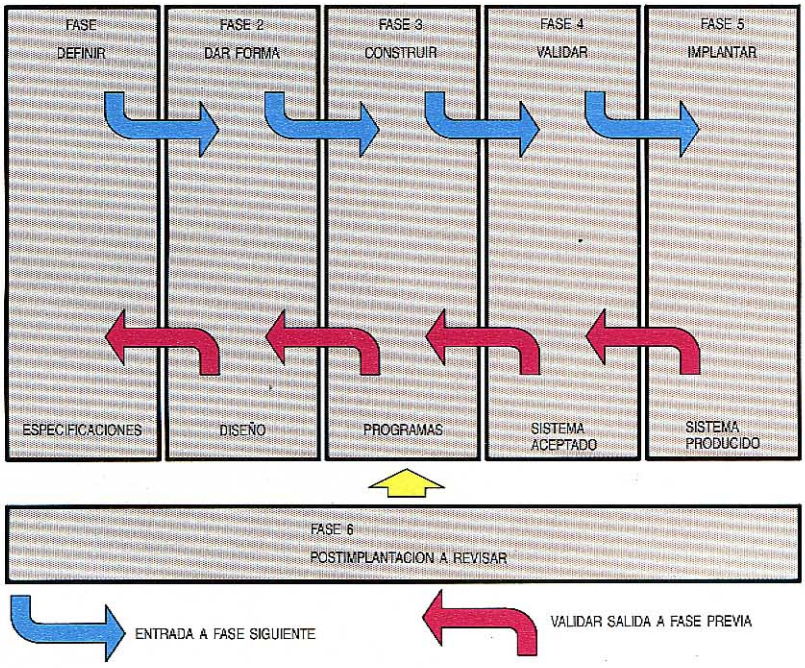
Primeros esfuerzos

Los primeros pasos serios de The Hartford para mejorar la productividad de la programación, en 1980, se dirigían a tres áreas principales.

Primeramente se instaló FOCUS, un lenguaje de alto nivel de la cuarta generación, para suplementar al COBOL, que no era apropiado para todas las aplicaciones. En segundo lugar, se formalizó el concepto de programas reutilizables, animando a los programadores a depositar buenas subrutinas y programas en una biblioteca, a disposición de otros programadores. En tercer lugar, las pruebas se mejoraron instalando nuevas herramientas, tales como generadores y ejecutores de pruebas. Estas acciones se complementaron con una mayor calidad de la instrucción.

Parte clave de este esfuerzo fue la dedicación de recursos permanentes para apoyar la productividad de los programadores. Inicialmente, siete consultores con plena dedicación llegaron a ser expertos en estas tres disciplinas, y después este grupo ha crecido hasta convertirse en el Centro

Figura 1
Fases del ciclo de vida del proyecto.



Soporte a la Productividad, con 35 personas que instruyen en el uso de nuevas herramientas y técnicas y supervisan el uso de las ya instaladas. El grupo administra también programas de incentivos y recompensas, y en esencia podría llamarse a sus miembros los "apóstoles" de la productividad.

Ciclo de vida del proyecto

The Hartford está entregada a la disciplina impuesta por su ciclo de vida del proyecto, que es un plan detallado para el desarrollo y mantenimiento de un programa o sistema (Fig. 1). Este ciclo de vida es dinámico, añadiéndose continuamente nuevas herramientas y técnicas para mejorar la metodología. Desde un enfoque directivo, su ventaja clave es el control.

El ciclo de vida tiene dos componentes. Uno, el *componente de dirección*, vigila el progreso de un proyecto en relación con criterios de gestión tales como necesidades de coste y recursos, fechas de terminación de actividad, revisiones de calidad e hitos de fases importantes en los cuales se examina el proyecto en detalle. El otro componente, el *técnico*, define las tareas, los productos que se obtienen de ellas y las herramientas a utilizar. Este componente debe ser dinámico y flexible; en efecto, el ciclo de vida de un proyecto en el que se crea un sistema de gestión de información de 300 transacciones por segundo será muy diferente del utilizado para producir un sistema sobre miniordenador.

The Hartford acomoda el ciclo de vida de los proyectos a sus necesidades específicas. Está orientado al cliente, que son los propios departamentos de la Compañía. El departamento de Sistemas de Información de Gestión es el contratante que usa el ciclo de vida en cuestión para desarrollar el producto. Sin embargo, los departamentos clientes tienen también un papel importante, definiendo sus necesidades en términos de tareas y actividades concretas.

Arquitectura automatizada

A finales de 1982, una revisión del progreso hasta la fecha reveló la necesidad de una arquitectura automatizada, servida por una estación de desarrollo, para implantar el ciclo de vida del proyecto. Se definieron objetivos de productividad expresados como funciones estándar de desarrollo de programas: análisis y diseño, desarrollo del programa, implantación e instrucción, salida en operación, documentación y controles, y mantenimiento. También se incorporaron en la arquitectura los aspectos de dirección y control del ciclo de vida.

Los objetivos específicos de la arquitectura automatizada incluyeron:

- Automatizar, dentro de dicha arquitectura, el máximo posible del ciclo de vida del proyecto.
- Desarrollar técnicas modernas de ingeniería de la información (p. ej., sistemas de diseño basados en gráficos) y transferir, desde los mundos del diseño y la fabricación informatizados, tecnología para realizar herramientas de diseño

capaces de crear electrónicamente diagramas de flujo de datos, cartas de estructura y modelos de datos.

- Dar flexibilidad a la arquitectura para que evolucione a medida que cambian las herramientas, técnicas y procesos del ciclo de vida.
- Integrar lo más posible el proceso del ciclo de vida y los elementos que han de entregarse, ligando entre sí los pasos del ciclo de vida de forma que una tarea conduzca naturalmente a la siguiente, y relacionando los elementos de tal forma que los datos producidos al principio del ciclo sean transferibles a un elemento posterior más detallado.
- Desarrollar una arquitectura común de circuitos y programas para sustentar el desarrollo y mantenimiento de la aplicación, incluyendo una estación de trabajo profesional para cada analista y programador, y ofreciendo conexión entre los miniordenadores a nivel de proyecto y los ordenadores principales.

El diseño de una estación de desarrollo que cumpliera estos objetivos se acometió en cooperación con un suministrador externo. La arquitectura de los equipos es en anillo, con estaciones interconectadas a través de una red de área local, y la comunicación con los ordenadores principales se hace por emuladores 3274 y 3271. Esta arquitectura permite a los programadores comunicarse entre sí y el que las herramientas se ejecuten en el nivel arquitectural más bajo (nivel de microordenador) al tiempo que se dispone de recursos compartidos en el segundo nivel (miniordenador). Además, la arquitectura de la estación de trabajo sirve para transferir las aplicaciones recién terminadas al ordenador principal donde tiene lugar la producción (Fig. 2).

La estación de desarrollo opera tanto en modo menú como en modo experto: un usuario nuevo puede acceder por pasos del menú a una tarea del ciclo, que le conduce a las herramientas apropiadas. Sin embargo, un programador o analista con experiencia puede operar en modo experto, accediendo directamente a las funciones requeridas.

La estación incorpora guiones de documentos de texto normalizado requeridos durante el desarrollo de un programa. Un ejemplo es el documento TREND, que es la norma de The Hartford para una pronta notificación, a sus grupos de operaciones y soporte técnico, del impacto de un nuevo programa o sistema de aplicación sobre el centro de datos. El guión del documento TREND se mantiene activo permanentemente, a fin de que tan pronto como se conozca información acerca de un nuevo sistema de aplicación, pueda el programador añadirla al documento.

Bases de datos

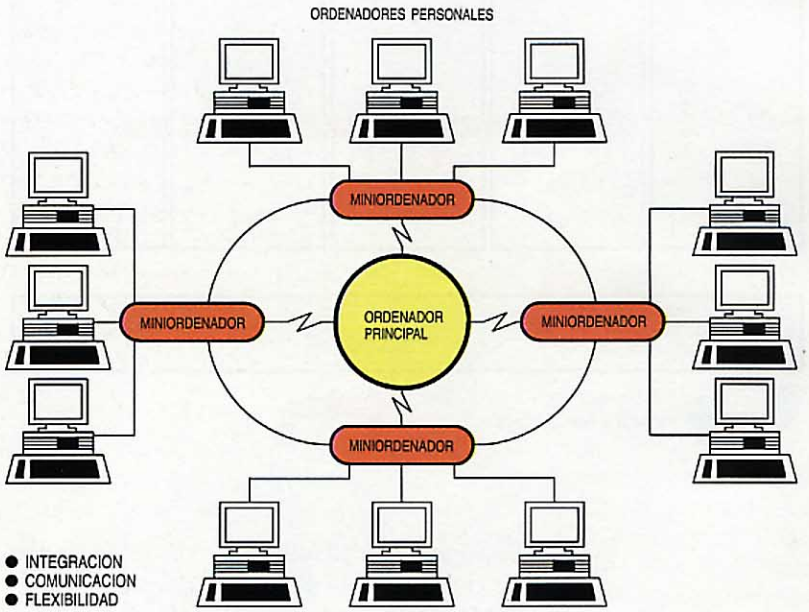
Las dos bases de datos en el núcleo del sistema residen en el segundo nivel (miniordenador) de la estación de trabajo. El archivo electrónico de proyectos, mantenido por cada grupo, es el almacén de los elementos terminados, tales como un documento general, análisis de mejoras de coste o plan de prueba. El archivo de elementos reutilizables pone los elementos genéricos a la disposición de otros grupos. Un elemento reutilizable puede ser visualizado como muestra o bien ser transferido realmente a la biblioteca de otro grupo. Los elementos reutilizables incluyen plantillas para estudios de mejoras de coste, gestión de proyectos y planificación de recursos.

Otra herramienta es el diario profesional que permite a los programadores seguir el estado de los trabajos en detalle y mantener al corriente sobre el progreso a sus supervisores.

Los generadores de código residen también en la estación de desarrollo. Un ejemplo es el generador de JCL (lenguaje de control de tareas). Cuando se presenta una tarea desde la estación al ordenador principal, va acompañada de un paquete JCL. El programador aporta los parámetros para generar el JCL, pero no necesita suministrar los formatos especiales ni la sintaxis asociada.

Se dispone de medios de composición, tanto a nivel de microordenador como de

Figura 2
Arquitectura en tres niveles de la estación de desarrollo.



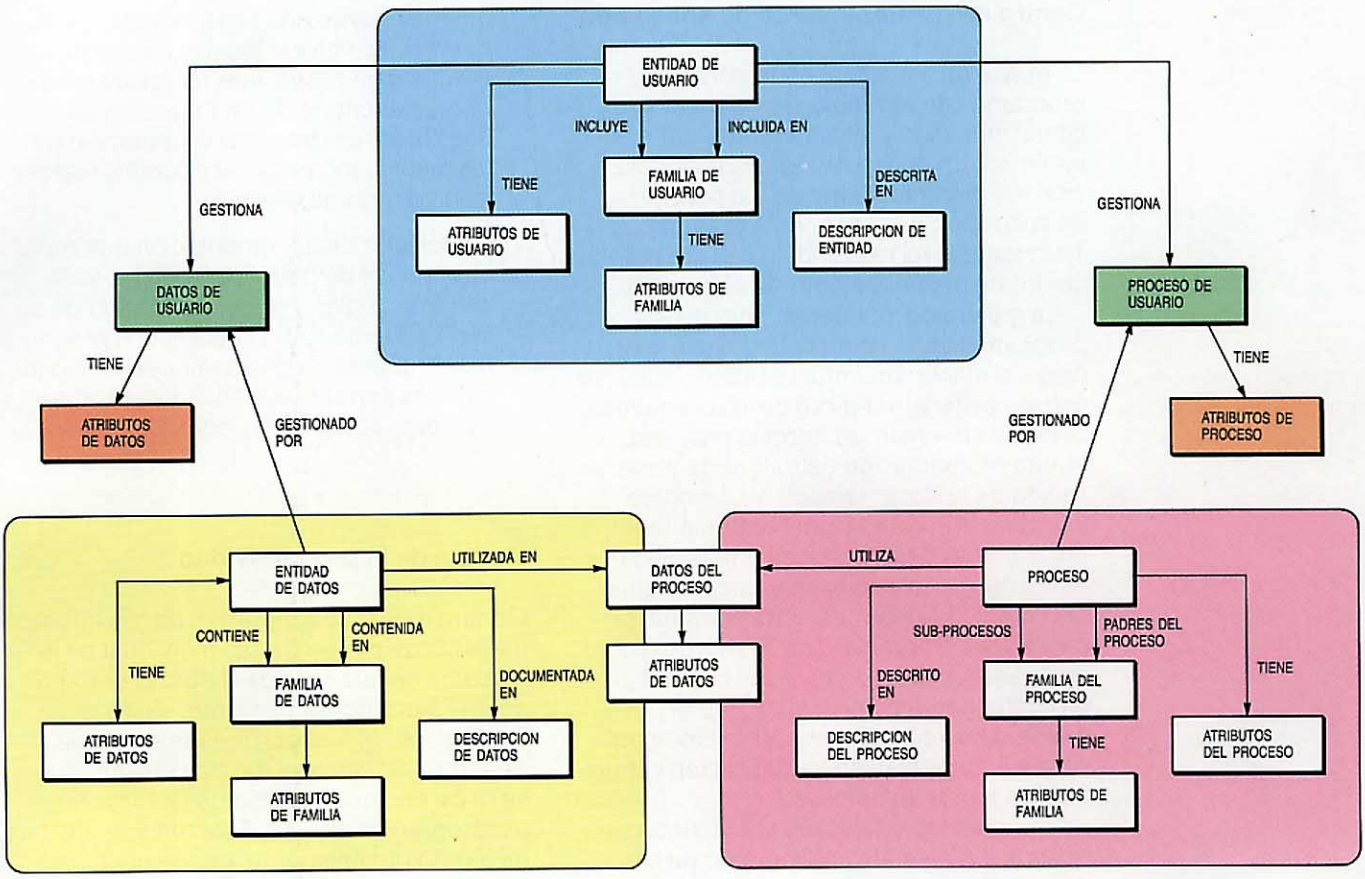


Figura 3
Diccionario del diseñador: modelo de relación de entidad.

miniordenador de manera que, en la mayoría de los proyectos, se incluye directamente un listado del contenido de una pantalla en la especificación de diseño. El soporte electrónico gráfico facilita la producción de diagramas de flujo de datos, cartas de estructura y modelos de datos.

Diccionario del diseñador

Esta reciente adición a la estación de trabajo usa una base de datos relacional para almacenar especificaciones de ingeniería de información rigurosa. Las entidades (procesos, ficheros, segmentos, elementos, pantallas, programas) que describen un sistema están representadas en el diccionario, con referencias cruzadas desde el nivel conceptual hasta el más detallado (nivel de elemento de datos). El diccionario acomoda cualquier clase de entidad que haya de ser definida para una ingeniería de información rigurosa; esta definición puede hacerla el usuario, lo que permite añadir fácilmente nuevas clases de datos y tipos de objeto. Los atributos asociados con cada clase de entidad son también definibles por el usuario. Si, por ejemplo, se define una clase de entidad de información llamada "diagrama de burbujas de flujo de datos" con 10 atributos asociados con cada entidad, el administrador de datos puede añadir fácilmente un 11° atributo en caso necesario.

La figura 3 muestra el modelo de relación de entidades del diccionario del diseñador. En la base de datos relacional las clases y las entidades de datos están relacionadas con sus procesos asociados, identificando qué dato utiliza cada proceso en una jerarquía completa. Las entidades de datos y procesos tienen también relación con las entidades de usuario, siendo así posible determinar qué departamento o función es responsable de un elemento de datos concreto, quién es su propietario, quién lo usa y qué proceso lo utiliza o modifica.

Otro elemento del diccionario es el control por el administrador de datos. Dentro del diccionario hay diferentes subconjuntos que permiten a un equipo del proyecto operar a un nivel con los datos que necesite, mientras que el administrador controla los datos de la Compañía.

Más bien que construir todas estas herramientas, lo que hizo The Hartford fue seleccionar los mejores productos e integrarlos en su concepto de ciclo de vida del proyecto. Sin embargo, sí que desarrolló las herramientas clave de integración, como el diccionario del diseñador, que ligan las herramientas en un marco de programación coherente.

Centro de mantenimiento de aplicación

La mayor eficiencia en el desarrollo de programas de aplicación tiene un efecto lateral indeseado: el incremento del mantenimiento. En consecuencia, la racionalización del mantenimiento de los programas de aplicación es el siguiente gran reto, y The Hartford ha respondido a él creando un centro de mantenimiento de aplicación.

La mejora del mantenimiento de los programas tiene dos facetas: la preservación y el mantenimiento. La preservación se dirige a evitar el deterioro de los programas. Si no se observan las normas para una buena programación estructurada y actualización de la documentación cuando se introduzca un cambio por mantenimiento, el programa se degradará y será más difícil de mantener, entorpeciendo y encareciendo los cambios siguientes. Para asegurar la calidad del programa, The Hartford creó un analizador estático, merced al cual los programadores comprueban que el programa cumple las normas de la Compañía sobre documentación y codificación estructuradas y baja complejidad.

Como ejemplo, el analizador estático restringe el nivel de anidamiento de las decisiones y el uso de expresiones matemáticas complejas para hacer los programas tan legibles y modificables como sea posible. Siempre que se actualiza un programa, automáticamente se le evalúa y analiza antes de ser entregado a producción. La puntuación obtenida sirve de entrada a un índice de productividad tabulado a nivel de aplicación de área, a nivel de sistema y a nivel de división. Un sistema de información a la dirección utilizará después esta información para determinar si los sistemas se están deteriorando.

Actualmente The Hartford usa más de 21 millones de líneas de código, habiéndose producido 4 a 5 millones de ellas antes de la introducción de las nuevas normas de programación estructurada a mediados de los años 70, por lo que adolecen de estructura y documentación deficientes. Para mejorar la mantenibilidad de estos viejos programas, The Hartford aplica un proceso de restauración, a cuyo fin se han instalado diversas herramientas:

- Herramientas de conversión de los programas viejos, con estructura deficiente, en programas estructurados.
- Herramienta de alineación para mejorar la legibilidad.
- Herramienta de división en módulos. Algunos programas viejos tienen tres o cuatro mil líneas, contra la norma de The Hartford de no superar las 300 líneas en un programa en COBOL. Estas herramientas trazan una carta de estructura que ayuda a dividir los programas en módulos menores, más fáciles de mantener. La escritura de un programa de control para supervisar la operación de los nuevos módulos hace posible retener la funcionalidad original.
- Generador de documentación que trabaja a nivel de párrafos COBOL, analizando el código y generando un guión de la documentación. Un analista o programador puede luego añadir descripciones de cada párrafo COBOL a nivel comercial, para completar la documentación.

Medida de la productividad

Generalmente se acepta que no hay buenas medidas de productividad individual en la industria del proceso de datos. El enfoque de The Hartford es, por tanto, vigilar el rendimiento global del departamento de Sistemas de Información de Gestión, que se mide en líneas de código desarrolladas por programador y año. Este rendimiento ha mejorado notablemente, desde poco más de 6.000 en 1982 hasta más de 11.000 en 1985. Se espera incrementar el desarrollo de nuevo código hasta un valor entre 18.000 y 25.000 líneas por persona y año en 1990.

La productividad del mantenimiento, en cambio, no presenta un cuadro tan favorable. Actualmente, una persona mantiene entre 30.000 y 32.000 líneas de código por año. Con el nuevo código que se está escribiendo a razón de 11.000 líneas por año, se necesitará un nuevo programador de mantenimiento por cada tres diseñadores, situación que forzosamente empeorará con el aumento de la productividad del desarrollo. Aunque los programas más nuevos, mejor documentados y estructurados sean más fáciles de mantener, es necesario mejorar marcadamente la relación entre diseñador y técnico de mantenimiento si The Hartford quiere satisfacer sus necesidades controlando los costes.

Análisis coste-beneficio

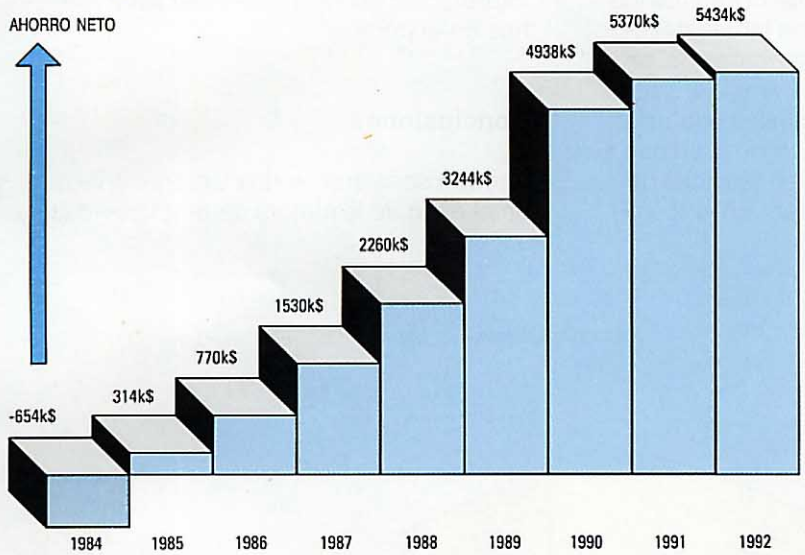
The Hartford asumió un compromiso importante en la estación de desarrollo, por lo que realizó un amplio análisis coste-beneficio para calcular el rendimiento de la inversión. La aplicación de este análisis al esfuerzo de mejora de la productividad se basó en un estudio de ingeniería industrial que registraba el tiempo invertido en cada tarea del ciclo de vida del proyecto. Cada herramienta

y componente esencial de la estación de desarrollo se calibró, comparando los trabajos requeridos para realizar una función con ayuda de la estación y sin ella. Se calculó la productividad media en cada tarea y se extrapoló para obtener economías adicionales, resultando un ahorro medio del 25 al 30% en el tiempo de desarrollo de nuevos proyectos.

Estas cifras sirvieron para calcular el ahorro en dinero. El grueso del ahorro resulta de requerir menos personal del que hubiera sido necesario sin el nuevo entorno de desarrollo. En cuanto las estaciones de desarrollo alcancen su plena expansión, The Hartford requerirá 175 programadores menos, lo que justifica sobradamente la inversión.

Se están logrando ahorros suplementarios al retirarse los terminales y ordenadores existentes. Miles de horas de funciones se transferirán desde grandes ordenadores a miniordenadores o, mejor aún, a microordenadores, con el objetivo de reducir en un 35% el uso del ordenador principal en tiempo compartido, consiguiendo así economías sustanciales.

Figura 4
Estación de desarrollo: ahorros netos anuales 1984–1992.



La gráfica de ahorros (Fig. 4) en 10 años muestra unos resultados todavía más impresionantes. Al contener el coste, el proyecto pronto fue rentable, rebasando el umbral de beneficios a mediados de 1986.

Instrucción y soporte

La instrucción y el soporte son componentes esenciales del esfuerzo de mejora de la productividad. De los 35 especialistas del Centro Soporte a la Productividad, 20 están relacionados con la instrucción y/o el servicio a los usuarios de las estaciones de desarrollo. Además, en cada una de las 17 divisiones del departamento de Sistemas de Información de Gestión, un coordinador de área trabaja con el personal del centro soporte antes citado para ayudar a la instrucción, además de detectar problemas y sugerir mejoras. Una comunicación interna directa y rápida proporciona a los usuarios de las estaciones de desarrollo auxilio inmediato para resolver los problemas.

Recompensas e incentivos

Para motivar al personal, The Hartford ha introducido un programa completo de recompensas e incentivos, administrado por el Centro Soporte a la Productividad, que comprende promociones internas, un boletín de productividad y la designación como "supervisor del mes" del individuo que más contribuya a la productividad y el mejor uso de la estación de desarrollo. Hay también ferias de la productividad, con mesas redondas donde los profesionales del sistema pueden discutir con expertos las nuevas herramientas y técnicas.

Estado del proyecto

Aunque ha habido problemas, The Hartford valora muy positivamente los resultados netos. Las estaciones de desarrollo se están instalando a razón de 80 a 100 por trimestre; 850 estaciones estarán en servicio a final de 1986 y 1.100 al fin de 1987, cuando está prevista la terminación. Se están asimismo incluyendo clientes de los servicios del sistema de información de gestión – directores de proyectos de automatización –, y algunos disponen ya de estaciones. También habrá personal de explotación, algunas de cuyas áreas han sido conectadas a la estación de desarrollo.

El ciclo de vida del proyecto sustenta el desarrollo de programas desde la identificación de los requisitos del usuario, y a través del diseño de sistemas y desarrollo de programas (donde hay un pico de necesidades de recursos), hasta la prueba del programa e integración inicial del mismo, en cuyo punto se pueden reasignar algunos recursos. Después siguen las pruebas de aceptación por el usuario, la instalación de los programas y la adopción del nuevo sistema de aplicación desarrollado.

Se hace necesario entonces preservar el sistema mediante técnicas de mantenimiento eficaces. Si no se toman medidas, el

sistema comienza a deteriorarse hasta que se hace necesario desarrollar otro en su lugar y se repite el ciclo de vida del proyecto.

La combinación de herramientas y técnicas contenidas en la estación de desarrollo, junto con el interés en composición, modelado de datos y uso del diccionario del diseñador, están adelantando el pico de recursos del ciclo de vida (Fig. 5). Puede dedicarse un mayor trabajo a definir los requisitos con precisión y al diseño de calidad de los sistemas. La integración de los generadores de código en la biblioteca de elementos reutilizables resuelve la mayoría de los problemas de construcción de sistemas. La conversión e instalación permanecerán esencialmente iguales, pero el mantenimiento debería aplanarse, e incluso descender, a medida que el sistema va madurando como resultado de las herramientas y técnicas de preservación.

Desarrollo futuro

La figura 6 es un modelo del ciclo dinámico de vida de un proyecto futuro. El dinamismo estriba en que pueden seleccionarse los elementos a entregar y las tareas del ciclo de vida con base en los atributos del proyecto que al principio se introdujeron en una ecuación, entre los cuales figuran el alcance del proyecto, el entorno en que se ejecutará y las directrices o técnicas de gestión que van a aplicarse. En virtud de tales informaciones, se seleccionarán de modo dinámico las tareas que conlleva un proyecto específico en un conocido entorno para esa clase de proyectos en el futuro.

Un caso concreto del ciclo de vida del proyecto (Fig. 6) invocará procedimientos específicos al ir aumentando el grado de automatización. Entonces las transferencias de datos de un elemento del ciclo de vida del proyecto a otro tendrán lugar de manera automática más bien que manual. A su vez, estos procedimientos contribuirán a la producción de elementos entregables. Cuando se haya terminado uno de estos elementos, el fin se señalará a un sistema activo de gestión del proyecto, mas no al sistema estático presente, lo cual por su parte dará paso a la tarea siguiente, dentro del control automático del progreso del trabajo.

La futura gestión de proyectos registrará también en una base de datos las medidas de productividad (tiempo requerido para ejecutar funciones específicas) relacionadas con el desarrollo de un producto, conduciendo a un mayor refinamiento de los estudios de rentabilidad y a una mejor información sobre la cual basar las inversiones en el porvenir.

Conclusiones

Pueden señalarse varios factores críticos en el éxito de la mejora de productividad de

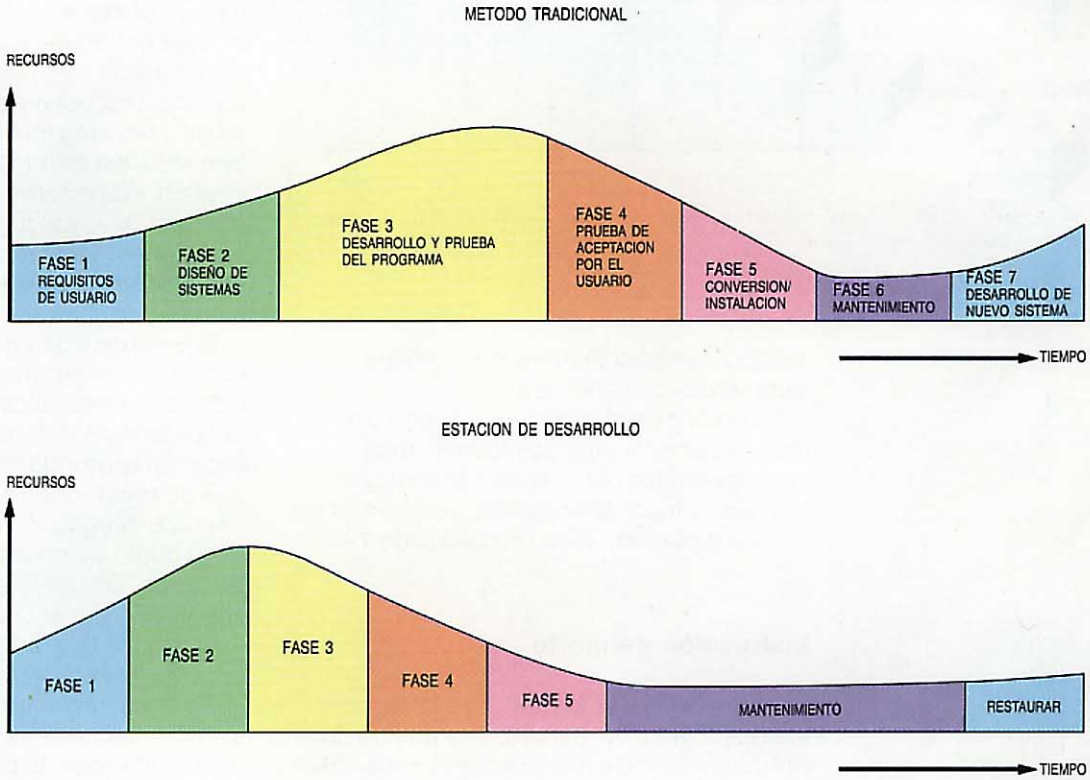


Figura 5
Asignación de recursos: método tradicional comparado con estación de desarrollo.

