Edmundo A. Cáceres

Análisis y Diseño de Sistemas de Información

2014

Análisis y diseño de sistemas de información

Conceptos preliminares

El análisis y diseño de sistemas de información consiste en *investigar sistemas y necesidades actuales para proponer sistemas mejores*. Por brevedad, suele decirse *análisis de sistemas*. En la descripción anterior hay implícitos varios conceptos, que vamos a tratar brevemente.

Sistemas de información. Son sistemas que procesan datos para producir información. Los datos son las entradas y la información la salida. Pueden ser manuales, cuando no hay intervención de computadoras, o computarizados.

Componentes de los sistemas de información. Son personas, datos, soportes de datos, máquinas, procedimientos, programas, controles, formularios, reglamentaciones, etc. Toda organización tiene sistemas de información que penetran y conectan las estructuras administrativas, del mismo modo que los nervios en el cuerpo.

Análisis*.* Consiste en recolectar e interpretar hechos sobre el sistema actual y las necesidades de información actuales y previsibles en el futuro, para detectar:

Lo que está bien
Lo que está mal
Lo que sobra
Lo que falta

Diseño. Consiste en planear y desarrollar un nuevo sistema que solucione los problemas detectados en el sistema actual y los supere ventajosamente. El nuevo sistema puede limitarse a remendar el sistema actual, pero también puede ser un cambio de grandes dimensiones.

Investigación. La investigación es el proceso utilizado por la ciencia y la tecnología para adquirir conocimiento, formular hipótesis, probarlas o rechazarlas. Es un proceso porque consta de una serie de etapas. Hay tres tipos de investigación: preliminar, descriptiva y causal.

INVESTIGACIÓN PRELIMINAR. También llamada investigación exploratoria, consiste en descubrir un problema y sus componentes; a veces puede formular hipótesis sobre ellos. Carece de estructura, porque no conociendo el problema, no se puede establecer de antemano qué se debe estudiar. Su propósito es esclarecer lo que hay que investigar posteriormente, mediante los otros tipos. En análisis de sistemas, este tipo de investigación se usa para definir el sistema a estudiar y sus componentes, y para planificar el análisis; raramente se formulan hipótesis.

INVESTIGACIÓN DESCRIPTIVA. Consiste en estudiar los componentes del problema descubierto mediante la recolección de hechos y su análisis, para describirlo con precisión. Como se conoce lo que se quiere investigar, esta investigación es estructurada, es decir, se pueden establecer las herramientas para recolectar los hechos. En análisis y diseño de sistemas, corresponde al análisis.

INVESTIGACIÓN CAUSAL. Consiste en probar hipótesis que postulan relaciones causa –

efecto entre los componentes de un problema. Generalmente, las pruebas toman la forma de *experimentos*. Es poco aplicable al análisis de sistemas.

Recolección de datos. Cualquier investigación necesita obtener datos sobre el problema. Hay cuatro técnicas para obtener datos: observación directa, entrevistas, cuestionarios, datos secundarios.

OBSERVACIÓN DIRECTA. El investigador observa atentamente los componentes del problema que estudia, registrando lo que ve. Es la mejor técnica, porque no hay nadie interpuesto. Los errores de observación son debidos a los instrumentos usados o a las deficiencias del investigador.

ENTREVISTAS. El investigador conversa con un informante, quien le cuenta sobre el problema. Además de los errores del investigador, se pueden producir errores del informante. Esta técnica es útil cuando el problema no puede observarse, porque no se da en el transcurso de la investigación. Como cada entrevista lleva tiempo, no puede aplicarse masivamente, sino a un reducido grupo de informantes. Al tener forma de diálogo, las preguntas pueden ampliarse para obtener mayores datos.

CUESTIONARIOS. Son preguntas por escrito, que se realizan simultáneamente a muchos informantes. Como no hay contacto directo entre ellos y el investigador, las preguntas son más estructuradas, careciendo de la flexibilidad de la entrevista.

DATOS SECUNDARIOS. Consiste en estudiar antecedentes escritos que puedan brindar datos sobre el problema. Es la técnica propia de la historia.

En análisis de sistemas se pueden aplicar todas estas técnicas en el análisis del sistema actual. En la investigación preliminar es poco utilizable el cuestionario, por ser muy estructurado.

Hechos. Son *datos verificados*, necesarios para conocer adecuadamente el problema. Verificados quiere decir que representan realidades. Cualquier dato sobre el sistema actual, mientras no sea verificado, es una *opinión*. Para verificar los datos, se recurre a la medición, el conteo, la observación de la ocurrencia del fenómeno, el cotejo de una opinión con documentación que la respalde, etc.

Analista de sistemas

El especialista en análisis y diseño de sistemas es conocido como analista de sistemas**,** o simplemente analista. No hay que confundirlo con el programador de computadoras, que es otro especialista. El analista puede trabajar sobre sistemas manuales o computarizados, mientras que el programador no puede prescindir de las computadoras.

Usuarios

Usuarios son las personas relacionadas con los sistemas de información, sin ser especialistas. Nos valdremos de dos criterios para clasificarlos: actuación y nivel organizativo.

Actuación

Según la forma de actuar sobre los sistemas, se clasifican en usuarios directos e indirectos.

Usuarios directos. Son aquellos que ingresan datos u obtienen informes, actuando directamente sobre los sistemas, es decir, operándolos. Donde no hay computadoras, por ejemplo, son los vendedores que llenan a mano las facturas; o los empleados que resumen las ventas por semana. En sistemas computarizados, son por ejemplo los vendedores que mediante terminales confeccionan facturas; o los operadores del centro de cómputos que capturan datos y obtienen informes. Con la invención de nuevas tecnologías, usuarios que antes no lo eran, se han convertido parcialmente en directos. Es el caso de los cajeros automáticos, operados por los clientes.

Usuarios indirectos. Son aquellos que emplean la información producida por los sistemas, pero no los operan. Por ejemplo, un gerente que usa los resúmenes de ventas o los clientes que reciben facturas por un servicio telefónico o eléctrico.

Nivel organizativo

Según su nivel dentro de la organización, los usuarios se clasifican en operativos, gerentes y directivos.

Usuarios operativos. Son los empleados y jefes de los niveles menores. Unos y otros pueden ser usuarios directos o indirectos. Lo común es que los empleados sean usuarios directos y los jefes usuarios indirectos.

Usuarios gerentes. Controlan el desempeño de los sistemas de información relacionados con sus áreas de competencia, por lo cual tienen un conocimiento adecuado de sus virtudes y defectos. Normalmente son usuarios indirectos de *informes por excepción* y de *resúmenes*. Pueden ser usuarios directos, especialmente si saben emplear cierto software. Así, desempeñan este papel cuando usan los resúmenes para procesarlos en planillas de cálculo o cuando obtienen datos directamente de las tablas, mediante consultas que ellos programan.

Usuarios directivos. Deciden invertir en sistemas de información, sea en su análisis y diseño, sea en nuevos equipos. Son básicamente usuarios indirectos de la información producida por los distintos sistemas, necesarios para determinar el estado interno de la empresa. También lo son de información externa, que les sirve para conocer el entorno, con lo cual aprovechan oportunidades y enfrentan riesgos.

Participación de usuarios

Antes de aplicar computadoras a los sistemas de información, cuando todo se resolvía en forma manual, se tenía clara la conveniencia de que los usuarios participaran del análisis y diseño, por lo que se recomendaba interesar a todos los usuarios. Con la llegada de las computadoras, pareció durante un tiempo que el lenguaje de los analistas era algo oscuro y poco entendible para los usuarios, situación abonada en cierta medida por la actitud un tanto soberbia de muchos de aquéllos.

A través de los años se ha dado un acercamiento en lo que respecta a los conocimientos de analistas y usuarios. Antes, los analistas poco sabían de organización;

pero actualmente un aspecto importante de su formación académica es el estudio de administración, contabilidad, economía, comportamiento organizacional. Los usuarios, por su parte, hace unos años no entendían de computadoras. Pero hoy, con el amplio uso de estas máquinas y la capacitación abundante en ellas, conocen el valor de los sistemas computarizados bien diseñados. Este mutuo acercamiento hace que puedan entenderse más que antes, lo cual facilita que los usuarios intervengan en el análisis y diseño de sistemas.

Es conveniente que el analista fomente la participación de los usuarios, porque la creación de un clima de entendimiento e interés facilita la recolección de hechos y el aporte de contribuciones valiosas. Los usuarios son quienes deberán emplear el nuevo sistema, por lo que su interés es legítimo. Por el contrario, un clima de desconfianza entorpece la investigación. La participación se da modo espontáneo en la medida que el analista sea una persona abierta y comunicativa. Pero puede formalizarse a través de comisiones integradas por personas con intereses en los sistemas bajo estudio. La participación está relacionada con el nivel de los usuarios, por lo que ellos pueden aportar.

Directivos. Tienen una visión panorámica sobre todos los sistemas de la institución y su eficiencia. Como son quienes deciden implantar nuevos sistemas e invertir en equipos y software, su opinión sobre lo que pretenden de los sistemas orienta al analista. Deben estar informados del avance e inconvenientes de la investigación. Su interés por un nuevo sistema abre las puertas de los niveles subordinados, porque si los directivos están interesados, difícilmente los niveles inferiores puedan ignorar la labor del analista.

Gerentes. Tienen una visión amplia de los sistemas actuales que afectan a sus áreas de responsabilidad. Brindan pautas generales sobre las necesidades que deben satisfacer los nuevos sistemas.

Jefes operativos. Tienen una visión más reducida, pero más detallada, de los sistemas actuales bajo su dependencia. Son una fuente de información útil en detalles.

Operadores. Su visión de los sistemas que operan es la más reducida, pero la más rica en detalles. Sus opiniones sobre cambios a introducir en un nuevo sistema son valiosas, porque son ellos los que van a sufrir o aprovechar las características operativas.

Grupos informales. Su existencia y poder, aunque no se reflejan en la estructura formal, son una realidad. Pueden ser, por ejemplo, un grupo de personas unidas por lo que consideran una amenaza o un sindicato con fuerte interés en impedir la reforma del sistema actual. El analista debe detectar estos grupos y tratar de cambiar su actitud.

Transacciones

Las transacciones se producen constantemente en el nivel operativo, como compras y ventas. En los sistemas de información, necesitamos de algunas nociones previas para precisar qué entendemos por transacción.

Entidades son tanto los actores de un sistema de información como las relaciones

que se dan entre ellos, por ejemplo clientes, vendedores, artículos, socios, alumnos, carreras, asignaturas, facturas, solicitudes de examen. Las entidades tienen *atributos*. Por ejemplo, los atributos de la entidad cliente son nombre, domicilio, teléfono, CUIT, etc.; los atributos de la entidad alumno son matrícula, nombre, domicilio, carrera, año de ingreso, etc.; los atributos de la entidad factura son número, fecha, cliente, artículos, cantidades, precios, etc. En un sistema de información hay que identificar las entidades y sus atributos. Los valores de atributo que corresponden a las entidades particulares se llaman *datos*. Así, si para un cliente dado el nombre es *Esteban Ruiz* y su domicilio *Salta 269*, estos valores son datos.

Hay dos tipos de entidades: maestras y transaccionales. Esta distinción tiene importantes consecuencias prácticas en sistemas computarizados, como veremos. Las *entidades maestras* son los actores de un sistema de información, que se relacionan entre sí en distintas circunstancias. En un sistema de facturación, por ejemplo, las entidades maestras son clientes, proveedores, vendedores, artículos. Cada entidad maestra particular, como el cliente *Esteban Ruiz* o el artículo *Lámpara de escritorio*, queda definida por sus propios datos. Las *entidades transaccionales*, o *transacciones*, son relaciones entre las entidades maestras, en circunstancias determinadas de tiempo, lugar, cantidad, propósito, modo, etc. En el sistema de facturación, por ejemplo, son facturas, notas de débito, recibos. Cada entidad transaccional, como la factura *19728*, queda definida por datos de las circunstancias, como fecha y cantidades de los artículos comprados, y por datos pertenecientes a las entidades maestras relacionadas, como el código de cliente y los códigos de los artículos facturados.

En sistemas computarizados cada entidad maestra se registra una sola vez mediante los datos de sus atributos. Uno de sus atributos o un grupo de ellos tomados como un todo deben poder identificar a cada entidad inequívocamente. Tratándose de clientes, este requisito lo cumple el código de cliente; en el caso de asignaturas de una facultad, para identificar a cada una hace falta la carrera, el plan de estudios y la asignatura. Los atributos que identifican la entidad maestra forman su *clave*, llamada clave intrínseca o interna. Los datos clave no pueden estar repetidos, porque dejarían de ser identificadores. En las transacciones se necesita una clave interna que las identifique, más las claves de los actores (llamadas aquí claves externas) y las circunstancias específicas. En una factura, por ejemplo, se registra el número de factura, que es su clave interna, más las claves externas del cliente, del vendedor y de los artículos, más las circunstancias de fecha, precio y cantidad de cada artículo. Registrar en la factura las claves externas hace posible obtener los demás datos de las entidades maestras mediante enlaces a ellas por tales claves, sin tener que introducirlos en la transacción. Esto evita errores y ahorra tiempo y espacio de almacenamiento. Al imprimir una factura, se activan los enlaces entre claves externas de la factura con las claves internas de las entidades maestras y se obtiene la totalidad de datos requeridos. La clave interna de las transacciones, como en las entidades maestras, puede ser un atributo o un grupo de ellos.

Para aprovechar la distinción mencionada, las entidades se almacenan en tablas distintas. Las entidades maestras se almacenan en *tablas maestras*, una tabla por cada tipo de entidad. Las transacciones se almacenan en *tablas transaccionales*, una o más tablas por cada tipo de transacción. En este último caso, usar más de una tabla supone ventajas que trataremos más adelante.

En sistemas manuales, la distinción entre entidades maestras y transaccionales no es rigurosa. Al llenar cada factura, hay que escribir la totalidad de datos requeridos, repitiendo una y otra vez los datos de los clientes y de los artículos. Esto, ob-

viamente, es más demoroso y muy sujeto a errores.

Actividades de los sistemas de información

Los sistemas de información, especialmente los que procesan transacciones, exigen la realización de diferentes actividades, que podemos tipificar como sigue.

Captura de datos

Es el ingreso original de datos al sistema. Por ejemplo, puede ser el llenado manual de un formulario, o la creación por teclado de un nuevo registro de una tabla, con datos tomados de documentos presentados por el usuario. Por el alto contenido de intervención humana, es una actividad sujeta a muchos errores, que afectarán la producción de información. Para reducir estos errores, cuando se emplean computadoras se han ideado varios recursos.

Transcripción

Consiste en copiar datos previamente capturados. Cuando se realiza manualmente, la probabilidad de errores es muy alta. En sistemas computarizados, la transcripción es automática: si hay errores en los datos, no se debe a la transcripción, sino a que han sido mal capturados.

Codificación

La codificación es una forma abreviada de representar datos mediante números, letras, caracteres especiales o una mezcla de ellos, llamados códigos. Por ejemplo, para el campo *Sexo* de la tabla *Alumnos*, se puede usar los códigos *M* para representar masculino y *F* para femenino.

Ventajas

La codificación tiene ventajas obvias, de las cuales mencionaremos las siguientes.

Reducción del espacio de almacenamiento. En el ejemplo, si para *Sexo* se escribiera la palabra *Masculino*, haría falta 9 posiciones; si usamos el código *M*, necesitamos 1 posición. Si en la tabla *Alumnos* tenemos 10.000 registros, si no usáramos codificación necesitaríamos 90.000 posiciones para el campo *Sexo*; en caso de usarla, nos bastarían 10.000.

Ahorro de tiempo de captura y transcripción. Es mucho más rápido escribir una *M* que la palabra *Masculino* o *F* que *Femenino*.

Reducción de errores. Escribir *M* es más fácil que escribir *Masculino*, porque ésta última podría escribirse como *Masculino*, *Macsulino*, *Maculino*, etc.

Posibilidad de control automático. Si el programa para ingresar estos datos al campo *Sexo* sólo acepta los valores *M* y *F*, los únicos errores posibles se reducen a dar a una mujer el código *M* y a un hombre el código *F*.

Normalización de la representación de valores. Si usáramos palabras para representar el sexo, serían posibles los valores *Varón*, *Hombre*, *Masculino*, *Mujer*, *Femenino*, más todos sus errores ortográficos, para representar dos valores. Si ordenáramos la tabla *Alumnos* por el campo *Sexo*, tendríamos tantos grupos como valores

distintos se hayan usado. Usando codificación, sólo habrá *M* y *F*, lo cual formaría dos grupos si ordenáramos por *Sexo*.

Características
Para que resulte práctica, la codificación deben tener ciertas características, entre las que conviene citar las siguientes.

Lógica. La codificación debe estar construida según un esquema racional. Por ejemplo, si queremos codificar los artículos de una ferretería, podría convenir que cada código indicara en posiciones predefinidas el rubro, la marca y el artículo específico. Esto facilitará realizar procesos de acuerdo a estos componentes, como listar un rubro particular o los artículos de una marca.

Flexible. La codificación debe permitir agregar nuevos códigos sin inconvenientes. Esto supone que se prevea adecuadamente el número de caracteres de cada componente del esquema.

Económica. La codificación debe usar la menor cantidad de caracteres posible, para ahorrar espacio y disminuir errores de captura o transcripción.

Validación

En los sistemas de información se realiza diversidad de controles sobre todas las actividades, para detectar errores y corregirlos mientras sea posible. Por ahora, nos interesa el tipo de control llamado *validación*. La validación es el control de admisión de los datos capturados o transcriptos. Decimos admisión y no exactitud, porque ésta no siempre se puede lograr. Por ejemplo, si tenemos que ingresar el código del departamento de San Juan donde vive un alumno, podemos controlar que varíe entre 1 y 19, rechazando otros valores; pero si el alumno vive en el departamento 6 y por error ingresamos 9, el código será admitido. La calidad de la información que produzca un sistema depende absolutamente de la calidad de los datos capturados o transcriptos. Si entran datos erróneos, las salidas que de ellos deriven necesariamente serán erróneas.

Como hay varios tipos de validación, conviene orientarse mediante el siguiente esquema:

- Validación simultánea
  - En un registro
    - A nivel de campo: tipo de datos, patrón, rango, lista de valores, valor específico, valor habilitado, acceso condicionado, dígitos verificadores
    - A nivel de registro: consistencia
  - Entre registros: integridad referencial
- Validación posterior

Validación simultánea
El control se realiza durante el ingreso de los datos, de modo automático. Es posible en sistemas computarizados. Por ejemplo, si el sexo se codifica como *F* o *M*, cuando se llena el campo *Sexo* con un valor que no sea *F* ni *M*, se puede construir un programa que detecte el error, dé un mensaje y no permita salir del campo hasta introducir un código aceptable. El ejemplo anterior es sencillo, pero es posible programar

procedimientos de gran complejidad para llevar a cabo el control.

La validación simultánea puede hacerse en un registro o entre registros. En un registro, puede ser a nivel de campo o a nivel de registro. La validación de campo consiste en controlar el valor ingresado a un campo, sin tener en cuenta los demás campos. Mientras el valor ingreso no sea correcto, no se puede ir a otro campo.

La validación de registro consiste en controlar el contenido de un campo contra el contenido de otro, de modo que ambos sean congruentes. Mientras el valor de esos campos no sea adecuado, no se puede ir a otro registro.

La validación entre registros, que necesita una relación entre dos tablas mediante un campo clave, controla que no queden registros huérfanos en la tabla hija.

Tipo de datos. El tipo de datos con que está definido un campo es un primer control básico de lo que se puede ingresar a ellos. Por ejemplo, si un campo es numérico, será imposible ingresar letras. Si el campo es de fecha, sólo admitirá fechas válidas, no un 31 de febrero, por ejemplo. Las mayoría de los lenguajes puede impedir estos errores sin mayores complicaciones.

Patrón. Un dato, como la matrícula de un automóvil, puede ser un compuesto de letras y números, pero siguiendo un patrón rígido: tres letras mayúsculas, tres números. Es fácil establecer estos patrones mediante códigos de formato y de máscara de entrada. Por más que se esfuerce, el usuario no podrá entrar una letra o un número donde no está permitido, ni escribir las letras en minúscula.

Rango. Un dato válido puede estar limitado a un rango de valores contiguos. Por ejemplo, si los códigos de las carreras de la Facultad de Medicina van del 18 al 25, al registrar datos de exámenes rendidos en esa Facultad no podremos ingresar carreras inferiores a 18 ni superiores a 25.

Lista de valores. Una lista es similar al rango, salvo que los valores válidos no son contiguos, sino que se indican en una lista. Por ejemplo, si los códigos de carreras de la Facultad de Medicina son 8, 11, 15 y 45, no se podrá ingresar carreras que no correspondan a estos números. Esto sirve cuando los valores son pocos, porque basta aplicar un mecanismo de soft sencillo. Por ejemplo, en VFP, la regla de validación se puede hacer con la función siguiente:

Inlist(carrera, 8,11,15,45)

En Access, al definir la estructura del campo *Carrera*, en la regla de validación se hace:

8 o 11 o 15 o 45

Cuando los valores son cientos o miles, es imposible listar todos los valores. En este caso, se usa una tabla que contenga todos los valores admisibles para buscar en ella el valor ingresado al campo que se quiere validar. Por ejemplo, si se está llenando el campo *Matrícula* de la tabla transaccional *Exámenes*, se puede tomar como origen de la lista el campo *Matrícula* de la tabla maestra *Alumnos*. Cada vez que se llene el campo *Matrícula* de *Exámenes*, el programa para bajas y cambios buscará en la tabla *Alumnos* el valor ingresado. Si lo encuentra, está bien. Si no lo encuentra, es un error que debe denunciar.

Valor específico. Los controles indicados anteriormente aumentan la probabilidad de que el dato ingresado sea correcto, no su certeza. Por ejemplo, si el código de carrera de la Facultad de Ingeniería que se ingresa es 20, sabemos que está dentro

del intervalo 18 – 25, por lo cual es probable que sea correcto; pero si en realidad es 21 y lo hemos escrito mal, el error no se detecta. Para disminuir estos errores *casi* totalmente, hay formas para controlar el valor específico del dato. Una es usar dígitos verificadores. Al dar de alta un nuevo cliente en un banco, el programa empleado determina el número de cuenta que le corresponde y calcula el dígito asociado. Posteriormente, toda vez que se ingresa esa cuenta en otros programas, hay que incluir el dígito asociado. Entonces se vuelve a aplicar el mismo cálculo. Si el dígito ingresado coincide con el calculado, el dato se considera correcto; de lo contrario, se rechaza. Una segunda forma es la clave personal que usa un usuario para acceder a un sistema. Hay dos claves, una inmodificable que corresponde al usuario y otra que él puede modificar a voluntad. Como esta segunda clave la establece el usuario sin que los demás la conozcan, es prácticamente imposible que otro pueda darla cuando le es solicitada.