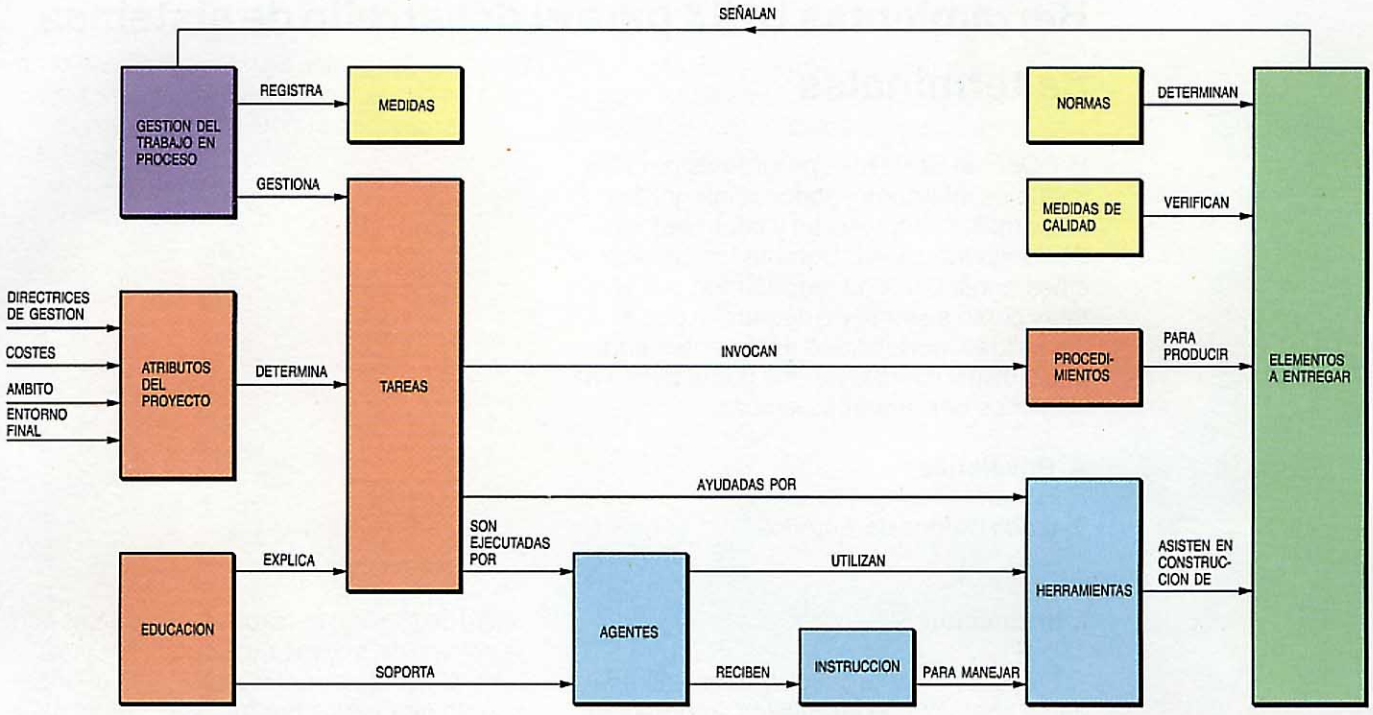


Figura 6
Modelo del ciclo dinámico de vida de un proyecto del futuro.

The Hartford. Por encima de todo, el proyecto no se podría haber realizado si la compañía no hubiera comprometido los recursos a gran escala que reclamaba este magno plan. El ciclo de vida del proyecto es la base para imponer las disciplinas de gestión necesarias para controlar el desarrollo de nuevos programas de aplicación, mientras que la estación de desarrollo proporciona una única arquitectura de equipos y programas para desarrollar y mantener las aplicaciones. No obstante, también era importante que este método estándar tuviera flexibilidad para acomodarse a futuros cambios.

Otros factores que contribuyeron al éxito del proyecto fueron la capacidad para medir (aunque toscamente) las mejoras de la productividad, así como una eficaz instrucción y asistencia al usuario.

En el futuro, a medida que se haga más dinámico el ciclo de vida del proyecto, las tareas se integrarán mediante procedimientos manuales y automatizados, y la transferencia de los datos de un proyecto desde los ordenadores principales a los miniordenadores y microordenadores se verá facilitada por herramientas nuevas. La estación de desarrollo será el factor determinante para lograr la coherencia de las tareas, técnicas y herramientas, al compás de su evolución.

John T. Crawford se graduó BA en contabilidad por la Universidad de Temple en 1966. Trabajó en Arthur Andersen Co como consultor y a continuación dedicó 10 años a INA, desempeñando un papel fundamental en el desarrollo de sistema electrónicos de seguros. En 1979 el Sr. Crawford se incorporó a Hartford Insurance Group, donde se ha responsabilizado del desarrollo de aplicaciones y soporte técnico, incluyendo el desarrollo de sistemas y programas de proceso de datos para todas las líneas de la empresa. Desde 1985 es presidente de Hartford Integrated Technologies, una filial creada para comercializar, llave en mano, los sistemas de desarrollo de aplicaciones introducidos satisfactoriamente en The Hartford.

Herramientas UNIX para el desarrollo de sistemas de terminales

ITT Courier desarrolla programas para los múltiples microprocesadores integrados en pantallas, impresoras y controladores de comunicaciones. Durante los últimos cinco años, UNIX ha demostrado sus ventajas como sistema de desarrollo por su flexibilidad, portabilidad a diferentes equipos y disponibilidad de una gama cada vez mayor de herramientas lógicas.

A. Povelones
ITT Courier, Tempe, Arizona,
Estados Unidos de América

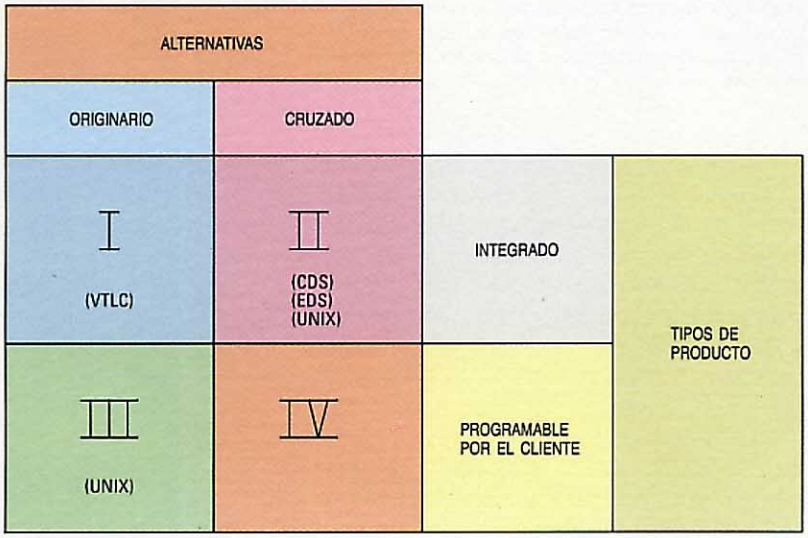
Introducción

ITT Courier crea, produce y mantiene una gama completa de terminales, impresoras, controladores y estaciones de trabajo compatibles con IBM. Los controladores y las estaciones de trabajo incorporan potentes microprocesadores y requieren cientos de k-octetos de programación, e incluso los terminales y las impresoras utilizan más de 100 k-octetos de programación para implantar la tecnología de comunicación del IBM 3270. Además, hay microprocesadores de un solo circuito integrado que controlan las funciones del teclado, y microprocesadores especializados que atienden los canales de comunicación de alta velocidad hacia el ordenador principal, compartidos por los productos ITT Courier con los periféricos de cinta y disco de gran volumen.

Todos estos productos deben desarrollarse y fabricarse del modo más económico posible. Hay además una persistente necesidad de mejorar los productos actuales con la adición de nuevas funciones. Por otro lado, el tiempo de anticipación para el lanzamiento de nuevos productos es ahora más corto que nunca.

Para cumplir estos importantes requisitos se necesita un entorno de desarrollo eficiente y económico. Al principio ITT Courier utilizaba sistemas de elaboración propia para el desarrollo de sus productos, pero en 1980 se vio claramente que costaría demasiado potenciar los dos sistemas de desarrollo entonces existentes al compás de las necesidades del momento, y que convendría más adoptar un sistema basado en el UNIX.

Figura 1
Tipos de desarrollo en ITT Courier.



Tipos de desarrollo de programación

Hay cuatro métodos para crear y mantener productos integrados y programables por el usuario (Fig. 1); abarcan las categorías básicas de desarrollo originario y cruzado. Los sistemas "integrados" tienen uno o más microprocesadores que proporcionan las funciones características del producto sin que el usuario conozca la tecnología de base ni pueda programar funciones adicionales. La posibilidad de programación por el usuario implica que el producto pueda utilizarse para desarrollar programación adicional, y no simplemente para selección de parámetros de operación. En este sentido un vídeo casero no es programable, aunque se diga que el usuario "programa" las funciones de grabación y reproducción.

Hasta hace poco todos los productos de ITT Courier usaban microprocesadores integrados, pero recientemente la compañía ha introducido su primer producto programable por el usuario: el procesador de aplicación. Es muy probable que esta facultad del usuario adquiera más importancia en

el futuro. En un entorno de desarrollo originario las herramientas se ejecutan en el producto mismo; en cambio, si el desarrollo es cruzado, se emplea un ordenador separado. A menos que el producto sea programable por el usuario, como lo es un ordenador personal o principal, las ventajas de codificar (escribir las instrucciones lógicas) y depurar (hacer converger el comportamiento real del producto con el deseado) sobre el mismo dispositivo, desaparecen ante los inconvenientes siguientes:

- Las herramientas de desarrollo son cada vez más complejas y pueden exigir más recursos que los ofrecidos por el propio producto; por ejemplo, hay microprocesadores especializados que no pueden albergar un ensamblador, mucho menos un compilador de lenguaje de alto nivel.
- El producto físico quizá no aporte las funciones necesarias de apoyo al desarrollo. Con frecuencia se introducen versiones específicas para admitir equipo adicional, tal como disco, impresora o un terminal de pantalla determinado.
- El diseño físico puede ser exclusivo, lo que implica tener que escribir las herramientas de desarrollo al mismo tiempo que el producto lógico. Además, éstas herramientas tal vez hayan de reescribirse a cada nuevo cambio de la arquitectura del producto.

Primeros sistemas de desarrollo de ITT Courier

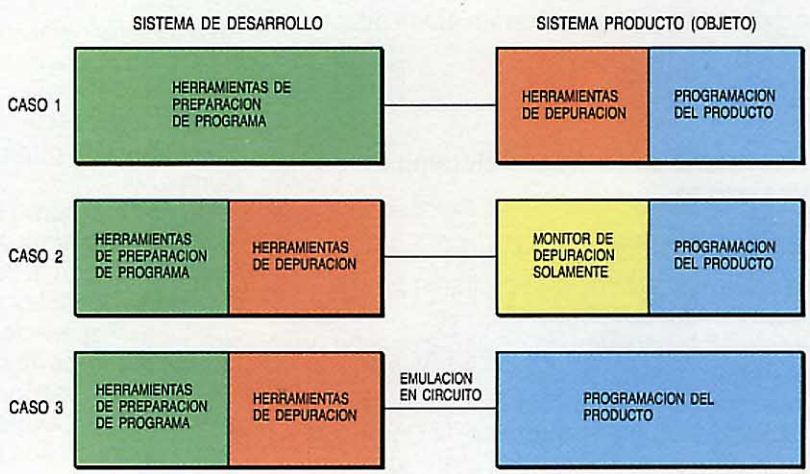
En 1975, ITT Courier utilizó un sistema originario de desarrollo para obtener un producto llamado VTLC (controlador de línea de terminal virtual). Primeramente escribió un juego de herramientas de desarrollo que funcionase en el VTLC, el cual sistema aparece en el cuadrante I de la figura 1.

En 1979 hubo que adoptar una tecnología de microprocesador más potente, y por ello se decidió utilizar métodos de desarrollo cruzado con el fin de que los futuros cambios en la arquitectura no afectaran a las herramientas genéricas de desarrollo. Esto se llevó a cabo diseñando la CDS (estación de desarrollo Courier) y unas potentes herramientas de desarrollo de microprocesadores. La productividad mejoró ostensiblemente, y el sistema originario no ha vuelto a utilizarse desde entonces. La CDS aparece en el cuadrante II de la figura 1.

La figura 2 ilustra diversos enfoques del desarrollo cruzado, siendo el "sistema objeto" el producto físico y su microprocesador. En todos los casos las herramientas de preparación de programas (editores de texto, gestores de ficheros, ensambladores, compiladores) residen en el ordenador del sistema de desarrollo.

Los sistemas de desarrollo cruzado se distinguen por las diferencias de complejidad del entorno de depuración que reside en el sistema objeto. Con la CDS, toda la depuración se realizaba mediante programas a medida ejecutados en el microprocesador objeto (caso 1 de la figura 2). Sin embargo, los programas necesarios para una depuración eficaz pronto llegaron a ocupar demasiada memoria en el producto. Eliminadas las restricciones impuestas por las herramientas de desarrollo, Courier afrontó otras análogas en las herramientas de depuración. Para minimizar el uso de memoria (caso 2 de la figura 2), estas últimas herramientas sólo admitían ciertos tipos de terminal de pantalla y había que conectar el terminal adecuado antes de la depuración. De este modo, para localizar un fallo había que instalar en las dependencias del usuario los terminales y programas oportunos, y después muchas veces los problemas no volvían a presentarse. Tal ineficacia movió a ITT Courier a idear otro procedimiento.

Figura 2
Métodos de desarrollo cruzado.



El caso 3 de la figura 2 ilustra esta tercera vía llamada ICE (emulación en el circuito), en la cual se quita el microprocesador del producto y se le sustituye por un conector al equipo ICE que realiza las funciones del microprocesador objeto; de este modo no se cambian el equipo físico ni la programación del producto, ya que la depuración se efectúa a través del ICE y no es necesario añadir funciones al producto. Este método presenta ventajas sobre las herramientas de depuración residentes en el sistema objeto: por ser externo al sistema que se está probando, el ICE puede notificar ano-

malías que habrían impedido que tales herramientas siguieran funcionando en el microprocesador del producto, y puede asimismo dar más información, como por ejemplo rastrear la secuencia de eventos que conducen a un fallo.

Sin embargo, la tecnología ICE fue de utilidad limitada porque en los productos de ITT Courier había circuitos que asignaban muchas direcciones de memoria a la misma dirección del microprocesador, por lo cual toda visión externa desde el ICE carecía de sentido. En consecuencia, este procedimiento no se utilizó hasta la introducción de la Serie 9000, si bien otros requisitos de esta Serie hicieron replantearse la CDS y el modo de escribir sus herramientas de desarrollo.

Necesidad de un nuevo sistema de desarrollo

En 1980, ITT Courier comenzó a investigar microprocesadores que pudieran programarse en lenguajes de alto nivel, con vistas a reducir la cantidad de código a escribir, y por tanto acelerar el desarrollo. Tras haber invertido cuatro hombres-año en escribir y documentar un ensamblador, la Compañía afrontó la tarea, mucho más ardua, de desarrollar un compilador adecuado pues ninguno del mercado podría trabajar con la CDS. Esto condujo a revisar la metodología CDS, obligando a escribir, documentar y mantener todas las herramientas de desarrollo.

Además, la CDS estaba ya al límite de su capacidad de memoria, y debido a su arquitectura no podía ampliarse. Por si fuera poco, la carencia de soporte de comunicaciones impedía utilizar la CDS con el equipo ICE y aislaba a los ingenieros del resto de la Compañía. Estos problemas se mezclaron a la falta de integración con otras funciones de la ingeniería, tales como publicaciones técnicas.

Hacia 1981, ITT Courier definió sus requisitos para un tercer sistema de desarrollo, denominado EDS (sistema de desarrollo de ingeniería), cuyo objetivo sería:

- Optimizar la productividad de al menos 60 ingenieros de programación (tanto de desarrollo como de mantenimiento) asignados a equipos de hasta 40 personas.
- Automatizar las actividades de soporte de otros 150 técnicos.
- Ayudar y controlar el desarrollo de bibliotecas de producto, con 500.000 líneas de código fuente (5.000 módulos fuente) como mínimo, para todos los microprocesadores usados en productos de ITT Courier.
- Probar los productos en un entorno IBM de equipo y programas.
- Hacer utilizables las herramientas de programación existentes en el mercado, evitando tener que acometer grandes desarrollos de herramientas.
- Trabajar con instrumentación de laboratorio, tal como el ICE, que se comunique por medio de protocolos ASCII asíncronos.

Al mismo tiempo se señalaron varios objetivos secundarios, como el uso dentro del EDS de terminales de pantalla e impresoras ITT Courier para que los ingenieros conozcan mejor los productos que crean, y el compartir los recursos de ordenadores existentes con otros departamentos de la Compañía.

Tanto CDS como EDS tenían la finalidad de proporcionar un entorno de desarrollo económico, con la flexibilidad necesaria para aprovechar las nuevas tecnologías que fuesen apareciendo. El EDS debe ofrecer recursos de desarrollo cruzado para los microprocesadores utilizados en productos de ITT Courier, y ser capaz de crear medios para dichos productos (disquetes y circuitos integrados programables). También ha de permitir a los ingenieros las pruebas tanto del prototipo como de productos de serie, y facilitar la automatización e integración de las pruebas de desarrollo y funcionales. Por añadidura, el EDS ha de apoyar la gestión de configuración y la documentación del proyecto, y ofrecer un conjunto de herramientas de uso general a los profesionales de la ingeniería y a la dirección de la misma.

Necesitándose equipos IBM para probar los productos de ITT Courier, parecía lógico utilizar circuitos y sistemas operativos de aquella marca. Al principio hubo un problema por no disponer de ningún compilador cruzado utilizable con Unix en un ordenador IBM. Afortunadamente, sin embargo, apareció entonces la implantación UTS del Unix para ordenadores de IBM; ésta ejecuta en VM/370, facilidad de máquina virtual que permite a diferentes sistemas operativos compartir la misma CPU. Gracias a UTS se ha podido instalar cualquier programación Unix ofrecida en forma de lenguaje fuente, aceptando por ejemplo programas escritos para Unix en equipo de otros fabricantes y distribuidos por ITT Programming.

La capacidad de ejecutar Unix en equipo IBM hizo posible la expansión del EDS, ampliando el centro de datos existentes y salvando el gran número de problemas que entraña el uso de sistemas múltiples.

Razones para comprar herramientas

A la postre, la adquisición de herramientas en vez de desarrollarlas ha dado buen resultado económico. El ensamblador para CDS consumió alrededor de cuatro hombres-año de desarrollo, incluyendo un año para preparar el manual del usuario. En menos de cinco meses, Courier compró e instaló la primera cadena de herramientas, compuesta de un compilador cruzado para microprocesador que optimiza los resultados, más un ensamblador, un montador y variados programas soporte de utilidad. Cada herramienta venía con su propia documentación.

La adquisición de herramientas tiene una serie de ventajas frente al desarrollo. En efecto, evita dificultades de mantenimiento y hace innecesarios los intercambios de capacitación, las pruebas de regresión y una fuerte dotación de personal de soporte. Además, se supone que el vendedor y los demás usuarios habrán detectado y resuelto muchos problemas. Se puede así empezar a considerar la programación exenta de faltas como un recurso asequible.

Alguno podría argüir que el desarrollo privado es el mejor medio de satisfacer las necesidades específicas de una compañía. Es cierto, por ejemplo, que el ensamblador CDS de 8 bit era inicialmente lo más avanzado, pero pronto muchos ensambladores comerciales igualaron o superaron sus posibilidades.

ITT Courier decidió aplicar sus recursos a la integración de herramientas comerciales de modo innovador, en vez de asignar demasiado peso a las necesidades "especiales" y, o bien esperar herramientas que nunca llegaban a aparecer, o bien invertir costosos recursos en volver a inventar algo ya conocido. Por ejemplo, ITT Courier adoptó el lenguaje "C" y rápidamente preparó su interfaz con un sistema operativo en tiempo real para la nueva Serie 9000, en lugar de intentar escribir el soporte en tiempo real en un compilador "C" o diseñar un nuevo lenguaje.

No obstante, se necesita elegir con cuidado las herramientas. A la hora de comprar una herramienta nueva habrá que preguntarse lo siguiente:

- ¿Soporta plenamente los métodos de desarrollo deseados?
- ¿Ayuda a organizar grandes proyectos?
- ¿Proporciona comunicación entre usuarios del departamento de ingeniería, de otros departamentos, de otras compañías y de otras redes?
- ¿Puede dirigirse el sistema resultante de manera económica?
- ¿Es posible ampliar el sistema elegido en todas las direcciones (usuarios, organizaciones, aplicaciones, tecnologías)?
- ¿Reduce al mínimo las dependencias de herramientas y tecnologías concretas?
- ¿Ofrece un entorno de desarrollo homogéneo?
- ¿Enlaza las actividades de desarrollo, integración, prueba, evaluación y documentación?

Estas mismas preguntas se repiten periódicamente para valorar los resultados de la estrategia[1]. En tanto que las respuestas sean afirmativas, aquélla será eficaz.

Conversión de CDS a EDS

Incluso mientras se estaban adoptando nuevos sistemas de desarrollo, había que continuar el soporte de los productos anteriores. Los productos basados en VTLC continuaron manteniéndose en ese sistema de desarrollo después de introducida la CDS, pero se constató que si se adoptara la misma táctica con los productos CDS al aparecer el EDS se retrasaría el provecho obtenido de este último sistema.

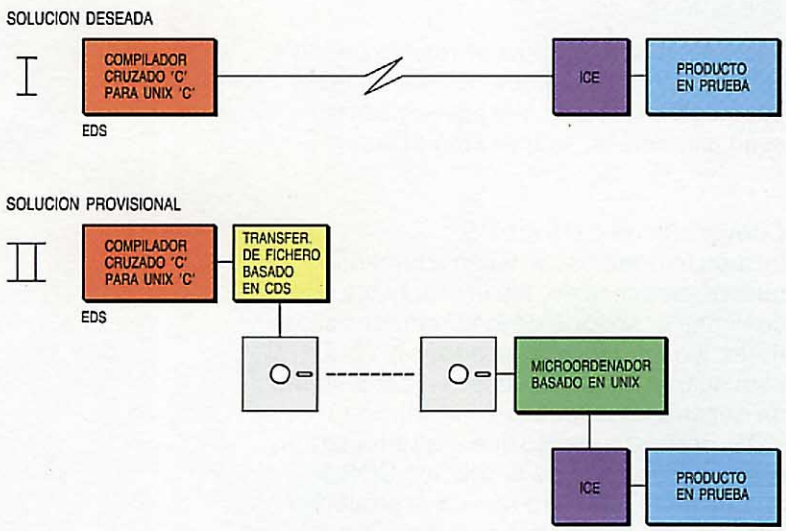
Mientras se seguía con los desarrollos en CDS el coste de mantenimiento de estas unidades aumentaba. El problema residía en instalar un ensamblador y montador a medida en EDS, pues la conversión parecía imposible sin reescribir por completo el ensamblador y el montador en un lenguaje con soporte en el EDS. Se decidió entonces escribir un programa capaz de emular un subconjunto de las instrucciones máquina para la CDS. Dicho emulador se escribió y probó en cuatro meses, con lo cual pudieron pasar a EDS todos los desarrollos de programación CDS. Debido a la potencia del EDS, las versiones objeto sin modificar del ensamblador y del montador se ejecutan más aprisa bajo emulación en EDS que en CDS.

Pronto se demostraron otras ventajas del EDS. Por ejemplo, las actualizaciones de la programación del sistema se instalan normalmente mediante una sola orden, frente a los dos días de trabajo que esto implica en las estaciones CDS. Análogamente, la obtención de copia de seguridad es automática y fiable, en vez de depender de cada ingeniero. Además el EDS elabora automáticamente estadísticas para la gestión del sistema.

La instalación inicial del Unix se vio limitada por la carencia de un soporte de comunicación de datos ASCII asíncronos. Pensando que esta limitación acabaría por desaparecer, se adoptó una facilidad provisional para enlazar el equipo ICE de labora-

torio con el EDS (Fig. 3), la cual requiere que los ingenieros transfieran la programación en disquete desde el EDS al ICE. Originalmente, un microordenador basado en Unix controlaba las unidades ICE, reduciendo al mínimo los costes de adiestramiento. En 1985 el UTS dispuso ya de medios de comunicación asíncrona ASCII y se suprimió el citado microordenador. El

Figura 3
Soluciones provisional y final para ICE.



equipo prototipo puede hoy depurarse por medio del ICE directamente conectado al EDS.

Herramientas disponibles

En un principio se temió que no bastaran las herramientas disponibles en Unix para satisfacer las necesidades de Courier. En 1982 se eligió un compilador cruzado que demostró su aptitud para el desarrollo del equipo Serie 9000. Desde entonces se han añadido otros compiladores y ensambladores, unidos a un rico conjunto de programación auxiliar, habiéndose centuplicado los productos Unix adecuados para aplicaciones de ingeniería.

El entorno de desarrollo de ITT Courier comprende dos compiladores cruzados de lenguaje "C" para los microprocesadores 68000/68008, y varios otros para los 8086/8088. El sistema de desarrollo es igualmente apropiado para las dos familias de micros, y ambas tienen soporte en ICE. Con vistas al desarrollo de lógica microprogramada, ITT Courier adquirió una serie de ensambladores para microprocesadores de 8 bit, y preparó en dos meses un ensamblador para microprocesadores de porciones de bits utilizando código fuente de dominio público.

El control de las fuentes está basado en el SCCS (sistema de control de código fuente) de Unix, añadiendo ITT Courier alrededor de un hombre-año de codificación a medida. Se obtiene así un sistema bien ajustado a la organización del proyecto y a las prácticas de gestión.

Los ordenadores personales, tales como el ITT XTRA*, proporcionan enlaces de alta velocidad al EDS con el fin de copiar disquetes y controlar dispositivos para programación de circuitos integrados.

La depuración comienza en el sistema de desarrollo, donde se prueban parcialmente los programas en lenguaje "C" con independencia del equipo objeto. Los rápidos ciclos de compilación y prueba se ayudan de una versión especial del compilador "C" que combina temporalmente los mensajes de error en un programa fuente y permite el uso de un editor de textos para corregir las instrucciones erróneas y los códigos contiguos. Automáticamente llama al compilador y el ciclo se repite hasta que la compilación quede exenta de errores.

Aunque el proceso a base de compiladores cruzados en EDS parezca más complicado que el desarrollo sobre el microprocesador objeto, las ventajas superan a los inconvenientes. Dado que el "C" es también lenguaje primario del Unix, se puede hacer mucha depuración inicial en EDS. En las primeras etapas de un proyecto se observaron relaciones de 5 a 1 entre las compilaciones en EDS y las compilaciones cruzadas, lo que indica claramente que los ingenieros preferían el nuevo sistema.

Los temores iniciales sobre dificultades semánticas y de sintaxis al cambiar del EDS a los compiladores cruzados no se confirmaron. Si acaso hubiera discrepancias semánticas, una facilidad especial de la cadena de herramientas (ilustrada en la figura 4) ayudaría a separarlas de los errores cometidos inadvertidamente al escribir la lógica del programa. El ingeniero no tiene sino que repetir cualesquiera pruebas que se hayan ejecutado en EDS (arquitectura de máquina IBM 370) y ejecutarlas ahora en ICE. Sólo si aparece una incoherencia entre unas y otras, podrá dudarse de la generación de código del compilador. La mayoría de los ingenieros se saltan confiadamente este paso y van directamente de la prueba en EDS a la prueba en el equipo prototipo.

Uso del VM

La programación VM/370 (facilidad de máquina virtual) de IBM permite que varios sistemas operativos se ejecuten concurrentemente en una máquina única, lo que hace posible desarrollar productos en Unix al mismo tiempo que se están calificando mediante los sistemas operativos MVS/CICS/VTAM y se preparan nuevas pruebas

* Marca registrada del Sistema ITT

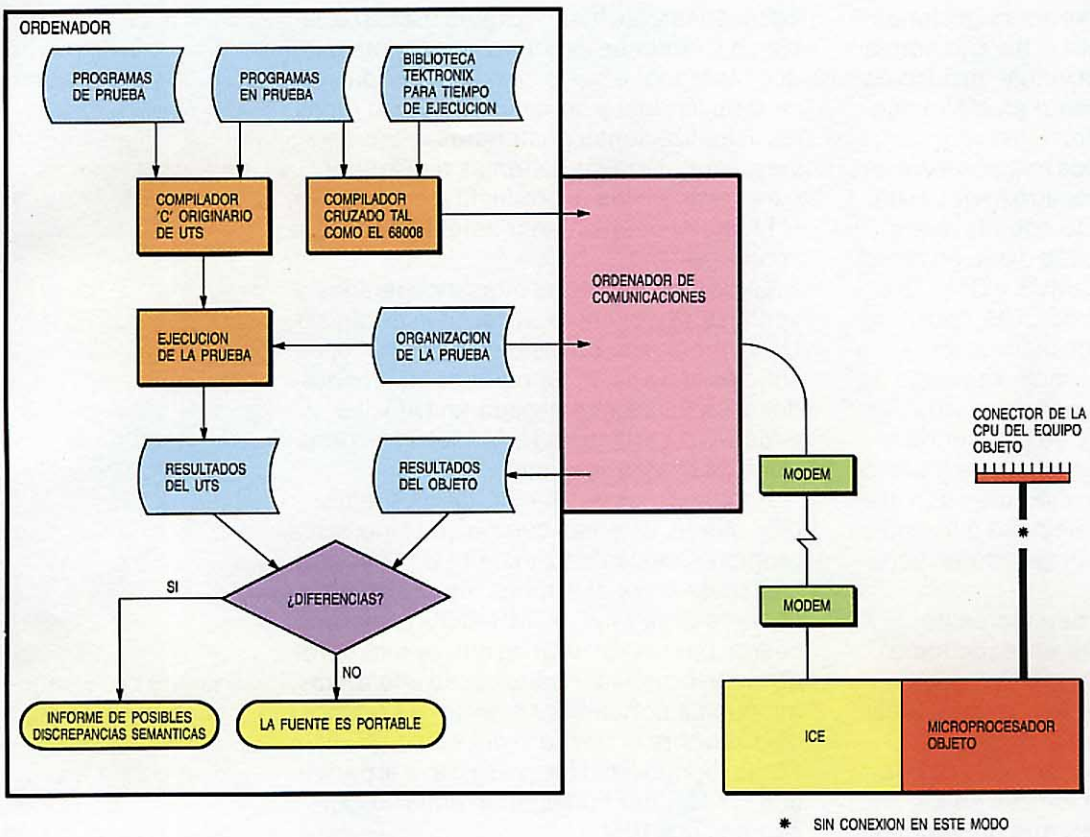
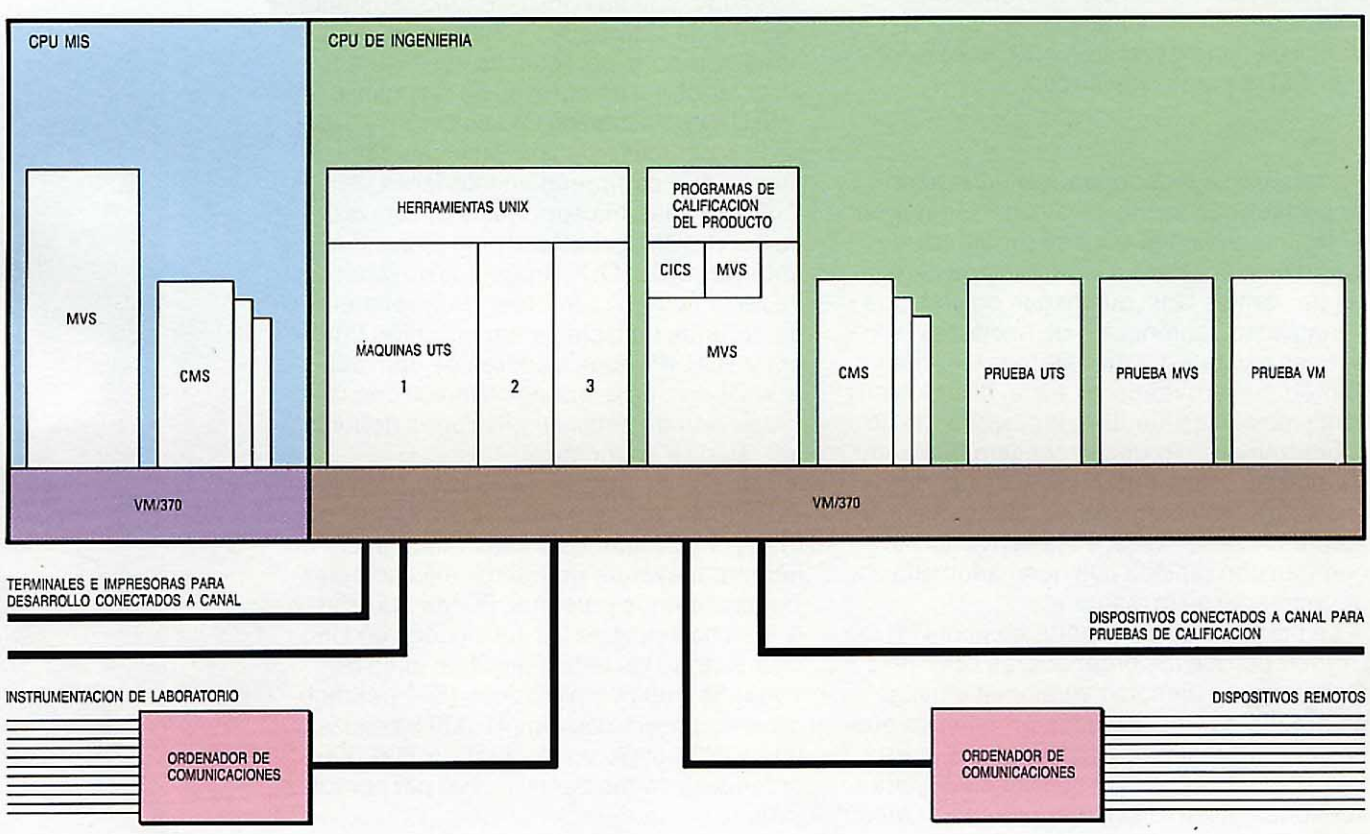


Figura 4
Eliminación de las dependencias objeto/ compilador.

de calificación con el sistema operativo CMS (Fig. 5). Cada sistema operativo gestiona sus recursos virtuales como si fueran reales. El VM mantiene la asociación entre recursos virtuales y físicos, facultando al personal para equilibrar las cargas entre sistemas operativos. También elimina la necesidad de mantenimiento fuera de

Figura 5
Entorno VM.



horas de trabajo, pues las actualizaciones pueden probarse durante el horario normal definiendo otra máquina virtual. Incluso es posible probar parcialmente en el VM nuevas versiones del mismo.

Con el VM se utilizan los mismos terminales e impresoras para desarrollo que para pruebas. Los servicios de soporte, tales como la copia de seguridad diaria en disco, son idénticos para Unix, MVS y CMS. Se comparten los recursos de CPU, haciendo así las veces de un procesador único y potente que atiende a las múltiples colas de peticiones de todas las máquinas virtuales. La solución de "servidor único" tiende a hacer los tiempos de respuesta de transacciones interactivas más uniformes que los observados cuando se ejecutan diferentes sistemas operativos en ordenadores separados[2].

Una función de los ordenadores de comunicaciones consiste en la recuperación automática de errores en las líneas, impidiendo así eficazmente que los dispositivos prototipo estropeen el trabajo de desarrollo. El VM además aísla al UTS de errores y trastornos funcionales en los MVS/CICS/VTAM y CMS que provengan de la prueba del producto. La protección que proporcionan el VM y el ordenador de comunicaciones permite realizar la mayor parte de la prueba del producto en el mismo ordenador y durante las mismas horas que el trabajo de desarrollo. Sin embargo, como no puede garantizarse protección contra fallos en los controladores de prototipo conectados al canal IBM, la prueba de la unidad de control conectable al canal se planifica en momentos en que no se necesite la EDS para el desarrollo.

Portabilidad e independencia del equipo

La portabilidad anima a los suministradores a ofrecer versiones Unix de productos de programación. Pese a la pretendida carencia de normas Unix que hagan posible una portabilidad "auténtica", de hecho las normas existen[3,4]. Los sistemas Unix de numerosos proveedores están basados en el mismo código fuente, y la mayor parte de la programación parece ser altamente compatible con esta norma fáctica. En general la programación comprada se puede fácilmente utilizar en UTS, y los vendedores corrigen con rapidez cualquier anomalía dependiente de la máquina.

La portabilidad es importante para ITT Courier porque los ordenadores IBM de la Compañía no ejecutan versiones binarias de productos Unix basados en equipos que no sean compatibles con IBM. Hay, pues, tres alternativas: o el vendedor suministra versiones objeto de las herramientas Unix adecuadas a las instrucciones IBM/370, o bien a Courier se le concede la fuente para que incorpore a sus sistemas el producto – recompilándolo y depurándolo – con todas las actualizaciones posteriores, o bien el vendedor utiliza los sistemas de Courier para llevar a ellos el producto y en ese caso ITT Courier debe obtener derechos sobre la versión binaria.

La portabilidad tiene otras inesperadas ventajas. Por ejemplo, la decisión de utilizar Unix como base de sistema de desarrollo anticipó la decisión de obtener un procesador de aplicaciones basado en Unix. La experiencia previa en Unix aceleró el desarrollo de este procesador.

A menudo es posible portar programación desde sistemas que no son Unix si los programas están escritos en "C". Así las analogías entre el sistema jerárquico de ficheros Unix y los del MS-DOS se aprovecharon para desarrollar un enlace microprocesador-ordenador principal de alto rendimiento. La portabilidad suaviza las restricciones sobre la elección del equipo a utilizar. La portabilidad fue pieza clave al permitir a ITT Courier conjugar un entorno Unix con equipos IBM.

Otras ventajas

Pese a la relativa sencillez que presenta el Unix para un ingeniero reciente, sus conceptos son lo suficientemente claros y concisos para que un técnico experimentado implante ideas sin pérdida de tiempo; ni siquiera es fácil que un ingeniero muy versado agote sus posibilidades. Sustenta la gama de actividades y funciones realizadas por todo el personal de ingeniería incluyendo los desarrollos de programación.

El Unix está todavía en evolución, y hace poco se anunciaron ampliaciones del mismo que comprendían interfaces al DISOSS (sistema soporte de oficina distribuida) de IBM, capacidad para ejecución en entornos MS-DOS[5], productos de sistemas de red y ficheros remotos, interpretadores de órdenes múltiples, comunicación Ethernet y TCP/IP, visualizadores de ventana, y el X/Open[6]. La experiencia indica que la instalación de estas ampliaciones deberá ser rápida y económica.

Aunque no haya medidas cuantitativas del aumento de la productividad en el desarrollo, ITT Courier está elaborando ahora un número mayor de productos más complejos, con menos personal. Puede apreciarse la marcha actual de las actividades en Unix examinando las estadísticas de junio de 1986: 51.000 compilaciones (850 por ingeniero de programación), 47.000 llamadas al editor (20 por persona y día) y 2.600.000 órdenes de todas clases (1.100 por persona y día).

Conclusiones

El objetivo de ITT Courier fue seleccionar un sistema capaz de dar pleno soporte a los proyectos de desarrollo de la Compañía. La experiencia revela que la elección de cambiar a Unix ha resultado altamente productiva y parece digna de que la estudien otras compañías de ITT con exigencias similares.

Agradecimientos

Deben agradecerse sus contribuciones a los siguientes empleados de ITT Courier: Doug Brown, Claudia Buckner, Robert-Flandrena, John Lessly, Irv Norfleet y Lee Steinbrenner.

Referencias

1 A. Povelones: A Study of Tools Strategies: *ITT Conference on Programming Productivity and Quality Proceedings*, ITT Programming, Stratford, 1983, págs. 2–9.

2 N. Macklin: Airport'84: A Parable, a Queueing Argument for Big Unix: *Unix/Review*, San Francisco, octubre 1984, págs. 28–30.

3 System V Interface Definition: *ATT Corporation*: Indianapolis, 1985.

4 B. W. Kernighan y D. M. Ritchie: The C Programming Language: *Prentice Hall*, Englewood Cliffs, 1978.

5 R. Cook: Unix Barters for Business: *Computer Decisions*, 15 julio 1986, pág. 34.

6 R. T. Gallagher: Europeans are Counting on Unix to Fight IBM: X/Open Group Works for a Standard Operating System: *Electronics*, 10 julio 1986, págs. 121–122.

Arthur Povelones nació en Filadelfia (Pensilvania, EE. UU.) en 1948. Se graduó en la Universidad Drexel de aquella ciudad, y en la Universidad Lincoln del mismo estado, en 1971, en ingeniería eléctrica y matemáticas. En 1985 obtuvo el grado Master en administración de empresas en la Universidad del Estado de Arizona. El Sr. Povelones trabajó para las compañías RCA, Addressograph-Multigraph y Harris antes de ingresar en ITT Courier, donde actualmente se encarga de los sistemas de soporte del desarrollo y del laboratorio de ingeniería.

Aportación de los factores humanos al desarrollo de productos

La ingeniería de factores humanos juega un papel fundamental en el desarrollo de productos de gran aceptación mediante la consideración de las características y necesidades del usuario, durante el proceso de diseño. Se puede mejorar notablemente la eficacia del producto y su aceptación asegurando que el diseño del interfaz del usuario sea de alta y uniforme calidad.

F. R. Brigham
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

La ingeniería de factores humanos consiste en la aplicación sistemática del conocimiento de las características y necesidades humanas al desarrollo de productos, sistemas y ambientes. Su objetivo es asegurar que los productos sean sencillos de aprender, eficaces y fáciles de usar, de operación cómoda y segura, y atractivos para el usuario.

La propiedad esencial de la ingeniería de factores humanos es su enfoque de diseño centrado en el usuario, lo que conlleva la aplicación de conocimientos de psicología cognoscitiva, fisiología y anatomía. Es importante notar que no hay tecnologías o soluciones de los factores humanos aplicables genéricamente, sino que su enfoque es más bien pragmático y comienza por analizar y definir las características del grupo (o grupos) de usuarios y su tarea. Por tanto, los dos primeros y más importantes preceptos son: *conocer al usuario* y *conocer la tarea*.

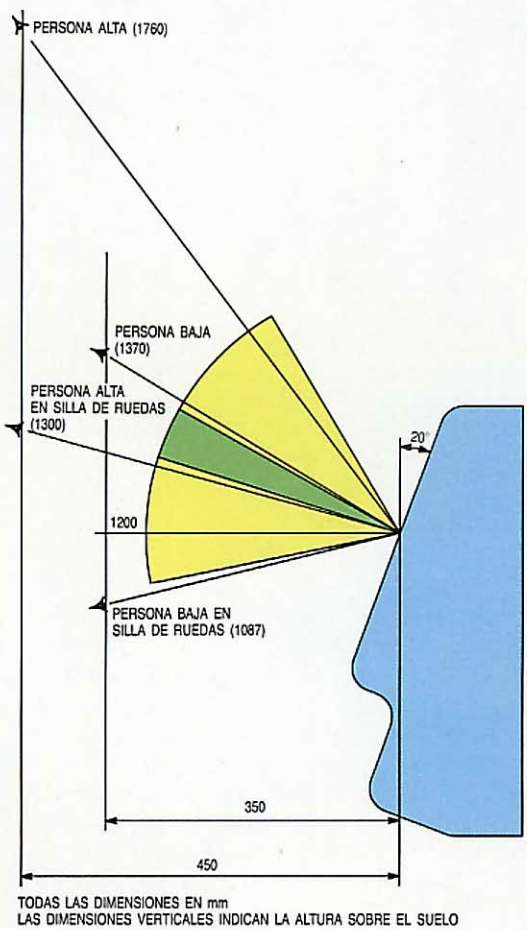


Figura 1
Campo de visión (área sombreada) de una pantalla de cristal líquido de un teléfono de pago, mostrando el ángulo de visión para varios tipos de usuarios.

Características del usuario

En el diseño de "productos inteligentes", o sea, productos que suponen un diálogo entre el usuario y el sistema, son de particular importancia las características del usuario como procesador de información. Por ejemplo, su memoria a corto plazo y su capacidad para procesar información adolecen de limitaciones bien conocidas. Las pantallas visualizadoras deben por tanto minimizar la información irrelevante y presentar, codificada y estructurada de manera apropiada, la necesaria. El interfaz con el usuario debe responder a un modelo sencillo que ayude al usuario a comprender la estructura del sistema y su funcionamiento.

Sin menoscabo del valor de la psicología cognoscitiva como aportación a la ingeniería de factores humanos, las características anatómicas y fisiológicas del usuario son también importantes para el diseño de muchos productos ITT. Por ejemplo, la figura 1 muestra que la pantalla de cristal

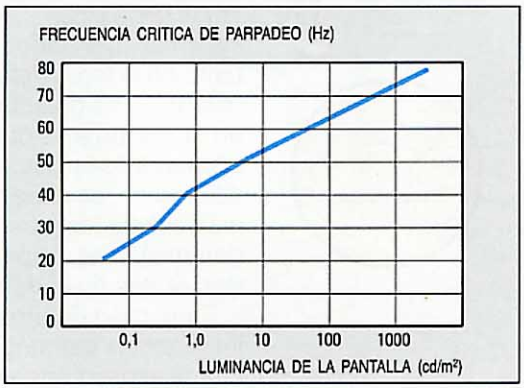


Figura 2
Frecuencia crítica de parpadeo en función de la luminancia de la pantalla (fósforo p4).

líquido usada en un teléfono de pago debe colocarse cuidadosamente en una posición e inclinación tal que ofrezca buena visibilidad, tanto para una señora bajita en silla de ruedas como para un hombre alto.

El parpadeo de un tubo de rayos catódicos sirve de ejemplo de problema en el que los mecanismos subyacentes son fisiológicos. La figura 2 muestra la frecuencia a la que empieza a percibirse dicho parpadeo en función de la luminancia de la pantalla (la perceptibilidad del parpadeo varía con las personas y depende de muchos otros factores, entre ellos el tipo de fósforo). El parpadeo es más perceptible cuando aumenta la luminancia de la pantalla y, por tanto, es un problema particular de las pantallas que presentan caracteres oscuros sobre un fondo claro. Se afirma que este tipo de pantallas causan menos fatiga ocular por disminuir el contraste entre la luminosidad de la pantalla y la del entorno (por ejemplo, documentos fuente), y hacerse menos perceptibles las reflexiones. Hay normativas en este sentido para los sistemas de datos suministrados en algunos países europeos (Escandinavia, Alemania Occidental). Las pantallas diseñadas para tales países deben tener una alta cadencia de refresco y utilizar un fósforo adecuado para eliminar el parpadeo.

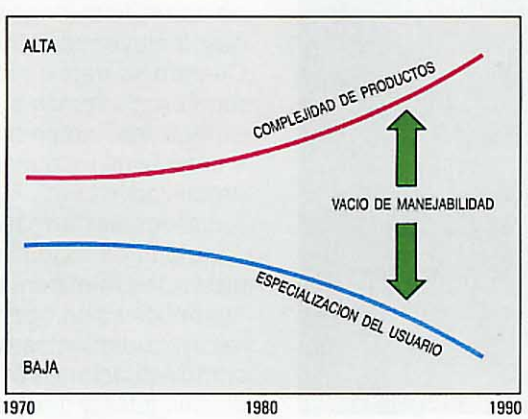


Figura 3
El *vacío de manejabilidad* aumenta con la complejidad de los productos y la menor especialización de los usuarios.

Concepto de manejabilidad

Office 2000 es un término genérico que cubre la gama de equipos de ITT Europa para usuario final. Uno de los criterios de selección más importantes para este tipo de productos es su manejabilidad, sobre todo en aquéllos que ofrecen un amplio abanico de posibilidades.

La manejabilidad, que corrientemente se denomina facilidad o comodidad de uso, abarca aquellos atributos que hacen al equipo sencillo de aprender, eficiente y de uso agradable, evaluados por el tiempo invertido en las tareas y el aprendizaje, las tasas de error y las opiniones de usuarios.

Varios estudios han demostrado las ventajas que pueden desprenderse de mejorar la manejabilidad del producto. Así, por ejemplo, un mejor diseño del interfaz de usuario en un paquete de programas puede originar un aumento medio del 25% en la productividad, y una disminución de otro 25% en las tasas de error. Estas mejoras benefician tanto al suministrador del sistema como al usuario final.

La tendencia actual en productos de usuario final es ofrecer cada vez más posibilidades, a la vez que exigir menos especialización del usuario. Por ejemplo, la transferencia de llamadas era facilitada por un operador de centralita bien capacitado, mientras que hoy se espera que los abonados inexpertos efectúen ésta y muchas otras funciones desde su aparato telefónico. El resultado es el "vacío" de manejabilidad mostrado en la figura 3. Para llenar este "vacío" se necesita tanto un buen diseño en cuanto a factores humanos como una asistencia de alta calidad al usuario.

Proceso de diseño

La figura 4 muestra un modelo simplificado del proceso de diseño, ilustrando dónde se necesita información sobre factores humanos. El modelo puede aplicarse con éxito a la mayoría de los desarrollos de productos y sistemas, incluida la programación, aunque puedan darse variaciones en casos concretos.

Análisis

A lo largo de las cuatro primeras etapas del diseño deben tenerse en cuenta las características de los usuarios y los requisitos derivados de las tareas. En el caso de productos inteligentes es necesario prestar especial atención al usuario como procesador de información, por lo que la concepción del diseño debe contar con las limitaciones de memoria y proceso de información propias de los humanos.

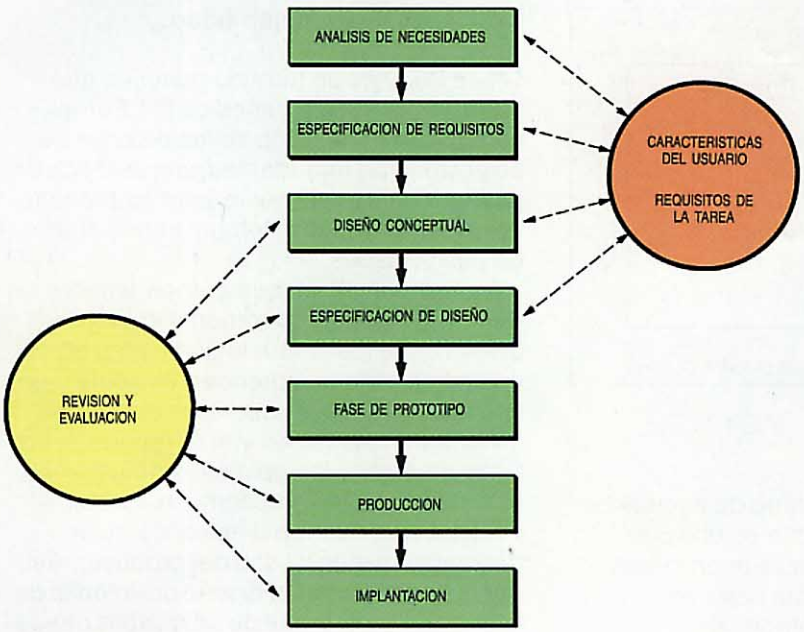


Figura 4
Modelo esquemático del proceso de diseño.

El análisis de tareas es el procedimiento normalizado para el estudio sistemático de la tarea y de las actividades básicas del usuario. Hay varias técnicas formales disponibles, como diagramas de flujo de la información, secuencias de operación y análisis de relaciones (véase la figura 5), y en otras ocasiones puede bastar un enfoque informal. En cualquier caso, formal o informal, el análisis de la tarea es fundamental en cada ejercicio de diseño de factores humanos.

Un análisis profundo de la tarea puede llevar a un tipo de interfaz de usuario radicalmente diferente del esperado. Por ejemplo, los paneles de información en la señalización ferroviaria han reproducido tradicionalmente la distribución espacial de las trayectorias de las vías. Sin embargo, en los sistemas avanzados, la parte más importante de la tarea del operador es la capacidad de decisión sobre los acontecimientos en el tiempo más que en el espacio (los sistemas de enclavamiento de seguridad se ocupan de los aspectos espaciales). Esto implica que los paneles de información deben ayudar al operador a tomar estas decisiones en el dominio temporal.

En el caso de productos complejos o de importancia estratégica es esencial involucrar al especialista en factores humanos en éstas y otras etapas del diseño. En cambio, para productos más sencillos puede bastar un ingeniero de desarrollo con alguna formación en los principios y métodos de factores humanos.

Pantalla obtenida de una demostración durante el desarrollo del interfaz de usuario para un sistema de pagos basado en ordenadores personales de un importante banco internacional.

Especificación de diseño

Dentro del diseño, el conocimiento de los factores humanos puede contribuir a desarrollar nuevos enfoques y generar especificaciones. Estas últimas deben abarcar los objetivos de manejabilidad, estándares aplicables, interfaz de usuario detallado y los procedimientos de operación. La especificación de manejabilidad ha de incluir criterios sobre el tiempo de instrucción y rapidez de aprendizaje, tiempo necesario para realizar una tarea, probabilidad de éxito y tasas de errores, y las opiniones de los usuarios. Estos criterios se definirán para un determinado grupo de usuarios, con un cierto nivel de experiencia, disponiendo de algún grado de asistencia, y otros factores pertinentes.

Hay que tomar en consideración todos los estándares nacionales, internacionales, y de la industria que sean aplicables, junto con cualquier requisito particular del mercado. En Suecia, por ejemplo, los sistemas de datos deben cumplir con requisitos locales rigurosos y en las gestiones de aprobación en Alemania Occidental (aprobación "GS") es necesario demostrar la conformidad con las normas de seguridad vigentes para las pantallas de uso prolongado y algunos otros productos de usuario final.

El diseño detallado supone la producción de especificaciones de programas y equipos, incluyendo al interfaz de usuario. Cuando se trata de sistemas completos, se requieren además otras especificaciones de factores humanos que abarcan aspectos ambientales (iluminación, ruido, etc.) y organizativos.

La ingeniería de factores humanos puede contribuir de modo importante al diseño de productos inteligentes, realizando una "especificación operacional" que detalle los procedimientos de operación – tales como actuaciones del usuario e información visualizada – y que permita estimar la mane-

Sala de control para el sistema de control ferroviario Videopult.

jabilidad de los mismos antes de la producción de un prototipo.

Las respuestas a muchas de las preguntas planteadas durante el diseño detallado pueden hallarse en el análisis de tareas previo. Una pregunta típica sería cómo estructurar los elementos de un menú, a lo que puede responderse mediante un listado alfabético o bien ordenando los elementos por la frecuencia de uso, sin que ninguna de estas soluciones sea intrínsecamente mejor que la otra. Solamente puede determinarse la más apropiada ponderando el significado de las opciones del menú con respecto a la tarea del usuario.

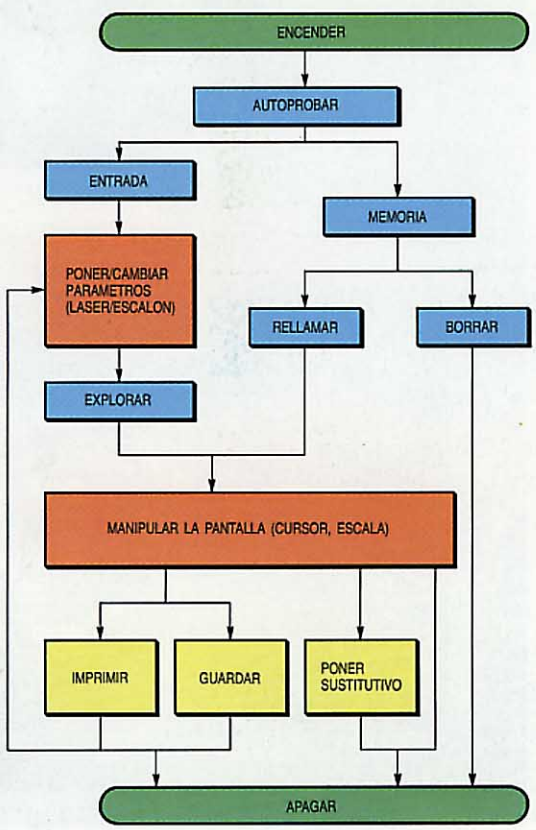


Figura 5
Diagrama de flujo simplificado de la operación de un reflectómetro de fibra óptica.

Evaluación

Para asegurar niveles altos de manejabilidad se necesita un procedimiento de revisión y evaluación, comenzando en la etapa de diseño conceptual y continuando a lo largo de las siguientes etapas, según indica la figura 4.

Durante el diseño conceptual y la producción de la especificación de diseño hay que llevar a cabo revisiones de factores humanos para asegurar que se cumplen los requisitos básicos.

Los ensayos con usuarios escogidos son valiosos siempre que se disponga de un modelo, simulador o prototipo, pues permiten evaluar las soluciones de diseño preferidas y comparar alternativas, mientras todavía es posible hacer cambios a un coste relativamente bajo. No es necesario que en dichas pruebas participe un gran número de personas, pero sí es esencial que éstas sean representativas del usuario típico. Normalmente esto impide utilizar personas del equipo de diseño u otros ingenieros de desarrollo.

En los casos en que se hayan establecido objetivos de manejabilidad, es posible un enfoque más formal de las pruebas de calificación, ya que, al haberse definido aquéllos como prestaciones funcionales, se pueden utilizar pruebas de referencia apropiadas para comprobar su cumplimiento.

La revisión y evaluación prosigue durante la producción e implantación, con pruebas de campo de las primeras series.

Implantación

La mayoría de productos no se usan aisladamente, sino que se integran en sistemas de oficina, redes de empresa, sistemas de mando y control, u otros. La implantación de un sistema completo debe involucrar a todas las fases del diseño, concentrándose en aspectos tales como la integración del equipo, el diseño de la estación de trabajo y de la consola, la distribución en planta y el diseño ambiental (control de iluminación y de ruido). También hay que considerar los factores sociales y organizativos, especialmente allá donde el sistema nuevo altere las prácticas establecidas.

Guías para el usuario

La ayuda al usuario implica diversas aportaciones desde el punto de vista de los factores humanos a lo largo de todo el proceso de diseño. Como muestra la figura 6, es útil distinguir tres tipos de ayuda al usuario, a saber, la "guía incorporada al producto" (basada en la programación), la "anexa al

producto" (físicamente unida al mismo), y la "exterior al producto" (utilizable por separado). Estos tres tipos de guías deben estudiarse conjuntamente, como parte de un enfoque global durante el diseño conceptual. De aquí que el diseño detallado de las guías incorporada y anexa al producto se considere incluido en el desarrollo de equipos y de programas, mientras que la guía exterior no podrá realizarse hasta que exista una especificación final para los procedimientos operativos y de interfaz de usuario.

Se contribuye de modo importante a la manejabilidad de un producto suministrando al usuario una guía bien estructurada y presentada de forma atractiva, y de ello tiene considerable experiencia el ITTE ESC (ITTE Europe Engineering Support Centre). Sin embargo, el objetivo primordial de la ingeniería de factores humanos es diseñar un producto que sea manejable por sí mismo, más que compensar un diseño pobre con una buena documentación de cliente.