Valor único por registro. A veces se necesita que un campo o un grupo de campos identifiquen cada registro de una tabla. Para ello, no se debe permitir que un mismo valor se repita en distintos registros. Es el caso de la matrícula en un archivo maestro de alumnos, el código de producto en una tabla maestra de artículos, el número de factura unido al código de producto en una tabla transaccional de facturas, etc. Para lograr esta unicidad, basta con crear un índice que no acepte duplicados en la tabla respectiva. En adelante, cuando se ingrese un nuevo registro o se cambie el valor del identificador de un registro existente, el soft controlará que el valor no se repita y dará un error en caso de transgredir esta exigencia.

Valor habilitado. En ocasiones, la prueba de valor específico puede ser insuficiente como control. Por ejemplo, si un número de cuenta es correcto según su dígito verificador, ello no significa que la cuenta esté habilitada en el archivo maestro de cuentas corrientes. Podría ser que hubiera sido dada de baja por mal cumplimiento, o que esté suspendida por sobregiro. Si el formulario permitiera una operación para una cuenta no habilitada, podría ser muy grave. En estos casos, hay que buscar el número de cuenta en el archivo maestro. Si está habilitada, no hay problema. En caso contrario, no se puede autorizar al cliente a operar.

Acceso condicionado. Hay veces en que el llenado de un campo depende del contenido de otro campo. Por ejemplo, sea un campo *Tipo* para indicar si una venta es al contado o a crédito y un campo *Cuotas* para ingresar la cantidad de cuotas. Si se ingresa venta al contado y un número de cuotas, es un error. Para impedirlo, el formulario puede deshabilitar el campo *Cuotas* si en *Tipo* se ha ingresado al contado; y habilitarlo en caso contrario. Otro ejemplo: al llenar los datos de los empleados, un campo *Estado* se codifica como 1 (Soltero), 2 (Casado), 3 (En Pareja), 4 (Divorciado), 5 (Viudo). En otro campo *Cónyuge* se escribe el nombre del cónyuge o pareja. Al dar de alta un empleado, si *Estado* se codifica como 1, 4 o 5, el formulario deshabilita el campo *Cónyuge* automáticamente, para impedir que sea llenado.

Dígitos verificadores. Una forma de validación muy eficaz de valores numéricos es usar *dígitos verificadores*. A un valor numérico dado se le asocia un dígito único, que se calcula mediante un procedimiento. El procedimiento es arbitrario; pero, una vez elegido, debe aplicarse siempre del mismo modo.

Supongamos que queremos calcular el dígito que corresponde a la cuenta bancaria 537847. Las cuentas pueden tener hasta seis posiciones. El procedimiento

ideado consiste en los siguientes pasos.

1. Definir cómo ponderar cada posición de la cuenta bancaria. Para ello se da a cada una de las seis posiciones un peso o *factor de ponderación* o *ponderador*: las unidades, decenas, centenas, etc. pesarán respectivamente 2, 3, 4, 5, 6 y 7. La ponderación es esencial a cualquier procedimiento para calcular dígitos verificadores.

Posición	Ponderador
centenas de mil	7
decenas de mil	6
unidades de mil	5
centenas	4
decenas	3
unidades	2

2. Multiplicar cada posición de la cuenta por su factor de ponderación. Observe que en la cuenta bancaria hay dos 7, pero que adquieren ponderaciones distintas, una vez 35 y otra 14.

Cuenta	Ponderador	Ponderación
5	7	35
3	6	18
7	5	35
8	4	32
4	3	12
7	2	14

3. Sumar las ponderaciones: 35 + 18 + 35 + 32 +12 + 14 = 146

4. Tomar el valor de las unidades de la suma anterior como dígito verificador. En este caso, el 6 de 146. El dígito verificador se asocia al número de cuenta: 537847–6.

El procedimiento se aplica a todos los números de cuenta, con lo que cada uno tendrá su propio dígito verificador. Una vez otorgado el dígito a una cuenta particular, cada vez que se la capture o transcriba se deberá incluir el dígito. En estas oportunidades se vuelve a calcular el dígito y se lo compara con el dígito ingresado. Si son diferentes es porque el número de cuenta ingresado, el dígito ingresado o ambos están incorrectos.

Probemos. Un error frecuente es transponer las cifras del número. Si el procedimiento es efectivo, deberá detectar este error. Sea que se introduce erróneamente el número anterior como 538747–6.

Cuenta	Ponderador	Ponderación
5	7	35
3	6	18
8	5	40

7	4	28
4	3	12
7	2	14
	Suma	147
	Dígito	7

En el ejemplo, el dígito calculado, 7, no coincide con el ingresado, 6, indicando un error. La detección de éste se debe a que la inversión de las cifras origina otras ponderaciones.

Como queda dicho, el procedimiento elegido es arbitrario. Así, para obtener el dígito verificador del ejemplo anterior, en vez de tomar el número que está en la posición de las unidades de la sumatoria de las ponderaciones, podríamos sumar los números componentes de ella repetidamente, hasta obtener un único número. El resultado 146 se trataría sumando 1 + 4 + 6, lo que da 11. Se aplicaría otra vez la suma de 1 + 1, obteniendo 2, que sería el dígito verificador. Otra forma podría ser sumar las posiciones pares, por un lado y las impares, por el otro; luego se podría hacer la diferencia entre ambas sumas y tomar el último dígito del resultado, o sumar repetidamente los componentes hasta obtener un único número, etc.

Consistencia. Sea un registro donde el dato de un campo es aceptable y el dato de otro campo también es aceptable. Pero puede ser que estos datos son inconsistentes entre sí. Por ejemplo, en una facultad se registran los datos personales de cada alumno en un archivo donde hay dos campos de fecha, *Ingreso* y *Egreso*. Por sí sola, *Ingreso* es válida si está en el período fecha de inicio de la facultad y última fecha de inscripción. Por su lado, *Egreso* debe estar entre fecha de primer egreso y fecha de último egreso. Sin embargo, aunque ambas por sí sola sean válidas, no se debe aceptar que *Egreso* sea anterior a *Ingreso*. Otro ejemplo, la casa central de una empresa tiene código 0 y la única sucursal código1. Si la casa central tiene los sectores A, B y C, y su sucursal los sectores D y E, no se puede aceptar como correcto una entrada de casa central con sector E.

Integridad referencial. Los sistemas de bases de datos tienen diversos componentes. El más importante son las tablas, porque en ellas se guardan los datos capturados y a partir de ellas se obtienen los demás componentes, como consultas, formularios, informes. Los datos de un tema se distribuyen en distintas tablas. Por ejemplo, los datos de un alumno se guardan en un tabla maestra de alumnos, en una tabla de regularidades vigentes y en una tabla de exámenes rendidos. Esta distribución permite administrar mejor distintos aspectos del alumno: datos personales, cursado y exámenes. Sin embargo, los datos totales de un alumno, distribuidos en tres tablas, necesitan ser resguardados para que se conserven íntegros. Para lograrlo se construyen relaciones entre las tablas, valiéndose de un campo común en las tres, llamado clave. En este caso, la clave es la matrícula. En la tabla *Alumnos*, una matrícula no se puede repetir en distintos registros. En las tablas *Regularidades* y *Exámenes*, la matrícula sí puede repetirse, porque un mismo alumno puede tener diversas regularidades y diversos exámenes. Una relación se crea desde la matrícula de *Alumnos* a la matrícula de *Regularidades* y la otra relación se crea desde la matrícula de *Alumnos* a la matrícula de *Exámenes*.

La integridad referencial, que viene incluida en el soft y sólo necesita ser puesta en vigencia, contempla la integridad de datos cuando se dan altas, bajas y cambios.

ALTAS. Se aplica a las tablas hijas. No se pueden incorporar nuevos registros en *Regularidades* ni en *Exámenes* con una matrícula que no exista en *Alumnos*, porque se producirían registros huérfanos en las dos primeras tablas. Tampoco se puede modificar una matrícula en *Regularidades* ni en *Exámenes* por una matrícula sin correspondiente en *Alumnos*, porque se produciría el mismo efecto.

BAJAS. Se aplica a la tabla madre. Hay dos casos posibles, de los cuales sólo se puede elegir uno. Caso 1: No se puede eliminar un registro de *Alumnos* que tenga registros correspondientes en *Regularidades* y/o en *Exámenes*, porque quedarían huérfanos. Caso 2: Se puede eliminar un registro en *Alumnos*, siempre que se eliminen automáticamente los registros correspondientes en *Regularidades* y *Exámenes*.

CAMBIOS. Se aplica a la tabla madre, cuando se modifica la clave. Hay dos casos posibles. Caso 1: No se puede cambiar una matrícula de *Alumnos* con registros correspondientes en *Regularidades* y/o en *Exámenes*, porque quedarían huérfanos. Caso 2: Se puede cambiar una matrícula de *Alumnos*, siempre que se cambien automáticamente la matrícula de los registros correspondientes de *Regularidades* y *Exámenes*.

La implementación de integridad referencial es fácil y garantiza que no se produzcan errores, que son denunciados automáticamente por el soft sin necesidad de programación especial.

Validación posterior

Consiste en calcular totales o conteos de los datos antes de capturarlos, que se compararán con los totales o conteos de los datos una vez capturados o transcriptos. Suponga, por ejemplo, que el sector contable de una empresa prepara manualmente formularios en papel de minutas, que luego deben ser transcriptos a la tabla *Minutas*, usando un programa *A*. Cada formulario lleva un número de hoja y al pie se escribe la cantidad de renglones llenos y las sumas de los códigos de cuenta, de los débitos y de los créditos. Después de capturar los datos, un programa *B* da un resumen, donde indica, para cada hoja, la cantidad de renglones y las sumas. Si todo se ha hecho bien, los cálculos deben coincidir; de lo contrario, habrá que buscar los errores.

Por ejemplo, sea un formulario para comunicar las altas de minutas a contabilizar en un sistema informatizado.

Minutas a contabilizar Fecha __/__/__ Hoja ____

Minuta	Cuenta	Descripción	Débito	Crédito

Total Líneas	Total Cuenta		Total Débito	Total Crédito

Quienes llenan las minutas a mano, suman los campos numéricos y cuentan las líneas, por cada hoja, escribiendo los totales al pie de cada hoja. Cada línea dará origen a un registro en la tabla *Minutas*.

La tabla *Minutas*, donde se dan de alta registros capturados del formulario anterior, tienen los campos *Fecha*, *Hoja*, *Minuta*, *Cuenta*, *Descripción*, *Débito* y *Crédito*. Terminada la transcripción con el programa *A*, el programa *B* produce un informe con los totales y la cantidad de registros de cada hoja. Estos totales sirven para una validación posterior a la captura, contra los formularios fuente. El informe para el control puede tener un aspecto como el siguiente.

Totales de control de movimientos contables Fecha 12/10/03 Hora 10:45

Hoja	Fecha	Registros	Cuentas	Débitos	Créditos
1	10-10-00	30	5.745.633	214.500,00	160.200,00
2	10-10-00	24	981.328	15.500,00	69.800,00
3	11-10-00	18	185.429	18.300,00	18.300,00

Habrá que puntear estos totales contra los correspondientes de cada hoja. Si hay errores de transcripción, se facilita la corrección porque se sabe en qué hojas se encuentran. Nótese que, aunque sumar números de cuenta parece no tener sentido, sí lo tiene a efectos del control.

Ordenamiento

Esta actividad consiste en disponer un conjunto de objetos por alguna característica común, siguiendo una secuencia ascendente o descendente. Ordenar es útil para encontrar un objeto de modo más rápido. Por ejemplo, para saber lo que vale un artículo cuyo código se conoce, si el archivo está ordenado por código la búsqueda será más rápida que si no tuviera ese orden. El ordenamiento a veces es necesario para realizar determinados cálculos. Por ejemplo, si queremos saber cuánto ha comprado cada cliente, debemos ordenar el archivo de facturas por cliente.

En sistemas manuales el ordenamiento puede ser arreglar facturas por fecha, o por cliente, o por cliente y fecha, etc.; o arreglar legajos de empleados por número de legajo o nombre de empleado; etc. Es una actividad lenta y trabajosa, porque significa colocar físicamente los objetos que se ordenan en el orden que les corresponde. Una forma mejorada, aunque consume mucho tiempo, es la que se usa en algunas bibliotecas. Cada libro se coloca en un estante y anaquel, siguiendo un criterio. Cada estante y anaquel tienen un código “topográfico”. Luego se crean distintos índices, por ejemplo, por título, autor, materia, etc., donde cada índice hace referencia al código topográfico. Luego, si se quiere buscar *Borges, Jorge Luis*, se recu-

rre al índice de autores. Habrá una entrada por cada libro de *Borges*, cada una indicando en que estante y anaquel está. Si se quiere buscar temas de *Estadísticas*, se recurre al índice de materias y se tendrán todos los libros disponibles de ese tema, cada uno indicando dónde está. Los libros se prestan a los lectores; y cuando son devueltos, se debe tener cuidado de ubicarlos en el lugar que les corresponde.

En sistemas computarizados el ordenamiento consiste en secuenciar los registros de una tabla por uno o más campos o por expresiones construidas con los campos. Por ejemplo, podemos ordenar los registros de la tabla *Alumnos* por los campos *Matrícula* o *Nombre* o *Carrera*. También podemos ordenar por más de una campo, como por *Facultad* y *Nombre*, o por *Sexo* y *Matrícula*, o por *Facultad*, *Carrera* y *Nombre*, etc. El ordenamiento no altera el orden físico de los registros, sino que se hace a través de índices, que se construyen y mantienen fácil, automática y velozmente.

Cálculo

Es someter datos numéricos a operaciones aritméticas o lógicas. Los cálculos aritméticos se hacen mediante operadores aritméticos (+ - * / **) o funciones, algunas de las cuales realizan cálculos trigonométricos, estadísticos, financieros, de fechas, etc. Los cálculos lógicos usan operadores lógicos (y, o, no) y relacionales (>, <, =, etc.), y funciones, obteniendo como resultado *verdadero* o *falso*. En computadoras, la potencia del cálculo es enorme.

Almacenamiento

Consiste en guardar los datos para conservarlos seguros y recuperarlos cuando se necesiten. El almacenamiento manual, tedioso y lento, es por ejemplo archivar documentos. En computadoras se usan dispositivos de gran capacidad y velocidad, como discos duros, amén de dispositivos masivos para conservar y transportar datos, como CDs u pendrives. Cuando se necesita encontrar un dato, se emplea un tiempo. Mientras menor sea ese tiempo, se dice que hay mayor *velocidad de recuperación*. Esta es mayor con computadoras que manualmente, pero en ambos casos es mayor si los datos están ordenados. Si no hay orden, la búsqueda es lenta; si lo que se busca no está, sólo se sabrá al final, luego de haber inspeccionado la totalidad de registros, documentos, etc. Habiendo orden, se sabe fácilmente si el registro o documento está o no, porque basta buscarlo en el lugar donde debería estar.

El almacenamiento de cierta documentación, como la comercial, registral o médica, es una obligación legal. Esto implica mucho trabajo y dinero. Por suerte hay empresas que se encargan del almacenamiento sistemático y confidencial de la documentación, reduciendo costos. Por otro lado, documentos cuyas copias había que almacenar en papel, ahora se almacenan en archivos de tipo PDF.

Comunicación

Es el intercambio de datos o información entre dos o más personas, máquinas, instituciones. Conversaciones cara a cara, teléfonos, correspondencia postal, faxes, redes de computadoras, satélites de comunicación, correo electrónico, etc., son ejemplos de medios para realizar esta actividad.

Producción de informes

Consiste en la elaboración de datos para brindar información a los usuarios de un sistema. Esta actividad es el fin último de los sistemas de información. Las formas

más comunes son combinación de correspondencia, tabulados y gráficos.

Tipos de sistemas de información

Se han intentado distintas clasificaciones de los sistemas de información, no siempre claras. Esto se debe a que se suelen mezclar distintos criterios. Veamos los tres más clásicos.

Sistemas transaccionales

Estos sistemas son empleados por el nivel operativo durante el largo plazo. Las transacciones son lo más rutinario y voluminoso de una institución y siguen procedimientos bien establecidos, donde se define con exactitud cada paso. Lo que escapa a lo establecido se resuelve por vía de excepción: si un empleado no puede tomar una decisión porque no está autorizado, debe elevar un informe de excepción, verbal o escrito, para que un superior dé solución al problema.

La alta estructuración de las actividades y el gran volumen de transacciones hizo que estos sistemas fueran los primeros en ser computarizados. La estructuración permitía traducir los procedimientos y decisiones a programas de computadora, aprovechando la capacidad de estas máquinas para procesar con rapidez grandes volúmenes de transacciones. Estos sistemas siguen siendo aquellos a los que se dedica más esfuerzo.

En ocasiones, es posible comprar sistemas ya hechos, especialmente en lo referido a software. En este caso, hay poco lugar para el analista, salvo en la evaluación de la eficiencia del sistema y aspectos complementarios. Pero como cada institución tiene características muy particulares, no hay sistema prefabricado que se adapte al cien por ciento de las necesidades. Estas particularidades suelen hacer necesaria la intervención del analista, sea para los aspectos manuales, sea para los aspectos informatizados de los sistemas.

Sistemas gerenciales

Son sistemas que producen información necesitada por los gerentes para administrar. Los gerentes tienen necesidad de dos clases de información. Una es predecible y se refiere a la actividad de los sectores a su cargo, mediante la cual controlan los resultados logrados, corrigen los desvíos y autorizan o no ciertas operaciones. La otra no es predecible, como la referida a las nuevas tendencias de los clientes, la competencia, los proveedores, nuevos productos y servicios, etc., que afectan a las áreas que administran.

La información predecible se refiere a *resúmenes* e *informes de excepción*. Ejemplo del primer tipo son los resúmenes diarios, semanales o mensuales de ventas, en pesos o unidades, por vendedor, por cliente, por artículo, por sección. Para ejemplificar el segundo tipo, sea un cliente que desea comprar una cantidad importante de un artículo con stock insuficiente, cantidad que el vendedor no está autorizado a vender. Como la venta es excepcional, el empleado debe recurrir al gerente, quien podrá autorizarla o no, según que se pueda o no cumplir con el cliente.

Los resúmenes son producidos por los sistemas transaccionales, por lo que son un tema de análisis y diseño. Si se quiere automatizar el manejo de las excepciones, éstas también son tema de análisis y diseño. En el ejemplo del vendedor que no puede vender por falta de existencia, el programa de facturación podría impedir que el vendedor haga la operación, quedando a cargo del gerente destrabar el impedimento mediante una clave.

La información no predecible, por ser variable, no permite establecer de antemano sistemas de información a largo plazo. Personal capacitado puede desarrollar sistemas de corto plazo y poca envergadura. El mismo gerente, con una capacitación adecuada, puede producir la información necesitada, preservando la confidencialidad.

Ejemplos de soft que ayuda a los gerentes son las planillas de cálculo, que consolidan datos, los grafican, los presentan en tablas y gráficos dinámicos y responden a cuestiones del tipo “qué pasa si”, que mediante fórmulas muestran resultados posibles variando el valor de las celdas intervinientes. Otro ejemplo en las bases de datos son las consultas, que obtienen resultados de gran complejidad a partir de las tablas maestras y transaccionales. Es importante que los gerentes sepan manejar las variedades mencionadas de soft, sin necesidad de terceros.

Sistemas directivos

En el nivel directivo también se da información predecible y no predecible. La primera son resúmenes a nivel más global de las distintas gerencias y excepciones planteadas por ellas. La información no predecible proviene principalmente del entorno, es la que más incide en este nivel y es a la que menos pueden satisfacer sistemas a largo plazo.

Muchas de las decisiones de los directivos son trascendentes y de riesgo, como ampliar actividades, comprar nueva maquinaria, abrir una sucursal o aumentar el capital social. Para reducir el riesgo se necesita información adecuada. Los sistemas de la institución pueden brindar parte de los datos necesarios, pero siempre habrá que recopilar datos de los que se carece y procesarlos en sistemas de corto plazo, desarrollados ad hoc para un problema.

Método para el análisis y diseño de sistemas

Consiste en una serie de etapas para obtener un nuevo sistema en funcionamiento. La totalidad de etapas aplicadas a una situación particular constituye un *proyecto*, cuya duración puede abarcar semanas, meses o incluso años. Cuando el proyecto es grande, se puede dividir en subproyectos, los cuales son desarrollados por distintos analistas. Las etapas, brevemente, son las que siguen.

Investigación preliminar. Consiste en una investigación exploratoria para saber cuál es el problema a resolver y en el planeamiento de las actividades y recursos del proyecto.

Análisis. También conocida como determinación de requerimientos, consiste en investigar el sistema actual y detectar las necesidades de información. Es una investigación descriptiva, consistente en recolectar hechos, analizarlos y sacar conclusiones. Las conclusiones son los *requerimientos* que debe satisfacer el nuevo sistema. Si no existe el sistema actual, porque el que se va a diseñar será el primero a crear, aún habrá necesidades que satisfacer, de modo que la etapa es necesaria.

Diseño. Es la planificación detallada del nuevo sistema, de forma que satisfaga los requerimientos establecidos en la etapa anterior. En sistemas manuales, el nuevo sistema queda materializado en esta etapa. En sistemas informatizados, el nuevo sistema queda diseñado aquí, pero no materializado: el analista es el arquitecto, los programadores son los constructores.

Desarrollo de software. Se aplica cuando se trata de sistemas informatizados. No es tema de los analistas, sino de los programadores, que siguen las pautas establecidas por aquéllos en la etapa de diseño. El software materializa el nuevo sistema.

Pruebas. Consiste en hacer funcionar el nuevo sistema bajo situaciones simuladas, con datos correctos y erróneos. Si hay funcionamientos incorrectos, se introducen los ajustes necesarios.

Implantación. Consiste en poner en vigencia el nuevo sistema, una vez probado con éxito. La implantación puede ser *paralela*, *paulatina* o *total*. En la implantación paralela se hace trabajar simultáneamente el sistema actual y el nuevo, para comparar resultados. En la paulatina se van reemplazando módulos del sistema viejo por el nuevo. En la total el reemplazo es de una sola vez. Las dos primeras modalidades dan tiempo para introducir ajustes de último momento, lo que no es posible en la última. En cualquiera de las tres es necesario capacitar a los usuarios directos en la operación del nuevo sistema.