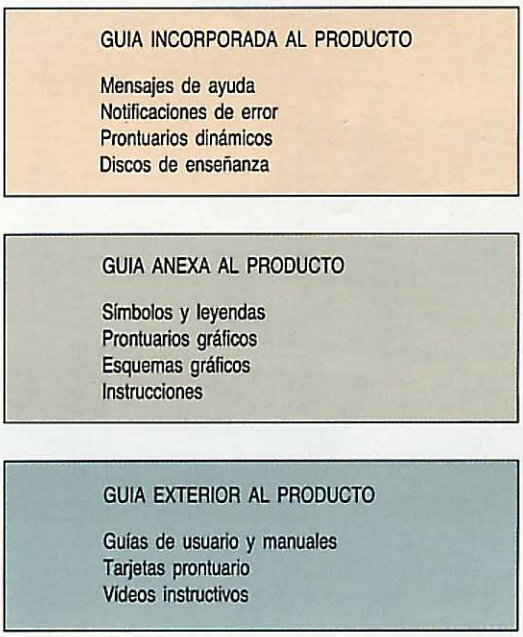


Figura 6
Tipos de guías al usuario con ejemplos.

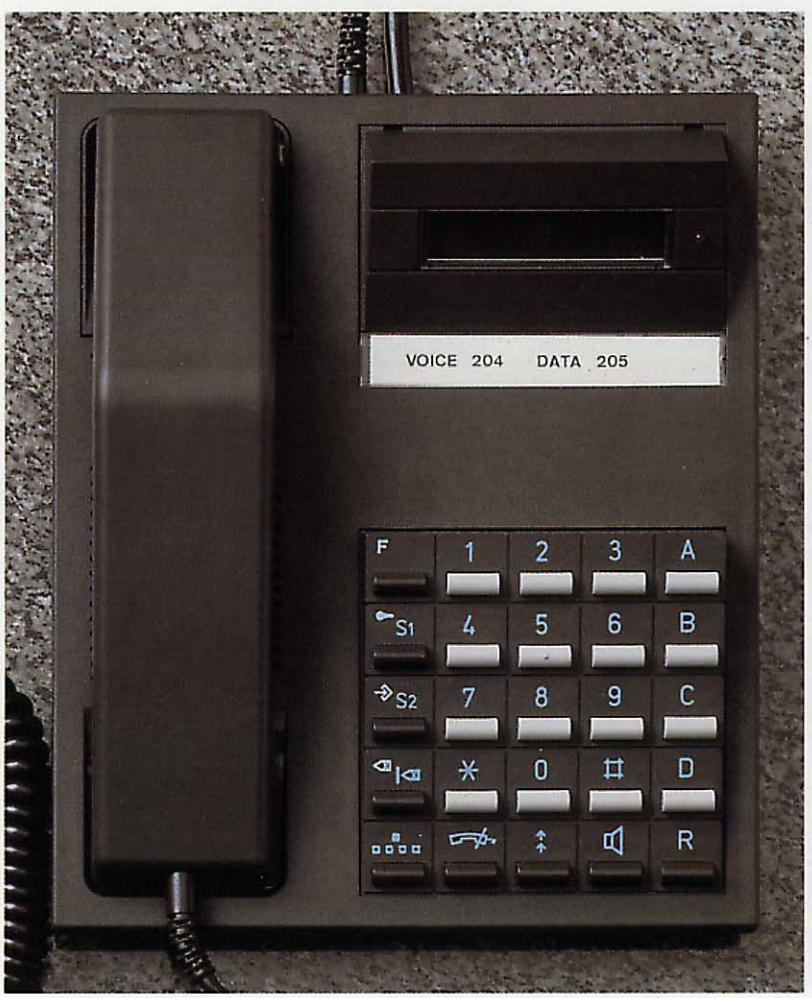
Contribuciones al desarrollo de productos

Desde 1979 el programa de factores humanos del ITTE ESC viene prestando soporte práctico para numerosos productos desarrollados por compañías europeas de ITT. Este programa ha cubierto todos los aspectos de diseño de interfaces de usuario (pantallas, controles, diálogo), ayuda al usuario (guías, tarjetas prontuario, vídeos instructivos, discos de enseñanza) y la implantación de sistemas (diseño de estaciones de trabajo y salas de control). Los ejemplos descritos seguidamente ilustran el alcance de este trabajo.

Aparatos de abonado Venturer

La familia de aparatos telefónicos Venturer* – incluida en la colección de estudio de diseño del Museo de Arte Moderno – juega un papel clave en la gama "Office 2000". El desarrollo de los Venturer ha supuesto una estrecha colaboración entre tres casas ITT: Bell Telephone Manufacturing Company (Bélgica), Standard Eléctrica SA (España) y FACE-Standard (Italia). La serie comprende el aparato básico Venturer 2100, el mejorado Venturer 2200, el multifunción Venturer 2400 y el de datos digitales Venturer 2600.

La aportación de la ingeniería de factores humanos a la familia Venturer tuvo dos objetivos clave. El primero, alcanzar niveles elevados de manejabilidad – especialmente en las variedades de multifunción – y el segundo, asegurar la uniformidad de procedimientos, gráficos, configuración de teclados y guías al usuario, en toda la gama.



Aparato telefónico de datos digitales Venturer 2600.

* Marca registrada del Sistema ITT

Especificación funcional
El aparato telefónico multifunción Venturer 2400 ofrece facilidades tales como un directorio para números telefónicos y los correspondientes nombres, y un diario con fechas y horas asociadas a números y nombres. Pese a su inherente complejidad, tales características deben ser sencillas de aprender y fáciles de usar, por lo cual se preparó una especificación funcional de este aparato que documentara los procedimientos propuestos y así permitiera evaluarlos antes de desarrollar la programación. En general, estas especificaciones constituyen una herramienta de valor incalculable para el diseño de procedimientos fáciles y agradables al usuario, en productos inteligentes.

La figura 7 muestra el procedimiento para encontrar una anotación en el directorio, con una secuencia completa de las actuaciones del usuario y el contenido detallado de la pantalla.

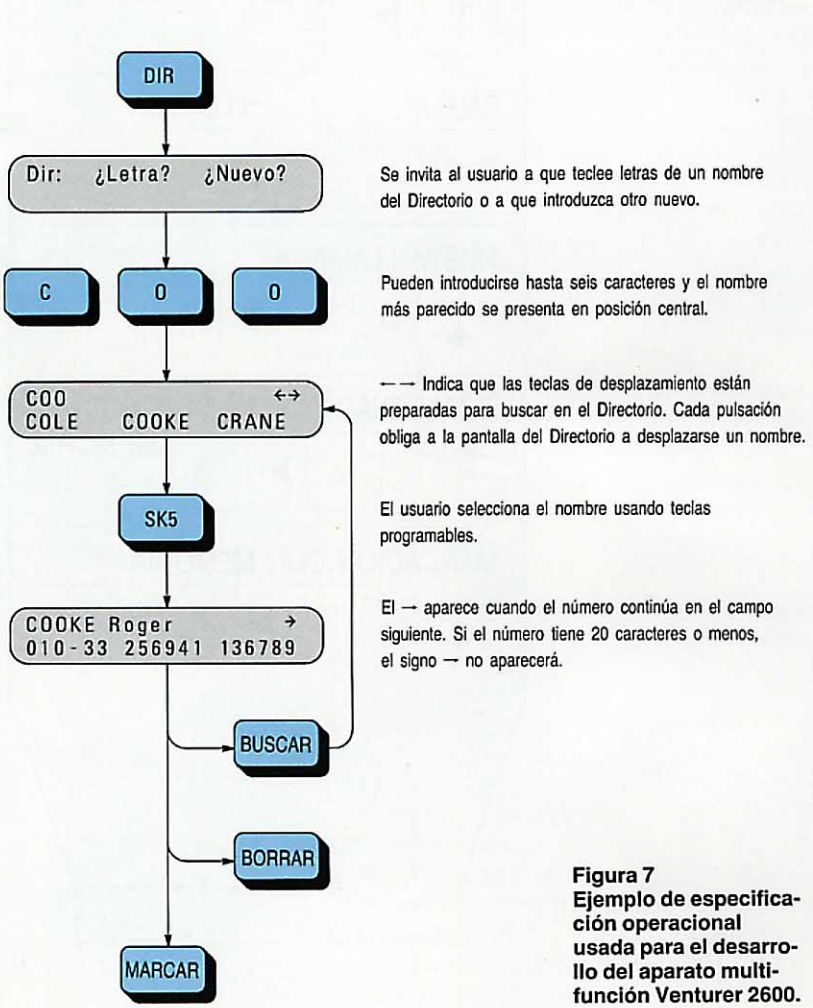


Figura 7
Ejemplo de especificación operacional usada para el desarrollo del aparato multifunción Venturer 2600.

Símbolos
En un producto de ámbito europeo como el Venturer conviene utilizar símbolos para designar las funciones de las teclas. Con este objeto el CCITT ha desarrollado un conjunto de símbolos, que se usan siempre que sea posible. Sin embargo, no existen símbolos del CCITT para funciones como el acceso al directorio o al diario. Por tanto, el grupo de factores humanos del ESC diseñó y evaluó nuevos símbolos para estas funciones, según indica la figura 8. La manejabilidad de los mismos se puede expresar por el porcentaje de usuarios que adivina su significado a la primera, y por el de aquéllos que lo recuerdan en veces sucesivas. Puede así cuantificarse la calidad del diseño de símbolos.

Guía al usuario
Los aparatos de abonado Venturer utilizan los tres tipos de guía indicados en la figura 6.

Los aparatos multifunción y de datos digitales tienen visualizadores de cristal líquido, y por tanto ellos mismos pueden guiar al usuario con mensajes, advertencias de error y avisos dinámicos.

El aparato mejorado ofrece una gama de facilidades que incluyen: marcación "manos libres", pausa interdígitos normalizada, señal silenciadora y una memoria de nueve números. Incluso estas pocas facilidades pueden crear problemas al usuario esporádico o no iniciado, y por ello se le adjunta una guía de usuario "anexa al producto" en forma de tarjeta deslizable bajo el aparato (Fig. 9), que resume los procedimientos principales. Además, hay una tarjeta reemplazable que muestra el procedimiento de marcación con memoria y las nueve números almacenados por el usuario.

Aparte del aparato básico, todos los modelos Venturer requieren también una asistencia "exterior al producto". Se han producido así guías de usuario en el estilo "Office 2000" sobre los principios de diseño siguientes:

- tamaño mínimo (A6), para un manejo cómodo
- uso de símbolos clave en el índice que se corresponden con los grafismos del teclado
- mínimo uso de texto, para facilitar la comprensión y mejorar el aspecto
- empleo de símbolos para resumir la secuencia de operación
- gráficos de alta calidad e impresión a dos colores para destacar diferentes tipos de información.

Figura 8
Nuevos símbolos desarrollados en el ESC para productos de ámbito europeo.

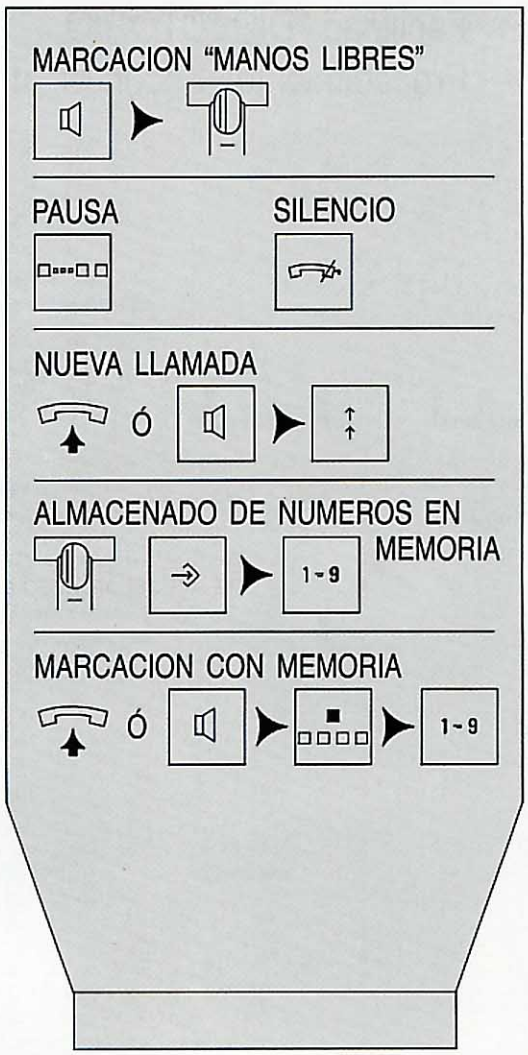
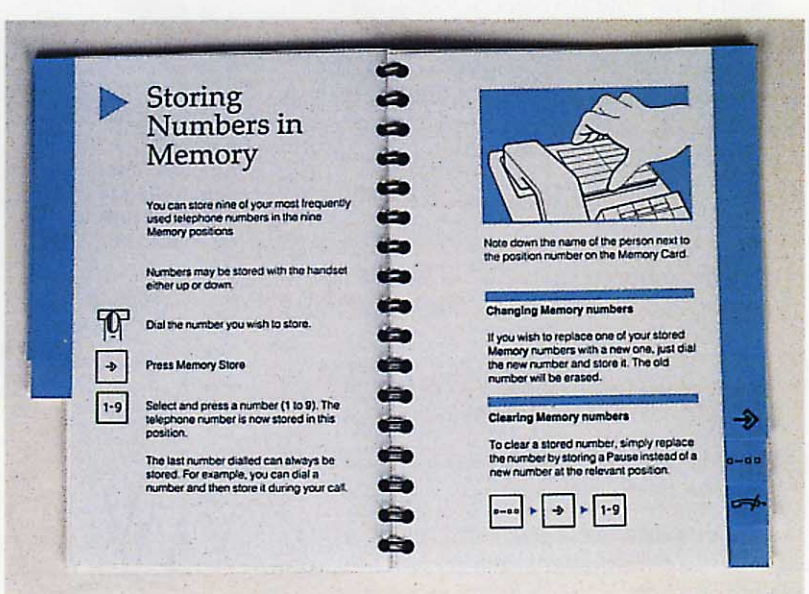


Figura 9
Tarjeta prontuario para el aparato telefónico mejorado Venturer 2200.



Páginas de la guía del usuario del nuevo aparato Venturer 2200 mostrando las instrucciones para almacenar números en la memoria.

Productos de programación

La perspectiva de los factores humanos es aplicable tanto a los equipos como a los paquetes de programación. Aunque los detalles específicos del diseño de programas puedan ser diferentes, es igualmente importante conocer al usuario y a la tarea.

En este campo el ESC ha diseñado un interfaz de usuario para un sistema de pago mediante ordenadores personales, encargado por un importante banco internacional. Una pantalla de muestra contiene:

- una banda donde se indican las opciones presentes, incluyendo "ayuda" y "atrás" (al anterior punto de elección en el diálogo)
- una plantilla bien diseñada, cuidando en especial el formato y el uso del color
- una ventana para la exhibición simultánea de la información de "ayuda" específica del contexto.

Antes de todo intento de especificar la distribución en pantalla y el diálogo, se hicieron una serie de visitas a los usuarios potenciales del nuevo sistema. Se había sugerido utilizar un "ratón", por ser el dispositivo de entrada de datos usual en aquel momento. Sin embargo, las informaciones recogidas sobre los usuarios, la tarea y el medio ambiente, indicaron que el "ratón" sería inadecuado por varias razones, entre ellas la escasez de espacio en la mesa.

Las ideas iniciales sobre diseño del interfaz de usuario se pusieron a prueba a la primera oportunidad, efectuando varias demostraciones comparativas con usuarios. Aunque el concepto básico del diseño probó su validez, hubo descubrimientos inesperados que permitieron introducir mejoras.

Un soporte lógico bien diseñado no sólo es fácil de aprender y utilizar, sino que puede demostrarse con mayor eficacia. Introduciendo la manejabilidad en los programas de aplicación, las técnicas de factores humanos pueden mejorar tanto el rendimiento como la aceptación por el usuario de este tipo de producto.

Implantación de sistemas

Las casas ITT se encargan no sólo de la fabricación de productos, sino también de suministrar sistemas completos, por ejemplo redes empresariales, sistemas de reserva de plazas aéreas y sistemas de control ferroviario. El diseño e implantación de sistemas de esta clase toca todos y cada

Pruebas de usuario para evaluar una maqueta a tamaño natural de una estación de trabajo de un importante sistema de mando y control (servicio de ambulancias de Londres).

uno de los aspectos de los factores humanos, desde el diseño de interfaces de programación al diseño del entorno de trabajo entero. El planteamiento básico de los factores humanos (considerar las características del usuario y los requisitos de las tareas) es aquí igualmente válido. Las pruebas con usuarios son importantes, y a menudo conviene ensayar las propuestas de diseño utilizando una maqueta.

La optimización del entorno e interfaz del operador mejora el comportamiento del sistema y su fiabilidad. Son igualmente importantes los beneficios derivados de la mejora de las condiciones del operador y la reducción de su carga de trabajo.

Conclusiones

El conocimiento de los factores humanos es una importante ayuda en el diseño de interfaces de usuario para todo tipo de equipos y sistemas. Esto sobre todo es válido en los productos multifunción de usuario final, que deben ser sencillos de aprender, eficaces, y fáciles de usar si han de gozar de una amplia aceptación de los usuarios. La ingeniería de factores humanos contribuye al diseño tanto del equipo físico como de la programación, incorporando el concepto de manejabilidad durante el desarrollo.

Además, dicha manejabilidad se puede potenciar mediante guías y manuales bien diseñados. Sea cual fuere el tipo de producto, el enfoque básico y las implicaciones a través del proceso de diseño son las mismas. Las ventajas son también idénticas: un superior rendimiento y mayor aceptación por el usuario.

Fred Brigham nació en Yorkshire, Inglaterra, en 1946. Se graduó en psicología en la Universidad de Liverpool en 1967. y un año más tarde consiguió un MSC en psicología aplicada en la Universidad de Aston en Birmingham. Empezó su carrera en factores humanos con EMI Electronics, y en 1970 fue miembro fundador del HUSAT (grupo de investigación de ciencias humanas y tecnología avanzada) en la Universidad de Loughborough. El Sr. Brigham pasó luego varios años en la TÜV Rheinland de Colonia trabajando en numerosos proyectos, como el diseño de la sala de control de una central de energía nuclear. Entró en el ESC, Harlow, en 1984, donde actualmente es director de factores humanos en la división de productos de usuario final.

Diseño de interfaces usuario-sistema mediante ayuda cognoscitiva

La psicología cognoscitiva ofrece una base para el diseño de interfaces de usuario eficaces, pero su aplicación por los diseñadores no resulta fácil. Una nueva ayuda informatizada permite evaluar la eficiencia de un interfaz usuario-sistema y señalar sus debilidades a fin de que el diseñador pueda corregirlas.

P. F. Byerley
R. G. Leiser
R. F. Saffin
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

Introducción

La clave para aprovechar todo el potencial que encierra la tecnología de la información se encuentra en el interfaz con el usuario. Numerosos dispositivos y paquetes de programas obsoletos dan fe de que las características avanzadas y la inteligencia de los sistemas sirven de poco si se mantienen inaccesibles al usuario.

Para que un interfaz sea bueno, las características de entrada/salida del sistema han de ajustarse con precisión a las del usuario, lo cual optimiza el diálogo entre ambos de tal manera que el usuario se pueda concentrar en sus objetivos finales (p. ej., preparar un documento) en vez de hacerlo en otros más inmediatos, como editar un carácter. Sistemas de este tipo, hechos a la medida, exigen un profundo conocimiento de las características de entrada/salida del usuario, y esto lo aporta la psicología cognoscitiva. En efecto, hay que sobrecargar lo menos posible la memoria y la atención del usuario, entendiendo qué concepto tiene del sistema y qué métodos aplica para completar las tareas.

La carga sobre la memoria se ve afectada por numerosos factores, entre ellos el número de órdenes existentes, la complejidad de la sintaxis de la entrada, y la extensión y accesibilidad de las respuestas del sistema. Por ejemplo, considérese un editor de textos que ofrezca una gran variedad de órdenes, cada una de ellas con muchos parámetros posibles, bien opcionales u obligatorios. Supóngase, además, que las respuestas del sistema muestren únicamente las líneas editadas con éxito o un mensaje de error único si la entrada es errónea. Cuando el usuario se proponga realizar una sola modificación en un fichero textual, deberá ante todo, tras invocar el editor, encontrar la línea que ha de corregirse; ello plantea inmediatamente otro objetivo secundario: dar con la orden que localiza textos. El usuario sabe cómo conseguir ayuda del sistema, pero obtiene una lista-resumen de órdenes que desborda la pantalla, y surge entonces un tercer paso a cubrir: averiguar cómo detener el listado en la pantalla. Una vez logrado esto y consultado el resumen de órdenes, el usuario debe seguir recordando la orden que busca y su sintaxis hasta que finalice el listado de todas ellas y el sistema admita de nuevo instrucciones. Todo esto puede requerir varios intentos si la sintaxis es compleja, o si el nombre de la orden es una sola letra que no guarda relación aparente con la función que representa.

Cuando el usuario recuerda el nombre y la sintaxis de la orden e intenta utilizarlo, el sistema le responde con una interrogación. El usuario no sabe si tal interrogación es una invitación del sistema a proseguir o si se trata de un mensaje de error, así que se le aparecen dos nuevos objetivos: averiguar cómo interpretar la respuesta del sistema y cómo leer la última línea.

A medida que avanza el diálogo, crece el número de etapas a cubrir por el usuario, y ello aumenta la carga de su memoria y le dificulta poder recordar los resúmenes de órdenes. Cualquier equivocación incrementa la tensión, y limita aún más la memoria, con lo que el usuario podría llegar a abandonar o aplazar ese trabajo.

En todo lo anterior, se han tenido muy poco en cuenta las aptitudes y limitaciones cognoscitivas del usuario, y esto sucede con harta frecuencia. La consecuencia es que el usuario ha de gastar un tiempo apreciable en averiguar cómo ejecutar sus

tareas. Por muy avanzadas facilidades que este tipo de editor ofreciera, inevitablemente caería en desuso, pues la mayoría de los usuarios prefiere alcanzar sus objetivos por un camino largo y sencillo mejor que por uno brillante pero complicado.

La psicología cognoscitiva en el diseño de interfaces

Para aprovechar todas las posibilidades que ofrece un sistema es imprescindible un buen interfaz. Sin embargo, diseñar y evaluar interfaces no es sencillo debido a la complejidad de la psicología cognoscitiva.

Una de las dos orientaciones básicas consiste en "encapsular" la psicología cognoscitiva para que la puedan utilizar los no psicólogos mediante la elaboración de normas directrices. Todo un conjunto de descubrimientos empíricos se analiza y condensa en una sola norma que aplica tales hallazgos a algún aspecto del diseño de interfaces. Por ejemplo, muchos estudios sobre lectura rápida, capacidad para recordar textos y búsqueda de ciertas palabras dentro de textos pueden condensarse en una norma única: los textos deben justificarse sólo en el margen izquierdo.

Las normas son muy valiosas como referencia rápida para evaluar interfaces o resolver dilemas durante su diseño. Sin embargo, como ayudas objetivas presentan ciertas desventajas:

- Se contradicen a menudo, hasta el punto de que la aplicación de una norma puede infringir otra.
- Algunas normas sólo sirven para aplicaciones concretas, pudiendo resultar perjudiciales en otros casos.
- La necesidad de claridad y referencia fácil impide indicar la importancia relativa de las normas. Algunas deben seguirse siempre, otras en algunos casos, y otras en fin son simples recomendaciones.
- Por extenso que sea el conjunto de normas, no podrá abarcar todas las decisiones de diseño posibles.

En todos estos casos el diseñador debe confiar en su propia intuición, y además las normas son suficientemente discutibles como para permitir la interferencia de decisiones prácticas o políticas con la evaluación objetiva.

Pese a ofrecer una buena orientación general, las normas no pueden conseguir una evaluación objetiva completa de un interfaz. Incluso un experimentado psicólogo enterado de las fuentes de tal normativa, no podría confiar en aplicar tan amplio cuerpo de conocimiento de una forma sistemática y objetiva.

El segundo método de evaluación de interfaces prescinde de las características de entrada/salida del sistema y se limita a observar cómo interacciona con el usuario, de manera que cualquier deficiencia de comunicación entre sistema y usuario se manifieste en la visión que éste tenga de aquél. Este método "conductista" (basado en el comportamiento) trata de evaluar la comunicación estudiando su eficiencia, y su atractivo intuitivo proviene de la conocida disparidad entre teoría (normativas) y práctica (observaciones fruto del mismo método). No carece, sin embargo, de inconvenientes.

En primer lugar, a la gente le cuesta mucho – y lo hace mal – el explicar el proceso de sus pensamientos. Los primeros estudios acerca de la introspección revelaron que ciertos procesos mentales son tan básicos que el sujeto no es consciente de ellos, y que las experiencias relatadas tienden a reflejar la idea preconcebida que el sujeto tuviera de lo que iba a suceder. La estrategia de "conocer a los usuarios" tiene, pues, poco valor si se basa en el conocimiento que éstos tengan de sí mismos.

Un segundo problema con los métodos conductistas consiste en que incluso los datos no basados en la introspección (tiempo empleado en realizar un trabajo, número de errores cometidos, preferencias del usuario) reflejan sólo el comportamiento de ciertos usuarios en ciertas situaciones. En consecuencia, es vital realizar las pruebas con usuarios que sean representativos de los que vayan finalmente a utilizar el producto.

No debe olvidarse tampoco que el usuario va adquiriendo experiencia, sus hábitos cambian y sus motivaciones varían hasta el punto de que lo observado en pruebas con usuarios "primerizos" puede ya no ser cierto a las tres semanas de utilizar el sistema. Esta situación se produce, por ejemplo, cuando un usuario acostumbrado a manejar ficheros con un lenguaje de órdenes empieza a usar un sistema en el que los ficheros se representan mediante iconos de páginas, los directorios mediante iconos de "carpetas" y el borrado de ficheros se realiza "tirándolos" a una "papelera" con el "ratón". Los anteriores usuarios de un lenguaje de órdenes sienten a menudo que no tienen comunicación directa con el sistema, pero esta insatisfacción sólo dura mientras sigan pensando en el sistema en términos de ficheros y directorios transformados. Una vez que aprendan a tratar el sistema como "mesa de despacho", les resultará mucho más fácil de usar.

En vista de la complejidad del diseño de interfaces y de los defectos de las técnicas actuales, un sistema de diseño asistido por ordenador ayudaría mucho a conseguir interfaces eficientes entre usuarios y sistemas.

Ayuda cognoscitiva al diseño

Se ha contratado con el ITTE Engineering Support Centre la parte principal de un proyecto ESPRIT orientado a desarrollar una ayuda cognoscitiva al diseño (CDA) que utilice técnicas de medida relativas a la compatibilidad de conocimientos hombre-máquina para evaluar la consistencia y los puntos débiles de los diseños de interfaces de usuario. Otros miembros del consorcio son General Electric Company (Hirst Research Centre), Logos Progetti y Medical Research Council Applied Psychology Unit.

La CDA no pretende reemplazar al diseñador o al especialista en factores humanos, sino ofrecer una evaluación objetiva y sistemática de los diseños de interfaces de usuario.

Se obtiene de la CDA una *descripción formal* del interfaz analizado tal que pueda aplicarse a distintos tipos de diseño de interfaces (ventanas, menús, gráficos de alta resolución, etc.), incluso si se ejecuta en máquinas con terminales alfanuméricos. Los módulos CDA aplican sistemas de medida basados en principios pertenecientes a diferentes áreas de la psicología cognoscitiva. El prototipo pretende explorar la factibilidad de tal ayuda al diseño y aportar un marco para posteriores desarrollos.

Programas y equipos de la CDA

La CDA tiene una arquitectura informática modular y utiliza el sistema operativo Unix. Todos sus programas están escritos en "C". Fue consideración fundamental durante todo el desarrollo la portabilidad de los programas, habiéndose ya trasladado éstos a tres máquinas diferentes basadas en Unix.

Tal como se muestra en la figura 1, la CDA consta de tres partes principales:

- Parte A: permite al usuario introducir toda la información precisa para que la CDA desarrolle una descripción formal del interfaz a evaluar.
- Parte B: aplica a la descripción formal del interfaz algoritmos basados en técnicas de medida de psicología cognoscitiva.
- Parte C: da información al usuario en términos de índices numéricos.

El aspecto novedoso de la CDA es la parte B y el tipo de información que recibe de la parte A. La parte B tiene una estructura modular y extensible (Fig. 1).

La técnica usada para la descripción debe satisfacer dos importantes requisitos: primero, proporcionar descripciones precisas y rigurosas que admitan evaluación automática; segundo, ser compatible con la forma humana de razonar sobre interfaces y dar descripciones suficientemente relevantes y completas para someterlas a un análisis de factores humanos.

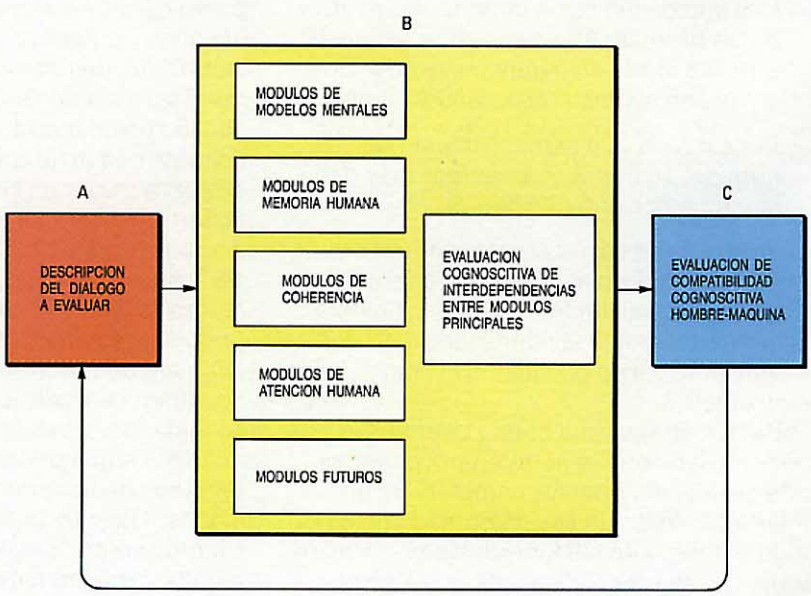
La CDA utiliza diagramas de transiciones de estado para la descripción de interfaces: los estados del interfaz se representan por nodos en el diagrama, y las acciones mediante arcos entre nodos que expresan así una transición de un estado a otro (Fig. 2). Este método permite contemplar el comportamiento global del interfaz. El diagrama se enriquece con definiciones de acciones y entidades de interfaz.