Investigación preliminar

El proceso de análisis y diseño se inicia a pedido de una persona o un grupo, generalmente directivos. La petición origina una investigación preliminar, primera etapa del método clásico, que descompondremos en tres partes: determinación del objetivo, determinación de factibilidad y planeamiento.

Determinación del objetivo

Lo primero que debe hacer el analista es entender para qué lo llaman. Quien recurre a él pocas veces expresa sus necesidades y problemas de información con claridad. El analista, entonces, tiene que iniciar una investigación preliminar para precisar cuál es el problema a investigar y resolver. Esto es lo que se conoce como determinación del objetivo del análisis y diseño.

Determinar el objetivo es fundamental, porque establece: (1) lo que hay que hacer y (2) las obligaciones del analista y del peticionante. El analista queda obligado al logro del objetivo, no a más. El peticionante no puede exigir más de lo establecido en el objetivo. Conviene que este aspecto contractual del análisis de sistemas sea formalizado por escrito. Esto es corriente cuando el analista es ajeno a la institución. Si el analista es un empleado, es preferible que la orden se dé por escrito, particularmente cuando la tarea a realizar sea de largo aliento.

La investigación preliminar es de naturaleza exploratoria. Como tal, carece de estructura. Sin embargo, a medida que avanza, el analista va orientando sus indagaciones a temas cada vez más definidos. Para recolectar datos, las técnicas adecuadas son la entrevista, la observación directa y el estudio de datos secundarios. El cuestionario rara vez es útil, por ser un instrumento muy estructurado.

Normalmente el objetivo involucra el estudio de varios aspectos. La investigación preliminar debe descubrirlos, para estimar cuánto trabajo, tiempo y recursos serán necesarios. La investigación preliminar debe ser realizada con rapidez. El criterio para terminarla es cuando la información adicional que se obtenga de los diferentes aspectos redunde con la información ya obtenida.

Determinación de factibilidad

Con la información obtenida en la investigación preliminar hay que determinar si es factible un nuevo sistema. Hay tres aspectos de factibilidad a considerar: técnica, económica y social.

Factibilidad técnica

Se refiere al conocimiento y a la tecnología.

Conocimiento suficiente. No todos los pedidos a un analista pueden ser satisfechos por él, porque involucran conocimientos que no tiene ni puede conseguir de otro profesional. Por ejemplo, en un sistema que aplica complejos cálculos matemáticos o financieros, si el analista desconoce las ciencias respectivas y no puede integrar a un especialista, seguramente fracasará. El analista puede ocuparse de sistemas siempre que tenga conocimientos suficientes o pueda adquirirlos en un tiempo que no afecte la tarea encomendada.

Disponibilidad de tecnología. Si el nuevo sistema requiere de una particular tecnología, está condicionado por la disponibilidad de ella. Si no es posible adquirirla, el sistema será impracticable. La tecnología implica no sólo su adquisición, sino tam-

bién su mantenimiento. Por ejemplo, si la empresa quiere que ciertas salidas del sistema vayan a microfilmes, será posible siempre que los equipos, repuestos e insumos se puedan comprar en la región y el servicio técnico se brinde dentro de las 48 horas.

Factibilidad económica

Es la estimación de beneficios y costos, realizada por el peticionante. Lamentablemente no es posible cuantificar todas las variables. Los únicos valores monetarios conocibles son los del nuevo sistema, cuando media un presupuesto, y el precio del equipamiento necesario. Obviamente, la factibilidad económica no se estima una sola vez, sino en distintas oportunidades durante el desarrollo del proyecto. Se pueden usar diversas relaciones, como las que siguen.

Beneficios del nuevo sistema contra costos del nuevo sistema. Si los beneficios del nuevo sistema son mayores a sus costos, se puede avanzar en su desarrollo. Los beneficios son estimaciones de las ventajas que traerá el nuevo sistema, como controles e información no considerados por el sistema actual, nuevos y mejores servicios a terceros, más rapidez, mayor seguridad, menos desperdicio, menos transcripciones, etc. Los costos son los correspondientes a las tareas de análisis y diseño, desarrollo y adquisición de software, inversión en máquinas y capacitación de usuarios.

Costos del sistema actual contra costos del nuevo sistema. Si los costos por los inconvenientes y oportunidades perdidas que provoca el sistema actual son mayores a los costos del nuevo sistema, se puede avanzar en el desarrollo de éste.

Factibilidad social

Los usuarios directos e indirectos pueden estar a favor o en contra de un nuevo sistema, por razones individuales o de grupos. Que el sistema nuevo tenga éxito dependerá de la aceptación de este conjunto social. El analista puede tener indicios de estas actitudes, logradas en la investigación preliminar; pero es el peticionante quien conoce mejor este aspecto. Si el sistema actual provoca quejas, rechazos, demoras y otros inconvenientes, la probabilidad de que un nuevo sistema sea aceptado es mayor. Si hay personas que perciben como amenaza un nuevo sistema, porque los hará prescindibles, les obligará a aprender cosas nuevas, serán mejor controlados, etc., encontrarán mil maneras de boicotearlo.

Planeamiento

Cuando queda bien definido el objetivo de la investigación, se puede planificar el proyecto. La planificación consiste en determinar las tareas a realizar, fijarles duración, encadenarlas según relaciones antes–después, asignarles costos y recursos. Aunque en nuestra cultura no tenemos la costumbre de planear en detalle y con precisión, es recomendable hacerlo, por los resultados que brinda. Si se producen desvíos durante el desarrollo del proyecto, habrá que reajustar la planificación. De esta forma, será una poderosa herramienta para administrar el proyecto.

Cuando el análisis se contrata externamente, los analistas pueden confeccionar el presupuesto definitivo a partir de la planificación. Puede que hayan debido presentar un presupuesto provisorio antes de ella, urgidos por el peticionante para evaluar la factibilidad económica. El presupuesto anticipado puede ser poco realista y des-

ventajoso para alguna de las partes. Para evitar pérdidas al analista en caso que el peticionante desista, aquél puede cobrar honorarios por actividades anteriores al planeamiento.

Análisis (Determinación de requerimientos)

Esta etapa consiste en una investigación descriptiva del sistema actual, mediante hechos, para conocerlo en profundidad e inferir cuáles son los requerimientos que debe satisfacer el nuevo sistema.

Hechos

Para llegar a un sistema mejor que el actual, debemos conocerlo adecuadamente. Sintéticamente, debemos saber qué hace bien, qué hace mal, qué le sobra y qué le falta. El nuevo sistema deberá incluir lo que hace bien, hacer bien lo que hace mal, eliminar lo que sobra, incorporar lo que falta. Mientras mejor se conozca el sistema actual, mejor será el nuevo sistema. La única forma de conocer adecuadamente el sistema actual es mediante hechos. Recordemos que un hecho es un dato verificado. La despreocupación por obtener hechos produce muchos inconvenientes, que podemos resumir como sigue.

Conocimiento incierto del sistema actual. Si los datos no son verificados pero se los toma como ciertos, el conocimiento del sistema actual no será cierto.

Requerimientos inciertos. El nuevo sistema resultará de requerimientos que se deducen del sistema actual. Pero si éste no se conoce con certeza, los requerimientos serán inciertos, lo que llevará a diseñar un nuevo sistema sobre bases inciertas.

Desperdicio. El trabajo del analista basado en opiniones tendrá poco valor, con lo cual se habrá malgastado tiempo y dinero.

Con respecto a la calidad de hechos a recolectar, deben estar en función del objetivo del proyecto, es decir, deben ser *pertinentes*. Hay que recolectar no cualquier hecho, sino aquéllos que contribuyan a conocer el sistema actual. Con respecto a la cantidad de hechos a recolectar, deben ser *suficientes*. Esto significa que no deben ser menos de los necesitados para conocer el sistema, pues el conocimiento sería incompleto; pero tampoco más, pues el costo de conseguirlos aumentaría innecesariamente, sin aportar conocimiento relevante. Acá vale el criterio que se aplica en la investigación preliminar: detener la investigación de un aspecto cuando la información adicional que se obtenga redunde con la información ya obtenida.

Para recolectar los hechos se puede aplicar cualquier técnica de recolección de datos o combinaciones de ellas. El tipo de hecho determina la técnica a usar.

Hechos a buscar

No hay una forma estipulada para buscar hechos, pero conviene comenzar por las salidas. Identificadas éstas, se pueden identificar los procesos que las producen, las entradas que brindan los datos y los archivos donde se almacenan éstos.

Salidas. La forma más habitual de las salidas son los tabulados, sean impresos o en pantalla. Otras formas también usuales son gráficos y textos personalizados. Muchas de las salidas que produce el sistema actual son “naturales”, por lo que resulta fácil conseguirlas. En un sistema de ventas, por ejemplo, seguramente hay facturas, listas de precios, resúmenes de ventas en cantidades y pesos. Otras salidas, que usan confidencialmente los jefes, son más difíciles de obtener. Sea para ejemplificar un informe mensual que usa el gerente de ventas, donde se resume por cada em-

pleado cuántos días trabajó, cuánto y cuáles rubros vendió, cuántas devoluciones y reclamos hubo. El informe es confidencial, porque sirve para calificar a cada vendedor. La manera de conocer estas salidas confidenciales es que el analista cuente con la colaboración efectiva y motivada de quienes las usan, para que las hagan conocer.

Observando las salidas, se puede saber si sus aspectos formales son adecuados y se pueden reconocer los datos utilizados para producirlas. Pero su validez funcional debe evaluarse en relación a las necesidades de los usuarios. Ellos son la clave para saber si las salidas actuales sirven o no, si deben modificarse y si se necesitan otras, ahora inexistentes. Con respecto a estas últimas, el analista y los usuarios afectados deben definir su contenido. Esto permitirá conocer los datos y establecer los procesos que las producirán.

Procesos. Como las salidas resultan de procesos, conociendo las salidas actuales se determinan los procesos generadores. Los procesos son un secuencia de pasos y pueden ser manuales o automáticos. Por procesos manuales queremos decir procedimientos de oficina, representables por cursogramas. Los procesos automáticos están contenidos en programas de computadora. El analista debe entender bien los procesos actuales, porque no son evidentes como las salidas y pueden ser complicados. Esto hará que pueda perfeccionarlos o eliminarlos en el diseño del nuevo sistema. Las salidas inexistentes actualmente, pero necesitadas, van a requerir nuevos procesos en el nuevo sistema.

La obtención de hechos sobre procesos debe centrarse en aquellos que sean complejos, requieran cálculos complicados y cuándo deben realizarse.

Entradas. Hay que determinar todos los datos que entran al sistema actual y cuáles documentos los soportan. Las salidas ayudan a encontrar la mayoría de esos datos, pero no todos. Muchos datos de salida son simples transcripciones de los datos de entrada, tal como fueron capturados; por ejemplo los nombres y domicilios de los clientes. Otros datos de salida resultan de cálculos con datos de entrada, como el total vendido en una semana, que es la suma de los productos entre cantidad y precio de cada factura. A veces se encuentran datos ingresados al sistema actual pero que no tienen efecto, porque no producen salidas ni intervienen en los procesos. Son datos cuya captura es inútil y deberán ser eliminados en el nuevo sistema.

Los soportes de datos de entrada normalmente son formularios, en papel o en pantalla. Ambos se deben examinar en sus aspectos formales y funcionales. A veces hay soportes distintos, como tarjetas de débito o crédito, archivos proporcionados por terceros o generados por medidores de consumos, etc. Observe que estos archivos no son propios del sistema, sino datos que se capturan en archivos del sistema, por lo que son equivalentes a formularios de entrada.

Además de identificar los datos de entradas, es necesario conocer los controles a que son sometidos. Si se usan dígitos verificadores, hay que conocer la fórmula para calcularlos. El nuevo sistema no debería cambiar los dígitos ya otorgados, pues provocaría molestias innecesarias, salvo cuando el procedimiento de cálculo actual no detecte un porcentaje alto de errores.

Archivos. En sistemas computarizados, los datos de entrada se guardan en archivos a través de captura. Es fácil obtener listados de los archivos del sistema actual, sus campos de datos y los índices que emplean. Si estos archivos usan campos que guardan códigos, es necesario conocer la ley de formación y si se llenan manual o

automáticamente. Si hay campos que guardan datos de control llenados por procesos posteriores a la captura, hay que averiguar en qué consisten esos controles.

En lo posible, los archivos del sistema actual deberán ser reciclados para obtener los archivos del nuevo sistema. Esto a veces es sencillo y otras veces necesita programas complejos para convertirlos.

Circunstancias. Adicionalmente a los hechos sobre salidas, procesos, entradas y archivos, hay otros que también van a influir en el diseño del nuevo sistema. Sólo podemos dar ejemplos de tales hechos, ya que dependen de las circunstancias en que se desarrolla cada proyecto. Así, la normativa legal que regula la actividad de la empresa; la decisión ya tomada de vender por Internet, cobrar a los clientes mediante débitos bancarios automáticos o abrir nuevas sucursales, etc. Muchas veces las circunstancias son las que llevan a necesitar un nuevo sistema, sea con adaptaciones menores del sistema actual, sea con cambios radicales.

Sugerencias

Los hechos son necesarios para inferir los requerimientos del nuevo sistema, pero no las únicas fuentes de ellos. Es común que los usuarios den sugerencias, especialmente cuando hay un clima de participación efectiva. El analista debe estar abierto a ellas, porque pueden servirle para establecer requerimientos adicionales. Por ejemplo, alguien puede sugerir que las pantallas del nuevo sistema tengan leyendas bien detalladas y claras, para no tener que recurrir a manuales impresos difíciles de manejar, como sucede en el sistema actual. Considerando valioso este comentario, el analista establece como requerimiento para el nuevo sistema que brinde dos clases de ayuda: una en la barra de estado, al entrar a campos que necesiten más explicación que la etiqueta asociada; la otra, de tipo Windows, en la tecla F1. Otro ejemplo puede ser que, dado que la impresión del balance siempre es completa, no pueden rehacer una sola hoja que se rompa o ensucie. Por ello opinan que sería más económico indicar las hojas a imprimir. Esta sugerencia lleva al analista a establecer como norma que todas las impresiones, no solo el balance, puedan ser parciales o totales.

Análisis de hechos

Hay varias herramientas que ayudan a obtener hechos y organizarlos en vistas a su análisis. Entre ellas, podemos mencionar cursogramas, tablas de decisión, lenguaje estructurado, diagramas de flujo de datos, diccionario de datos, diagramas de sistemas, etc. Como la exposición de estas herramientas requiere más detalle del que conviene acá, la dejamos para más adelante.

El propósito del análisis es considerar los hechos para determinar qué del sistema actual se debe conservar, mejorar o eliminar y qué de nuevo se debe incluir en el nuevo sistema. Durante la obtención de hechos, el analista seguramente ha ido sacando conclusiones sobre estos aspectos. Sin embargo, son conclusiones al pasar e incompletas. Llega un momento en que debe concentrarse en analizar críticamente la documentación obtenida sobre el sistema actual y sobre las necesidades de información no satisfechas. Si hay lagunas en la documentación, deberá retroceder a la búsqueda de hechos, para completar, corregir o aclarar lo que haga falta. Esto retrasará el proyecto, pero será inevitable.

El análisis se aplica a todos los hechos: salidas, procesos, entradas, archivos y circunstancias. Un análisis a fondo implica la consideración de muchos aspectos. A

continuación exponemos algunos aspectos importantes a que debe someterse el sistema actual.

Calidad de información

Los procesos toman datos de entrada y producen datos de salida. Si los datos de salida son útiles a alguien para conocer o decidir, reciben el nombre de información. Este aspecto subjetivo es la característica que distingue la información. La información está referida a un usuario determinado, en circunstancias determinadas. Lo que es información para A, puede no serlo para B. Tampoco puede serlo para el mismo A, en otras circunstancias.

La utilidad de la información radica en que es la única forma de conocer el resultado de las operaciones rutinarias y de tomar decisiones adecuadas. Como las decisiones conllevan el riesgo de equivocarse y ocasionar perjuicios, es importante disminuir el riesgo, lo cual se consigue con información. Si la información es completa, la decisión se toma en condiciones de certidumbre, con riesgo nulo. A medida que aumenta la carencia de información, aumenta la incertidumbre para tomar la decisión, con riesgo creciente.

Para que la información sea de calidad, debe ser oportuna, completa y tener formato adecuado.

Oportunidad. La oportunidad consiste en que el usuario disponga de la información cuando la necesita. Pasado ese tiempo, quizás ya no sea información. Si la información del sistema actual no es oportuna, puede deberse a archivos mal diseñados o incompletos, a procesos complicados, a software pobre o mal aprovechado, a equipamiento obsoleto, etc. El analista debe tomar nota de factores como los enumerados, para evitarlos en el nuevo sistema.

Compleción. La compleción se refiere a que el usuario disponga de la toda información necesitada. Esto no quiere decir muchos datos, sino datos pertinentes a la necesidad. Si el gerente de ventas quiere conocer cuánto se vendió el mes pasado, sólo le hace falta saber el total, no el detalle de todas las ventas. El analista debe considerar con los usuarios específicos cada salida, para determinar cuáles son los datos necesitados. Esto puede haberlo hecho al recolectar hechos de salidas, pero quizás deba insistir hasta que quede claro. Muchas veces va a encontrar que los usuarios se adaptan a las salidas, completándolas con cálculos y anotaciones manuales, porque no satisfacen lo que requieren.

Formato adecuado. El formato adecuado se da cuando los datos componentes de la información están dispuestos de modo que sean fáciles de encontrar y entender. El buen formato contribuye a la utilidad de la salida, porque ahorra tiempo, facilita la comprensión, evita errores, reduce la fatiga.

Redundancia mínima

Este aspecto se refiere a que los datos se repitan lo menos posible en los archivos permanentes. Lo ideal sería que cada campo se almacene una sola vez en una sola tabla, excepto las claves. Las transacciones necesitan incluir claves externas que permitan enlazarlas con las claves internas de los registros maestros. La repetición de claves debe ser la única redundancia permitida en archivos permanentes. Los demás campos redundantes ocupan espacio innecesario. También provocan pro-

blemas de actualización, porque si se cambia el contenido de un campo, hay que actualizarlo en todos los archivos donde se repite. Finalmente exigen procesos adicionales cuyo propósito es salvar los inconvenientes producidos por el diseño inapropiado de tablas. Si entre las tablas del sistema actual hay redundancias, habrá que eliminarlas en el nuevo sistema, mediante un diseño más eficiente.

Facilidad operativa

Un buen sistema debe ser fácil de operar y debe operar con facilidad. Esto se logra con procedimientos manuales o automáticos precisos, claros y sin retrocesos innecesarios; formularios adecuados; menús racionales; manuales claros y rápidos de consultar; muebles y máquinas ergonómicos; etc. La facilidad operativa estimula el uso de un sistema; su carencia inclina a los usuarios a evitarlo. El sistema actual puede tener méritos y desventajas a este respecto. Los méritos deben potenciarse en el nuevo sistema y las desventajas eliminarse.

Control

Los controles son esenciales en los sistemas de información. Hay distintos tipos, de los que veremos algunos. Estudiar los controles que emplean los procesos actuales sirve para descubrir fallas que hay que subsanar y mecanismos seguros que deben ser mantenidos en el nuevo sistema. Sin embargo, esto no basta. La importancia de los datos y los procesos puede llevar a establecer nuevos controles futuros.

Controles de datos. Son los que hemos visto como validación simultánea o posterior, mediante códigos, rangos, consistencia, dígitos verificadores, integridad referencial, etc. Se aplican a la captura o transcripción de datos, procurando reducir el ingreso de errores al sistema. Como esos datos originales son la materia prima para la elaboración de datos calculados y de información, es vital controlarlos.

Controles de procesos. Se aplican cuando se deben realizar distintos procesos según un orden prefijado, sin omitir ninguno. En los sistemas manuales, por ejemplo, son los vistos buenos que se incorporan a un formulario, a medida que se va procesando en distintas unidades funcionales. Permiten saber si falta alguna intervención.

En sistemas computarizados, los procesos se pueden controlar mediante una tabla diseñada al efecto. Por ejemplo, sea una empresa que fabrica a pedido productos que requieren tres etapas sucesivas de diseño, fabricación y pintura. Al recibir un pedido, la Gerencia Industrial da de alta un registro en una tabla de productos en proceso, con el número de pedido, otros datos complementarios y tres campos de fecha de terminación, uno para cada etapa. Cuando el sector Diseño completa su trabajo, manda un parte a Gerencia Industrial, que llena la fecha de cumplimiento de diseño en el registro correspondiente. Cuando el sector Fabricación completa su trabajo, manda un parte a Gerencia, que llena la fecha de fabricación. Cuando el sector Pintura completa su trabajo, manda un parte a Gerencia, que llena la fecha de pintura. Si Fabricación comunicara haber terminado un producto todavía no diseñado o Pintura haber pintado un producto todavía no diseñado o no fabricado, el programa de carga debe detectar la anormalidad y no permitir llenar la fecha correspondiente. El gerente puede consultar el registro de un pedido particular para ver su estado de avance, obtener un informe de todos los productos en proceso, etc.

Controles de seguridad. Son los que permiten acceder al sistema a quienes tienen

autorización para hacerlo, impidiendo el ingreso de extraños. En sistemas manuales, por ejemplo, para ingresar al tesoro de un banco, se establece que la clave para abrir la puerta sea conocida solamente por el tesorero y el gerente general, quienes pueden tener dos llaves que en conjunto abran la puerta.

En sistemas computarizados en red se usan claves de usuario para entrar a ellos. El supervisor del sistema, que tiene acceso total a él, define a cada persona autorizada en una tabla de usuarios. La definición incluye: (1) datos del usuario; (2) una clave invariable que lo identifica, accesible al supervisor pero inaccesible al usuario; (3) una clave personal del usuario, que por razones de seguridad puede o debe cambiar periódicamente; (4) los permisos otorgados al usuario: leer archivos, modificarlos, copiarlos, destruirlos, imprimirlos, etc. Para ingresar, cada usuario debe proporcionar su clave personal, que está asociada a la clave invariable. Si hay correspondencia entre ambas, la persona puede ingresar, con los permisos que tiene otorgados.

Además de la tabla de usuarios, suele llevarse otra tabla de conexiones. Cada vez que un usuario se conecta a la red, se agrega un registro a esta tabla, llenando su clave invariable y la hora de conexión. Cuando termina su trabajo, en el registro que le corresponde se llena la hora de desconexión. Esta tabla permite saber quiénes han estado conectados un día y a una hora determinados.

Como el lector puede inferir, mantener usuarios y permisos es una tarea de administración delicada. El supervisor de este sistema debe ser cuidadosamente elegido, porque se le exige confidencialidad máxima.

Controles de responsabilidad. Son los que permiten conocer quién intervino en un proceso. En sistemas manuales, por ejemplo, son el agregado de las iniciales de quien escribió una nota o llenó un formulario, o la firma y el sello de quien dio una autorización.

En sistemas informatizados, cuando es importante controlar quién realizó una tarea riesgosa, se desarrolla un mecanismo de control más refinado. Por ejemplo, sea que se generan registros en una tabla de transferencias bancarias. Obviamente, es necesario evitar la comisión de fraudes. Sea que varios empleados que tienen permiso de realizar transferencias trabajan en el mismo horario. Si se detecta un error o una transferencia sospechosa, ¿cuál de esos empleados generó el registro? Como el sistema que maneja la red sabe qué usuario está trabajando en cada computadora, cada vez que uno de ellos da de alta un registro, el programa que se utiliza llena campos ocultos automáticamente. Estos campos son la clave invariable del usuario y la fecha y hora en que genera cada registro. Cuando se investiga un problema, el supervisor de la red puede ver el registro sospechado, quién lo llenó y a qué hora.

Controles de autenticidad. Son los que procuran que los datos, informes, credenciales, etc., no sean alterados o falsificados. Veamos algunos ejemplos. Para asegurar la autenticidad de una nota, es común el uso de membretes y la aclaración de las firmas mediante sellos. En los documentos donde se escriben importes y otros datos importantes, se pueden usar tramados para dejar en evidencia si el papel ha sido borrado; asteriscos de protección en lugar de espacios izquierdos, como en ***.**7.247,29; signos flotantes adheridos al dígito izquierdo de una cifra, como en $7.247,29 o $5.744.300,00; impresión de importes en números y en palabras; etc. En los mensajes y documentos enviados por correo electrónico, se usan firmas digitales que garantizan la autenticidad del remitente y la no adulteración del mensaje.

Controles de funcionamiento. Son los que comprueban que las distintas actividades de un sistema funcionen correctamente. No son parte constitutiva del sistema, sino que forman una auditoría sobre él. Se aplican intensamente durante la etapa de pruebas, sometiendo el sistema a datos simulados, correctos y erróneos. Debido que estas simulaciones, por meticulosas que sean, pueden no agotar todas las combinaciones de circunstancias, quizás queden casos sin probar. Estas omisiones pueden saltar como fallas una vez que el sistema está implantado. Es conveniente, entonces, que los usuarios, asesores, clientes, proveedores, etc., controlen su funcionamiento, para corregir los desperfectos. Tomar nota de las circunstancias de los desperfectos facilita la identificación de sus causas. Demás está decir que las fallas de un sistema pueden afectar severamente la información que produce, llevando a decisiones equivocadas, pérdidas monetarias, desperdicios de esfuerzos y tiempo, daños a terceros, deterioro del prestigio, etc.

A menudo, el mal funcionamiento del sistema actual motiva a los directivos a querer un nuevo sistema. Para el analista, conocer las fallas actuales de funcionamiento es un punto fuerte, porque le permite idear un nuevo sistema que las supere.

Eficiencia

No basta que un sistema sea eficaz, sino que debe ser eficiente. Se dice que es eficaz cuando logra el propósito para el que fue creado, sin tener en cuenta lo que tarda, cuesta, consume, etc. La eficacia es una apreciación cualitativa. Cuando se tienen en cuenta los insumos, se puede hablar de cuán eficiente es. La eficiencia es una comparación entre lo que se logra y lo que es necesario para lograrlo, siendo de naturaleza cuantitativa. Puede expresarse como un cociente, mediante la siguiente fórmula

$$\text{Eficiencia} = \text{Salida} \,/\, \text{Entrada}$$

Si dos sistemas A y B producen información por un valor de $100, siendo que A necesita $50 y B $20 de insumos, podemos calcular las respectivas eficiencias.

$$\text{Eficiencia}_A = \$100 \,/\, \$50 = 2 \qquad \text{Eficiencia}_B = \$100 \,/\, \$20 = 5$$

Obviamente, B es más eficiente que A. Note que, en este ejemplo, la eficiencia no tiene especie, porque tanto en el numerador como en el denominador la especie es $, que se elimina en el resultado. Si los insumos hubieran sido horas, la eficiencia hubiera tenido como especie $ / h:

$$\text{Eficiencia}_A = \$100 \,/\, 50\ h = 2\ \$ \,/\, h \qquad \text{Eficiencia}_B = \$100 \,/\, 20\ h = 5\ \$ \,/\, h$$

Note que B sigue siendo más eficiente que A, pues produce mayor valor monetario por hora. La especie del resultado, entonces, depende de lo que comparemos. Tanto numerador como denominador pueden estar expresados en cualquier especie (pesos, tiempo, trabajadores, energía, espacio, kilos, etc.).

La eficiencia es una medida que se usa mucho, aunque con distintos nombres. Por ejemplo, cuando decimos que un auto puede andar 18 Km. con un litro de combustible, estamos midiendo su eficiencia, comparando su salida (kilómetros recorri-

dos) con su entrada (combustible).

Siendo la eficiencia de naturaleza cuantitativa, sólo es posible aplicarla cuando la entrada y la salida pueden medirse o contarse. Incluso cuando sea posible, conviene medirla cuando represente relaciones significativas, porque las mediciones pueden llevar demasiado tiempo. Si los usuarios juzgan que el sistema actual "no da para más", medir eficiencias es una pérdida de tiempo.

Requerimientos

Concluido el análisis del sistema actual, se tiene suficiente conocimiento de él como para describirlo. Sin embargo, el propósito no es describir algo que va a dejar de existir, sino concluir qué características de él deben sobrevivir en el nuevo sistema y qué nuevas características debe tener éste. Algunas de las nuevas características surgen por la comprobación de su inexistencia en el sistema actual, siendo que debería poseerlas necesariamente; otras nacen de la imaginación, el conocimiento y la experiencia del analista y de los usuarios. Generalmente, éstos necesitan que el nuevo sistema no sea un remiendo del actual, sino una creación original, para circunstancias internas y ambientales quizás muy diferentes a aquéllas que dieron origen al sistema actual . Esto hace necesario la incorporación de muchas características nuevas.

La totalidad de características que debe incluir el nuevo sistema es lo que se conoce como sus *requerimientos*. Como es natural, hay requerimientos que pueden haber sido intuidos en etapas previas de la investigación. Por ejemplo, si en la recolección de hechos el analista descubrió que hay un alto porcentaje de errores al capturar o transcribir números de clientes, pudo concluir que el nuevo sistema deberá emplear dígitos verificadores para tales números. Si bien esta intuición puede ser útil, no todo requerimiento puede ser determinado así, ni es conveniente que lo sea. Las intuiciones circunstanciales y los juicios rápidos pueden distorsionar o sesgar la investigación y no siempre dan buenos resultados. Los requerimientos deben resultar del análisis cuidadoso de los hechos y serán el nexo con la etapa de diseño.

El analista deberá tener presentes los requerimientos cuando diseñe el nuevo sistema. Hay requerimientos comunes a cualquier sistema, de modo que no es necesario registrarlos. Otros, específicos al sistema a diseñar, deben ser anotados, para no olvidarlos.

Requerimientos comunes

Los sistemas deben cumplir con ciertos requerimientos comunes, de modo que son una guía para cualquier analista. Veamos algunos.

Calidad. El nuevo sistema debe ser valioso en cuanto a calidad de información, redundancia mínima, facilidad operativa, controles suficientes y eficiencia. Mientras mejor cumpla estos aspectos, más calidad tendrá el nuevo sistema en sí mismo y más calidad tendrá el servicio que preste a los usuarios directos e indirectos. Las consideraciones realizadas al analizar los hechos del sistema actual contribuyen a aplicar este criterio.

Flexibilidad. Los cambios internos y externos de la institución pueden ser radicales, convirtiendo en obsoleto el sistema actual de un momento para otro. Pero la mayoría de los cambios son menores, de modo que el nuevo sistema debe ser lo suficientemente flexible para poder adaptarse a ellos. Veamos un ejemplo. Sea un proceso de

un sistema que liquida un impuesto. Si las escalas de montos y tasas se introducen como instrucciones de programación, cuando se produzca un cambio en las escalas el proceso el programa quedará inutilizado. Esto se da porque el sistema tiene una rigidez en tal proceso. En cambio, si las escalas se registran en una tabla que puede ser modificada en otro proceso, los cambios de escala obligarán a modificar esa tabla, pero no afectarán el proceso de liquidación. Este sencillo recurso ha introducido flexibilidad, permitiendo que el sistema resista el golpe. Resulta obvio que, mientras más flexible sea un sistema, más vida útil tendrá. Lógicamente, la acumulación paulatina de cambios menores puede ser tanta que, llegado un momento, el sistema no resista más y deba ser reemplazado por otro nuevo.

Adaptación tecnológica. La calidad del nuevo sistema depende de la tecnología empleada. La disponibilidad de tecnología en hardware y software puede cambiar radicalmente la forma de encarar el nuevo sistema. Si la tecnología ideal tiene un precio inalcanzable, habrá que considerar la mejor tecnología dentro de la capacidad financiera de la institución, que se adecue a los requerimientos. La carencia de tecnología adecuada puede hacer imposible un nuevo sistema de calidad.

Realismo. Los requerimientos deben adecuarse a la realidad de la empresa. No es posible pensar en procesos que exijan lenguajes desconocidos por los programadores disponibles; en el manejo de software complejo por empleados cuyo nivel intelectual es insuficiente; en formas de trabajo que impongan cambios culturales improbables; etc.

Requerimientos específicos

Estos requerimientos dependen de cada sistema particular, por lo que deben anotarse para no olvidarlos al diseñar el nuevo sistema. Para ordenar su exposición, usaremos las categorías ya vistas de salidas, procesos, entradas, archivos y circunstancias.

Salidas. Hemos visto que, al considerar con los usuarios los informes producidos por el sistema actual, el analista trabajó sobre los impresos o pantallas, anotando errores, sugerencias y correcciones. Al tratar informes inexistentes pero necesarios, seguramente bosquejó con el usuario cómo deberían ser, con anotaciones sobre cómo obtener determinados resultados. Estos papeles de trabajo sirven de base para establecer los requerimientos de salida del nuevo sistema. Obviamente los deberá volver a estudiar y pasar en limpio. Si hace falta aclarar cómo se obtienen ciertos datos de estas salidas, puede incluir notas aclaratorias. Estos pre diseños le van a servir para precisar el diseño definitivo, más exacto, que usarán los programadores. Si hay que producir archivos requeridos por otras instituciones, deberá determinar sus datos y formatos, como las declaraciones de retenciones impositivas a presentar a AFIP y DGR, o las normas de ciertos bancos para cobranza de facturas.

Procesos. Los requerimientos de procesos en parte se obtienen de lo que hacen los programas del sistema actual. En el nuevo sistema estos procesos pueden sufrir fuertes modificaciones para hacerlos más efectivos, unificándolos, dividiéndolos, asignándolos a programas diferentes. Pueden necesitarse nuevos procesos para producir nuevas salidas o controlar aspectos ignorados en el sistema actual. El analista debe establecer principalmente en qué consistirán los controles y los métodos de cálculos, con todas las variedades de condiciones a que están sometidos. Si es-

tos controles y cálculos usan datos auxiliares, como escalas salariales, impositivas, tarifarias, debe precisar la estructura de estos datos, lo que le servirá para introducirlos como parte de los nuevos programas o como tablas externas. Para no demorar su tarea, debe considerar los requerimientos que no sean de rutina. Estos procesos deberán ser detallados al especificar los nuevos programas, para que los programadores los desarrollen. Para no olvidarlos, puede usar listas que haya ido confeccionando en experiencias anteriores.

Entradas. Se refieren a los formularios de papel y las ventanas usados para capturar o transcribir datos. Similar a lo que sucede con las salidas, los requerimientos pueden expresarse como bosquejos en limpio, con notas aclaratorias. Los procesos de validación pueden incluirse aquí o en los procesos de control. Si la validación se hace por rangos, listas de valores, dígitos verificadores u otros mecanismos, éstos son un requerimiento. Si los datos de entrada se ingresan como códigos, los códigos son un requerimiento.

Archivos. Estos requerimientos son los datos capturados que va a manejar el nuevo sistema, que se conservarán en tablas. Esos datos están entre los que posee el sistema actual más los que deban ser incorporados para producir información ahora inexistente. El analista tiene los primeros en los listados de las estructuras de las tablas actuales. Los datos nuevos surgen de la nueva información a producir y conviene listarlos. En cuanto al diseño de los registros, es una tarea reservada a la etapa de diseño.

Circunstancias. Cuando haya leyes, decretos, resoluciones, contratos, normas internas, etc., que condicionen el diseño del nuevo sistema, son requisitos que no deben olvidarse. Para ello, puede usarse directamente esta documentación; pero puede ser más práctico resumir lo esencial, evitando la búsqueda complicada en los textos originales.

Los directivos pueden querer que el nuevo sistema posea determinadas características, como el uso de claves de seguridad, o que permita vender a través de Internet, transformando en usuarios directos a los compradores; o que automatice la captura de datos de pago mediante códigos de barra; etc. Estas circunstancias son también requerimientos.

Las sugerencias de los usuarios aceptadas por el analista, forman también requerimientos para el nuevo sistema.

Herramientas del analista

La sola recolección de hechos, no los hace directamente analizables. Para apreciarlos con claridad, detectar fallas e idear soluciones, normalmente hay que describirlos mediante textos o gráficos, ordenarlos, contarlos, resumirlos, tabularlos, etc. Estas son soluciones de sentido común, que inventa el analista en cada ocasión.

Por otro lado y complementariamente, existen herramientas con propósitos específicos, que han sido desarrolladas por académicos y especialistas. Son ejemplo de ellas los cursogramas, adecuados para exponer gráficamente procedimientos manuales, y las tablas de decisión, que sintetizan procesos para realizar distintas acciones, dependiendo del estado de varias condiciones. Las herramientas tienen la ventaja de ser precisas y sintéticas. Son también versátiles, porque pueden aplicarse en varios momentos del proyecto, como se explica a continuación.

EXPOSICIÓN DE HECHOS. Permiten expresar con claridad un hecho o un conjunto relacionado de ellos, constituyendo una forma de documentarlos.

ANÁLISIS DE HECHOS. Facilitan el análisis de los hechos del sistema actual, posibilitando el hallazgo de errores, complicaciones y carencias.

DISEÑO DEL NUEVO SISTEMA. Sirven para diseñar el nuevo sistema, ya sea en forma general o en aspectos particulares.

PROGRAMACIÓN. Permiten expresar con exactitud muchas de las instrucciones que el analista debe escribir para los programadores.

En lo que sigue, veremos algunas de las herramientas. Advierta que hay muchas más que las tratadas aquí. Tenga también en cuenta que, por ser herramientas, sirven de ayuda al analista, pero no reemplazan por sí solas el esfuerzo, la agudeza ni la inventiva requeridas de este especialista.

Lenguaje estructurado

Esta herramienta consiste en usar estructuras lógicas y sintácticas para controlar la ejecución de un conjunto de acciones. Las estructuras lógicas se refieren a modos eficientes de controlar la ejecución de acciones. Las estructuras sintácticas son la traducción de las estructuras lógicas a fórmulas lingüísticas precisas, en la lengua nativa del país. Las estructuras sirven para controlar rigurosamente la ejecución de las acciones. La herramienta fue inicialmente desarrollada para lenguajes de programación, pero dada su potencia ha sido ampliada a otros ámbitos.

El lenguaje estructurado sirve para documentar y analizar procedimientos de cualquier grado de complejidad. También es útil como herramienta de diseño, para dar instrucciones a los programadores.

Estructuras lógicas

Tras años de programar computadoras, se concluyó que todos los programas pueden ejecutar sus comandos siguiendo tres modalidades. Estas fueron llamadas estructuras lógicas de *secuencia*, *selección* e *iteración*. Gran parte de las complicaciones y fallas de programación se debían a no haberlas respetado adecuadamente. A partir de allí, cambió no sólo la forma de escribir los programas, sino que fueron modificados los mismos lenguajes de programación para incluir comandos explícitos

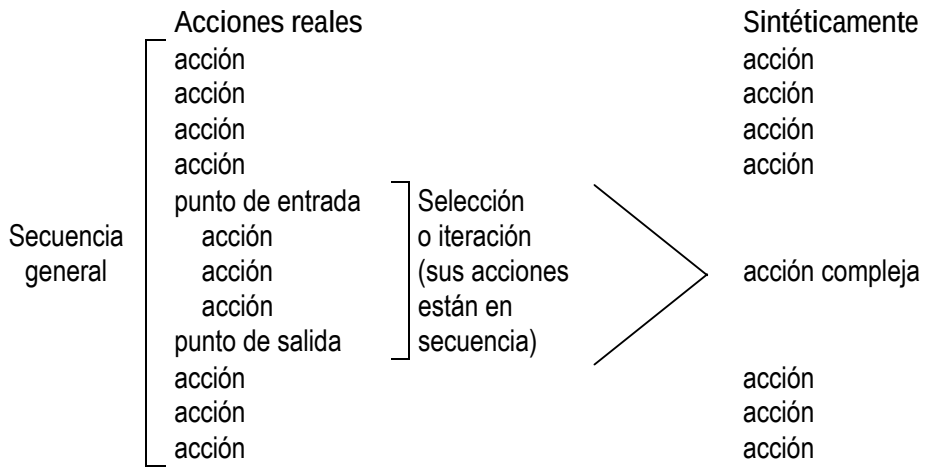
que materializaran estas estructuras. Un programa que observa las estructuras lógicas es más fácil de escribir y menos propenso a fallar, porque ellas controlan mejor el funcionamiento y facilitan hallar errores debidos a la mala ubicación de un comando. Como los comandos de programación no son sino condiciones y acciones, las estructuras lógicas pueden aplicarse también a cualquier conjunto de condiciones y acciones para realizar un procedimiento.

Secuencia. La forma básica de realizar algo es ejecutar la acción A, luego la B, luego la C, etc. La secuencia es esencial a los procesos, que por definición son una serie ordenada de pasos. Para hacer una casa, primero hay que cavar las zanjas, luego los echar cimientos, luego levantar las paredes y finalmente poner el techo. No podemos construir el techo si no hemos levantado las paredes, ni levantar éstas si no están los cimientos. La mayoría de las acciones necesariamente preceden o siguen a otras. Muy pocas pueden permutarse, como el caso de poner las ventanas, las cuales pueden colocarse antes o después de las puertas.

Selección. Hay veces en las cuales la ejecución de una acción o un grupo de ellas depende de condiciones. Por ejemplo, si el terreno está desnivelado, antes de echar los cimientos habrá que nivelarlo. Aquí, la condición "terreno desnivelado" lleva a la acción "nivelar"; es decir, el valor verdad de la condición "selecciona" la acción. También puede ser que una condición lleven a ejecutar un grupo de acciones u otro, pero no ambos. Por ejemplo, para pintar la casa, si el presupuesto de los pintores incluyendo materiales conviene más que comprar los materiales nosotros y pagar a los pintores la mano de obra, se les dejará la tarea de comprarlos y pintar; pero si es más económico que nosotros compremos los materiales, deberemos encargarnos de esa tarea y dejar a los pintores que pinten.

Iteración. Muchas acciones deben repetirse varias veces, hasta que se cumpla una condición. Si un marco no mantiene su cuadratura ni está a plomo una vez presentado en la pared, el inspector de la obra no dará la orden de fijarlo con hormigón. Mientras no esté en las condiciones requeridas, el inspector exigirá nuevos ajustes.

La selección y la iteración pueden llevar a ejecutar una o varias acciones. Si son varias, éstas también deben ser ejecutadas siguiendo una secuencia entre ellas. La selección y la iteración que incluyen varias acciones pueden considerarse como una acción compleja, pero acción al fin en la secuencia general de acciones. Para identificar dónde comienza y dónde termina una acción compleja, es necesario explicitar un punto de entrada y un punto de salida. El siguiente esquema pone en evidencia lo que decimos.



Estructuras sintácticas

Son formas normalizadas del lenguaje común, mediante las cuales se representan estructuras lógicas. La secuencia no necesita de una estructura sintáctica especial, pues basta con escribir cada acción en una línea por separado. Para la selección definiremos las estructuras sintácticas SI y CASOS y para la iteración la estructura sintáctica MIENTRAS.

SI, CASOS y MIENTRAS deben responder a un formato, es decir, a una suerte de "fórmula", de varios componentes. Los formatos siguen convenciones que indican cómo interpretar los componentes al aplicar las estructuras en un problema. Las convenciones son las siguientes.

- PALABRAS EN MAYÚSCULA. Deben escribirse textualmente al aplicarlas en un problema, sea en mayúsculas o minúsculas. Si no van entre corchetes, son obligatorias. No confunda: las mayúsculas aparecen en el formato para indicar textualidad; en la aplicación puede escribirlas en mayúsculas o minúsculas.

- <palabras entre paréntesis agudos y en minúscula>. Describen información que el usuario debe proporcionar. En la aplicación no se escriben los paréntesis agudos. Cuando aparece la forma <condición>, la condición puede ser simple o compleja. Estas últimas se construyen usando operadores lógicos (Y, O, NO) para unir condiciones simples. Para agrupar condiciones cada vez más complejas, se admite el uso de paréntesis. La forma <acción> indica una acción simple.

- [componentes entre corchetes]. Indican que todo lo que está entre los corchetes es opcional. El usuario puede o no usarlos, según lo que necesite. En la aplicación no se escriben los corchetes.

- ... Los puntos suspensivos indican que el componente al que siguen puede repetirse indefinidamente. En la aplicación no se escriben los puntos suspensivos.