Sólo se describe el interfaz; no se representa ni el usuario ni los procesos del sistema. El enriquecimiento del método de transiciones utilizado para entrada a la CDA consiste en una notación detallada para las salidas del interfaz (p. ej., facilidades para atraer la atención tales como vídeo inverso o destellos), para las entradas del usuario (ej., acciones que obliguen al usuario a volver a solicitar información), y para asignar entradas del usuario a ciertas partes de la pantalla del terminal (ventanas).

A partir de la descripción del interfaz cabe formular diversas preguntas:

- ¿en cuántos niveles se puede dividir el interfaz?

Figura 1
Estructura modular de la ayuda cognoscitiva al diseño.



- ¿tienen siempre las órdenes el mismo efecto, sea cual fuere el contexto?
- ¿cuántas órdenes diferentes hay?
- ¿son los estados del interfaz siempre el resultado de órdenes particulares?
- ¿existen indicadores que llamen la atención del usuario sobre la información necesaria para la ejecución de cada orden?

Por el momento, estas cuestiones constituyen la base de la evaluación por CDA de un interfaz, pero en los futuros desarrollos se introducirán descripciones de tareas y objetivos del usuario para que la evaluación sea más detallada.

La CDA basa su evaluación de factores humanos en dos tipos de descripciones:

- División jerárquica de las facilidades de sistema para dar una descripción aplicable a las propiedades de agregación y abstracción que tiene la organización del conocimiento. La agregación se refiere a la forma en que el usuario agrupa cosas relacionadas entre sí, y la abstracción al proceso por el cual dicho usuario abstrae características de las cosas.
- Diagrama de transiciones de estado, enriquecido con las descripciones de los tipos de acciones del usuario y de estados de la pantalla que tengan valor para el análisis.

Ambos tipos de descripciones se aplican solamente al interfaz.

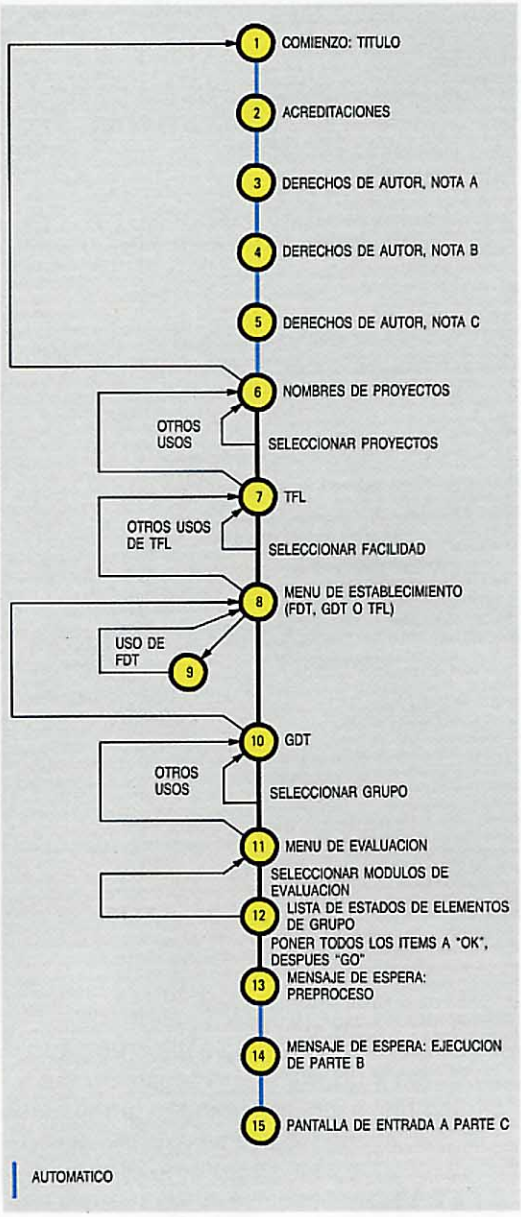


Figura 2
Diagrama de transiciones de estado para las dos primeras partes de una sesión CDA.

Sesión CDA

La estructura de una sesión CDA normal se ilustra en las figuras 2 y 3. El proceso es iterativo e interactivo, permitiendo al diseñador modificar las descripciones del diseño según vaya respondiendo el sistema. Aunque la CDA no proponga cambios específicos, cada módulo de evaluación calcula índices para un aspecto concreto de la compatibilidad cognoscitiva, basándose en la información aportada por el diseñador, y sugiere en qué áreas podría flaquear el diseño, indicando las normas oportunas y posibles fuentes de nuevas informaciones. Tales índices individuales se combinan en un valor global denominado índice de compatibilidad cognoscitiva, que da una medida aproximada de la solidez de las diferentes descripciones de diseños de interfaz. Cada sesión se divide en tres partes que se corresponden con los tres tipos de módulos: entrada de datos, evaluación y resultados.

Entrada de datos

Un proyecto, que es el nivel estructural más alto considerado por la CDA, comprende una o más "facilidades de sistema" tales como proceso de textos, correo electrónico o planificación de proyectos. Mediante el menú de "Nombres de Proyectos" el usuario puede seleccionar, crear o modificar detalles de hasta 15 proyectos distintos.

Seleccionado un proyecto, el usuario señala la facilidad a examinar, y la descompone en una jerarquía de elementos que serán la entrada a la CDA y a cuyo nivel se realiza gran parte del trabajo básico. El usuario aporta entonces los oportunos detalles sobre los elementos a evaluar, y la CDA calcula los índices correspondientes. Como el usuario a menudo se interesará por más de un elemento, la CDA permite

1. Facilidad de sistema (FS):	El usuario de la CDA selecciona una FS a partir de una lista, e identifica los elementos principales que la componen.
2. Selección de un grupo de elementos de FS:	El usuario elige un grupo de elementos de la FS seleccionada. Un grupo puede ser una FS completa.
3. Tablas de datos:	El usuario completa las tablas de datos precisas para el grupo FS elegido. La CDA completa luego varias "matrices".
4. Evaluación CDA:	La CDA evalúa el diálogo para el grupo de elementos de FS elegido en base a los módulos de evaluación seleccionados. Se realiza la evaluación para cada elemento y para el grupo entero.
5. Resultados:	La CDA proporciona al usuario informaciones acerca de la evaluación.
6. Fin del Grupo	
7. Fin de la Sesión	

Figura 3
Prototipo de sesión CDA.

definir uno o más grupos de elementos, produciendo índices para cada grupo además de los individuales de cada elemento. Por ejemplo, si un determinado grupo tiene un índice de coherencia bajo, el usuario puede estudiar sus elementos componentes para encontrar cuáles son los causantes.

Tras seleccionar un grupo de elementos, el usuario puede elegir los módulos de evaluación a ejecutar.

Si bien la CDA proporciona un marco para describir el interfaz de usuario, su eficacia depende del grado de coherencia o normalización alcanzado en el uso de términos. Por tal razón, la CDA incluye una serie de bibliotecas (y listas) de términos que pueden aportar la información necesaria para completar las distintas tablas de datos. Estas bibliotecas son ampliables, creciendo a medida que se utiliza la CDA.

Las bibliotecas detallan asimismo la división del sistema por facilidades y los mapas del sistema almacenados para uso posterior, cual son las definiciones de órdenes genéricas en otros contextos. Ello permite incluir "buenos" mapas para elementos normalizados de tareas (p.ej., "cut and paste", "cortar y pegar") como entradas normalizadas en una biblioteca, a fin de poder utilizarlos en nuevos interfaces de usuario.

Evaluación

Los módulos de evaluación abarcan un amplio espectro de áreas de investigación. La opción elegida fue tomar descripciones concisas de descubrimientos en psicología cognoscitiva e interpretarlos en cuanto a sus consecuencias al interaccionar con un sistema. Ello produjo varias escalas de medida que se convirtieron a módulos informáticos separadamente. Al extenderse la base de la psicología cognoscitiva y descubrirse más interrelaciones entre las diferentes áreas de investigación, tales interrelaciones estarán reflejadas en los enlaces entre módulos.

Actualmente están realizados los siguientes módulos:

- coherencia (1 y 2)
- modelos mentales (1 y 2)
- memoria
- atención.

Las descripciones que siguen sobre el modo de utilizar estos módulos se aclaran por referencia al interfaz usuario-sistema para operación y mantenimiento en las centrales digitales Sistema 12 de ITT. Tal interfaz, diseñado y desarrollado en ESC, da acceso al lenguaje de órdenes del CCITT, evitando que el usuario tenga que recordar y teclear nombres de órdenes (completos o abreviados), nombres de parámetros y argumentos válidos. Los datos se introducen en formularios accesibles mediante una jerarquía de menús.

Módulo de coherencia 1

La coherencia facilita la interpretación del diálogo del interfaz y la memorización de la información ofrecida, contribuyendo así a la adquisición rápida de un "modelo mental" unificado del sistema. Asimismo hace más probable la automatización de secuencias de acciones, aumentando la velocidad de ejecución y favoreciendo un comportamiento más inteligente. Finalmente, la coherencia es vital para organizar la información y aprender órdenes. Las directrices de diseño siempre han exigido "guardar coherencia", aunque se ha dejado al arbitrio de los diseñadores decidir cómo y en qué circunstancias.

Hace tiempo que se espera una especificación precisa de la coherencia, por lo que cualquier avance en este área durante el desarrollo de la CDA ayudará al proceso de diseño. Además, como estimar la coherencia del diálogo puede exigir bastante tiempo y esfuerzo, no es probable que los diseñadores adopten un criterio sistemático para su evaluación. Aquí también puede ser de

gran valor la CDA, ya que proporciona una comparación metódica de los índices de coherencia.

Los módulos de evaluación de la CDA se centran en la coherencia de obtener cierto resultado a partir de una acción determinada del usuario. Por ejemplo, si se pulsa una tecla en particular o se pronuncia cierta palabra,

- ¿se llega siempre al mismo nodo genérico (p. ej., al menú principal)?
- ¿se llega siempre al mismo tipo de nodo (a un menú, aunque no necesariamente el principal)?
- ¿se llega siempre al mismo tipo de resultado (p. ej., a la operación anterior, como con la orden "undo", "deshacer")?.

El prototipo de la CDA trata los dos primeros niveles, pero las versiones posteriores se ocuparán de otros tipos de coherencia.

El interfaz usuario-sistema (USI) del Sistema 12 es coherente en ambos aspectos. Pulsar la tecla del "ratón" cuando el puntero está sobre el primer bloque de la lista de opciones sucesivas de menús, conduce siempre al menú de más alto nivel, y dos pulsaciones consecutivas de dicha tecla llevan siempre a una página de ayuda.

Módulo de coherencia 2

Este módulo se ocupa de la coherencia merced a la cual, por ejemplo, se llega a un menú haciendo siempre las mismas o parecidas cosas, tales como pulsar la tecla de función 1. Se calculan índices más o menos similares a los obtenidos por el módulo de coherencia 1.

El USI del Sistema 12 es muy coherente en este aspecto. Así, la facilidad "desk", que permite al usuario leer o imprimir informes del Sistema 12 y examinar la lista de tareas ejecutadas, se alcanza siempre pulsando la tecla "ratón" con el puntero colocado en la zona de pantalla rotulada "Desk".

Módulo de modelos mentales 1

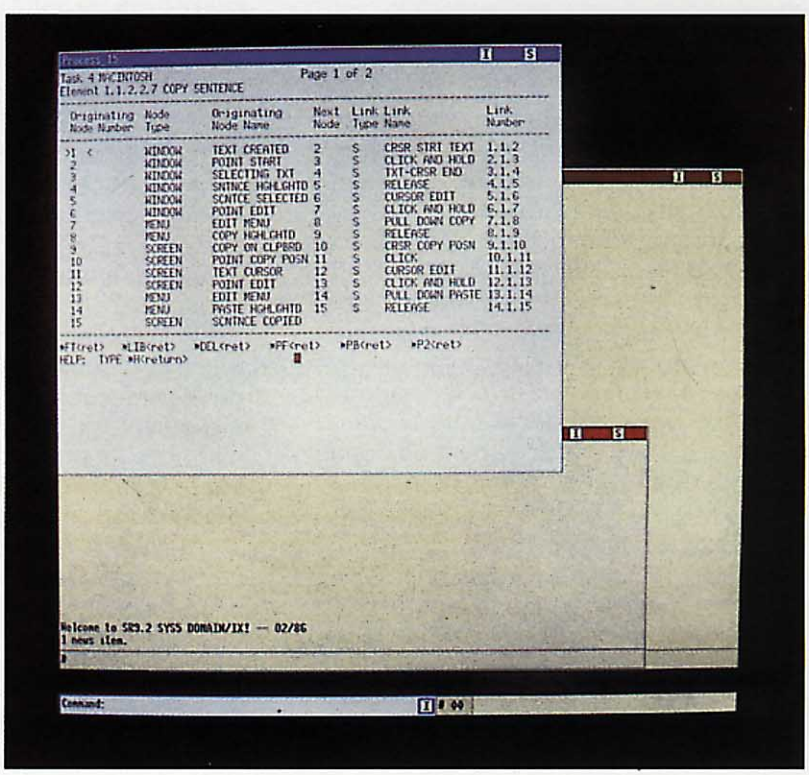
Un principio importante de la psicología cognoscitiva es que el comportamiento no depende tan sólo de la asociación de los estímulos ambientales con las respuestas de la conducta individual, como podría postular una sencilla teoría de estímulo-respuesta. El comportamiento se concibe mejor pensando en la representación interna del mundo que se construye una persona, pues se cree que es tal interpretación (modelo mental) la que inspira su conducta, más bien que una traducción literal y reflexiva de ella.

La operacionalización (puesta en forma ejecutable, operacional, algorítmica) del concepto de "modelo mental" se ha resistido bastante hasta ahora, así que este módulo operacionaliza especulativamente sólo un pequeño aspecto. Lo importante, no obstante, es que cubra una faceta del "modelo mental" aplicable al contexto de la interacción hombre-máquina.

Se afirma que la facilidad con la que un usuario elabora una representación interna útil del interfaz con un producto de oficina, se ve afectada por la organización de las tareas que dicho interfaz puede controlar. Más concretamente, para cualquier tarea de cierta complejidad se pronostica que:

- Es más difícil el desarrollo de un modelo mental cuando la estructuración es *vertical estrecha* (muchos niveles jerárquicos con pocos elementos en cada uno) que cuando es *piramidal*.
- De forma similar, elaborar un modelo mental de estructura *horizontal estrecha* (pocos niveles con número relativamente alto de elementos en cada uno) será también más difícil que una estructura *piramidal*.

Se considera que ambas estructuras, la horizontal y la vertical, ocasionarán dificultades al desarrollar un modelo mental útil, pues implican el aprendizaje de largas listas de elementos poco relacionados entre sí. Sería piramidal una estructura en la que hubiera varios o incluso muchos niveles jerárquicos, de modo que cada elemento cubriese un moderado número de elemen-



Fotografía de pantalla mostrando la evaluación de una función editora mediante la CDA.

tos (tres o cuatro) del nivel inmediato inferior. Los grupos de tres o cuatro elementos constituyen "aglomerados" naturales que ayudarían al usuario a recordar cómo se organiza la tarea en el interfaz.

El módulo en cuestión compara la estructura propuesta con la piramidal, calculando un índice que será tanto más favorable cuanto más piramidal se estime la jerarquización presentada.

El USI del Sistema 12 tiene en este aspecto una buena estructura de menús. Se accede a los formularios mediante una jerarquía de menús de hasta seis niveles, cada uno de los cuales representa con respecto a los inferiores un nuevo agrupamiento lógico de las áreas funcionales del Sistema 12. Como media, hay cinco elementos por menú, es decir, un número de opciones lo bastante reducido como para que la selección sea rápida, a la vez que el número de niveles jerárquicos es suficientemente bajo para acceder con rapidez al formulario preciso.

Módulo de modelos mentales 2

El uso de "órdenes genéricas" se acepta generalmente como un buen principio de diseño. Se trata de órdenes utilizables en muchos contextos distintos para realizar una tarea, cuyo uso evita que el usuario tenga que recordar grandes series de órdenes, así como los contextos en los que éstas pueden utilizarse o no. Dentro del concepto cognoscitivo de "modelos mentales", es válida la hipótesis de que la existencia de órdenes genéricas facilita el desarrollo de una representación interna del interfaz.

El módulo de modelos mentales 2 evalúa la proporción de órdenes genéricas y su número. Los índices calculados son favorables si el usuario puede confiar en unas pocas órdenes genéricas en vez de tener que aprender un gran número de órdenes específicas del contexto.

Las órdenes genéricas del USI del Sistema 12 incluyen pulsar la tecla "Home" para desplazar el cursor al área de atajo ("shortcut", en la cual se pueden teclear los números de sucesivas entradas de menús para alcanzar rápidamente el nivel deseado de la jerarquía), y pulsar la tecla "ratón" que selecciona el elemento de pantalla donde se encuentre el puntero en ese momento.

Módulo de memoria

Un enlace, tal como se define en la CDA, consta de tres elementos: entrada, mediación y acto. El primero se define de modo que admita una entrada desde la memoria del usuario. Por ejemplo, para ejecutar una orden se advierte al usuario de que se está esperando que genere dicha orden, exigiéndole a la vez que recuerde el nombre adecuado. Desde luego cuantos más "enlaces" de este tipo haya, mayor será la carga de la memoria del usuario. Esta crítica es aplicable a todos los diseños de diálogo a base de lenguajes de órdenes, especialmente cuando el interfaz está destinado a usuarios inexpertos.

El módulo de memoria evalúa la proporción de "enlaces" del sistema que requieren aportaciones de la propia memoria del usuario. La evaluación debe considerar varios factores, en particular si el "enlace" forma parte de una orden genérica, en cuyo caso y en cuanto el usuario haya aprendido la información precisa para efectuar las entradas necesarias desde su memoria, puede utilizarse tal información en muchos casos específicos. La carga mental es mayor cuando el usuario ha de aprender información que puede variar para cada una de las órdenes de un numeroso grupo "ad hoc".

El USI ha reducido enormemente el número de "enlaces" de memoria requeridos en operación y mantenimiento del Sistema 12. Los nombres de órdenes se muestran en pantalla para que el usuario pueda seleccionarlos. Los nombres de parámetros se visualizan automáticamente en formularios, de modo que no sea preciso recordarlos. Cuando se requieren nombres de argumentos, aparecen también en pantalla.

Módulo de atención

En muchos casos, la entrada apropiada para un enlace exige que el usuario fije su atención en alguna parte concreta de la pantalla, o en una señal auditiva o de otro tipo. Por ejemplo, el usuario puede borrar del sistema sin darse cuenta una información valiosa si deja de ver algún aviso que aparezca en la pantalla, o bien teclear mayúsculas si no advierte que está encendido el bloqueo de mayúsculas. Podría seleccionarse un menú incorrecto si el usuario no prestara plena atención a los datos pertinentes.

En cualquier punto de un diálogo, hay muchos aspectos de la pantalla que rivalizan en reclamar atención. Este módulo define un conjunto de características normalmente utilizadas en el diseño de interfaces para atraer la atención del usuario, y evalúa la probabilidad de que éste se fije en las partes adecuadas de la pantalla. Las características de este tipo se denominan *indicadores*.

Se ha puesto un gran cuidado al diseñar el USI del Sistema 12 para asegurar que la atención del usuario se dirija a los lugares apropiados de la pantalla. Cuando el proceso incorporado de validación de plantillas encuentra un error en alguna de ellas,

aparece un mensaje de error justamente debajo del área reservada para la plantilla, en vez de usar un nuevo contenido de pantalla que pudiera distraer la atención del usuario. Este podrá, tras haber advertido el error, encontrar fácilmente el campo de entrada en cuestión, que estará en color azul. En rojo se presentan aquellos sucesos que requieran atención inmediata. De todas maneras, el color no se utiliza pródigamente a fin de conservar su eficacia.

El módulo necesitaba un esquema que describiera el nodo a definir, y se ha elegido uno basado en el concepto de nodo como jerarquía de objetos identificables, cada uno de ellos con características que atraen la atención (indicadores).

Indicadores de nodo: señalan que el usuario debe prestar atención, pero no concentrarla en ningún objeto determinado. Por ejemplo, a menudo se utiliza un pitido para dirigir la atención hacia un nodo de error pero sin asociarlo con ningún objeto concreto de la pantalla, sino más bien con el nodo en conjunto. El usuario tiene todavía que explorar el nodo para detectar qué características podrán revelar el fallo. A veces un mensaje en pantalla puede complementar al indicador auditivo, pero incluso en este caso tendría el usuario que localizar el problema.

Indicadores de objeto: indican objetos concretos en los que hay que fijarse. Ejemplos son el parpadeo de un mensaje de error, o el mayor brillo de un icono.

Por el momento el módulo toma en consideración tan sólo un aspecto de atención; en versiones posteriores se tratarán otros.

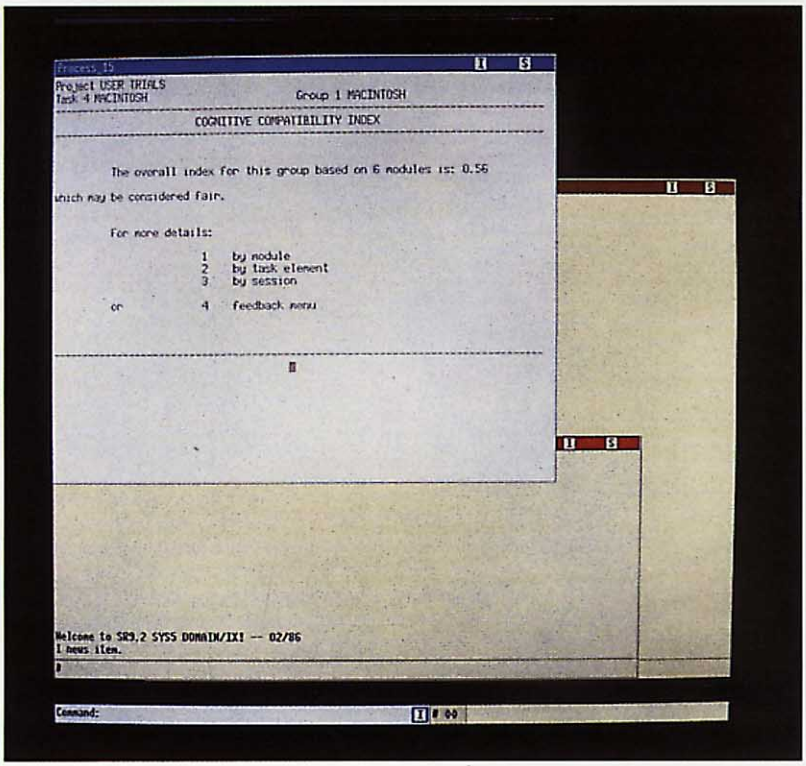
Salida de resultados

El resultado final del uso de los módulos de evaluación del interfaz es un conjunto de índices que se presentan al diseñador agrupados según su nivel específico de detalle. El diseñador tiene la opción de ejecutar sólo uno o varios módulos para una evaluación determinada, pudiendo después retroceder, cambiar elementos de las tablas de descripción y volver a ejecutar la evaluación.

La mayoría de los índices se calculan en dos niveles: para el grupo seleccionado y para elementos individuales del grupo. Cuando la evaluación de un grupo es baja, el usuario puede examinar los índices de elementos individuales a fin de identificar las áreas débiles.

Indice de compatibilidad cognoscitiva
Este índice es un indicador de evaluación global, basado en los restantes índices. Se calcula para cualquier nivel que se solicite (nivel de elemento o de grupo).

La base del cálculo consiste en multiplicar cada uno de los otros índices calculados en la sesión por un factor de ponderación variable, y hallar a continuación la media. Estos factores de ponderación se determinan empíricamente y pueden ser ajustados a la luz de las pruebas de validación a que se está sometiendo actualmente la CDA.



Salida de la CDA, indicando el índice de compatibilidad cognoscitiva.

Conclusiones

La CDA "encapsula" por primera vez psicología cognoscitiva en un formato que puedan manejar los diseñadores de interfaces. Es, pues, un primer paso hacia un instrumento informático de diseño de factores humanos predictivo y automatizado. Mucho queda, no obstante, por hacer antes de que llegue a ser un producto enteramente utilizable.

El prototipo ofrece un entorno estructurado para la investigación de varias áreas de la psicología y del propio proceso de diseño. Se ha llevado a cabo algún trabajo de validación cuyos resultados inciden sobre el desarrollo de la programación, con el fin de refinar las bases psicológica e interpretativa de los módulos de evaluación cognoscitiva.

En mejoras futuras se desarrollarán nuevos módulos que evalúen otros aspectos del interfaz usuario-sistema. Otros desarro-

llos posteriores podrían también potenciar de manera provechosa el proceso de introducción de datos en las tablas descriptivas (p. ej., mediante un diálogo de preguntas y respuestas), y posiblemente unir la CDA con herramientas rápidas de modelado en estaciones de trabajo de alta resolución con pantalla de mapa de bits, lo cual permitirá crear y evaluar de forma concurrente prototipos ejecutables de interfaces.

Paul F. Byerley tiene los grados BSc y PhD en psicología, y MSc en informática. Su tesis de PhD se refirió a la influencia de factores motivadores en el proceso humano de la información, y su tesis de MSc al análisis de programación en lenguaje declarativo dentro del área de sistemas de gestión de bases de datos. Actualmente trabaja en el ITTE ESC, en Harlow, en el área de sistemas tutores inteligentes y en la gestión del proyecto 234 de ESPRIT de ayuda cognoscitiva al diseño.

Robert G. Leiser nació en Londres en 1958. Se graduó MA en psicología por la Universidad de Glasgow (1982) y, después de invertir tres años en su tesis de postgraduación sobre la resolución de la ambigüedad en el lenguaje natural, ingresó en el Human Factors Technology Centre de ITT, donde ahora trabaja en diseño de interfaces ordenador-usuario. En este momento es responsable de los aspectos de factores humanos del USI Sistema 12 y del trabajo de desarrollo de diálogos de entrada/salida en forma verbal.

Raymond F. Saffin nació en Sunninghill, Ascot (Reino Unido) en 1947. Estudió física teórica en la Universidad de Lancaster, en la que se graduó BA en 1968 y PhD en 1972. Es Chartered Physicist y miembro del Institute of Physics. En 1973, el Dr. Saffin ingresó en el Technical Publicacions Centre de ITT Europe (que luego pasó a integrarse en ITT ESC) como escritor técnico, trabajando en la documentación de centrales telefónicas. En 1979 alcanzó la categoría de analista de sistemas. Actualmente es jefe del grupo de soporte a la programación en el Human Factors Technology Centre del ESC, y dirige dentro de ITT el proyecto 234 de ESPRIT.

Sistema vídeo de análisis asistido por ordenador

Las cintas de vídeo son un medio ideal para registrar el comportamiento de las personas ante las máquinas, tanto en laboratorios como en ambientes normales de trabajo. El sistema CAVAS da un medio fácil, rápido y fiable para interpretar las grabaciones de vídeo resultantes.

M. de Alberdi
ITT Europe Engineering Support Centre,
Harlow, Inglaterra

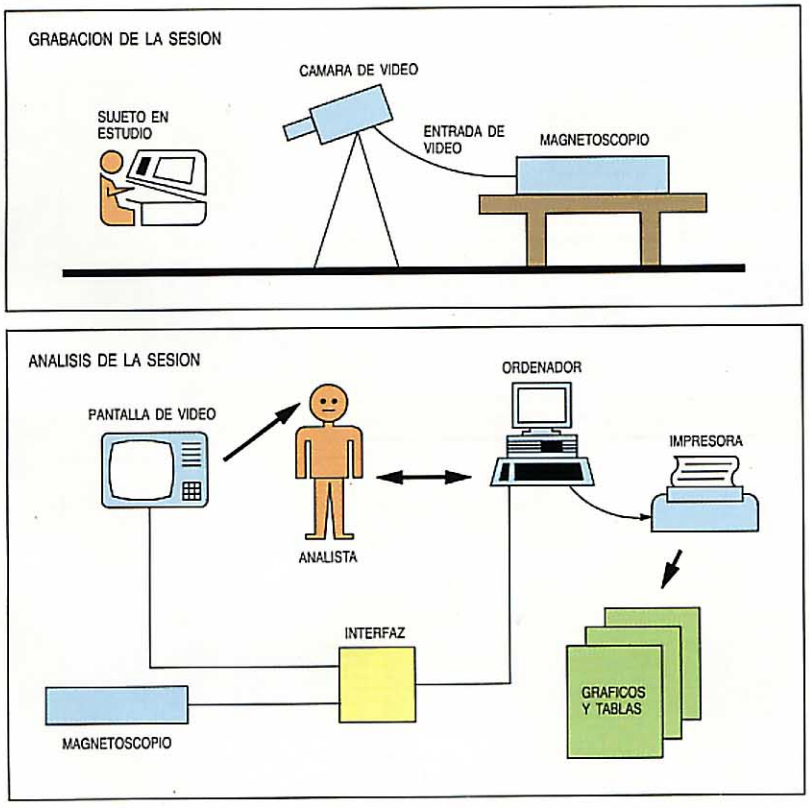
Introducción

Una actividad esencial de los psicólogos especializados en factores humanos es la observación metódica y el análisis de los sistemas hombre-máquina, ya sea ante un prototipo de posición de trabajo en ordenador para oficina o en un complejo centro de control de ambulancias en horas punta. El magnetoscopio acoplado a una videocámara es muy adecuado para esta tarea, ya que funciona con autonomía durante largos periodos y reproduce al instante las grabaciones. Utilizando a la vez varias cámaras de vídeo, se obtienen diferentes perspectivas que pueden ayudar mucho al análisis del comportamiento de operadores y sistemas en centros de control. Por otra parte, las observaciones grabadas molestan menos que las directas y permiten medir con calma cualquier número de parámetros.