Estructura sintáctica SI. Tiene el siguiente formato:

```
SI <condición>
    <acción 1> . . .
[DE LO CONTRARIO
    <acción n> . . . ]
FIN SI
```

SI identifica el punto de entrada y FIN SI el punto de salida. Cuando la <condición> es verdadera, se ejecuta <acción 1>. Como ésta va seguida por puntos suspensivos, puede haber una <acción 2>, <acción 3>, etc. Cuando <condición> es falsa, si se quiere ejecutar <acción n>, <acción n + 1>, <acción n + 2>, etc., hay que introducirlas mediante las palabras DE LO CONTRARIO. La ejecución de este segundo grupo de acciones no es obligatorio, sino opcional, lo cual se manifiesta en la estructura por los corchetes. Dado que las acciones incluidas en la estructura están subordinadas a ella, es conveniente que se escriban con mayor sangría, la cual hace patente tal subordinación. Veamos alguno ejemplos de esta estructura. Las líneas de guiones al comienzo y final indican que hay o puede haber otras acciones.

```
1   - - - - -
    si el cliente no ha pagado en término
      contar los días de mora
      calcular el recargo
      sumar el importe de la cuota y el recargo
      confeccionar el recibo, discriminando cada concepto
    fin si
    - - - - -
```

```
2     - - - - -
      si el obrero no ha faltado
        calcular el sueldo completo
        agregar el premio por asistencia
        registrar ambos conceptos
      de lo contrario
        calcular el sueldo por los días trabajados
        registrar el concepto
      fin si
      - - - - -
```

```
3     - - - - -
      si el pago está autorizado y hay dinero en caja chica
        confeccionar el recibo
        pagar
        hacer firmar el recibo
      fin si
      - - - - -
```

Observe que las acciones subordinadas a las estructuras SI comienzan un poco más a la derecha. Esta sangría indica visualmente la subordinación. Aunque la sangría no es obligatoria, da mayor claridad.

Una estructura SI puede incluir otra estructura SI, cuando la condición sea ver-

dadera o cuando sea falsa, según la necesidad del problema. En este caso, se tiene un nido de SI. El proceso puede repetirse cuantas veces se desee. Ejemplos.

```
4     - - - - -
      si el pago está autorizado
        si hay dinero en caja chica
          confeccionar el recibo
          retirar el dinero
          pagar
          hacer firmar el recibo
        de lo contrario
          confeccionar el recibo
          confeccionar el cheque
          entregar el cheque
          hacer firmar el recibo
        fin si
      fin si
      - - - - -
```

```
5     - - - - -
      si el pago está autorizado
        si hay dinero en caja chica
          confeccionar el recibo
          retirar el dinero
          pagar
          hacer firmar el recibo
        de lo contrario
          confeccionar el recibo
          confeccionar el cheque
          entregar el cheque
          hacer firmar el recibo
        fin si
      de lo contrario
        informar al interesado que falta autorización
      fin si
      - - - - -
```

```
6     - - - - -
      si el pago está autorizado
        si hay dinero en caja chica
          confeccionar el recibo
          retirar el dinero
          pagar
          hacer firmar el recibo
        de lo contrario
          confeccionar el recibo
          confeccionar el cheque
          entregar el cheque
          hacer firmar el recibo
```

```
    fin si
  de lo contrario
    si ha llegado la factura por el pago reclamado
      informar al interesado que falta autorización
    de lo contrario
      informar al interesado que aguarde el arribo de la factura
    fin si
  fin si
  - - - - -
```

Note que, cuando hay SI anidadas, cada SI debe tener su correspondiente FIN SI. Para saber qué SI se corresponde con qué FIN SI, se sigue la siguiente regla: se recorre la estructura desde el comienzo, hasta hallar un FIN SI; cuando se encuentre, éste se hace corresponder al SI inmediato anterior sin correspondiente. El proceso se repite hasta llegar al final de la estructura. Si entre un SI y su correspondiente FIN SI hay un DE LO CONTRARIO, éste pertenece a ellos. Apliquemos esta regla al último ejemplo.

```
7       - - - -
    si el pago está autorizado
      si hay dinero en caja chica
        confeccionar el recibo
        retirar el dinero
        pagar
        hacer firmar el recibo
      de lo contrario
        confeccionar el recibo
        confeccionar el cheque
        entregar el cheque
        hacer firmar el recibo
      fin si
    de lo contrario
      si ha llegado la factura por el pago reclamado
        informar al cobrador que falta autorización
      de lo contrario
        informar al cobrador que debe aguardar a que arribe la factura
      fin si
    fin si
    - - - - -
```

Todos los ejemplos anteriores han incluido acciones interiores a las estructuras SI, sugiriendo acciones externas con líneas de guiones. En situaciones más reales, SI es una acción compleja dentro de una secuencia general, como indica el siguiente ejemplo.

```
8   recibir el dinero y la boleta de depósito del tesorero
    controlar el dinero con el importe a depositar
    si el dinero coincide con el importe a depositar
```

```
   ir al banco
   depositar el dinero
   controlar sello del banco y firma del cajero en el
duplicado de boleta de depósito
de lo contrario
   comunicar al tesorero que el dinero no coincide
con el importe a depositar
fin si
continuar con las tareas habituales
```

Estructura sintáctica CASOS. Tiene el siguiente formato:

```
CASOS
    CASO <condición 1>
        <acción 1> . . .
  [CASO <condición 2>
        <acción j> . . . ] . . .
    [DEMAS CASOS
        <acción n> . . .]
FIN CASOS
```

CASOS y FIN CASOS indican los puntos de entrada y salida. La estructura es similar a SI, ya que responde a la selección, pero es más potente. SI sólo permite considerar dos situaciones: que la condición se cumpla o no se cumpla. CASOS contempla tantas condiciones como uno desee. Cada CASO introduce una condición diferente. Cuando una de ellas es verdadera, se ejecutan las acciones subordinadas a ella. No hay ejecución cuando es falsa, como se hace en SI con la opción DE LO CONTRARIO. Para considerar tal situación debe usarse otro caso o el general DEMAS CASOS. DEMAS CASOS no tiene condición y permite tratar todas las demás condiciones no contempladas específicamente en cada caso. Las condiciones deben ser preferentemente excluyentes; si no lo son, se ejecutarán las acciones del caso que se cumpla primero, es decir, el que en la secuencia está antes.

Note cuidadosamente el uso de puntos suspensivos en la porción

```
[CASO <condición 2>
    <acción j> . . . ] . . .
```

Los puntos que siguen a <acción j> indican que puede haber una <acción j+1>, <acción j+2>, etc. Los puntos suspensivos que siguen al corchete de cierre, indican que todo el caso se puede repetir las veces que haga falta.

En los siguientes ejemplos, note que también se usa adecuadamente la sangría, para poner de manifiesto las distintas subordinaciones.

9

```
- - - - -
casos
  caso cantidad pedida < 200
    facturar a precios corrientes
  caso cantidad pedida entre 200 y 499
    facturar a precios corrientes
    bonificar con el 5 % de descuento sobre el to-
tal facturado
  caso cantidad pedida entre 500 y 999
```

```
      facturar a precios especiales
    otros casos
      facturar a precios especiales
      bonificar con el 3 % de descuento sobre el to-
tal facturado
fin casos
- - - - -
```

10
```
- - - - -
casos
  caso solicitante socio y antigüedad < 5 años
    entregar solicitud
    entregar papeles para el garante
  caso solicitante socio y antigüedad >= 5 años
    entregar solicitud
  caso solicitante no socio e ingreso > $15.000
    entregar solicitud
    entregar documentación para dos garantes
  demás casos
    informar que no puede obtener préstamo
fin casos
- - - - -
```

Al igual que en SI, dentro de CASOS puede haber otros CASOS. Del mismo modo, dentro de SI puede haber CASOS y a la inversa. Estas posibilidades de anidamiento hacen muy poderosas a las estructuras. Vea el siguiente ejemplo.

11
```
- - - - -
casos
  caso alumno normal y cumple correlativas
    si regular en materia solicitada
      inscribir en acta de examen como regular
    de lo contrario
      si materia solicitada admite exámenes libres
        inscribir en acta de examen como libre
      de lo contrario
        rechazar solicitud
      fin si
    fin si
  caso alumno por equivalencia parcial y cumple
correlativas
    inscribir en acta de examen complementario
por equivalencia
  caso alumno vocacional
    inscribir en acta de examen como vocacional
  demás casos
    rechazar solicitud
fin casos
- - - - -
```

Observe que el anidamiento y el uso de condiciones compuestas permiten expresar un problema de distintas formas. Lo importante es que, cualquier forma que se use debe exponer precisamente la totalidad del problema. Así, el ejemplo anterior puede ser expresado también como sigue.

```
12  - - - - -
    casos
     caso alumno normal y cumple correlativas y regular en materia solicitada
      inscribir en acta de examen como regular
     caso alumno normal y cumple correlativas y libre en materia solicitada
      si materia solicitada admite exámenes libres
       inscribir en acta de examen como libre
      de lo contrario
       rechazar solicitud
      fin si
     caso alumno por equivalencia parcial y cumple correlativas
      inscribir en acta de examen complementario por equivalencia
     caso alumno vocacional
      inscribir en acta de examen como vocacional
     demás casos
      rechazar solicitud
    fin casos
    - - - - -
```

Podemos incluso eliminar el nido de IF, como sigue.

```
13      - - - - -
        casos
         caso alumno normal y cumple correlativas y regular en materia solicitada
          inscribir en acta de examen como regular
         caso alumno normal y cumple correlativas y libre en materia solicitada;
         y materia solicitada admite exámenes libres
          inscribir en acta de examen como libre
         caso alumno por equivalencia parcial y correlativas previas cumplidas
          inscribir en acta de examen complementario por equivalencia
         caso alumno vocacional
          inscribir en acta de examen como vocacional
         demás casos
          rechazar solicitud
        fin casos
```

El caso en que el alumno cumpla las correlativas, sea libre en la materia solicitada y ésta no pueda ser rendida en tal calidad, queda contemplado en el general "demás casos", porque no es tratado específicamente en los "casos" específicos.

Se puede notar la semejanza que hay entre la estructura CASOS y las tablas de decisión. Esto es así, con la salvedad que las tablas de decisión consideran siempre una cantidad fija de condiciones, que se enuncian en la respectiva matriz. Cada regla exige el cumplimiento o no de tales condiciones, o es indiferente a algunas de ellas. Pero cada regla considera la totalidad de condiciones. La estructura CASOS es más flexible, porque en un CASO puede considerar condiciones que no figuran

en ninguno de los demás casos. La regla Otras de las tablas de decisión equivale a la opción DEMAS CASOS de la estructura CASOS.

Estructura sintáctica MIENTRAS. Tiene el formato siguiente:

```
MIENTRAS <condición>
  <acción 1> . . .
FIN MIENTRAS
```

MIENTRAS y FIN MIENTRAS marcan los puntos de entrada y salida. Esta estructura controla que se ejecute la secuencia de acciones subordinadas mientras la <condición> sea verdadera. Si inicialmente la <condición> es verdadera, se ejecuta un ciclo de las acciones internas. Al llegar a FIN MIENTRAS se vuelve a probar la <condición>. Si ésta sigue siendo verdadera, se ejecuta otro ciclo de las acciones internas, y así. Cuando la condición se hace falsa, se abandona la estructura y se continúa con la acción que sigue a las palabras FIN MIENTRAS, fuera de la estructura. Si inicialmente la <condición> es falsa, no se ejecuta ninguna acción interna.

Puede que queramos repetir el ciclo una cantidad X de veces o puede que no sepamos cuándo <condición> se hará falsa. Ambas situaciones se pueden manejar con MIENTRAS. En el primer caso, hay que llevar la cuenta de las repeticiones y controlar en <condición> si se ha llegado al límite. Para ello, se define el contador de vueltas antes de la estructura. Por ejemplo:

```
14  - - - - -
    veces = 1
    mientras veces < 4
      hablar por teléfono a un cliente
      veces = veces +1
    fin mientras
    - - - - -
```

En el caso anterior se hacen tres llamadas telefónicas. La cantidad de veces está controlada por un contador llamado "veces". Note que el valor inicial de veces, 1, se ha establecido en una acción anterior a la estructura MIENTRAS.

```
15  - - - - -
    mientras haya usuarios en cola
      llamar al siguiente usuario
      atenderlo
    fin mientras
    - - - - -
```

En el ejemplo anterior, no hay una cantidad determinada de repeticiones. La condición es indeterminada, como es la cola de usuarios que esperan ser atendidos, la cual puede aumentar o disminuir a lo largo del horario de atención.

Las acciones a repetir pueden ser cualesquiera. Pero hay dos predefinidas, con un propósito especial: ITERAR y SALIR. Ambas son opcionales. ITERAR vuelve a probar la <condición> de MIENTRAS, sin ejecutar las acciones que le siguen dentro del ciclo. SALIR saca de la estructura MIENTRAS, sin esperar a que <condición> se haga falsa e ignorando las acciones que le siguen dentro del ciclo.

Cuando se llega a FIN MIENTRAS, la estructura vuelve automáticamente a probar la <condición>. Esto equivale a un "iterar" implícito antes de FIN MIENTRAS. Del mismo modo, si al probar la <condición> ésta resulta falsa, hay un "salir" implícito que saca de la estructura. Sin embargo, al hablar de ITERAR y SALIR no nos referimos a esas dos acciones implícitas, sino a las que podemos escribir explícitamente dentro del ciclo, entre las demás acciones y como una más de ellas.

Para que ITERAR y SALIR funcionen correctamente, deben estar condicionadas, ya sea usando SI o CASOS. De lo contrario, las acciones que le siguen serán inalcanzables y por ende, no podrán ser ejecutadas. Vea los siguientes ejemplos.

16
```
- - - - -
ordenar las solicitudes por fecha
mientras haya solicitudes a procesar
  tomar la próxima solicitud
  revisar los datos
  iterar
  consultar la fecha de viaje
  salir
  emitir la orden de compra del pasaje
  emitir el cheque para gastos
  reservar habitación en el hotel de destino
fin mientras
- - - - -
```

En el ejemplo anterior, ITERAR hace que nunca se ejecuten las últimas cinco acciones (consultar fecha de viaje, salir, emitir orden de compra del pasaje, emitir el cheque para gastos y reservar habitación en el hotel de destino). Aunque están escritas, es como si no lo estuvieran. Del mismo modo, SALIR impediría la ejecución de las dos últimas acciones (emitir el cheque para gastos y reservar habitación en el hotel de destino). Para corregir estos errores, hay que condicionar ITERAR y SALIR, por ejemplo, como sigue.

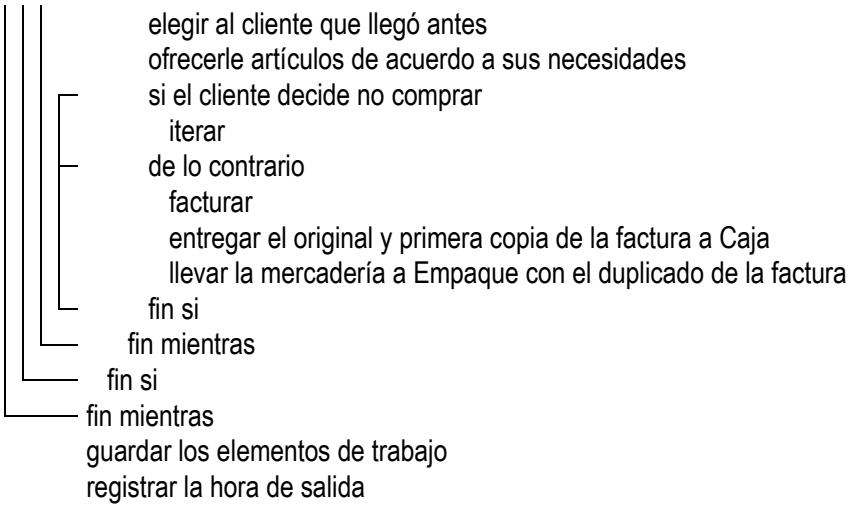
17
```
- - - - -
ordenar las solicitudes por fecha
mientras queden solicitudes a procesar
  tomar la próxima solicitud
  revisar los datos
  si hay datos incompletos
    iterar
  fin si
  consultar la fecha de viaje
  si la fecha de viaje es para más de tres meses
    salir
  fin si
  emitir la orden de compra del pasaje
  emitir el cheque para gastos
  reservar habitación en el hotel de destino
fin mientras
- - - - -
```

Del mismo modo que SI y CASOS pueden anidarse, la estructura MIENTRAS puede incluir otras MIENTRAS, SI o CASOS o estar incluida en ellas.

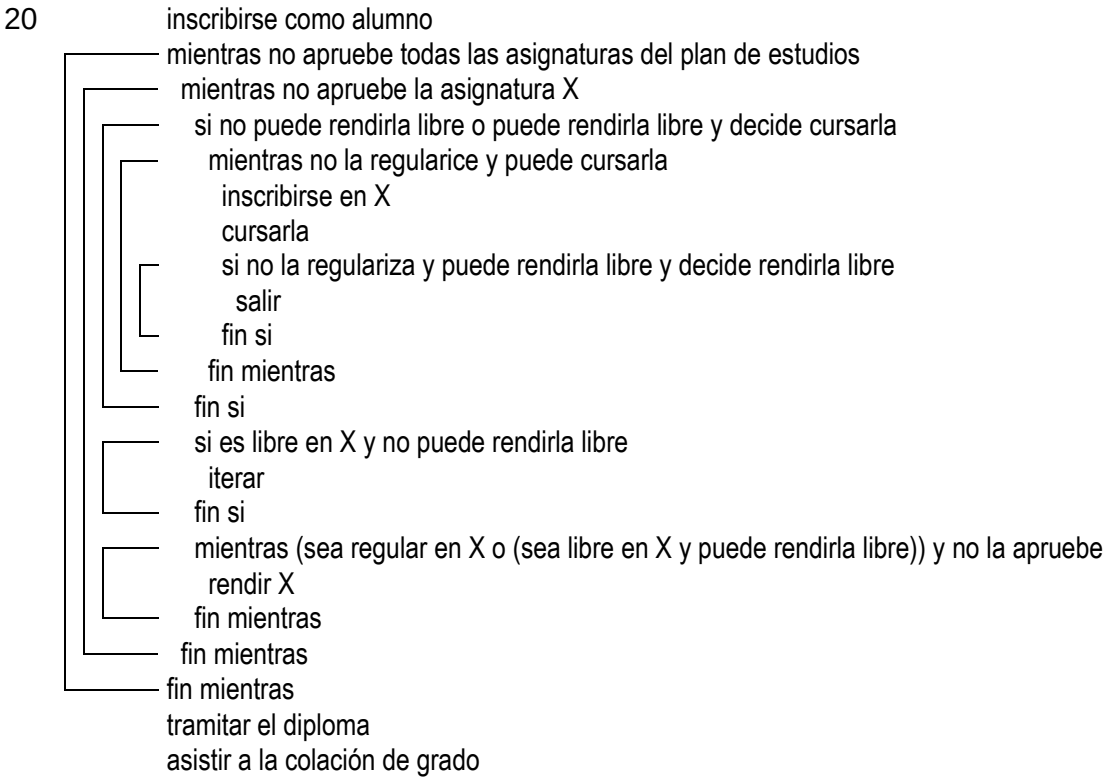
18
```
registrar la hora de entrada
preparar los elementos de trabajo
mientras no termine el horario de atención
  si no hay clientes que atender
    hacer lo que indique el encargado
    esperar a que lleguen clientes
  de lo contrario
    mientras haya clientes que atender
      elegir el cliente que llegó antes
      ofrecerle artículos de acuerdo a sus necesi-
dades
      si el cliente decide no comprar
        iterar
      de lo contrario
        facturar
        entregar el original y primera copia de la
factura a Caja
        llevar la mercadería a Empaque con el du-
plicado de la factura
      fin si
    fin mientras
  fin si
fin mientras
guardar los elementos de trabajo
registrar la hora de salida
```

Las sangrías ayudan a ver qué se corresponde con qué. La correspondencia, no obstante, no está relacionada con la sangría, que es una ayuda visual, sino que sigue una lógica eficaz. La regla de correspondencia, cuando hay estructuras anidadas, puede ahora ser expresada de la siguiente forma: desde el principio del nido se van buscando puntos de salida; cada uno se hace corresponder al punto de entrada de la estructura a que pertenezca que todavía no tiene correspondiente. En el último ejemplo, las correspondencias se indican a continuación mediante líneas.

19
```
registrar la hora de entrada
preparar los elementos de trabajo
mientras no termine el horario de atención
  si no hay clientes que atender
    hacer lo que indique el encargado
    esperar a que lleguen clientes
  de lo contrario
    mientras haya clientes que atender
```



El lenguaje estructurado es muy poderoso, pudiendo aplicarse a procesos muy complejos. El diagrama que sigue, claro para un universitario, ejemplifica tal poder.

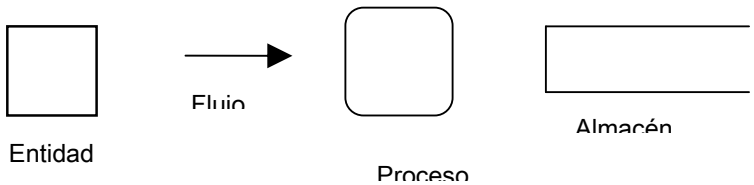


Diagramas de flujo de datos

El lenguaje estructurado, como los cursogramas y las tablas de decisión, sirven para documentar y analizar decisiones y procesos. Pero no dan una idea de los sistemas. El diagrama de flujo de datos tiene por objeto, precisamente, describir sistemas.

Símbolos

Estos diagramas utilizan los siguientes cuatro símbolos:



Entidades. También llamados terminales, representan usuarios de un sistema, que le entregan datos o reciben datos o información de él. Cuando entregan, se les llama entidades *origen*. Cuando los reciben, entidades *destino*. Las entidades pueden ser personas, unidades de organización, instituciones, máquinas, etc. Están fuera del sistema.

Flujos. Representan datos que transitan entre los demás símbolos, conectándolos. Nacen en entidades origen, procesos o almacenes. Terminan en entidades destino, procesos o almacenes. Con respecto a los procesos, los que terminan en ellos son *flujos de entrada* y los que nacen de ellos son *flujos de salida*. Pueden ser datos orales o tomados de notas, formularios, registros magnéticos, gráficos, etc.

Procesos. Representan transformadores de los flujos de entrada en flujos de salida. Pueden ser puestos de trabajo, unidades orgánicas, programas, etc.

Almacenes. Representan depósitos de datos. Por ejemplo, carpetas o ficheros que guardan documentos, archivos en discos magnéticos.

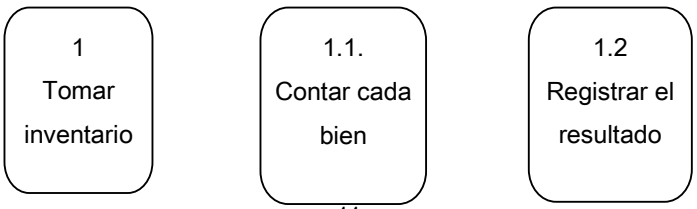
Reglas formales

1. Los símbolos pueden tener cualquier tamaño.

2. Los símbolos deben tener un nombre significativo. Las entidades, flujos y almacenes son nombres de cosas: *Cliente*, *Departamento de Personal*, *Factura*, *Remito*, *Facturas revisadas*, *Archivo de proveedores*, etc. Cuando los flujos no presten lugar a dudas sobre su contenido, pueden carecer de nombre. Los nombres de procesos deben ser una frase iniciada con un verbo en infinitivo: *Calcular comisiones*, *Facturar*, *Revisar mercaderías*, etc. Los nombres van dentro de los símbolos, excepto en los flujos, donde van sobre ellos o junto a ellos, a izquierda, derecha, arriba o abajo.

3. Si hay que volver a representar una entidad ya identificada, para evitar considerarla como nueva usaremos el borde más grueso.

4. Los procesos se numeran en la parte superior. Si un proceso es expandido en varios, los componentes se numeran en forma decimal. La numeración facilita identificar los procesos.



5. Los almacenes se numeran para facilitar su identificación. Si un almacén vuelve a dibujarse, para no considerarla como nuevo la dibujaremos con borde más grueso.

Reglas lógicas

1. Todo proceso debe tener flujos de entrada y salida.

2. A un proceso sólo deben entrar los flujos necesarios para ejecutarlo.

3. Los flujos que salen de un proceso deben ser el resultado exclusivo de los flujos de entrada.

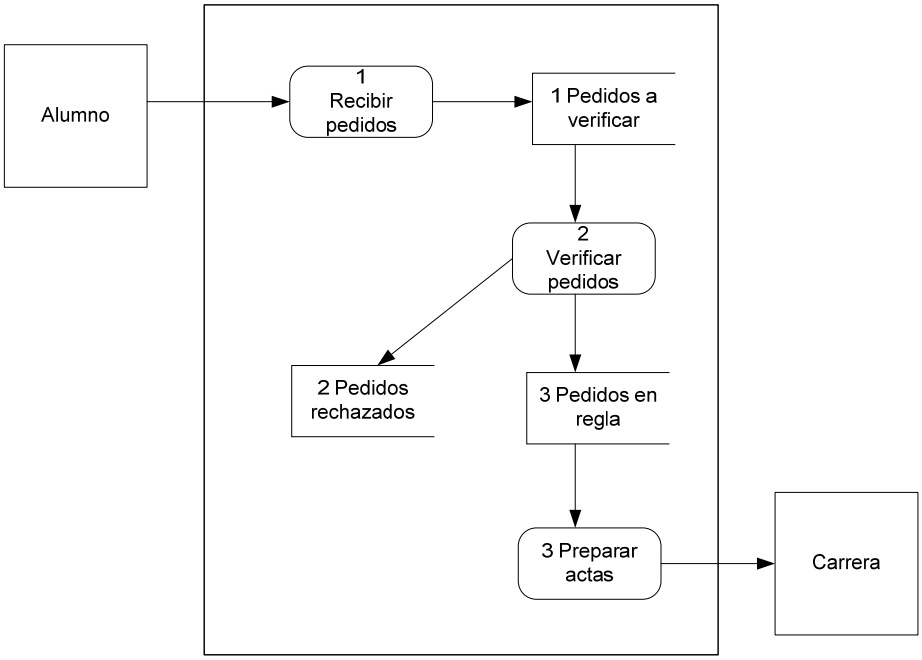
4. Un proceso puede estar compuesto internamente por procesos menores, almacenes y flujos. Cuando por claridad no convenga representar la complejidad en un solo diagrama, los detalles se exponen en diagramas subordinados, llamados *expansiones* del anterior. Esto puede repetirse las veces que haga falta.

5. Una expansión manifiesta detalles que están implícitamente contenidos en el proceso que se expande. No puede agregar componentes que no están implícitos en él. Si la expansión no agrega detalles, es innecesaria. Si tiene tantos detalles que complican su entendimiento, conviene expandirla a su vez.

Ejemplo
Sea un analista que debe estudiar el sistema actual de procesar pedidos de exámenes en la oficina de alumnos de una facultad. El sistema es manual. La primera información que ha obtenido, es la siguiente:

Al llegar el período para inscribirse en un turno de exámenes, el alumno interesado presenta un pedido de examen en Departamento Alumnos. Este revisa que esté bien llenado. Al terminar el tiempo de inscripción, verifica los pedidos bien confeccionados para determinar si están en condiciones. Con los que están en regla, confecciona las actas de exámenes, que entrega a cada carrera. Los pedidos rechazados se conservan para dar explicaciones a los interesados, por si hay reclamos.

Con esta información, el analista puede desarrollar el siguiente diagrama de flujo de datos:



Este primer diagrama representa el sistema total de exámenes. La línea de puntos, no obligatoria de representar, es la frontera del sistema. Lo que está afuera, las entidades *Alumno* y *Carrera*, usan el sistema, pero no son parte de él. *Alumno* inicia los procesos del sistema. *Carrera* recibe el resultado. Se han detectado tres procesos y tres almacenes.

Indudablemente, faltan muchos detalles. El analista se pregunta qué pasa realmente en cada proceso y si hay flujos, almacenes y procesos que no ha detectado. Profundizando su recolección de hechos, descubre que son bastante más complejos. Sin embargo, introducir todos esa información adicional en el diagrama lo haría sumamente complicado. La solución es expandir cada proceso mediante nuevos diagramas, que representarán subsistemas del sistema total. La expansión está en correspondencia con la forma natural en que el analista va descubriendo hechos y conociendo el sistema. Estará, entonces, en un segundo nivel de diagramas, de acuerdo a su nivel de conocimiento más detallado. Si en este nivel no alcanza a entender o expresar con claridad los detalles de cada proceso, los desarrollará en un tercer nivel, y así siguiendo. La expansión es un caso particular de un método para enfocar problemas según una dirección descendente: de lo más general a lo más particular, por refinamientos sucesivos.

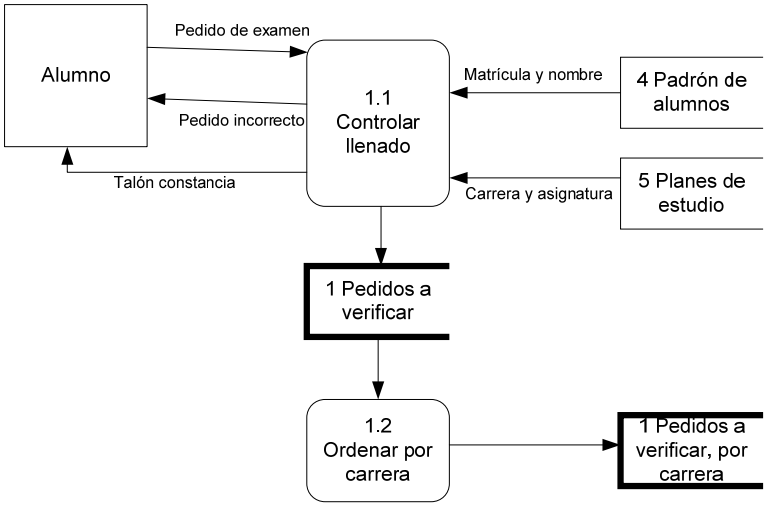
Observe que en ningún momento se ha dicho en el diagrama si el trabajo se hace manualmente o con computadoras. Tampoco se indican nombres de personas, lugares, etc. Lo que importa es captar la esencia del sistema. Esta abstracción permite decir que son diagramas lógicos. A veces, para que los usuarios entiendan mejor, se los puede hacer más "sensibles", incorporando lugares, nombres de personas, referencias a máquinas, etc. En tal caso, se dice que son diagramas físicos.

El analista, luego, descubre que el proceso *Recibir pedidos* se hace de esta forma:

Se controla que el número de matrícula y el nombre del alumno que figuran en el pedido de examen esté de acuerdo al padrón de alumnos. También se controla que el nombre de la carrera y la asignatura que el alumno pide rendir corresponda al

plan de estudios en que está inscripto. Si estos datos son incorrectos, se producirán varios inconvenientes en otros procesos. Pasada estas pruebas, se controla y firma el talón del pedido, duplicación de los datos anteriores, para que sirva de constancia de inscripción al alumno. Si hay errores en el pedido, se lo devuelve al alumno para que lo corrija o llene otro. Al final del período de inscripción, se ordenan los pedidos bien llenados por carrera.

El diagrama correspondiente a este proceso, es el que sigue.

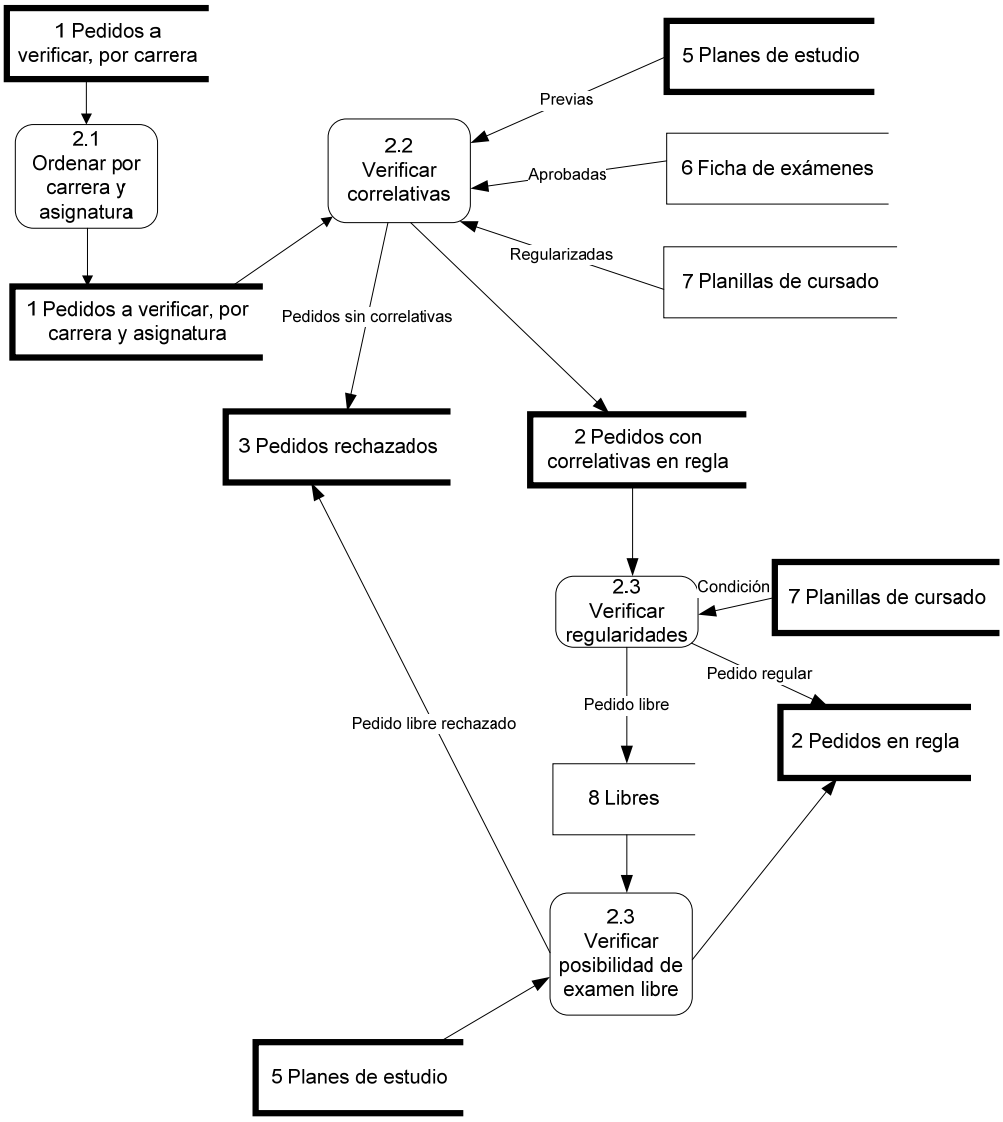


Observe que en esta expansión se han detectado dos nuevos almacenes y que el proceso 1 se ha dividido en 1.1 y 1.2.

El proceso *Verificar pedidos* del primer nivel, consiste en lo siguiente:

Se ordenan los pedidos a verificar por carrera y por asignatura. Se verifica que cada pedido cumpla con las asignaturas correlativas anteriores que el alumno debe tener aprobadas o regularizadas, para lo cual se consulta el plan de estudios, su ficha de exámenes y las planillas de cursado. Si no cumple este requisito, se anota en el pedido qué asignaturas previas no ha aprobado o regularizado. Los rechazados por esta causa se colocan en un lote de rechazos. Se someten los pedidos que pasan el filtro anterior a la prueba de regularidad usando las planillas de cursado. Si tienen regularidad, se anota en ellos esta condición y se los coloca en un lote de pedidos en regla. Si no la tienen, se anota en ellos la condición de libre y se los coloca en un lote de libres. Se comparan los pedidos de este lote con el plan de estudios correspondiente, para saber si las asignaturas pedidas pueden rendirse en condición de libre. Si es así, se los agrega al lote de pedidos en regla. En caso contrario, se les anota que no pueden rendirse como libre y se los coloca en el lote de rechazos.

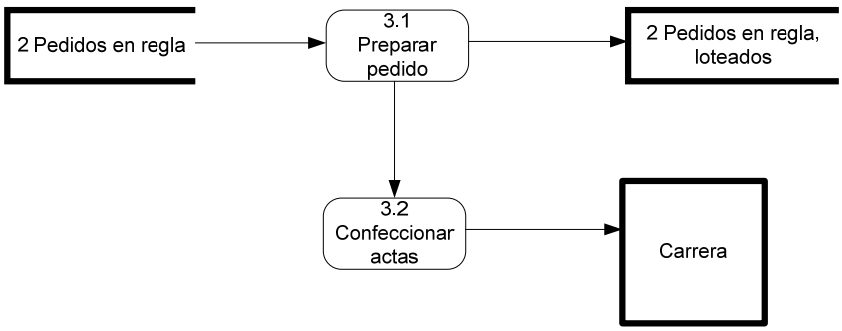
El diagrama para este proceso, más complicado, se presenta a continuación.



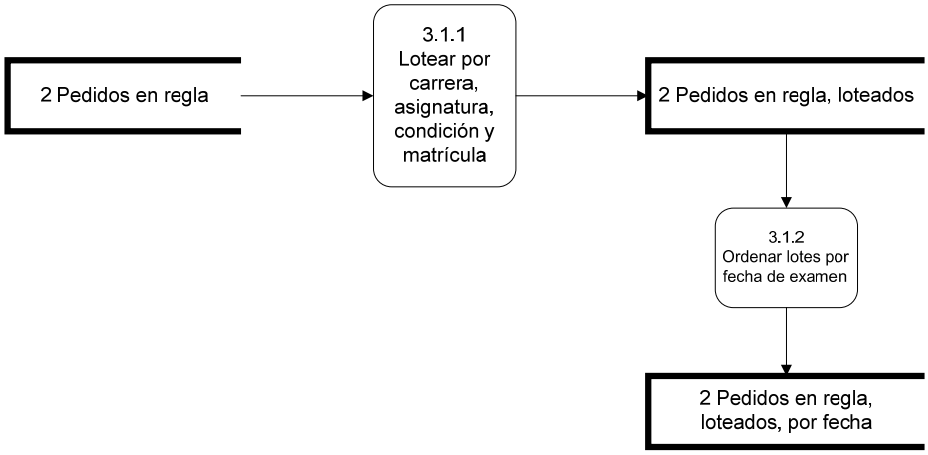
Para el proceso *Preparar actas*, los hechos adicionales dicen los siguiente:

La preparación de las actas de examen tienen dos etapas. En la primera se preparan lotes de pedidos en regla. Cada grupo de pedidos con igual carrera, asignatura y condición de examen es un lote. Cada uno se ordena internamente por matrícula. Luego se ordenan los lotes según el calendario de examen. En la segunda etapa se confeccionan las actas por duplicado a partir de cada lote, tomando los nombres de los miembros del tribunal examinador del calendario de examen. Se confecciona un remito de las actas que se envían a cada Carrera. Cada Carrera firma el remito y retiene las actas.

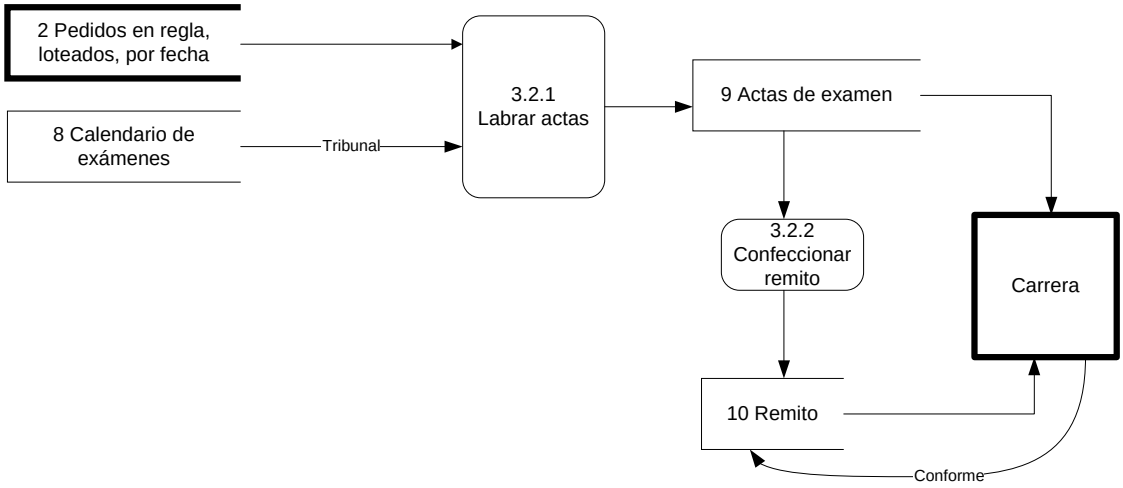
Para simplificar la diagramación de estos hechos, conviene primero un nivel para los dos procesos: *Lotear pedidos* y *Confeccionar actas*, según sigue.



Ahora es más fácil expandir cada proceso. Para *Preparar pedido*, el diagrama es:



Para *Confeccionar actas*, el diagrama es:



Los diagramas anteriores se han desarrollado para el sistema de pedidos de exámenes actual. Pero pueden aplicarse a un nuevo sistema, como resultado del análisis del sistema actual, se haya o no desarrollado mediante diagramas de flujo de datos.

Diccionario de datos

Un diccionario de datos, como indica el nombre, consiste en la definición de cada dato de un sistema de información. En este apartado, por "dato" nos referirnos al espacio de un formulario o al campo de un registro que se llena con un valor. Llamemos soporte al formulario o archivo.

En un soporte puede aparecer el dato *Artículo* y en otro soporte el dato *Producto*, pero resulta que son dos nombres para el mismo dato. También puede ser que el dato *Nombre* aparezca en dos soportes, pero en uno es el nombre de cliente y en el otro el nombre de empleado. Estas confusiones pueden evitarse creando un diccionario de datos, donde cada dato se defina una sola vez, mediante una serie de propiedades. El diccionario será entonces la fuente de consulta autorizada para analistas, programadores y usuarios.

En muchos lenguajes, cada campo de una tabla se define mediante un nombre, tipo, tamaño, etc., y tiene espacio para introducir comentarios. Esto podría considerarse un rudimento de diccionario. Pero hay algunos inconvenientes: los datos están dispersos en distintas tablas; un mismo dato puede aparecer en varias tablas con diferentes comentarios; no se subsana el inconveniente de distintos nombres para un mismo dato y del mismo nombre para datos distintos. Estos inconvenientes se superan con un diccionario, que centraliza la definición de cada dato una sola vez.

El diccionario puede incluir otros apartados para estructuras, flujos, almacenes y procesos. A los efectos prácticos, estos apartados no nos interesan.

Una forma rústica de construir un diccionario podría ser un documento de Word, donde a cada dato se le dé estilo de título. Todas los datos deberían tener las mismos propiedades (nombre, nombre simbólico, etc.). En Word, los títulos permiten construir tablas de contenido (similar a lo que entendemos por índice en un libro), con hipervínculos al lugar donde se desarrolla cada dato. Ordenar alfabéticamente los datos, como se hace en el diccionario convencional de un idioma, lleva algo de trabajo, pero se consigue.

Una forma más efectiva de construir un diccionario es usar una tabla cuyos registros tengan dos campos: un campo *Dato*, de caracteres, para escribir el dato que se define, y otro campo *Definición*, de tipo memo, para desarrollar sus propiedades. Construyendo un índice por el campo *Dato*, se presentan los datos ordenados alfabéticamente y se accede rápidamente a cualquiera de ellos. Un formulario sencillo permite dar altas, bajas y cambios en esa tabla, y buscar cualquier dato.

En el diccionario se deben definir los datos elementales. Un grupo de datos elementales, reunidos con algún propósito, constituyen una estructura de datos.

Datos elementales

Un dato elemental es aquél que no se subdivide en otros, tal como el nombre de un artículo, el documento de identidad, el código de un cliente, el precio de un bien, etc. A veces, que un dato se considere elemental depende de una convención. Por ejemplo, se puede considerar el dato *Fecha* como elemental; pero se la puede considerar una estructura compuesta por los datos elementales *Día*, *Mes* y *Año*. Del mismo modo, se puede definir convencionalmente que el dato *Nombre*, que incluye nombre y apellido, es elemental; pero se puede considerar que es una estructura compuesta por los datos elementales *Nombre* y *Apellido*.

Un dato elemental se debe definir mediante una serie de propiedades, de las cuales mencionaremos las que siguen. Dependiendo de la necesidad, pueden definirse otras propiedades.

Nombre. Es el denominación oficial que se da al dato. Debe indicar con precisión el dato que representa. Por ejemplo, *Matrícula universitaria*, *Código único de identificación tributaria*, etc.

Nombre simbólico. En sistemas computarizados, se puede establecer el nombre simbólico para referirse al dato elemental en los archivos y programas. Puede ser igual al nombre, pero generalmente es una abreviatura clara. Por ejemplo, para simbolizar el nombre *Matrícula universitaria* se podría usar el nombre simbólico *MU* y para *Código único de identificación tributaria* el nombre simbólico *CUIT*.

Título. El nombre simbólico es útil para manejarlo cómodamente en los programas, pero puede resultar difícil para los usuarios. En este caso, se puede explicitar mejor en la propiedad *Título*, que es lo que los usuarios verán en formularios e informes.

Descripción. Es una breve explicación del significado del dato elemental. Por ejemplo, para *Matrícula universitaria*, se puede describir como “número que identifica a cada alumno”.