Para obtener resultados estadísticos, los psicólogos deben observar una serie de eventos (el parpadeo de los ojos, por ejemplo) y registrar exactamente cuándo ocurren y en qué relación con otros eventos. En la construcción de los diálogos es muy útil la posibilidad de marcar todos los casos de fallo para corregirlos en la siguiente oportunidad. Este proceso suele ser lento, tedioso y muy sujeto a inexactitudes. El sistema CAVAS (sistema vídeo de análisis asistido por ordenador) aporta una solución que no exige al usuario más que identificar los eventos, pues su etiquetado, marcación de tiempos y presentación de resultados estadísticos en forma gráfica o tabular se realizan automáticamente (Fig. 1).

El sistema CAVAS es una de las herramientas elaboradas por el Centro de Tecnología de Factores Humanos para diseño y evaluación de sistemas de uso agradable.

Figura 1
Funcionamiento básico del sistema CAVAS.



Descripción del sistema

En la figura 2 se muestra la configuración física del sistema. Sus componentes principales son el magnetoscopio con pantalla y el ordenador personal. Al canal de entrada de audio se conecta un generador de códigos de tiempo con objeto de marcar cada trama de vídeo con uno de tales códigos, sea durante la propia grabación o posteriormente, lo cual permite clasificar los eventos en periodos de 40 ms. En la reproducción, el lector de códigos de tiempo extrae estos códigos, y los pasa por un interfaz que los adapta al puerto paralelo de usuario del ordenador. El ordenador dispone de un conjunto de programas CAVAS que ayudan al usuario a identificar los eventos en cuanto aparecen en la pantalla, marcando su tiempo y duración, y que posteriormente procesan los datos correspondientes para

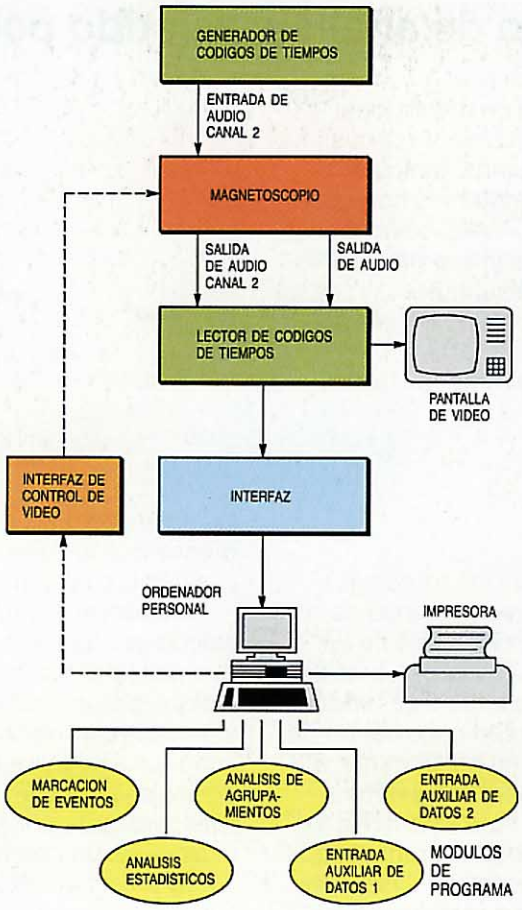


Figura 2
Diagrama de bloques del sistema CAVAS.

formar estadísticas. La presentación gráfica o textual de los resultados se obtiene en una impresora de matriz de puntos.

El soporte lógico del CAVAS consta de cinco programas independientes. El primero proporciona la entrada de tiempos y el almacenamiento de descripciones de eventos predefinidas, y permite que el usuario registre, mediante teclado, la aparición en la pantalla de eventos específicos. Los dos programas siguientes analizan luego los datos recogidos por el primer programa, mientras que los dos programas finales permiten al usuario introducir descripciones más flexibles de los eventos, a medida que vayan apareciendo en la cinta de vídeo.

Marcación de eventos

Durante su funcionamiento el sistema CAVAS recibe dos clases de señales de entrada: la primera es una sucesión de códigos de tiempo procedentes del magnetoscopio, y la segunda procede del usuario, quien señaliza (mediante el teclado del ordenador) los instantes en los que ocurren eventos notables. La marcación de eventos se ejecuta sobre la base de actividades predefinidas, en las que el usuario, al obser-

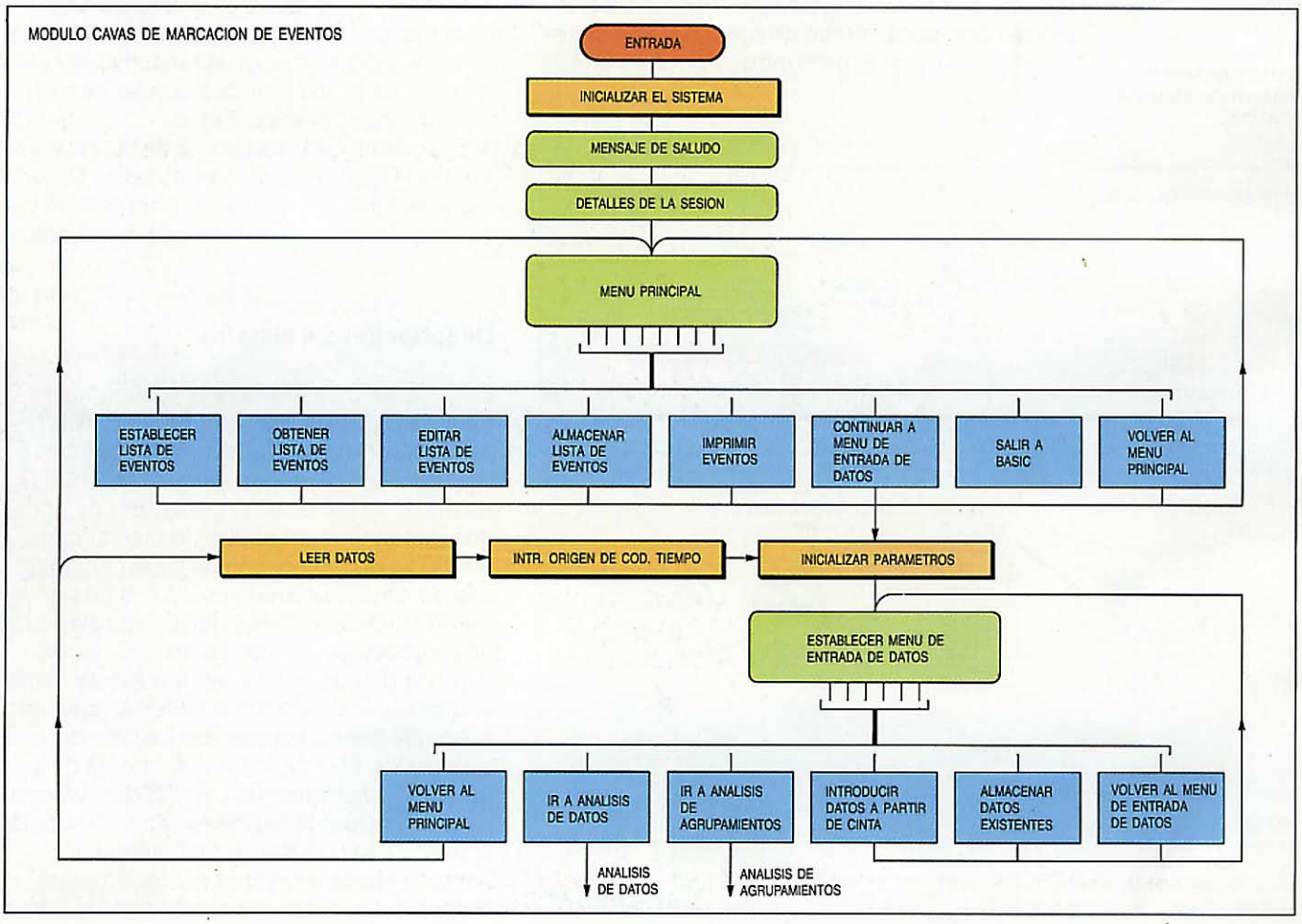


Figura 3
Diagrama de flujo para la marcación de eventos.

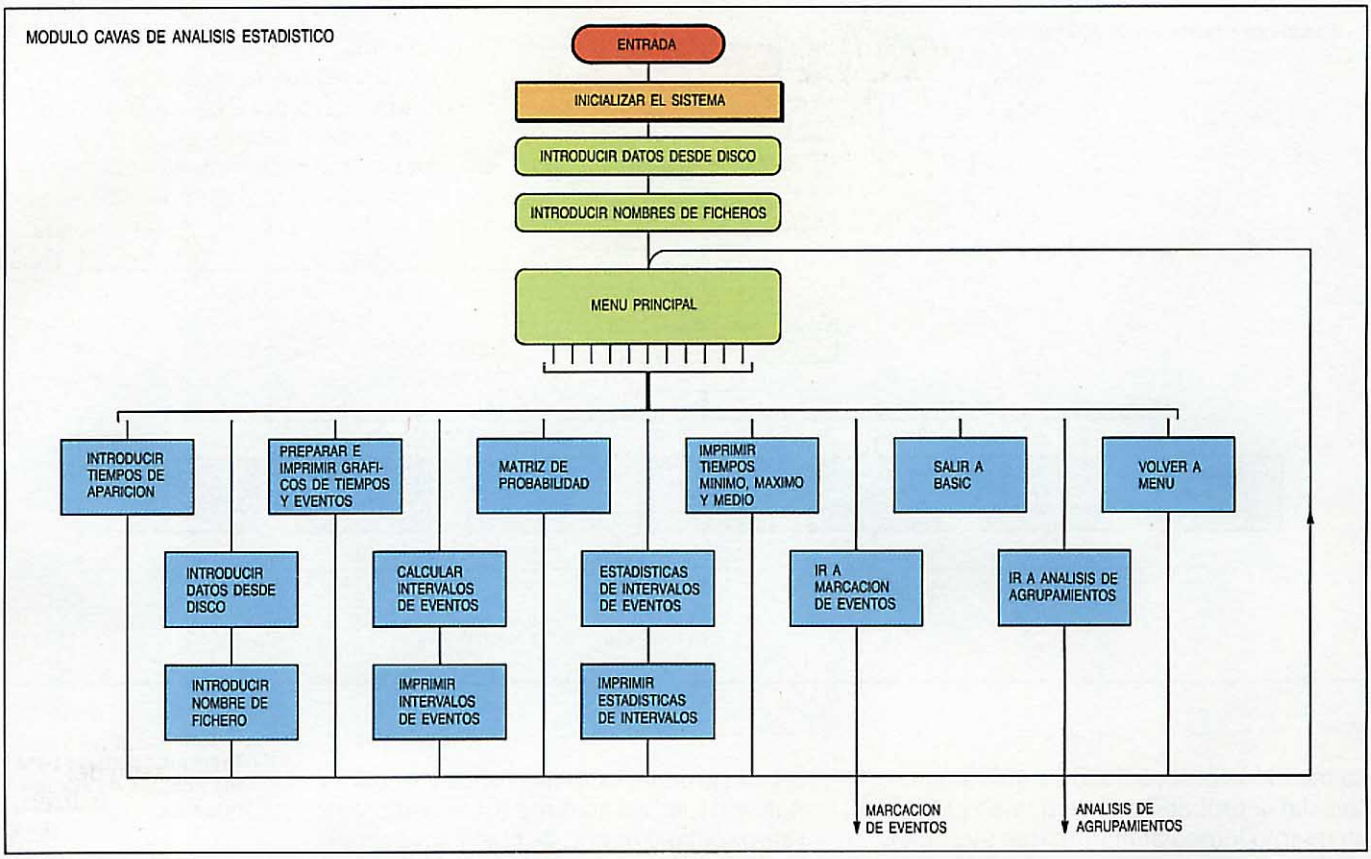


Figura 4
Diagrama de flujo para análisis estadísticos.

var la cinta de vídeo, busca aspectos específicos de comportamiento. El usuario debe establecer y almacenar en disco un conjunto inicial de descripciones de eventos, cada una de las cuales se identifica mediante una sola pulsación de tecla. Pueden señalarse dos tipos de eventos: instantáneos, representables por un único carácter, o bien de una determinada duración, en los que se utilizan distintos caracteres para marcar el instante de comienzo y el de final.

En la figura 3 se muestra un diagrama de flujos operacional para la marcación de eventos. Al empezar, el sistema se inicializa automáticamente, y presenta al usuario un mensaje de saludo y la petición de detalles concretos sobre la sesión. El menú principal ofrece opciones para establecer, almacenar, buscar y editar las listas de eventos. Una vez completadas las listas, el usuario puede avanzar al menú de entrada de datos para marcar los eventos mientras contempla la grabación de vídeo. Tras la entrada de datos, el menú presenta opciones para continuar con otros módulos de programas.

Análisis estadístico

Los datos recogidos en el módulo de marcación de eventos se utilizan en el de análisis estadístico para calcular los intervalos, efectuar análisis estadísticos y crear un grafo de tiempos y una matriz de probabilidades (Fig. 4). El programa relaciona los registros de eventos con los de tiempo a través de sus identificadores comunes, e imprime las descripciones completas de los eventos con indicación de los instantes en que aparecen. Pueden seleccionarse los siguientes resultados estadísticos:

- lista de intervalos de eventos
- número total de intervalos de eventos
- tiempo total consumido por un evento determinado
- duración máxima de un evento
- duración mínima de un evento
- márgenes de duración de los eventos
- intervalo medio de eventos
- tiempo medio gastado en un evento especificado
- varianza de los intervalos de eventos
- desviación estándar de los intervalos de eventos
- moda estadística de los intervalos
- margen de los semi-cuartiles
- histograma de la distribución de intervalos de eventos en el margen de los tiempos disponibles.

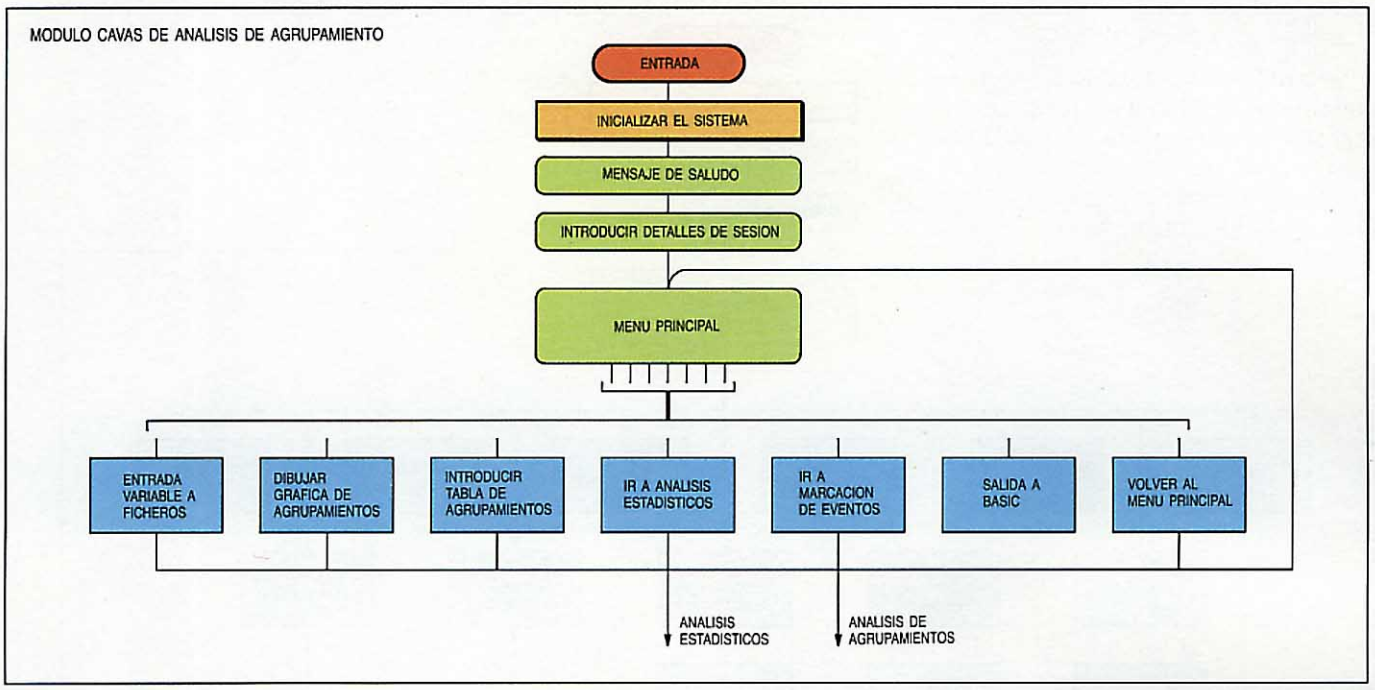


Figura 5
Diagrama de flujo para los análisis de agrupamientos.

La matriz de probabilidad se utiliza para calcular la probabilidad de que se presente un evento inmediatamente tras otro. En esta matriz se contienen las identidades de eventos impresas por filas y columnas, lo que permite hallar la probabilidad de que a un evento de una columna siga otro de los enumerados en una fila.

Análisis de agrupamientos

El tercer módulo CAVAS es un programa de análisis automático de agrupamientos, el cual permite extraer del disco los registros tomados por varios observadores independientes y comparar sus resultados (Fig. 5). El ordenador reconoce cualquier coincidencia apreciable en la aparición de los eventos e imprime una gráfica con el grado de agrupamiento.

El usuario debe introducir dos variables relacionadas directamente con los cálculos de agrupamientos: el tiempo máximo, que impone límite a cualquier agrupamiento, y el número mínimo de observadores necesario para lograr un consenso significativo.

Los agrupamientos se obtienen comparando los ficheros de cada observador, que contienen valores de tiempo escogidos en orden ascendente. El primer valor de tiempo del primer fichero se compara con todos los demás valores de todos los ficheros. Dicho tiempo inicial se toma como límite inferior del agrupamiento; el límite superior depende del valor que haya introducido el usuario para indicar el margen de tiempo máximo de ese agrupamiento particular. Una vez tratados todos los valores de tiempo, el agrupamiento queda identificado si el número total de ficheros (observadores) representado dentro de cualquier consenso de eventos es igual o mayor que el número requerido por el usuario. Se toma después como punto de comparación el segundo valor de tiempo del primer fichero, considerando únicamente los tiempos mayores que éste como parte del siguiente agrupamiento. El proceso continúa hasta que todos los valores de tiempo de todos los ficheros se hayan tomado como límite de tiempo inferior de los posibles agrupamientos.

A continuación se calculan los tiempos medios por agrupamiento y se clasifican en orden ascendente. Los resultados se imprimen por último en un gráfico (asteriscos para los tiempos de los eventos, y líneas para el tiempo medio de cada agrupamiento). Suele darse información complementaria con mayor grado de detalle en forma de tabla impresa.

Entrada de datos auxiliares

Durante la observación de las cintas de vídeo, pueden presentarse eventos interesantes que no estén definidos previamente en el módulo de entrada de datos. El primer módulo auxiliar de entrada de datos permite hacer descripciones completas de eventos mediante el teclado del ordenador al tiempo que se observa la cinta de vídeo y reciben códigos de tiempo. Al aparecer uno de estos eventos en la pantalla, el usuario puede teclear una sentencia que lo describa, creando así una base de datos de descripciones de eventos específicos, posible-

mente útil para la valoración general del vídeo. El uso de este módulo se facilita conectando un interfaz de control de vídeo que permita controlar las funciones de reproducción mediante el teclado del ordenador.

El segundo módulo auxiliar de entrada de datos permite también la preparación de descripciones mientras se pasa la cinta de vídeo. Sin embargo, en este caso las descripciones se construyen partiendo de una serie de listas de palabras, agrupadas por categorías gramaticales (listas de verbos, adjetivos, nombres, preposiciones y adverbios). Como estas listas pueden variar notablemente según el tema de la grabación, el programa permite también que el propio usuario defina, almacene y extraiga listas de palabras específicas. Aunque este método dé una descripción de eventos más limitada que con el primer módulo, permite desarrollar los ficheros de datos a partir de los modelos teórico y conceptual del tema en estudio. Las palabras pueden tratarse y agruparse con mayor facilidad, y es menos probable que haya confusiones al comparar las impresiones de distintos observadores del sistema CAVAS.

Aplicaciones

El sistema CAVAS se desarrolló específicamente para ayudar a evaluar las necesidades de los usuarios y facilitar su interacción con los sistemas en el diseño de los interfaces hombre-máquina. El examen de estas interacciones es una tarea multidimensional que debe correlacionar el binomio estado/comportamiento del usuario con el del sistema. En el mundo real, a menudo hay que ver esta interacción en un contexto más amplio; por ejemplo, la actuación de un operador de consola dentro de un sistema mayor, cual es un centro de control de ambulancias. En estos casos es imprescindible poder reproducir varias veces una misma escena, como permiten las grabaciones de vídeo. Sin el CAVAS, costaría enormemente obtener y analizar datos.

El sistema CAVAS es de un especial valor en muchas áreas de psicología aplicada, como la clínica y la educacional, así como en los factores humanos. También es aplicable a muchas otras materias técnicas: análisis del tráfico de vehículos, operación de máquinas, estudio del trabajo, etcétera.

El Grupo de Factores Humanos ha utilizado el sistema CAVAS en las siguientes áreas:

- pruebas en el puesto de trabajo del ordenador personal y periféricos
- análisis de tareas en centros de control
- comparación del comportamiento de los usuarios en condiciones diferentes, tales como entornos convencionales y electrónicos
- diseño de modelos de diálogos y pruebas de usuarios para productos telefónicos.

Los resultados han demostrado que el sistema CAVAS ofrece evaluaciones objetivas en entornos naturales, revelando en algunos casos tendencias subjetivas en los métodos convencionales de valoración.

Conclusiones

El sistema CAVAS está demostrando ser una herramienta eficaz en el ámbito de los factores humanos, pues agiliza las sesiones de valoración de las cintas de vídeo y permite obtener informaciones estadísticas directamente de los datos de entrada. El CAVAS es un sistema abierto que sin duda experimentará sucesivas mejoras para ampliar sus campos de utilización, pero que no obstante trabaja ya en la práctica con eficacia comprobada, tanto en la cuantificación de ciertas categorías predefinidas de comportamiento como en la definición consensuada de otros tipos de comportamiento.

Marco de Alberdi nació en Londres en 1950. Estudió psicología en el Politécnico de Plymouth, donde se graduó BA en 1978. Ingresó en el Grupo de Psicología del ITT Engineering Support Centre de Harlow en 1982, haciéndose cargo de la aplicación de los estudios sobre factores humanos a las centrales digitales Sistema 12. En 1985 el Sr. de Alberdi fue nombrado director del Centro de Tecnología de Factores Humanos.

Análisis de planificación, modelado y fijación de precios

Una planificación eficaz de la comercialización y producción de centrales se apoya en la inmediata disponibilidad de una gran cantidad de datos. El sistema SCAMP centraliza todos los datos relativos a una central, desde su pedido hasta su entrega, y permite controlar la gestión de todas las etapas.

F. Giambalvo
D. Vignazia
Industrie FACE Standard, Milán, Italia

Introducción

En 1984 FACE comenzó a introducir en el mercado las centrales telefónicas Sistema 12 de ITT. El departamento comercial, ante la complejidad que suponía el controlar los numerosos aspectos relacionados con la producción y venta de estos sistemas tan versátiles y avanzados, comenzó a buscar herramientas apropiadas de gestión, basadas en ordenador. Concretamente, el SCAMP se diseñó para ayudar en el modelado y planificación de la producción, así como en la determinación de los precios de las centrales. El sistema debía además cumplir otros requisitos, como por ejemplo el poder acceder a información disponible en centros de FACE distantes más de 1.000 km.

En el momento de iniciar el desarrollo, la producción de programas del Sistema 12 ocupaba prácticamente toda la capacidad del ordenador principal de FACE, por lo que era esencial asegurar que esta herramienta no entorpeciera el sistema informático. Así, el objetivo fue elaborar un plan a medio plazo para automatizar las operaciones del ciclo de producción, con la garantía de que el nuevo procedimiento no consumiera demasiado tiempo de ordenador. Además, había también otros condicionantes sobre el desarrollo, tanto económicos como de plazos.

Un estudio preliminar confirmó dos puntos que resultaron básicos en el desarrollo del SCAMP: por un lado, el sistema tenía que residir en el ordenador principal para facilitar la comunicación con otras unidades; por otro lado, sin embargo, ubicar todas las facilidades en dicho ordenador exigiría una excesiva capacidad de proceso. Este segundo punto era crucial, y sólo cabía una solución: ejecutar en el ordenador principal una parte del sistema y distribuir el resto de las funciones en equipos más pequeños, como ordenadores personales.

También era conveniente desarrollar un procedimiento que se pudiese introducir por etapas, ya que algunos departamentos necesitaban de forma inmediata los programas de ayuda basados en ordenador, mientras que otros no estaban preparados todavía para utilizar este tipo de procesos. En cualquier caso, lo más importante era evitar la aparición de cuellos de botella durante el periodo de utilización simultánea de procedimientos tradicionales y automatizados.

Una de las primeras actividades era la de registrar y consultar todos los datos relativos a las centrales y al tipo de tareas a realizar en cada una de ellas (por ejemplo, si se trataba de una nueva central o de una ampliación). Los programas destinados a este tipo de tareas requieren técnicas eficientes de acceso a archivos, así como un sistema avanzado de entrada/salida capaz de tener en cuenta todas las necesidades de seguridad y recuperación de datos.

Utilización del sistema SCAMP en FACE, Milán.

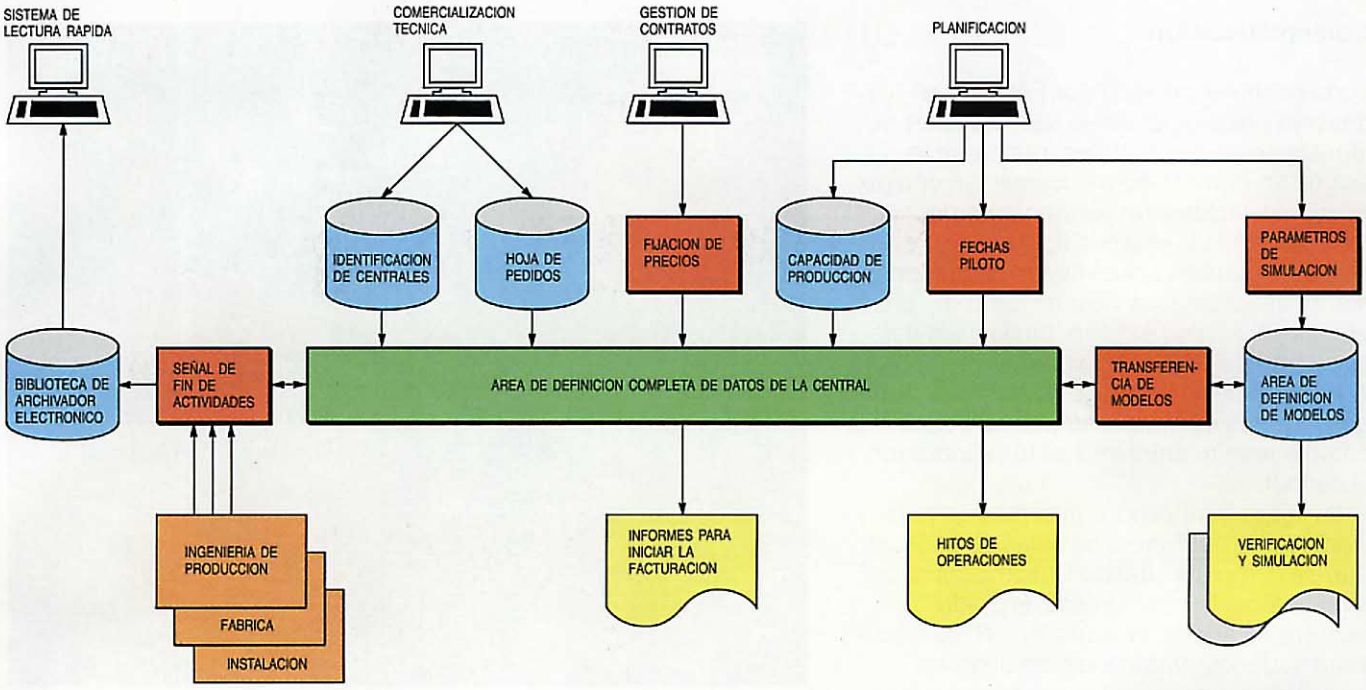


Figura 1
Arquitectura del sistema SCAMP para el análisis de cronogramas, modelado y fijación de precios.

Era evidente que los sistemas de gestión de bases de datos podrían simplificar estos programas, aunque en realidad había pocas bases de datos disponibles. Sin embargo, en aquellos momentos se instaló en FACE el lenguaje FOCUS, de cuarta generación, que incluía un sistema de gestión para control de bases de datos y que prometía alcanzar una productividad diez veces mayor que la del COBOL. Aunque algunos ingenieros no estaban convencidos de que con el FOCUS se pudieran efectuar las mismas funciones que con los lenguajes tradicionales, se tomó la decisión de probarlo, dada la gran mejora de productividad que se esperaba. En una primera fase, el FOCUS se utilizó en el desarrollo de un sencillo subsistema de archivo de una central, con el fin de comprobar la flexibilidad del lenguaje.

FOCUS

Es un lenguaje de la cuarta generación que comprende un sistema de gestión de bases de datos y un interpretador. Aunque los interpretadores son de uso más exigente que los lenguajes compilados, son más rápidos de programar y de probar.

Otro factor en favor del FOCUS fue su sencillez para presentar información en pantallas como las de los ordenadores personales. Esto era importante, ya que los procedimientos de aplicación se pueden describir mediante una serie de pantallas (o paneles) que facilitan a los analistas de sistema el acceso a funciones variadas.