Tipo. Es importante en sistemas computarizados, porque el tipo conlleva una validación automática del dato elemental. Por ejemplo, un campo de datos numérico no aceptará letras. Conviene usar las categorías de datos del lenguaje a usar.

Longitud. Es el tamaño que necesita el campo para almacenar los diferentes valores que pueda tener el dato. Como los datos numéricos generalmente requieren exactitud, la longitud debe ser definida por el valor mayor que se pueda representar. Si el depósito que puede hacer un cliente en una cuenta bancaria puede llegar a los cientos de miles de pesos, la longitud deberá tener seis posiciones enteras. Si la longitud fuera menor, no podría ingresarse un depósito de $425.372. Para datos alfabéticos, la longitud es la que hace falta para representar la mayoría de los valores. Por ejemplo, si el dato es el nombre de alumno, no hay que tomar como guía el nombre con más caracteres, que puede ser exageradamente largo. Esto llevaría a desperdiciar espacio de almacenamiento, porque el campo sería de igual tamaño para todos los alumnos. La longitud, en este caso, es una medida aceptable para escribir la mayoría de los nombres. Si resulta corta para algunos de ellos, se los podrá abreviar. La longitud es una exigencia en sistemas computarizados. También es útil al diseñar formularios de papel, porque permite asignar correctamente el espacio necesitado por cada dato.

Decimales. Es la cantidad de decimales fijos que tiene un dato de tipo numérico.

Sinónimos. Son los distintos nombres, además del oficial, usados para referirse al dato. *Matrícula universitaria*, por ejemplo, puede ser mencionada también como *Matrícula*, *Registro de alumno*, *Número de matrícula*, *Legajo*, *Número de registro*, *Número de alumno*, etc. No advertir esta variedad de nombres para el mismo dato, puede causar confusiones. Como el diccionario debe ser fuente para aclarar dudas, es importante que estos sinónimos se incorporen como equivalentes al nombre. Incluso se pueden incorporar en el diccionario de datos como entradas en un diccionario corriente, siempre que remitan al nombre oficial.

Formato. Muchas veces los datos no se presentan tal cuales son, sino “disfrazados” mediante un formato o máscara que facilita su lectura y llenado. En lenguajes de computación, hay diversas convenciones para definir formatos. Por ello, conviene usar las convenciones del lenguaje a usar.

Codificación. Si los valores posibles del dato se representan mediante unos pocos códigos, éstos pueden incluirse en esta propiedad. Cuando los códigos son muchos, es mejor incluir una referencia al anexo o tabla donde se los desarrolla.

Valor predeterminado. Cuando un dato posee un valor que es predominante, para facilitar su captura conviene que aparezca escrito al agregar un nuevo registro. Esto es lo que se conoce como valor predeterminado.

Validaciones. Se describen las validaciones que debe superar el dato para ser aceptado. Incluye cualquier validación descripta anteriormente, sea simultánea o posterior.

Diseño del nuevo sistema

El diseño es la etapa en que se desarrolla conceptualmente el nuevo sistema y se preparan las especificaciones para su posterior construcción física (desarrollo de programas, en casos de sistemas computarizados). Tiene similitud con el diseño de los ingenieros o los arquitectos, que hacen planos y cálculos de edificios, puentes, motores, pero no los construyen.

En esta etapa, la creatividad es fundamental. La creación del nuevo sistema no se hace de una sola vez. Habrá que pensar, imaginar, corregir, desechar, rehacer. Un nuevo sistema trae aparejada nuevas complicaciones y nuevas posibilidades de perfeccionamiento. Advierta que nunca se llegará al mejor sistema, porque todo sistema es perfectible por naturaleza.

Para estudiar el diseño del nuevo sistema, vamos a dividir nuestra tarea en las ya conocidas fases salidas, entradas, archivos y procesos.

Diseño de salidas

Una salida de un sistema de información son datos procesados, que pueden o no ser información para alguien. Un proceso que tome un archivo y lo ordene por algún criterio produce como salida un índice, difícilmente sea información para alguien. No obstante, la ordenación se hace para que, encadenada con otros procesos que quizás tampoco produzcan información, lleguen finalmente a producirla. En última instancia, hay salidas que sí son información: éstas son el propósito final de un sistema y son las que nos interesan. Las salidas no informativas son pasos intermedios necesarios para producir salidas informativas.

Las salidas informativas toman el aspecto sensible de impresos en papel o despliegues en pantalla. Siempre que sea posible, prefiera las salidas en pantalla, con la opción de imprimirlas. El uso indiscriminado de papel es costoso y tiene un impacto ambiental negativo.

Características de las salidas

Hemos visto al analizar los hechos que se debía examinar el sistema existente para juzgarlos en términos de calidad de información: oportunidad, compleción y formato. El nuevo sistema debe cumplir por norma este requisito.

Oportunidad

La oportunidad se debe lograr diseñando procesos que permitan producir a tiempo la información requerida. La oportunidad no es una características de las salidas en sí, sino de los procesos que las producen. Note que la oportunidad no sólo se refiere a usuarios internos de la institución, sino también a terceros. Por ejemplo, una empresa de servicios telefónicos debe entregar en término las facturas de un período. Esto implica un conjunto de procesos que registren los consumos, calculen los cargos, impriman las facturas y las distribuyan entre los clientes para que las paguen en fecha.

La compleción y el formato sí son características propias de las salidas. Algo hemos hablado de ellos anteriormente, pero agregaremos lo que sigue.

Compleción

Cuando se diseña una salida, el criterio fundamental es satisfacer a quien la va a usar. Sea para tomar una decisión, para enterarse de un resultado o para tener una fuente de consulta, los datos de la salida deben ser los *necesarios* para el usuario.

Esto supone que el analista ha estudiado el propósito de la salida, determinando cuáles son tales datos. Una salida que contiene todos los datos necesarios se dice que es *completa*. La carencia de un dato necesario obligará a que el usuario pierda tiempo consultando otras salidas o haciendo cálculos para completar la deficiencia. A veces pueden incluirse datos *convenientes*, porque facilitan la interpretación o búsqueda de datos relacionados en otras salidas. Pero hay que ser cuidadosos: con el argumento de la conveniencia, una salida puede incluir datos sin demasiado sentido, que complicarán el diseño, aumentarán el consumo de papel impreso y molestarán al usuario.

Como las salidas impresas son cerradas, pues lo que está escrito no se puede ampliar, la compleción debe lograrse en el impreso mismo. Tratándose de salidas en pantalla, a partir de una primera ventana y mediante menús o botones de comando se pueden obtener otras ventanas con información adicional. La compleción, entonces, se puede lograr mediante distintas ventanas, siempre que su tiempo de producción sea insignificante para el usuario. Esto, además, hace que la inclusión de datos convenientes sea más permisiva.

Formato

Hay varios factores que afectan la calidad del formato de una salida. Veamos algunos ejemplos.

- Cada dato tiene un ancho determinado, que afecta el espacio que ocupará en la salida. Como una salida normalmente tiene varios datos, puede que hacerlos entrar todos complique el diseño. Hay varios recursos a que se puede apelar. Así, si un dato es muy extenso, puede reducirse el espacio ocupado por él disminuyendo el tamaño de la fuente. Si no se exige la totalidad del dato, éste puede truncarse a una porción inicial, recurso común con datos de caracteres. A veces, dado el ancho de un campo de informe, se puede escribir un dato de caracteres en varias líneas dentro de ese ancho. Esta solución depende del soft usado. Incluso contando con el soft, si el tamaño vertical de la salida está estipulado, quizás no se pueda desarrollar un dato en varias líneas.
- Para datos de fecha, se puede ganar espacio expresando el año en dos cifras en lugar de cuatro. Para datos numéricos, si se requiere que sean expresados con precisión, es imposible abreviarlos; pero si no se exige exactitud, puede ganarse espacio expresándolos en cientos, miles, millones, etc., o en notación exponencial.
- Para facilitar la lectura de datos complicados, se los puede presentar usando máscaras. Así, el importe 7456320275, el número telefónico 02644232532 y el código de motor A72C47X329B son más fáciles de leer si se los presenta enmascarados como

  $7.456.320.275
  0264–423–2532
  A72–C4–7X–329–B.

  Los caracteres de intercalación de las máscaras también ocupan espacio, incrementando el ancho de los datos que las llevan.
- La disposición de los datos puede facilitar o entorpecer la tarea de quienes empleen una salida con fines de consulta. Por ejemplo, si diariamente se producen salidas que se encarpetan y se consultan frecuentemente por la columna de fecha, ésta debería ir colocada a la derecha, para hojear la carpeta con facilidad, sin tener que abrirla demasiado.

- Si de una salida impresa se van a transcribir algunos datos a un formulario, conviene que ambos tengan la misma disposición de estos datos, para que la transcripción sea lineal, evitando pegar saltos en la salida buscando los datos a llenar en el formulario.
- Si un informe contiene o puede contener varias páginas, conviene numerarlas. Esto permite colocarlas en secuencia correcta en caso de desordenarse. También es útil introducir el número total de páginas, porque asegura que no se extravíe ninguna. El lector recordará que la numeración puede tener la forma

  Página i de x
- Si un informe se produce todos los días, es conveniente que en el encabezado o pie se incorpore la fecha de emisión. Si se produce varias veces en el día, además de la fecha conviene introducir la hora.
- Para distinguir los encabezados, conviene resaltarlos por medio de letra en negrita, de mayor tamaño, separándolos por una línea o un rectángulo, etc.
- Para que el usuario ubique rápidamente datos como fecha de vencimiento o importe a pagar en una factura, conviene resaltarlos con negritas, un fondo, un rectángulo, etc.

Estos ejemplos son aspectos de formato, con los que el lector seguramente estará de acuerdo. No son los únicos, como se ha dicho, pero conviene pensar con detenimiento en este tema, para facilitar la lectura y la manipulación de las salidas.

Tipos de salidas

Las formas comunes de salidas informativas son textos, tabulados y gráficos.

Textos

Las salidas de texto que nos interesan son las repetitivas, con muchos destinatarios, como es el caso de certificaciones, reclamos, notificaciones, ofertas, etc. Puede ser necesario que, dentro del texto, se intercalen datos relativos a cada destinatario, como nombre, domicilio, importe reclamado, saldo de la cuenta, etc. Se dice entonces que el texto está “personalizado”. Cuando estas salidas se envían por correo, es necesario preparar sobres con los datos de cada destinatario, lo cual también es otra salida en forma de texto.

La tarea de confeccionar el sobre y la carta puede ser realizada a mano. Para facilitar el llenado de la carta, se la puede convertir en una especie de formulario, donde se completen solamente los datos personalizados. El llenado del sobre no tiene forma de abreviarse. Si la cantidad de destinatarios es elevada o si la tarea de preparar las cartas es frecuente, el trabajo manual es agobiante y lento.

Para facilitar la producción de estos tipos de texto, hay varias soluciones automáticas. La más común es usar un procesador de textos, como Word, que emplea dos archivos. Un archivo contiene el texto común a todos los destinatarios, donde se intercalan campos para los datos personalizados. El otro archivo está formado por campos que contienen los datos que varían por cada destinatario, constituyendo registros. Los campos usados en el primer archivo deben corresponder a campos del segundo. Éste puede ser preparado con el mismo procesador de textos, o puede provenir de una tabla o una planilla de cálculo. En el proceso de impresión, los campos se llenan automáticamente con los datos que se van tomando de cada registro del segundo archivo, un registro por carta. Para confeccionar los sobres, el proceso es similar. La siguiente figura ilustra lo dicho.

Archivo con el texto común

San Juan, **Fecha**

Sr./Sra./Srta. **Nombre**
Domicilio

La presente es para recordarle que, según nuestros registros, su deuda asciende a la suma de **Deuda**. Le rogamos ponerse al día, para evitar reclamos por otra vía, lo que produciría gastos innecesarios.

Saludamos a Ud. muy atte.

Archivo con datos de los destinatarios

Nombre	Domicilio	Deuda
García, Pedro	Salta 27	237
Garay, Mario	Colombia 448	1.215
Vital, Margarita	Haití 2749	680
Etcétera		

Salida combinando ambos archivos

San Juan, septiembre 30 de 2000

Sr./Sra./Srta. Garay, Mario
Colombia 448

La presente es para recordarle que, según nuestros registros, su deuda asciende a la suma de $1.215. Le rogamos ponerse al día, para evitar reclamos por otra vía, lo que produciría gastos innecesarios.

Saludamos a Ud. muy atte.

En el ejemplo anterior, el campo *Fecha* se toma del sistema.

Tabulados

Los tabulados son las salidas más empleadas. Hay varias modalidades.

Tabulados sin orden requerido, sin totales. Listan los registros de una tabla, la cual puede estar relacionada con otra u otras tablas para obtener datos complementarios. Cada campo forma una columna, bajo un encabezado. Los registros pueden estar ordenados o no, pero el ordenamiento es irrelevante para producir la salida.

Listado general de ventas				
Factura	Cliente	Artículo	Cant.	Importe
A000000000025	González, Manuel	Guardabarros del. izq.	1	45,00
A000000000025	González, Manuel	Guardabarros del. der.	1	45,00
A000000000025	González, Manuel	Paragolpes delantero	1	38,50
A000000000026	Martínez, Alberto	Neumático	4	240,00
A000000000026	Martínez, Alberto	Llanta	2	87,00
A000000000026	Martínez, Alberto	Gato neumático	1	187,00
A000000000026	Martínez, Alberto	Tornillos para llantas	12	18,00

Tabulados sin orden requerido, con totales. Son como el caso anterior, pero agregan datos al final, obtenidos de los registros procesados. Estos datos no forman parte de los registros, sino que resultan de ellos, como sumas, conteos o promedios. También se puede incluir totales al final de cada página impresa.

Listado general de ventas				
Factura	Cliente	Artículo	Cant.	Importe
A000000000025	González, Manuel	Guardabarros del. izq.	1	45,00
A000000000025	González, Manuel	Guardabarros del. der.	1	45,00
A000000000025	González, Manuel	Paragolpes delantero	1	38,50
A000000000026	Martínez, Alberto	Neumático	4	240,00
A000000000026	Martínez, Alberto	Llanta	2	87,00
A000000000026	Martínez, Alberto	Gato neumático	1	187,00
A000000000026	Martínez, Alberto	Tornillos para llantas	12	18,00
			Total	660,50

Tabulados con agrupación simple. Necesitan que los registros estén ordenados por algún campo, lo que permite formar grupos por ese campo. Por ejemplo, el archivo de facturas se ordena por factura, lográndose un grupo con los registros componentes de la factura X, otro con los registros componentes de la factura Y, etc. Hay tantos grupos como números de facturas distintos. El criterio de agrupación ha sido el campo factura. El listado puede tener totales por grupo, por página y finales.

Listado de ventas, ordenado por factura				
Factura	Cliente	Artículo	Cant.	Importe
A000000000025	González, Manuel	Guardabarros del. izq.	1	45,00
		Guardabarros del. der.	1	45,00
		Paragolpes delantero	1	38,50
			Subtotal	128,50
A000000000026	Martínez, Alberto	Neumático	4	240,00
		Llanta	2	87,00
		Gato neumático	1	187,00
		Tornillos para llantas	12	18,00
			Subtotal	532,00
			Total	660,50

Tabulados con agrupación múltiple. Necesitan que los registros se ordenan por varios campos, subordinados jerárquicamente, formando grupos, subgrupos, etc. Por ejemplo, el archivo de facturas se ordena por sucursal, cliente y artículo. Se lo-

gra un grupo de nivel 1 por cada sucursal; dentro de cada uno, un grupo de nivel 2 por cada cliente; dentro de cada grupo 1, grupo 2, un grupo de nivel 3 por cada artículo. Se puede dar totales de grupo 3, grupo 2 y grupo 1, más totales de página y finales.

Listado de ventas, por sucursal, cliente y artículo

	Factura	Cant.	Importe
Sucursal: Mendoza			
Cliente: Gutiérrez y Cía.			
Artículo: Juego sábana 1 plaza			
	A000100151798	12	2.880,00
	Total Artículo		*6.240,00*
Artículo: Acolchado 2½ plaza			
	A000100151798	15	4.500,00
	Total Artículo		*6.900,00*
	Total Cliente		13.140,00
Cliente: La Casa de lo Blanco			
Artículo: Almohada chica			
. . . . .	. . . . .	. . . . .	. . . . .
	Total Artículo		*2.358,00*
	Total Cliente		15.738,00
	Total Sucursal		*32.500,00*
	Total General		58.238,00

Sumarios. Necesitan registros ordenados por uno o más campos. No listan registros, sino encabezados y totales. Estos tabulados se llaman *resúmenes* o *sumarios*.

Resumen de ventas, por sucursal y cliente

Sucursal	Cliente	Total
Mendoza		
	Indiana	250,00
	El Hogar Feliz	398,40
	Total Sucursal	648,40
La Rioja		
	El aromo	1.200,00
	Total Sucursal	1.200,00
Tucumán		
	Azul y Oro	699,00
	Más Vale Pájaro...	575,80
	Total Sucursal	1.274,80
	Total General	3.123,20

Como se infiere de los ejemplos, los tabulados presentan varias regularidades:

- Cada página normalmente lleva datos de encabezado, como el nombre del informe, la fecha o los títulos de los datos encolumnados. También puede llevar un pie para colocar el número de página, totales por página, etc. Los datos mencionados ocupan porciones de la hoja, formando lo que se llama bandas, una de

encabezado de página y otra de pie de página.

- Dentro de la página, los datos se disponen en columnas, que corresponden a campos de datos o a expresiones calculadas a partir de ellos. Cada registro produce una o más líneas de salida, todas de igual formato, que ocupan un espacio. Esto es la banda de detalle.

- Los registros pueden formar grupos, entendiéndose por grupo aquellos conjuntos de registros que tengan igualdad de contenido en un campo. Para agrupar por un campo, es necesario que los registros estén ordenados por él. Puede necesitarse un encabezado para cada grupo, donde se escriben datos que lo identifican, y un pie para cada grupo, donde se dan datos que lo resumen, como totales, promedios, conteos. El encabezado y el pie de grupo, de ser incluidos, generan dos bandas, que encierran la banda de detalle.

- Dentro de cada grupo, los registros pueden constituir grupos subordinados por otro campo, si también están ordenados por él. Para el subgrupo puede necesitarse una banda de encabezado y otra de pie. Estas bandas rodean la banda de detalle, de la siguiente forma: banda de encabezado del grupo 1, banda de encabezado del grupo 2, banda de detalle, banda de pie del grupo 2, banda de pie del grupo 1. El proceso de crear grupos subordinados a un grupo mayor puede repetirse varias veces, siguiendo el mismo esquema.

- Puede ser que el tabulado necesite un título general y un resumen general. En este resumen puede darse el total general de la suma de cada columna, o el promedio, o la cantidad de registros listados, etc. Hay entonces una banda para título y otra para resumen, que pueden ocupar páginas por separado. La ilustración siguiente ejemplifica diferentes bandas.

Zeta S.A. Informe de ventas, por cliente Período: Enero / Agosto 2000	Título

Informe de ventas, por cliente		
Factura	Importe	Encabezado de página
Abadía, Jorge		Encabezado de grupo
A00001258	500	Detalle
A00001348	100	Detalle
Subtotal	600	Pie de grupo
Álvarez, Gerardo		Encabezado de grupo
A00001194	150	Detalle
A00001232	300	Detalle
A00001301	250	Detalle
Subtotal	700	Pie de grupo
Página 1		Pie de página

Total	124.345
Factura promedio	732
Cantidad de facturas	204

Resumen

Observe estas recomendaciones importantes, que hacen al formato:

- Evite el uso de códigos; en su lugar, escriba lo que significan. La codificación es útil porque ahorra espacio, normaliza distintas expresiones de un valor y facilita procesos en el interior del sistema; pero el usuario final no tiene por qué conocerla. A veces, ni los usuarios directos la conocen, pues son los programas los que codifican automáticamente. En los ejemplos anteriores, no se han empleado los códigos de sucursal, artículo, cliente, sino que se los ha sustituido por los valores que representan, logrando claridad. Con relaciones adecuadas entre diferentes tablas, esta sustitución es muy sencilla.

- Destaque los encabezados y totales de grupos, subgrupos, etc.

- Use títulos y encabezados que resuman el contenido y propósito del tabulado.

- Incorpore la fecha e incluso la hora en que se emite el tabulado. La carencia de estos datos puede confundir, cuando hay varios tabulados similares, por no poder determinarse cuándo fueron emitidos.

- Si el tabulado tiene varias páginas, numere cada una.

- Elimine datos repetidos que dificultan la lectura. Por ejemplo, este tabulado

Ventas del mes de febrero 1996

Factura	Cliente	Artículo	Cant.	Importe
A000000000025	González, Manuel	Guardabarros del. izq.	1	45,00
A000000000025	González, Manuel	Guardabarros del. der.	1	45,00
A000000000025	González, Manuel	Paragolpes delantero	1	38,50
			Total	128,50
A000000000026	Martínez, Alberto	Neumático	4	240,00
A000000000026	Martínez, Alberto	Llanta	2	87,00
A000000000026	Martínez, Alberto	Gato neumático	1	187,00
A000000000026	Martínez, Alberto	Tornillos para llantas	12	18,00
			Total	532,00

es mucho más claro si se lo presenta de esta otra manera:

Ventas del mes de febrero 2000

Página: 1
Fecha: 02-03-1996

Factura	Cliente	Artículo	Cant.	Importe
A000000000025	González, Manuel	Guardabarros del. izq.	1	45,00
		Guardabarros del. der.	1	45,00
		Paragolpes delantero	1	38,50
			Total	128,50
A000000000026	Martínez, Adalberto	Neumático	4	240,00
		Llanta	2	87,00
		Gato neumático	1	187,00
		Tornillos para llantas	12	18,00
			Total	532,00

Los tabulados, si no se apartan mucho de las regularidades descriptas, pueden producirse fácilmente con herramientas de diseño incorporadas a los lenguajes de programación. A veces, sin embargo, necesitan variar este formato de modo mucho más complejo. En tales casos, debe desarrollarse programación específica para producirlos.

Gráficos

El lector sabe por experiencia qué es una recta o una parábola, si las ve dibujadas; más difícil le resulta identificarlas si están escritas como expresiones analíticas. Del mismo modo, hay veces en que los datos tabulados no son fácilmente entendibles, por lo que su graficación agiliza interpretarlos. Los gráficos, por ejemplo, permiten advertir tendencias, comparar varias series de datos y notar irregularidades. No obstante, son una ayuda sensorial y no reemplazan el análisis cuantitativo.