Organización del SCAMP

El SCAMP se desarrolló partiendo de los siguientes supuestos básicos:

- la Administración pide todos los años un cierto número de nuevas instalaciones y ampliaciones de centrales
- se han de evaluar, sincronizar y planificar las actividades de cada departamento implicado en estas operaciones
- para evitar pérdidas de tiempo, la información ha de transferirse con rapidez
- es preciso simular la carga de trabajo esperada sobre los departamentos de producción, para poder evitar paradas y sobrecargas
- deben evaluarse con precisión los ingresos, o sea, las cantidades de dinero esperadas en determinadas fechas
- una vez aceptadas las centrales, hacen falta herramientas que produzcan la información de facturación necesaria.

Ciertamente, el SCAMP ha de tener en cuenta todas las áreas de producción y de gestión, empezando por la comercialización.

La figura 1 presenta un panorama completo del sistema SCAMP. Su fichero central se denomina *datos completos de la central* (o *definición completa de los datos de la central*), recogiendo y combinando información procedente de tres bases de datos menores: *identificación de centrales*, *hoja de pedidos* y *capacidad de producción*.

Comercialización

Toda central producida por FACE está descrita y catalogada en la base de datos de identificación de centrales, donde se le asigna un nombre convencional, un código de identificación y un elemento de definición geográfica (véase la figura 2). En esta etapa las centrales nuevas no están plenamente descritas, puesto que son sólo "proyectos", y aún no existen. La pantalla deja sitio también para otros parámetros, como el número de abonados o de enlaces, aunque algunos de estos campos no podrán completarse mientras no se haya instalado la central.

La figura 3 muestra la información más significativa del fichero de hojas de pedidos, que se completa cuando la Administración pide una central con un determinado número de líneas, enlaces, etc. Si es necesario, los programas que gestionan las hojas de pedidos pueden crear automáticamente nuevas entradas para el fichero de identificación de centrales. En la práctica, este fichero contiene un registro de todas las centrales de FACE, mientras que el de hojas de pedidos indica todas las fases de producción para las nuevas centrales o ampliaciones.

En esta fase, el SCAMP prepara también una proyección económica con fechas, para que la dirección tenga una idea de los ingresos esperados.

Figura 2
Información de identificación de la central obtenida de la base de datos.

Figura 3
Información aportada por la base de datos sobre las hojas de pedido.

Planificación

El departamento de planificación es responsable del mantenimiento de la base de datos de producción — que indica la capacidad de la fábrica —, actualizando los valores que dependen del número de módulos de equipo o de la cantidad de personas necesarias para la producción de programas (o sea, para popular la base de datos de la central telefónica), así como para la instalación de los equipos y programas. Normalmente se mide la capacidad de producción por el número máximo de placas de circuito impreso que se pueden fabricar con los medios y las personas disponibles.

La capacidad de producción se presta bien a la simulación, ya que ofrece numerosas posibilidades: las personas o máquinas que se necesitarían para producir *n* placas de circuito impreso, el efecto de que los equipos de programación o instalación tuviesen *x* personas. Sin embargo, en cuanto los datos reflejen el estado real de la fábrica o de otro departamento (o sea, cuando se haya alcanzado un entorno típico), las simulaciones pasarán a ser "predicciones".

La segunda aportación básica del departamento de planificación son las fechas principales de cada proyecto. Cuando se recibe un pedido, sólo se conocen tres fechas clave o fechas piloto: cuándo estará preparado el local de la central, cuándo podrán enviarse los equipos a la central y cuándo ha de terminarse la instalación. Con estas tres fechas, el SCAMP aplica algoritmos sencillos para calcular todos los hitos intermedios de las diferentes

actividades de producción, lo cual asegura un pleno control de la gestión.

La figura 4 muestra un ejemplo de la actividad de planificación. Cada fecha se expresa por un número de tres cifras; las fechas piloto se indican en la parte superior, mientras que la columna COM (computado) da las fechas calculadas para cada actividad.

Modelado y planificación

Todos los datos de entrada, junto con los analizados, constituyen el fichero de datos completos de la central. Esta información se puede transferir a la base de datos del área de definición de modelos, de donde la pueden extraer los planificadores para hacer simulaciones, probando con distintas fechas piloto y capacidades de producción.

Es normal que los planificadores repitan estas simulaciones varias veces, por ejemplo, para obtener resultados óptimos con una capacidad de producción constante, o para hallar la forma de ajustar la mano de obra a determinada fecha de entrega.

La simulación puede también servir para calcular los ingresos, susceptibles de clasificarse de varias maneras o en periodos diferentes.

La figura 5 muestra una pantalla con el menú de gestión del proyecto, que da acceso a diversas opciones de listados y gráficos comerciales, y puede ocupar varias pantallas. Actualmente, ciertas partes del SCAMP se escriben en lenguajes de programación tradicionales, pero se emplea el FOCUS para la obtención de informes y gráficos de empresa. Este menú con los diferentes tipos de informe se pone constantemente al día, y de modo especial para los que afectan a la planificación.

Cuando los planificadores están satisfechos con el modelo, todos los datos se transfieren de nuevo al fichero de datos completos de la central y la planificación se considera "oficial".

Modificación de los planes

Sería una limitación que tan sólo se tuviera un control anual de las actividades, puesto que no se dispondría de guías ni de predicciones en el caso de que las cosas no fuesen como se esperaba. Por lo tanto, cada tres meses los directores de los departamentos involucrados se reúnen para examinar el estado de las centrales que no se han terminado. En caso necesario, se toma la decisión de corregir las fechas motrices y revisar el modelo, y, una vez confirmada esta decisión, se edita un nuevo modelo estratégico que sustituye al anterior en el fichero de datos completos de la central.

En el caso de las centrales aún no completadas, y sólo para ellas, pueden recuperarse fácilmente los datos reales dentro del modelo para efectuar simulaciones.

Informes generales de salida

El SCAMP ofrece una amplia variedad de informes. La figura 1 incluye *hitos de operaciones*, que de manera continua proporcionan listados con formato variado para adecuarse a las necesidades de los diferentes

Figura 4
Pantalla de control de fechas de la planificación ofrecida por la base de datos completos de la central.

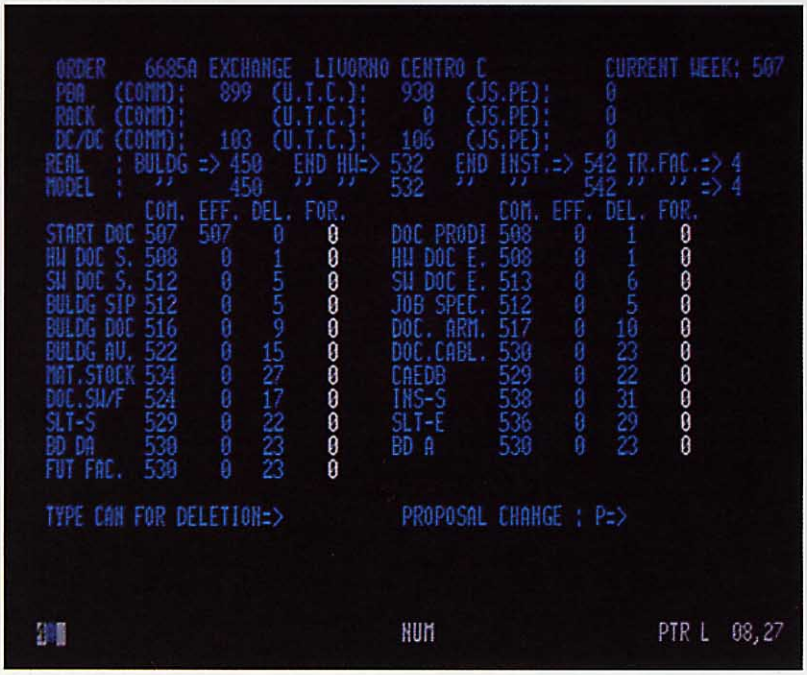


Figura 5
Información dirigida por menú con algunos de los resultados que proporciona el sistema SCAMP.

departamentos. La información de planificación es lo más concisa posible, ya que el trabajo consiste en controlar la actividad completa. Por esta razón, tales informaciones se presentan como gráficos de barras y tablas estadísticas.

Producción e instalación

Los tres departamentos principales que cooperan en la producción de una central son el de ingeniería del producto, el de fabricación y el de instalación. El SCAMP se ha basado en los hábitos de trabajo actuales, excepto cuando era esencial cambiarlos. Así, por ejemplo, las actividades de producción están caracterizadas por series de documentos producidos por cada departamento, algunos de los cuales se reducen a una simple tarjeta de trabajo con unas pocas firmas mientras que otros son documentos complejos, como planos de edificios o dibujos técnicos.

La figura 4 enumera todas las fechas en las que deben completarse las actividades (COM), y junto a ellas se indican las respectivas fechas reales de terminación (EFF). Un cero en esta columna indica que la actividad todavía no se ha terminado.

El SCAMP tiene dos maneras de suministrar estas fechas de fin de actividad: directamente en la pantalla de control de fechas (Fig. 4), o archivando el documento que concluye la actividad en una biblioteca especial, denominada *archivador electrónico*. Como a la pantalla de control de fechas solamente tiene acceso la oficina de planificación, en los departamentos donde todavía no se ha introducido el SCAMP deben ser los planificadores quienes observen las actividades e inserten la señal de final. En cambio, cuando los departamentos de producción utilizan el archivador electrónico para sus documentos de trabajo, la señal de *fin de actividad* se inserta automáticamente al archivar el documento de conclusión.

La tercera columna de la pantalla de control de fechas indica el "factor retraso" (DEL), o sea, el tiempo entre la fecha de terminación según el plan y la actual. Las cifras negativas indican que la actividad no se completó a tiempo, mientras que los valores positivos indican las semanas que faltan para la terminación planificada.

Final del ciclo

El departamento comercial puede comprobar en cualquier momento el progreso de las centrales que están en fabricación a través de una serie de informes en los que se incluye la señal de fin de actividad. Esta señal se inserta al entregar una central; a partir de ese momento, el SCAMP considera "estables" los datos de la central y ya no se pueden utilizar para efectuar simulaciones. Además, esta información no puede transferirse al *área de definición de modelos*.

Al llegar a esta fase, toda la información necesaria para la facturación habrá sido ya registrada en el fichero de datos completos de la central, y por lo tanto podrá servir para obtener una salida automática adecuada a los procedimientos contables tradicionales de la Compañía. Al mismo tiempo, los valores finales del número de líneas, enlaces, etc., se graban automáticamente en el registro de identificación de centrales para señalar el nacimiento de una central nueva.

Centralización y distribución

El prototipo completo del SCAMP se desarrolló en el ordenador principal, utilizando al máximo las operaciones en lotes para minimizar la cantidad de memoria requerida por el lenguaje FOCUS. En la actualidad el fichero de datos completos de centrales, así como todos los archivadores electrónicos, residen en el ordenador principal. Estos últimos archivadores se crean en sistemas automáticos independientes del SCAMP, y se transmiten a él mediante una red que trabaja por lotes.

La definición de los modelos se ejecuta íntegramente en el ordenador central, pero está previsto transferir los procesos de simulación y modelado a una estación de trabajo. No hay razón para que estas actividades se realicen en el ordenador central, ya que sólo existe un departamento de planificación y, con las ventajas del proceso local, es posible tener mayor libertad para ejecutar las simulaciones que con el modo interactivo de ahora.

Actualmente se estudian varios métodos para distribuir ciertos datos esenciales entre los departamentos que los necesiten. También se está examinando la posibilidad de usar hojas de cálculo electrónicas de tipo comercial y programas de gráficos para empresas, aunque ello obligaría a incluir algunos controles para evitar que la actualización de datos dañase la lógica del fichero de datos completos de la central.

En resumen, el enfoque ha consistido en utilizar el ordenador principal para ficheros centrales y bases de datos, y distribuir la información mediante procesos residentes en equipos remotos. Por ejemplo, la consulta de aquellos documentos relativos a la producción de centrales que interesen a diferentes departamentos de la Compañía, supondría mover el fichero con todos sus

documentos, duplicándose la información. Sin embargo, el archivador electrónico con su facilidad de lectura rápida permite que cualquier departamento haga consultas al fichero de un modo más o menos inmediato, sin tener que mover ni duplicar el mismo. Los documentos del referido archivador están conectados a un subsistema de lectura rápida que ayuda a efectuar las consultas.

Se ha desarrollado una conexión a través de un ordenador personal ITT XTRA* equipado con una placa para gráficos, que posibilita la circulación del archivador electrónico con la información actualizada; los usuarios pueden así tener los datos que necesiten impresos en su mismo lugar de trabajo.

Conclusiones

El desarrollo del SCAMP ha hecho posible evaluar, sincronizar y planificar las actividades de todos los departamentos involucrados en la producción y comercialización de centrales telefónicas. Es evidente que el flujo de información se ha acelerado, debido al cuidado puesto en minimizar los retardos en cada uno de los componentes del sistema SCAMP, incluida la lógica de la base de datos, así como en identificar aquellas actividades que introducían duplicaciones innecesarias.

Se ha prestado especial atención al desarrollo de herramientas que permitan a los departamentos de producción trazar cronogramas y ejecutar simulaciones.

El departamento de contabilidad dispone con facilidad de todos los datos financieros relevantes, tanto de valores reales como previstos, por lo que hay información de facturación preparada en cuanto la central se haya entregado a la Administración.

Se han obtenido resultados excelentes con el lenguaje FOCUS de cuarta generación. Después de dos años de desarrollo, el SCAMP es ya plenamente operativo. Sin embargo, se están estudiando varias mejoras entre las que se incluye el uso de estaciones de trabajo para ejecutar simulaciones. Aunque el archivador electrónico ya se está utilizando, todavía no está enteramente automatizado y parece que se tardará un año más en perfeccionarlo en un ordenador personal.

De todas formas, el SCAMP no es un sistema estático, sino que evolucionará a compás de las centrales Sistema 12, así como de las técnicas de producción y la propia organización de la compañía. Los usuarios actuales del SCAMP están pidiendo mejoras que les proporcionen herramientas para afrontar los nuevos desafíos tecnológicos en los campos de redes, bases de datos distribuidas y tratamiento estadístico.

Franco Giambalvo nació en Turín, Italia, en 1944. Posee un diploma del Instituto Avogadro de Turín, y se capacitó en programación de ordenadores trabajando en Olivetti, Sperry y otras compañías. Tiene experiencia fundamentalmente en los campos de desarrollo CAD/CAM y sistemas de bases de datos. El Sr. Giambalvo entró en 1982 en FACE, donde coordina actualmente el grupo de herramientas de programación.

Danilo Vignazia nació en Premosello, Italia, en 1945. Trabaja en FACE desde 1969, y participó en las pruebas de campo del sistema de señalización CCITT nº 6. Ha trabajado en el desarrollo de centrales METACONTA* y ha sido jefe de programación para el sistema de centrales ITT 1220. Desde 1982 es jefe de programación en FACE, con responsabilidad del soporte y desarrollo de herramientas para VAX e IBM.

* Marca registrada del Sistema ITT

Ayudas de ordenador para el diseño mecánico en la industria electrónica

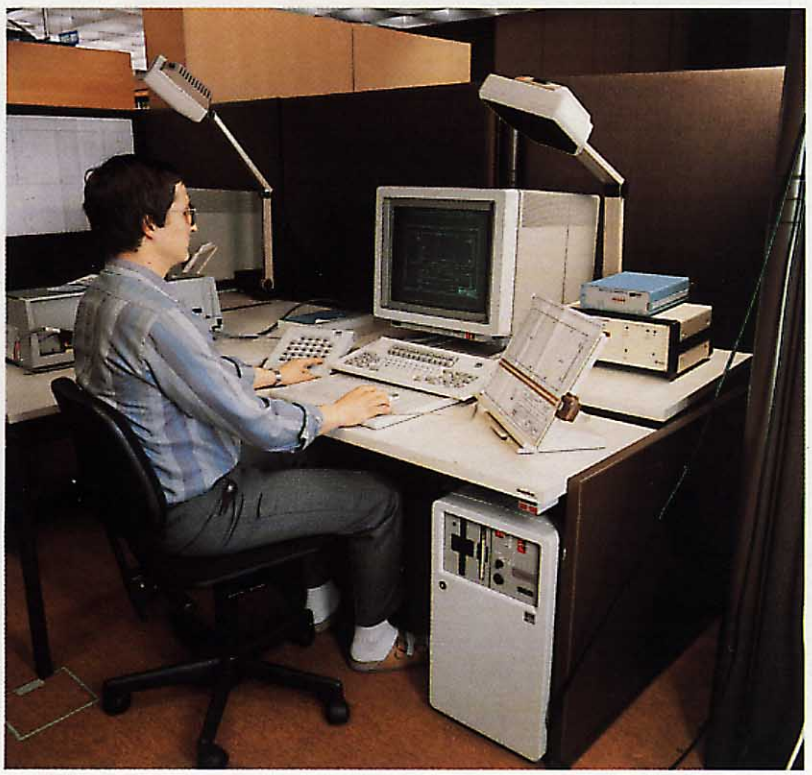
El diseño mecánico es parte esencial del desarrollo de la mayoría de los sistemas electrónicos. Como sucede con otras tecnologías, ha pasado a depender críticamente de la introducción de ayudas por ordenador.

W. Drtil
G. Klause
Standard Elektrik Lorenz AG, Stuttgart,
República Federal de Alemania

Introducción

El diseño mecánico en la industria electrónica implica mucho más que el mero empaquetamiento de circuitos electrónicos en una caja o bastidor, y es cada vez más difícil. En efecto, la complejidad de las tareas de diseño mecánico en la industria está creciendo de un 10 a un 15% al año debido a la necesidad de aumentar las densidades de empaquetamiento, mejorar la calidad de los productos y diversificar más los productos a fabricar en un periodo de tiempo más corto. En consecuencia, los sistemas CAD (diseño asistido por ordenador) son tan necesarios para el diseño mecánico como lo son para el diseño y trazado de placas impresas.

Estación de trabajo de concepción ergonómica para el diseño mecánico.



Decidida ya la introducción de CAD en el área de diseño mecánico, hay que afrontar una elección entre alrededor de 200 sistemas. Para asegurarse de elegir correctamente en una aplicación particular, es esencial realizar un análisis completo del área de diseño y posteriormente redactar una lista detallada de exigencias para evaluación del sistema CAD. Esta especificación de requisitos del usuario debe incluir:

- facilidades básicas para delineación, dibujo detallado y acotado
- medios para modelación tridimensional (modelo alámbrico)
- análisis térmico para ayudar a elegir un sistema óptimo de refrigeración en el diseño de equipos tales como bastidores, unidades de alimentación o equipo militar muy compacto
- capacidad de admitir un número creciente de variantes de productos y de evaluar conceptos nuevos, cumpliendo las tareas en un tiempo más corto
- representación tridimensional, usando detección de conflictos con ayuda de ordenador para poder alcanzar densidades de empaquetamiento elevadas (en el futuro).

Aunque los sistemas disponibles ofrecen un amplio repertorio de facilidades, la elección del sistema resulta difícil por el rápido avance de la tecnología. Generalmente la elección es un compromiso entre las exigencias del usuario y el coste. Se ha elegido un sistema tal que cumple los requisitos inmediatos y constituye una base sólida para aspectos futuros; este sistema se ha introducido paso a paso, empezando con tareas relativamente fáciles en un área de prueba, y extendiéndose a una aplicación

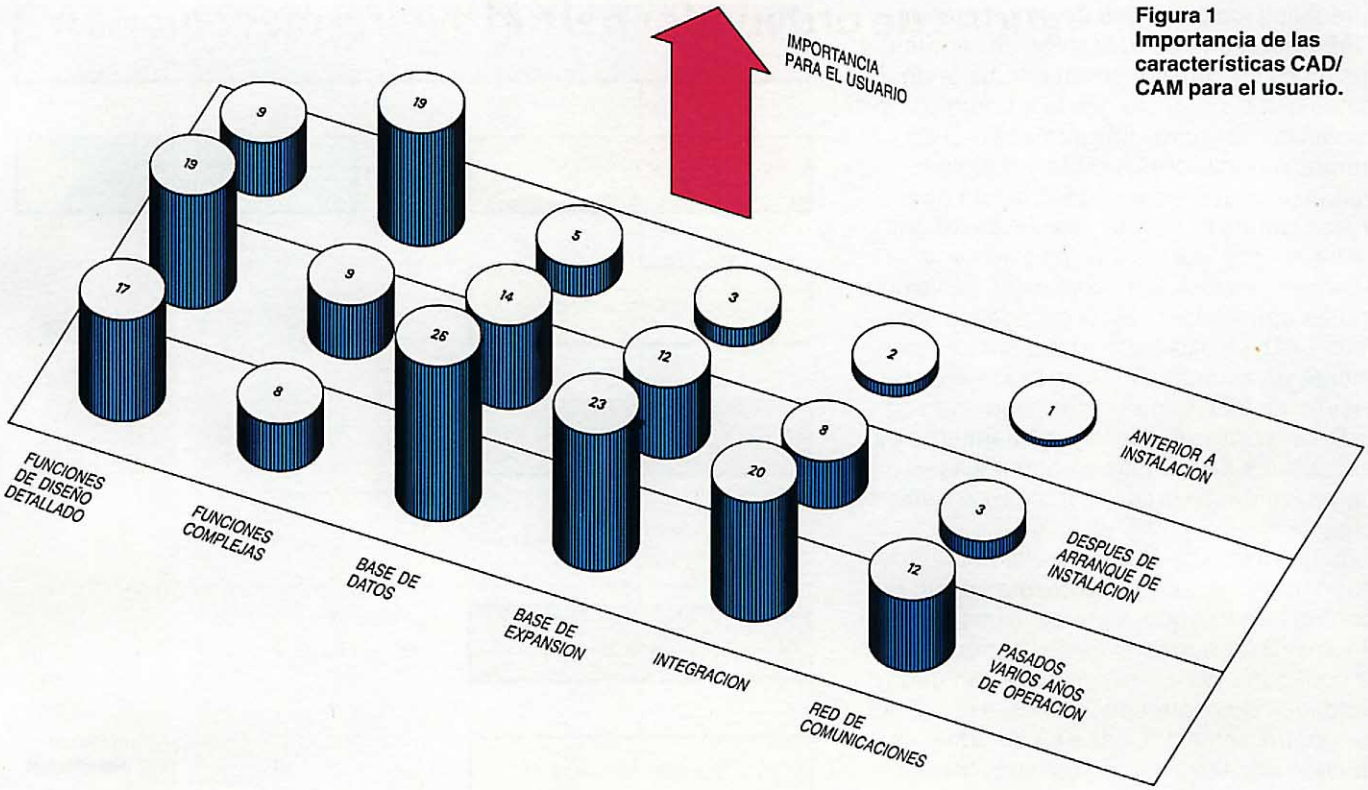


Figura 1
Importancia de las características CAD/CAM para el usuario.

amplia en el diseño de la práctica de equipos.

La figura 1 muestra cómo ven los usuarios las características del sistema CAD durante las diferentes etapas de introducción:

- antes de la instalación del sistema
- inmediatamente después de la instalación
- transcurridos varios años de uso.

Puede observarse que la capacidad de un sistema para integrarse en el entorno existente es un factor clave en el aumento de productividad de los procesos de desarrollo y fabricación.

Configuración del sistema

Los objetivos de desarrollo de SEL necesitaban un sistema CAD/CAM integrado, elegido entre los productos disponibles en el mercado. Idealmente la elección debería limitarse a un sistema único. Además, debería evitarse siempre que fuera posible el desarrollo local de programación. Dos criterios, en particular, fueron determinantes de la decisión final:

- capacidad para integrar sistemas CAD por medio de estaciones de trabajo locales y remotas, evitando así islas de CAD ineficaces
- almacenamiento normalizado de ficheros de datos de diseño coherentes en una base de datos integrada.

El sistema facilita el intercambio de datos entre compañías europeas de ITT, incluyendo Bell Telephone Manufacturing Company en Amberes, Standard Eléctrica en Madrid y Standard Telephon og Kabelfabrik en Oslo, todas las cuales tienen terminales CAD y utilizan sus instalaciones para el diseño de la práctica de equipos del Sistema 12 y de aparatos de abonado.

Los requisitos locales afectaban naturalmente a la configuración del sistema. La mayoría de los departamentos de diseño mecánico de SEL tienen ubicación centralizada, pero algunos están en oficinas que distan hasta 5 km de la planta principal. Consecuentemente había que examinar dos posibles configuraciones. La primera consistía en instalar en la localización remota un ordenador especializado que albergara toda la programación CAD y el equipo periférico (terminales, unidades de control, trazadores), y se comunicara con el ordenador central por una línea de datos de baja velocidad. La otra alternativa era conectar los terminales, unidades de control y trazadores a una unidad de control remota, que pudiera mantener comunicación con el centro de ordenadores a través de una línea de datos de alta velocidad (ej., 2 Mbit/s).

Se escogió la segunda opción por razones de coste: esta configuración permite

que todos los paquetes de programación CAE (ingeniería de aplicación de clientes) y los datos se almacenen en una base de datos única, mientras que la primera opción requería instalar los programas CAD en ambos ordenadores remoto y principal. Había de preverse la posibilidad de que instalaciones futuras tendieran al uso de ordenadores locales más pequeños, y asimismo en dichas instalaciones había que contar con la rápida evolución de los sistemas CAD, en particular la necesidad, obviamente en aumento, que los ingenieros de diseño de SEL tienen de un modelado en 3-D, así como del análisis y tratamiento de estructuras de superficies complejas. La figura 2 muestra la configuración de sistema elegida.

El programa CAD se ejecuta en un ordenador principal conectado a una unidad de control de canal que, a su vez, alimenta una línea de 2 Mbaud a través de un módem. En la configuración actual, se conectan diez unidades de control de terminal a la unidad de control remota. Cada una de aquéllas atiende a un terminal con teclado alfanumérico, un teclado separado de funciones programables, una tableta de digitalización y un cursor de cuatro botones. Un trazador de pluma local se conecta a una de las unidades de control de terminal.

Actualmente sólo hay un terminal de color, conectado a la unidad de control de canal remota a través de una unidad de control de terminal. Se le utiliza para soporte y mantenimiento del sistema, adiestramiento y estudios de diseño mecánico.

Esta configuración de sistema la utiliza SEL principalmente para el diseño mecánico de la práctica de equipo del Sistema 12. Puesto que existe una colaboración muy estrecha entre SEL, BTM y SESA, se ha instalado una línea para conectar las bases de datos del sistema en las tres compañías, permitiendo un directo intercambio de datos.

Hoy día están en uso los siguientes módulos de programación:

- diseño interactivo (2 1/2-D)
- diseño interactivo (modelado alámbrico en 3-D)
- interfaz geométrico (subrutinas macro, lotes, entrada/salida de datos geométricos de diseño)
- diseño paramétrico
- gestor de visualización de datos gráficos
- facilidad de consulta de datos gráficos
- especificación inicial de intercambio de gráficos (IGES).

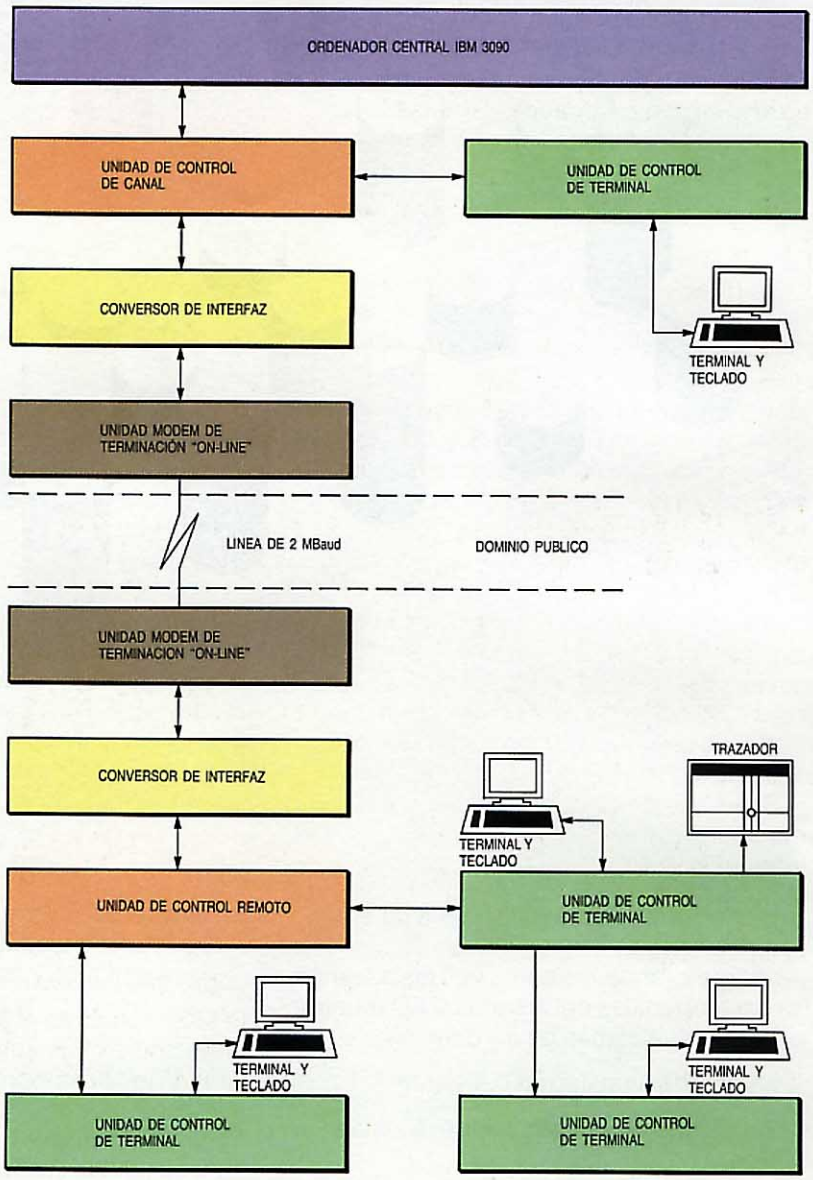


Figura 2
Configuración CAD utilizada en SEL para diseño mecánico.

El módulo de diseño paramétrico sustenta el diseño de variantes y la definición de una biblioteca normalizada de piezas, mientras que la facilidad de consulta de datos gráficos junto con el gestor de visualización de datos gráficos, permiten visualizar dibujos en pantallas gráficas de bajo coste, o bien sacarlos por impresora o trazador. Para que el trazado sea directo, se transfieren los datos "off-line" por cinta magnética entre el ordenador principal y un trazador por láser. Desde 1985 se ha usado con éxito este método en la producción de microfilms para archivo, directamente desde los dibujos CAD.

Entorno del sistema

Además del existente en Stuttgart, se ha instalado otro sistema más para el soporte de otro gran departamento de diseño de

SEL en Pforzheim. Este último sistema se emplea para el diseño y trazado de placas de circuito impreso, además de tareas de diseño mecánico. Ambos sistemas CAD se han equipado con módulos IGES por un periodo de prueba, a fin de que puedan transferirse datos entre sí.

Experiencia

Se ha acumulado considerable experiencia durante la planificación, instalación y operación del sistema que lleva en uso cerca de dos años, incluyendo la instalación piloto. Fundamentalmente se ha utilizado para el diseño mecánico de la práctica de equipo del Sistema 12, bien partiendo de cero o utilizando dibujos recibidos de BTM por la línea de datos.

A la vista de los rápidos avances tecnológicos en sistemas CAD, hay una gran preocupación por reducir los periodos de planificación, toma de decisiones e instalación, con el fin de conseguir un rápido beneficio sobre la inversión.

Se ha comprobado que merece la pena la introducción paso a paso y orientada a tareas específicas. No obstante, ha resultado ser un problema la limitada disponibilidad de bibliotecas de piezas que cumplan las normas DIN/ISO; no existen cintas con piezas normalizadas. Durante la fase de introducción se demostró que era esencial el prestar soporte sobre aspectos del sistema y de aplicaciones a través de una línea directa. También se demostró que instalando un trazador cerca de las estaciones de trabajo CAD se incita al diseñador a que abandone el tablero de dibujo y utilice el terminal CAD. Esto no sólo aumenta las posibilidades de aceptación, sino que suele ser también más económico que la instalación de otro terminal.

Un sistema CAD es de instalación relativamente costosa, y puede ser difícil cuantificar por adelantado los beneficios que reportará. No obstante, era evidente que un departamento de diseño mecánico no podía seguir utilizando métodos manuales si quería sobrevivir. Apenas transcurrido un breve periodo de tiempo desde la instala-

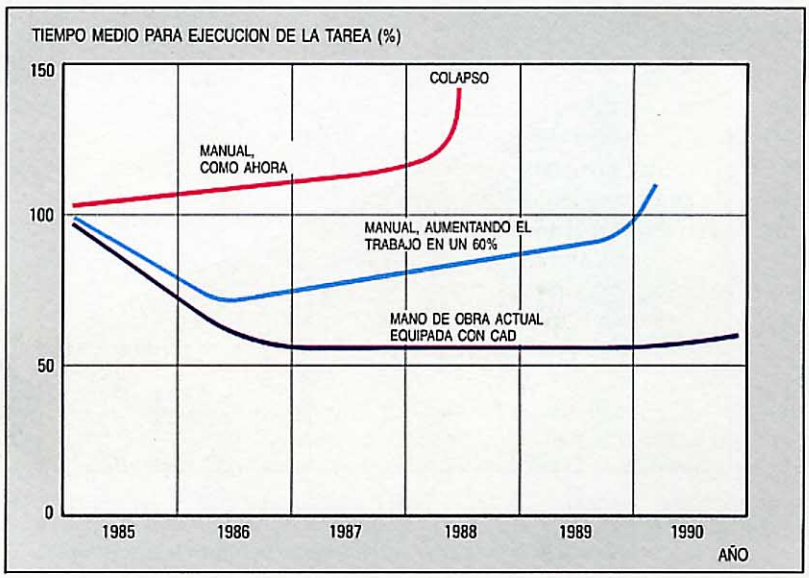


Figura 3
Tiempos medios invertidos en el diseño de equipo mecánico.

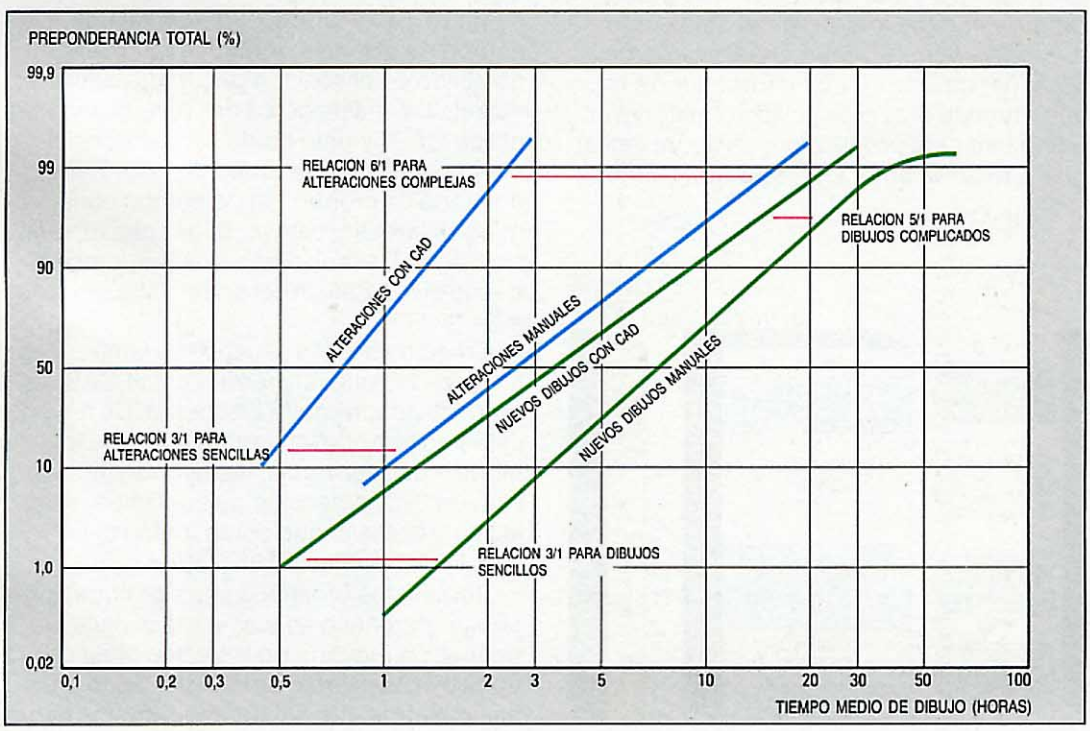


Figura 4
Productividad para delineación y dibujo detallado.

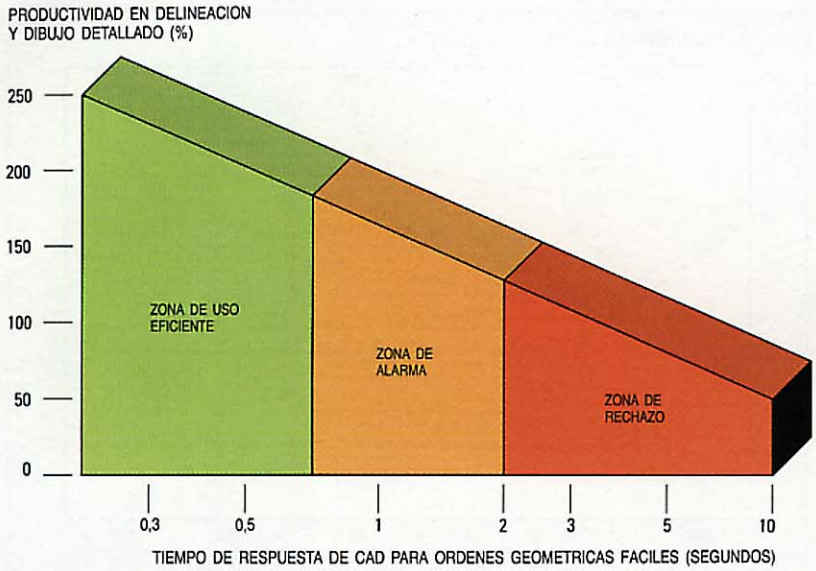


Figura 5
Relación entre productividad y tiempo de respuesta de un sistema CAD.

ción del sistema CAD, se comprobó que los tiempos totales para trabajos de diseño mecánico se habían reducido apreciablemente, tal como indica la figura 3. La experiencia inicial ha sido muy positiva en varios aspectos. En efecto, se han podido resolver tareas complejas de diseño imposibles de realizar manualmente. En segundo lugar, el sistema ha probado ser capaz de distribuir rápidamente información a grandes distancias. Finalmente, todos los datos están almacenados sin errores y de una forma coherente y uniforme. Ninguna de estas ventajas podía haberse conseguido con un sistema manual mejorado[1], que en cualquier caso obligaría a aumentar en un 60% el personal de diseño.

El incremento de productividad para delineación y dibujo detallado, mostrado en la figura 4, debe interpretarse sólo como tendencia general, dado el relativamente corto periodo de tiempo (alrededor de un año) durante el cual ha estado el sistema en pleno uso para producción. La figura indica que la relación entre los tiempos requeridos para diseño manual y diseño asistido por ordenador es próxima a tres para tareas sencillas, bajando hasta cerca de uno y medio para diseños complejos. El aumento de productividad conseguido depende mucho del contenido del dibujo. Las relaciones de tiempo estimadas para diversas modificaciones de diseño están alrededor de tres para cambios y adaptaciones sencillas, y crecen hasta más de seis para las complejas.

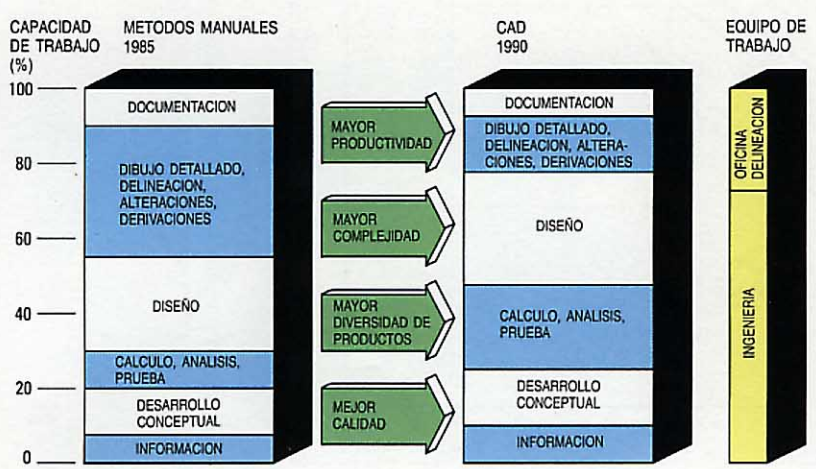
Los tiempos de respuesta del sistema y la comodidad con que se utilizan las facilidades (es decir, la ergonometría del equipo y los programas) son factores decisivos para asegurar que los usuarios acepten el sistema y sean capaces de manejarlo eficazmente[1,2,3]. La figura 5, que refleja la experiencia en SEL y BTM, muestra un factor de productividad para delineación y dibujo detallado cercano al 200%, con un tiempo medio de respuesta del sistema CAD inferior a 0,5 s cuando se realiza un diseño nuevo. Se ha demostrado que un sistema CAD tiene similar productividad a la obtenida por delineación manual (es decir, 100%) cuando el tiempo de respuesta alcanza los 3 s. Además, la figura 5 indica que los usuarios rechazarían la nueva herramienta CAD si las respuestas se demoraran más de 2 s. La línea de transmisión de 2 Mbaud, unida a la alta prioridad concedida al CAD por el ordenador central, garantiza tiempos de respuesta muy cortos.

Es evidente ya que las actividades de la ingeniería de diseño mecánico y del departamento de delineación no cambiarán fundamentalmente al introducirse ayudas por ordenador. No obstante, también está claro que el foco de interés ha cambiado (ver figura 6). En el futuro, el porcentaje de tiempo de diseño invertido en el dibujo detallado y delineación decrecerá desde el actual 35% hasta cerca del 15% cuando se utilice CAD, y esto como consecuencia directa del factor de productividad. Por el contrario, la proporción de tiempo consumida en tareas creativas, tales como diseño conceptual, simulación y análisis, crecerá desde el 22% hasta cerca del 38% o quizás más.

A medida que los usuarios se familiaricen más con las nuevas herramientas CAD, la productividad seguirá creciendo. De hecho, están solicitando ya herramientas de ingeniería más avanzadas, incluyendo modelación en 3-D, detección de conflictos, simulación y análisis, que serán tomadas en consideración hacia 1987/1988.

Durante los próximos años se conectará equipo periférico adicional a la instalación central. La mayoría de las estaciones de trabajo se instalarán en la localización principal, mientras que en lugares remotos habrá

Figura 6
Cambios de tareas como resultado de la introducción de un sistema CAD para diseño mecánico.



otros equipos conectados a través de la línea de 2 Mbaud o de otras líneas disponibles para instalaciones remotas menores.

Habrá que adquirir bibliotecas de piezas normalizadas y adaptarlas a los requisitos específicos de SEL; asimismo habrá que actualizarlas y aumentarlas con regularidad. A este proceso contribuirán con su trabajo diversas organizaciones de normalización que están implicadas en la definición y realización de las citadas bibliotecas.

La búsqueda rápida de documentos y piezas exigirá el establecimiento de un sistema adecuado de extracción, el cual está ya planificado y se realizará de acuerdo con el sistema de clasificación codificada de piezas DIN 4000.

El sistema se está ampliando también para permitir la programación interactiva de máquinas de control numérico basada en datos de diseño existentes. Se generará una biblioteca de herramientas de control numérico (herramientas existentes, datos de tecnología, datos de geometría, etc.) para utilizarse en departamentos de fabricación, y para que el ingeniero de diseño tenga acceso a ellas "on-line" a través de un terminal CAD.

En 1987/1988 se probarán estaciones de trabajo inteligentes y multifunción con el objeto de integrarlas en la instalación CAD existente. Se evaluarán, además, en una fase piloto, herramientas de distribución, archivo y control de documentación para documentos y datos geométricos CAD, que utilizarán terminales gráficos de bajo coste e incluirán gráficos y texto integrados para la generación de documentación de clientes. Finalmente, se tiene el propósito de integrar todavía más el proceso de la información CAD/CAM y comercial para ayudar al tratamiento de listas de piezas.

Conclusiones

La implantación del CAD como ayuda al diseño mecánico puede mejorar notablemente la productividad en diseño, pero ello depende sobre todo de la elección de un sistema adecuado. Es, pues, importante analizar los requisitos de los usuarios y evaluar metódicamente los sistemas CAD disponibles en comparación con tales requisitos. Cuando sea posible, se ha de procurar recomendar un sistema CAD único, con objeto de evitar los problemas de transferencia de datos entre sistemas diferentes.

Un tiempo de respuesta suficientemente corto, un eficaz adiestramiento y un soporte en "línea directa" a los usuarios ayudan a garantizar que las ventajas propias del CAD se logran con relativa prontitud tras la instalación. Es también importante asegurar la coherencia de las bases de datos y bibliotecas porque ellas forman, junto con un sistema asociado de búsqueda, la espina dorsal de cualquier sistema CAD.

La experiencia en SEL ha demostrado la conveniencia de usar el CAD para diseño mecánico, y se está ya preparando el plan de expansión de este sistema.

Referencias

1 G. Klause (editor): *Technical Equipment in Human Service: Volumen 3 - Use of CAD as a Tool in Equipment Design, a Handbook of Practice*, 1985, Expert-Verlag, D7031 Sindelfinger, 150 páginas.

2 J.-D. Foley y V.-L. Wallace: The Art of Natural Graphic Man-Machine Conversation: *Proceedings of the IEEE 62*, 1974, n° 4, pags. 462–471.

3 A. Marcus: Designing the Face of an Interface (Human to/from Computer): *IEEE Computer Graphics*, enero 1982, págs. 23–29.

Waldemar Drtil nació en Alemania, en 1940. Estudió física en la Universidad Justus-Liebig, Giessen, donde se graduó Dipl-Ing en 1965 y Dr-Rer-Nat en 1969. Durante los siguientes cuatro años trabajó para Zuse KG en ingeniería de programación de sistemas de aviación controlados por ordenador. En 1971 el Dr. Drtil ingresó en SEL, donde se especializó en ingeniería de programación para control de procesos, incluyendo fabricación de placas impresas y señalización ferroviaria. Actualmente dirige proyectos en el área de CAE, así como en la implantación del sistema CAD para diseño mecánico.

Gerhard Klause nació en 1931, en Alemania. Estudió ingeniería de telecomunicación y mecánica en la Universidad Técnica de Berlín Oeste, entre 1950 y 1957. Durante su carrera trabajó en muy diferentes áreas técnicas, incluyendo frecuencias ultraelevadas, instrumentación y control, y en 1973 fue nombrado responsable del diseño mecánico de aparatos telefónicos en SEL. Obtuvo el grado de Dr-Ing en 1981, y al año siguiente se encargó de la introducción de herramientas CAD/CAM en los departamentos de diseño mecánico de SEL.

En este número

Danneels, J. M.; Guebels, P.-P.; Rahier, M. C.
Metodología y herramientas CAD para el diseño de circuitos VLSI a medida
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs 196–206.

En los últimos 20 años la complejidad de los circuitos integrados ha aumentado drásticamente con la introducción de las tecnologías VLSI. Hoy, estos circuitos no podrían diseñarse sin la ayuda de los ordenadores. Los autores han investigado las metodologías y herramientas de los sistemas CAD más avanzados para diseños VLSI a medida, y proponen una serie de requisitos para las futuras herramientas que hagan posible el diseño en ITT de circuitos VLSI mucho más complejos. Entre los requisitos del IVDS (nuevo sistema de diseño ITT para circuitos VLSI), actualmente en desarrollo, se encuentra la formalización de los modelos de datos de ingeniería para la integración del sistema y el uso coherente de un método de diseño jerárquico.

Prazic, B.
Entorno soporte para desarrollos VLSI parcialmente a medida
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 207–215

El diseño parcialmente a medida es un procedimiento rentable para la producción de circuitos VLSI de complejidad baja y media. El autor describe el ISDS – sistema ITT de diseño de circuitos VLSI parcialmente a medida – que proporciona a los ingenieros de sistemas un juego completo de herramientas CAD para todos los aspectos del diseño, desde la captación de esquemáticos al trazado. Estas herramientas se apoyan en bibliotecas de células de familias lógicas clave (CMOS y ECL). El ISDS da también acceso a avanzadas tecnologías de proceso de suministradores seleccionados. Según los datos obtenidos, el sistema ISDS ha mejorado apreciablemente la productividad respecto a los anteriores métodos de diseño.

Sánchez Moreno, C.
Galileo: modelo, lenguaje y herramientas
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 216–224

La metodología Galileo es de gran utilidad para modelar, analizar y simular equipos y sistemas en los que intervienen procesos concurrentes. La autora describe la forma de modelar los sistemas concurrentes como redes Galileo, utilizando para ello un lenguaje gráfico basado en las redes de Petri para los problemas de sincronización, y Pascal para describir los aspectos funcionales o de datos. Las propiedades de sincronización pueden analizarse formalmente y el comportamiento funcional, comprobarse mediante simulación. Además es posible asignar tiempos o duraciones a las actividades y obtener así una evaluación de las prestaciones del sistema a partir del modelo.

Cunningham, J. J. F.
Sistema de información de componentes de ITT
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 225–231

La magnitud y complejidad de la información necesaria para elegir y comprar componentes eléctricos modernos, y emplearlos en la fabricación de productos, hacen que la distribución y almacenamiento en papel sea ineficaz. El autor describe la base de datos, solución desarrollada por el ITT Europe Engineering Support Centre, y cómo la emplea ITT en soporte de todas las etapas del ciclo de vida del producto.

Illi, D. R.
Desarrollo de programación para el Sistema 12
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 232–238

El método de diseño y documentación de la programación del Sistema 12 contiene las tres fases de diseño de programas: diseño de alto nivel, diseño detallado y codificación. Está basado en refinamientos graduales de la definición del producto, utiliza el lenguaje de especificación y descripción del CCITT, y sirve de base a los conceptos de máquina de estados y de mensajes finitos, así como al modelado de datos. Los procedimientos y herramientas de diseño se han mejorado continuamente para aumentar la productividad, mejorar la calidad, formalizar el proceso de diseño y obtener toda la documentación relevante de diseño y del cliente a partir de ficheros procesables en ordenador. Unos sistemas informatizados de distribución de información, controlados por gestión de la configuración, apoyan al desarrollo del Sistema 12 en los centros europeos de diseño de ITT.

Page, A.; Grützmann, S.
Sistema integrado de diseño de placas impresas
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 239–247

El IPDS (sistema integrado de diseño de placas impresas), es un sistema de diseño asistido por ordenador de ITT, que proporciona un entorno integrado para el desarrollo de placas impresas. Maneja el proceso completo, desde la generación del esquemático hasta el trazado final de la placa. Los datos generados durante las etapas de diseño se pueden luego enviar a los equipos de fabricación y pruebas. Los autores describen el IPDS y su aplicación para el diseño de placas, esbozando asimismo los planes para futuras mejoras e inclusión de nuevas facilidades.

Haque, T.; Montes, J.
Soporte de productos basados en microprocesadores durante su vida útil
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 248–255

Los entornos de desarrollo pueden considerarse como un marco dentro del cual se conciben, realizan, prueban y lanzan al mercado productos, y puede dirigirse la gestión de su ciclo de vida útil. Proporcionan los mecanismos de gestión de la información por los cuales se interrelacionan y controlan conceptos, métodos, herramientas de realización y necesidades nuevas o modificadas. Los autores describen el sistema de gestión de la configuración de productos desarrollado en ITT para proveer la extensa gama de facilidades necesarias para el soporte de los productos con microprocesadores durante toda su vida útil. Por su naturaleza general, el sistema puede también ofrecer tal soporte a muchos otros productos diversos.

Hunter, D.; Kobitzsch, W.
Módulo interfaz de medidas para el entorno de desarrollo de programas
Comunicaciones Eléctricas, volumen 60, n° 3/4, págs. 256–258

Los proyectos se pueden controlar con mucha más eficacia disponiendo a tiempo de información sobre los parámetros principales que rigen su desarrollo. Precisamente esta información la aporta el módulo interfaz de medidas, elaborado como parte del entorno de desarrollo de programas de ITT, permitiendo a los responsables de proyectos una rápida evaluación del estado de los mismos. Los autores describen la estructura de este nuevo módulo alrededor de una base de datos, y analizan los tipos de información que puede suministrar en ayuda de los responsables de proyectos. También consideran brevemente futuras mejoras posibles.

Baradello, C. S.; Carloni, G.
Generación de sistemas ejecutivos en tiempo real
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 259–265

El GERTEF, sistema normalizado de soporte lógico para generación de programas ejecutivos en tiempo real "a medida" de una arquitectura determinada de equipo físico, actúa combinando módulos pequeños de programas para formar un único componente lógico que proporcione todas las funciones requeridas, análogamente al diseño de equipo físico en el que un circuito VLSI sustituye a miles de componentes. La reutilización de módulos de programas ya comprobados ofrece considerables ventajas tanto en productividad como en fiabilidad del sistema. Los autores describen los principios del GERTEF y su empleo para generar programas ejecutivos en tiempo real. El sistema está ya integrado en el entorno de desarrollo de programas de ITT, y por tanto a disposición de todas las casas asociadas.

Wellens, W.
Entorno distribuido de pruebas de programación
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 266–269

Al introducir estaciones de trabajo para desarrollo de programación, han aparecido nuevos medios para el soporte de pruebas eficientes de programas en proyectos de telecomunicaciones, a menudo basados en redes de microprocesadores. El autor describe los condicionantes de este enfoque, y expone cómo los cumple el GESTOM, entorno distribuido de pruebas de programación. El GESTOM se puede utilizar en las tres fases principales (prueba de módulo, prueba del sistema y prueba de verificación), liberando al probador de la necesidad de conocer sobre cuáles circuitos actúan los programas.

Crawford, J. T.
Desarrollo y mantenimiento de programas de aplicación
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 270–277

La productividad es un problema para cualquier productor en gran escala de programas de aplicación. The Hartford, un importante grupo de seguros de los Estados Unidos, decidió en 1980 afrontar este problema introduciendo nuevas técnicas y herramientas. El autor describe el ciclo de vida de desarrollo obtenido, del que resulta un ahorro de tiempo muy importante al desarrollar nuevos programas. También trata de las medidas tomadas para mejorar el mantenimiento de programas antiguos con la intención de evitar que tal actividad llegue a ser la enorme carga que se presumía.

Povelones, A.
Herramientas UNIX para el desarrollo de sistemas de terminales
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 278–285

Al comienzo de los 80, ITT Courier comprobó la evidente necesidad de un nuevo entorno para el desarrollo de sistemas de terminales de pantalla provistos de varios microprocesadores y con un soporte físico propio capaz de emular la tecnología IBM. El autor describe las razones para cambiar de un entorno de desarrollo a medida a otro basado en el sistema operativo UNIX, y expone las ventajas de este último. Hoy, el sistema soporte de desarrollo de Courier da servicio a 210 personas, entre ellas a 60 ingenieros de programación que han utilizado el UNIX para crear bibliotecas de productos de 500.000 líneas de código (5.000 módulos fuente) en tres años.

Brigham, F. R.
Aportación de los factores humanos al desarrollo de productos
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 286–293

La ingeniería de factores humanos se ocupa de mejorar la calidad del interfaz del usuario durante el diseño, teniendo en cuenta sus necesidades. En el caso de productos de usuario final, lo más importante es asegurar la máxima aceptación, diseñando productos de rápida comprensión y uso fácil. Para conseguirlo, debe obtenerse la apropiada información de factores humanos en todas las etapas del diseño, desde el diseño conceptual hasta la puesta en marcha del sistema. El autor reseña las etapas principales del diseño e ilustra con ejemplos prácticos la variedad de aportaciones de los factores humanos al desarrollo de productos.

Byerley, P. F.; Leiser, R. G.; Saffin, R. F.
Diseño de interfaces usuario-sistema mediante ayuda cognoscitiva
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 294–302

El diseño de interfaces usuario-sistema debe tener en cuenta las características de los usuarios para que el diálogo pueda comprenderse y aprenderse fácilmente. La psicología cognoscitiva ofrece la base necesaria para diseñar tales interfaces, pero se la debe presentar de un modo que sea asequible a los diseñadores del interfaz. Los autores describen el desarrollo de una ayuda cognoscitiva al diseño, adecuada para evaluar, de forma sistemática y objetiva, la eficiencia de varios aspectos del interfaz usuario-sistema. Por el momento existen módulos de medida para evaluar coherencia, modelos mentales, memoria y atención, pero esta gama se extenderá en el futuro.

de Alberdi, M.
Sistema vídeo de análisis asistido por ordenador
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 303–307

El análisis de tareas y el diseño de diálogos en los que intervengan factores humanos requiere la observación de las personas y de las máquinas, así como de su interacción, tanto en los laboratorios como en condiciones normales. Los magnetoscopios son ideales para esta función, ya que permiten la grabación ininterrumpida de largas sesiones así como su reproducción instantánea y repetitiva. Las diferentes secuencias de grabación pueden codificarse fácilmente con marcas de tiempo para identificar con precisión los eventos que aparezcan en la pantalla. El sistema CAVAS (sistema de análisis de vídeo por ordenador) asegura la utilización óptima de estas grabaciones y facilita la identificación y etiquetado de los eventos, anotando el instante en que ocurren, así como el análisis de su significación estadística y la impresión de los resultados, tanto en forma gráfica como textual. El CAVAS se ha utilizado ya en diversos estudios por el Grupo de Factores Humanos del ITT Europe Engineering Support Centre, revelándose como una valiosa herramienta para mejorar el diseño del interfaz entre hombre y máquina.

Giambalvo, F.; Vignazia, D.
Análisis de planificación, modelado y fijación de precios
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 308–313

Para optimizar los costes de producción, comercialización e instalación de las centrales telefónicas se ha de disponer de información bien organizada y fácilmente accesible a los departamentos involucrados. Los autores describen el sistema SCAMP, herramienta desarrollada para recoger y controlar datos de las centrales, desde que se recibe el pedido hasta que se entrega la nueva central a la Administración. El artículo expone asimismo cómo se pueden utilizar estos datos para planificar la producción y efectuar simulaciones que determinen la eficacia de introducir cambios, en la capacidad de producción por ejemplo. Finalmente, el SCAMP suministra de modo automático toda la información necesaria para fijar precios a la central ya completada.

Drtil, W.; Klause, G.
Ayudas de ordenador para el diseño mecánico en la industria electrónica
Comunicaciones Eléctricas (1986), volumen 60, n° 3/4, págs. 314–319

La introducción de ayudas de ordenador para el diseño mecánico en la industria electrónica sólo tendrá éxito si el sistema elegido se acomoda a las exigencias concretas del departamento de diseño. Los autores describen la experiencia de SEL en especificar requisitos, elegir una configuración de CAD adecuada e instalarla en el entorno de diseño existente. Comentan asimismo las experiencias iniciales en el sistema, que ya han revelado un desplazamiento de las responsabilidades, pasando la oficina de delineación a realizar un diseño más detallado y quedando libres los ingenieros para dedicarse al diseño conceptual, análisis, simulaciones y cálculos.

Oficinas Editoriales

La correspondencia relacionada con las diferentes versiones de Electrical Communication debe dirigirse al editor correspondiente:

Rod Hazell
Electrical Communication
Great Eastern House
Edinburgh Way
Harlow, Essex
England

Wolfgang Schmid
Elektrisches Nachrichtenwesen
Lorenzstrasse 10
7000 Stuttgart 40
Bundesrepublik Deutschland

Antonio Soto
Comunicaciones Eléctricas
Ramírez de Prado, 5
28045 Madrid
España

Jean-Pierre Dartois
Revue des Télécommunications
30 Avenue George V
75008 París
France