Hay grandes posibilidades de graficación. Esto debe hacernos cuidadosos. El uso de un gráfico inapropiado, la pobreza o el exceso de recursos empleados, la ausencia de texto aclaratorio, la desproporción de escalas, etc., pueden despojarlo de valor informativo. Existe cantidad de software generador de gráficos. El más conocido es el brindado por Excel, que los producen a partir de series numéricas. Estas series pueden introducirse manualmente o importando datos desde ciertos archivos. Los archivos a importar se producen con procesadores de texto o lenguajes de bases de datos. Estos últimos, por sí solos, también pueden generar gráficos, más pobres, sin necesidad de planillas de cálculo. Suponemos que el lector conoce los tipos de gráficos que puede lograr, por ejemplo, usando Excel: de barras, de líneas, de dispersión, circulares, de anillos, de burbujas, etc.

Otro tipo de gráficos son los íconos. Una variedad es usar íconos proporcionales a la magnitud que representan. Supongamos que una empresa telefónica ha instalado nuevos servicios durante tres años, por estas cantidades:

Año	Cantidad
1993	5.000
1994	7.500
1995	10.000

Usemos la figura de un teléfono. Su tamaño depende de la cantidad:

1993 1994 1995

El tamaño de 1995 es visualmente impactante, pero no se sabe exactamente cuánto mayor que 1993 es, a menos que se sepa la cantidad.

Otra variedad, en vez de variar el tamaño del ícono, suele usar uno de igual tamaño, repetido las veces que haga falta para dar idea de la magnitud. El ejemplo anterior, supuesto que cada ícono representa 2500 teléfonos, podría expresarse así:

1993

1994

1995

Salidas impresas

La impresión es la forma más tradicional de presentar información, si bien ahora se tiende a reducir su incidencia, reemplazándola con presentaciones en pantalla. Al diseñar salidas impresas hay que considerar varios factores.

Impresoras

En pocas décadas, las impresoras de computadoras se han presentado en gran cantidad de formas. Si bien algunas llegaron a ser maravillas de ingeniería, sus costos eran muy elevados. Esto hacía que se usara una sola impresora para todas las salidas. Hoy las cosas han cambiado: son baratas y en una instalación en red o incluso en una sola computadora se pueden usar muchas.

Velocidad. Para medir la velocidad se usa un cociente, donde el numerador es la cantidad de impresión y el denominador el tiempo. De acuerdo a la rapidez, el cociente se puede expresar como cantidad de caracteres por segundo, líneas por minuto, páginas por minuto.

Hasta los setenta, la velocidad era un factor crítico para grandes empresas e instituciones gubernamentales, que debían producir un gran volumen de impresos. Para terminar las tareas a tiempo, hacía falta impresoras cada vez más veloces, lo cual las hacía cada vez más caras. La PC revolucionó los precios y popularizó la computación, de modo que hasta las empresas pequeñas pudieron aprovecharlas.

Aparecieron impresoras modestas, pero baratas. La tendencia se acentuó con los años, de modo que se hizo común poseer varias impresoras. Por otro lado, la conexión de las PC en red también se hizo común. Hoy es posible obtener gran cantidad de impresión usando computadoras en red y muchas impresoras, de modo que un gran trabajo puede subdividirse en fragmentos, ocupándose cada máquina de imprimir uno de ellos.

Producción de la impresión. Para lograr la impresión, hay impresoras que se valen de impacto y otras que no lo emplean. Las primeras son las más antiguas y trabajan como las viejas máquinas de escribir: cada carácter se logra golpeando una tela entintada que tiñe el papel. Los caracteres pueden estar tallados en negativo (en la cabeza de un pequeño martillo, en una lengüeta, en el eslabón de una cadena, en una banda metálica, etc.), y son golpeados contra la tela. Puede que los caracteres no estén tallados en negativo, sino que haya una cantidad de agujas que, al golpear la tela, producen puntos sobre el papel. La distribución oportuna de estos puntos forma distintos caracteres, rayas, tramados y plenos, figuras, etc. A mayor cantidad de puntos, más definido será un carácter o una imagen, pero más lenta la impresión. La impresión por impacto tiene desventajas: no produce impresión de calidad y es ruidosa. La ventaja es que permite obtener copias al carbónico.

Entre las impresoras que trabajan sin impacto, las de mejor rendimiento son las de chorro de tinta y las láser. Las primeras arrojan minúsculas partículas de tinta, de rápido secado. Las láser emplean cargas electrostáticas, produciendo un negativo eléctrico de la página sobre un rodillo, el que se rocía con *tonner* que se adhiere donde hay carga. El rodillo se hace girar sobre el papel, obteniendo el positivo en tonner. Para fijar lo impreso, se derrite el tonner mediante calor, con lo que el impreso queda adherido a la hoja. El proceso es similar al fotocopiado.

Las impresoras sin impacto son silenciosas y logran gran calidad. No pueden obtener copias al carbónico. Ambos tipos pueden imprimir en colores. Decidir entre impresoras láser o de chorro de tinta se determina no sólo por el precio de estas máquinas, sino por el precio de los insumos que utilizan.

Una tercera forma de impresión sin impacto son las impresoras térmicas, poco usadas porque lo impreso se va desvaneciendo con el tiempo. El proceso consiste en producir la impresión sobre un papel sensible al calor: sólo recibe calor la parte que representa letras o números.

Fuentes y gráficos. Las impresoras más antiguas, todavía en uso, solamente imprimían los caracteres del teclado, con pobreza de fuentes. En esos días se usaban máquinas especiales para producir gráficos, llamadas *plotters*. Las impresoras modernas pueden imprimir distintos tipos de fuentes y gráficos de alta calidad.

Color. El color de un impreso puede resaltar datos importantes y proteger ciertos datos mediante tramados. La forma más común de emplear colores es valerse de formularios pre impresos donde ciertas áreas lleven este recurso. También se puede imprimir directamente con impresoras que manejan colores. Las tintas de color son más caras que las negras y las impresoras láser a color son bastante caras.

Si se usa color, hay que emplear discreción y gusto. Demasiados colores en una hoja resultan chocantes, especialmente si contrastan violentamente. Son preferibles tonos suaves para la mayor parte de un impreso, dejando los fuertes para destacar algo especial, como un error, un total o una fecha de vencimiento.

Pre impresos. Muchas salidas son formularios, como facturas, notas de débito, etc. Las impresoras modernas pueden imprimir simultáneamente el formulario y los datos. Pero hay normas del Estado que exigen usar formularios pre impresos, impidiendo aprovechar esa ventaja. También puede ser que las impresoras a usar no tengan posibilidades de imprimir colores, fondos, logotipos, etc. En estos casos hay que usar pre impresos, que son fabricados por imprentas con máquinas adecuadas. Además de la impresión en sí, las imprentas pueden troquelar el papel si hay que fraccionar los formularios; también pueden pre perforarlos si necesitan ser encarpetados. Hay que diseñar los pre impresos con exactitud milimétrica y dar instrucciones precisas a las imprentas. Para aceptar el trabajo solicitado, es conveniente exigirle pre impresos de muestra y hacer impresiones de prueba para asegurarse que responden a las instrucciones.

Impresión con códigos de barra. La mayoría de los impresos que conocemos son de salida exclusiva, es decir, tienen como destinatarios a usuarios humanos, los únicos que los pueden leer. No obstante, se ha desarrollado tecnología para hacerlos "legibles" a ciertas máquinas, lo cual los convierte en entradas de otros procesos. Los datos que van a leer las máquinas se escriben en códigos de barra.

Esta impresión, que suele llamarse “de retorno”, evita la transcripción de datos, actividad lenta y sujeta a errores. Suponga que una empresa de gas confecciona las facturas por consumo a partir de dos tablas: una contienen los datos del cliente (número, nombre, domicilio) y otra contiene número de cliente, período, consumo, fecha de vencimiento, importe a pagar. Cada factura tiene tres talones. En el talón del cliente figuran en forma convencional datos como número de cliente, nombre, domicilio, consumo, tarifa, vencimiento e importe. En el talón del banco figuran, también en forma convencional, datos para el control del cajero y para rendir cuentas de lo recaudado a la empresa. En el talón de la empresa figuran datos de la transacción en forma convencional; pero, en códigos de barra, número de usuario y período. El banco sólo cobra facturas no vencidas. Lotea los talones de la empresa por día de cobro y le rinde cuentas. Para registrar los pagos de un lote, la empresa introduce una vez la fecha del lote y usa un lector (*scanner*) que lee los talones y registra en el archivo de facturas, para cada usuario, la fecha de pago. Para las facturas vencidas se usa otro procedimiento, que no interesa desarrollar.

El ejemplo pone de manifiesto que, al usar códigos de barra, el impreso ya no es una salida final del proceso facturación, sino que sirve como entrada al proceso de registración de pagos. El talón legible por una máquina se llama documento o formulario de retorno. La gran ventaja de esta tecnología es que evita la transcripción manual de los pagos, con mayor rapidez y sin errores.

Una forma mejor de los últimos años es usar una factura con un solo cuerpo, donde el código de barras incluye código de empresa, código de concepto que se paga, número de cliente, período, fecha de vencimiento e importe. Al pagar, el cajero del banco hace leer el código de barras y obtiene un talón que acredita el pago, que entrega al cliente. Internamente, el programa que utiliza el cajero crea un registro con los datos del pago. Periódicamente, el banco produce distintos archivos, uno para cada empresa asociada al sistema de cobros, donde se transcriben los registros que le corresponden. Cada empresa usa el archivo correspondiente de pagos para registrarlos en el archivo de facturas.

Especificaciones de salidas impresas

Especificar las salidas impresas es establecer con precisión el diseño que van a tener, las fórmulas a aplicar para calcular datos que no están en los registros de entrada, los agrupamientos de datos, las condiciones para imprimir ciertos datos, las fuentes y sus tamaños, etc. Las especificaciones van dirigidas principalmente al programador, para que desarrolle los programas que produzcan tales salidas. También pueden tener como destinatarios las imprentas de pre impresos y los encargados de comprar el papel para las impresoras. La especificación de cada salida particular formará parte de la documentación del programa correspondiente.

En el diseño se distribuyen los datos requeridos de una salida en el formato de hoja a usar, aplicando los criterios generales del diseño de formularios. El ancho asignado a cada dato en la hoja es una limitación física, porque no se puede modificar en tiempo de ejecución del programa. De aquí que, para que un dato quepa, se puede variar la fuente, truncarlo o escribirlo en varias líneas, lo cual debe especificarse. En los tabulados, cuando los datos a escribir son tantos que no caben en una línea, se puede usar dos o más líneas para cada registro. Como esta solución no siempre resulta clara de entender, se suele dividir la salida en varios tabulados, cada uno con un subconjunto de los datos, repitiendo en cada uno los datos que sirven como clave de unión.

El diseño se facilita si se precisan las bandas a utilizar y se definen los controles de cada una (etiquetas, campos de informe, líneas, rectángulos, imágenes). Si se requiere mucha precisión, se pueden utilizar reglas horizontales y verticales para cada banda, de modo que proyectando los controles sobre las reglas se sepan sus posiciones y tamaños. Incluso, por cada banda, se puede indicar el alto, ancho y ubicación del vértice superior izquierdo de cada objeto.

Si el diseño se explica por sí mismo, no hace falta agregar nada. Pero si se requiere explicar cómo se obtiene el dato de un campo, se puede incluir una llamada que se desarrolla en un cuadro de referencias anexo.

Salidas en pantalla

Las salidas en pantalla se usan cada vez más, conforme los usuarios se acostumbran a ellas y advierten sus ventajas, porque:

- Se obtienen más velozmente que las salidas impresas.
- Permiten recursos no disponibles en impresión, como zoom y cambio instantáneo de página.
- Evitan el amontonamiento de papel y el trabajo de archivarlo.

Las ventajas incentivan el empleo de estas salidas. El usuario descubre que puede prescindir del papel si necesita información que no debe conservar. La desventaja viene dada por las dimensiones de las pantallas, menores que una hoja de papel.

Tipos de salidas en pantalla

Hay dos tipos de salidas en pantalla. El primero consiste en salidas cuyo destino es la impresión, pero que pueden ser visualizadas previamente en pantalla. Cualquiera de las variedades conocidas de textos, tabulados y gráficos pueden tener este destino. El buen soft posee herramientas que permiten desplazar el contenido de la pági-

na visualizada en sentido horizontal y vertical, avanzar o retroceder páginas, aumentar o disminuir lo que se ve, etc.

El segundo tipo son salidas cuyo destino es la pantalla. Una ejemplo es un formulario que despliega el contenido de uno o varias tablas, mediante distintos objetos. Son muy útiles los marcos de página, que en diferentes páginas muestran distintos datos, y las cuadrículas, que presentan varios registros con campos de una tabla o de tablas relacionadas. Al ejecutar los formularios, se puede cambiar el orden de los registros, el ancho de las columnas y su posición relativa, aplicar filtros, etc. Desde un formulario se puede ejecutar otro, mediante un botón de comando. Si se permite modificar los datos visualizados, el formulario no sólo es de salida, sino también de entrada. Otro ejemplo es mostrar un archivo que resulte de ciertos filtros establecidos al comienzo de la ejecución de un programa, como una vista con parámetros.

Especificaciones de salidas en pantalla

Lo dicho para salidas impresas tiene aplicación aquí. El cuadro de referencias es conveniente para precisar los detalles.

Lo que hay de nuevo sobre las salidas en pantalla es que, dado las menores dimensiones de ella con respecto al papel, hay que considerar la necesidad de desplazamientos para alcanzar datos no visibles. Esto es innecesario cuando el soft posee barras de desplazamiento que permiten tales movimientos. De no ser así, el analista deberá establecer cuáles teclas, íconos, menús, etc., se usarán para lograrlos. En este caso, es muy importante que estas convenciones sean iguales en todas las salidas. De este modo el usuario las automatizará para todo el sistema, no para cada salida particular.

Diseño de entradas

En los procesos fabriles, el producto que se quiere obtener determina los insumos a emplear. Lo mismo acontece en los sistemas de información. La información que se quiera producir determina los datos de entrada.

Tipos de entradas

Previo a considerar el diseño de las entradas, veamos algunos tipos de ellas, que nos darán pistas sobre el diseño.

Origen

Podemos distinguir dos clases de entradas: las generadas en el entorno y las generadas internamente.

Entradas generadas en el entorno. Aportan datos del entorno que deben almacenarse en tablas. Son como la materia prima en los procesos fabriles. Las actividades principales son la captura de datos y el control de su corrección (validación). Es el caso de la documentación que presentan quienes ingresan a la universidad, de donde se toman datos para crear registros maestros de alumnos; o las solicitudes para cursar o rendir materias, generadas por los alumnos, que van a crear registros de transacciones entre los alumnos y las materias; o las facturas enviadas por los proveedores. En el segundo ejemplo, si bien se usan las tablas de alumnos y materias para tomar algunos datos, que ya existen, lo que proviene del entorno es la voluntad de un alumno de cursar o rendir ciertas materias.

Entradas generadas internamente. Son archivos producidos por ciertos procesos, que posteriormente sirven de entrada a otros. Son semejantes a los productos semielaborados de los procesos fabriles. Acá no hay captura de datos, sino otras actividades, como transcripción, cálculo, control, ordenamiento, comunicación. Por ejemplo, sea que se quiera obtener un archivo con el total mensual comprado por cada cliente, durante un período de doce meses, el cual servirá para producir un informe. Para generar el archivo hace falta, como entrada generada en el entorno, el mes y año inicial; pero es necesario tomar datos de los archivos de clientes y de facturación, que ya están generados. El archivo obtenido es la salida del proceso de totalización, pero será la entrada del proceso que produce el informe.

A los efectos del diseño, las entradas que importan son las generadas en el entorno, porque son la fuente de todas las salidas. El diseño de estas entradas se materializa como formularios que sirven para capturar los datos.

Complejidad

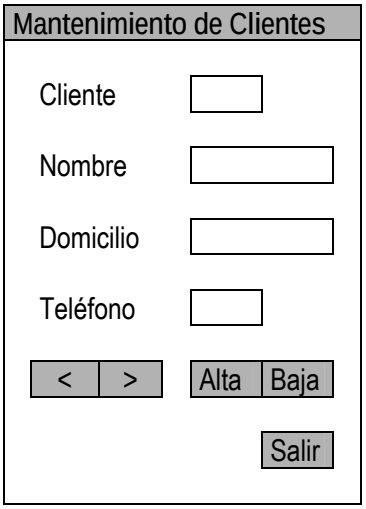
Las entradas a un sistema de información son flujos de datos de distinta complejidad: datos elementales, estructuras y subestructuras.

Datos elementales. Un dato elemental es aquél que no puede descomponerse en datos menores. Un dato elemental puede provenir de una estructura o ser independiente de ella. El primero puede considerarse una subestructura de un solo dato. El segundo es un dato que se usa como filtro para llevar a cabo un proceso. En sistemas manuales, por ejemplo, si se quiere calcular el total facturado en un período de

tiempo, hace falta una fecha inicial y otra final. Estas fechas son datos elementales que sirven como filtros para las facturas: toda factura emitida dentro del intervalo definido por ambas fechas se totalizará, ignorándose en caso contrario. En un sistema computarizado, el programa que hace el mismo cálculo pedirá inicialmente ambas fechas, que servirán de filtro para los registros de facturas a totalizar.

Estructuras. Las estructuras son datos relacionados que definen una entidad maestra o una transacción. En sistemas manuales son los formularios y en sistemas computarizados los registros de tablas. Como un formulario es un soporte de datos con espacios contenedores para ellos y textos indicativos del dato a llenar en cada espacio, los registros pueden concebirse como formularios, donde los campos son los contenedores y los nombres de los campos los textos indicativos.

Las estructuras son tema del diseño de archivos. Pero como hay archivos (tablas) que se alimentan con entradas generadas en el entorno, hay un aspecto de ellos que afecta el diseño de entradas. Para capturar los datos en las tablas, no es cómodo ni recomendable hacerlo directamente en ellas, usando por ejemplo la ventana *Examinar* de VFP o la vista *Hoja de datos* de Access. Por un lado, no todos los lenguajes tienen estas posibilidades; por otro, el control de los datos puede requerir programación difícil o imposible de definir en las reglas de validación de las tablas, como el cálculo de un dígito verificador. Es más conveniente capturar los datos a través de formularios en pantalla, porque admiten controles de cualquier complejidad. Los formularios en pantalla pueden servir directamente para la captura, o pueden ser una réplica de formularios en papel para transcribir a pantalla los datos capturados en ellos. Ambas clases de formularios sí son tema del diseño de entradas. La ilustración siguiente ejemplifica un formulario para modificar los registros y campos de una tabla.



Formulario en pantalla

Cliente	Nombre	Domicilio	Teléfono

Estructura del archivo

Subestructuras. Una subestructura es un subconjunto de datos de una estructura. El número de factura y la fecha, tomados del archivo de facturación, son un ejemplo. Cuando una subestructura está formada por un solo dato elemental, como se dijo, se puede considerar indistintamente subestructura o dato elemental. Las subestructuras se usan para controlar y completar entradas generadas en el entorno. Por ejemplo, al registrar una factura, hace falta el código y el nombre del cliente, que se toman de

la estructura *Clientes*, y el código, nombre y precio de los artículos comprados, que se toman de la estructura *Artículos*. Las subestructuras se usan para generar internamente tablas que serán entradas en muchos procesos.

Las subestructuras interesan al diseño de formularios de entrada cuando se procesan parte de los datos de un registro. En la ilustración anterior, si la tabla estuviera compuesta por más campos que los indicados, pero hiciera falta procesar los que muestra el formulario, sería necesario diseñar éste. Se estaría diseñando un formulario para una subestructura.

Procesamiento

Las entradas generadas en el entorno se producen con independencia del sistema. Pero éste debe capturarlas en algún momento. El tiempo que transcurre entre la generación y la captura se puede estirar dentro de ciertos límites o debe ser instantáneo. Esto permite distinguir entre procesamiento por lotes y en tiempo real.

Procesamiento por lotes. Consiste en tratar en un momento dado un grupo o lote de entradas similares, que fueron previamente acumuladas durante un período. Se usa cuando no se requiere un resultado inmediato por cada entrada. Usando computadoras, un ejemplo común son las novedades de empleados que el Departamento de Personal remite a un centro de cómputos: faltas, tardanzas, nacimientos, casamientos, suspensiones, etc., se acumulan durante un mes en la oficina de Personal y se procesan en conjunto en vistas a la liquidación de sueldos. El procesamiento por lotes suele requerir el diseño de formularios con este propósito.

Procesamiento en tiempo real. Consiste en procesar cada entrada en el momento en que se presenta. Así, el registro de una operación bancaria debe ser instantáneo. Por ejemplo, cierto día un cliente puede depositar en su cuenta una cantidad que, sumada a su saldo anterior, da $1000. Luego, ese mismo día, se presentan tres portadores de cheques contra esa cuenta por $500, $600 y $200. El banco debe rechazar el pago del segundo y tercer cheques, por falta de fondos. Si el banco procesara por lotes los movimientos al final del día, advertiría el descubierto cuando fuera demasiado tarde. La necesidad de contar con información actualizada tan pronto como se produzca la transacción en casos como el del ejemplo, estimuló el desarrollo del procesamiento en tiempo real, que históricamente fue posterior al procesamiento por lotes.

Movimientos

Toda tabla, maestra o transaccional, sufre variaciones a lo largo del tiempo. Estos cambios, que afectan a los registros de la tabla, se llaman movimientos o novedades. Los movimientos son de tres tipos: altas, bajas y cambios.

Altas. Son nuevos registros que se incorporan a una tabla. Una tabla maestra de artículos, por ejemplo, se incrementa con un nuevo registro por cada artículo nuevo que empezamos a vender, donde se describe su código, nombre, precio unitario, existencia, punto de pedido, etc. En una tabla transaccional, como la de facturación, cada nueva factura originará uno o varios registros nuevos.

Bajas. Son registros que se eliminan de la tabla, porque ya no forman parte de ella. Por ejemplo, si dejamos de comercializar un artículo, cuando ya no se hagan más

referencias a él, su correspondiente registro maestro debe ser eliminado.

Cambios o modificaciones. Son registros que sufren actualizaciones en uno o más campos. Por ejemplo, si un artículo cambia su precio, no debemos eliminar su registro, sino cambiar el contenido del campo correspondiente. Las actualizaciones normalmente se dan en campos que no son clave.

Una clave es uno o más campos que identifican un registro. Cada registro tiene su propia clave. Para clientes, artículos o personas, basta con un campo para identificar cada registro, como son el número de cliente, el código de artículo y el documento de identidad. En otros casos hace falta un grupo de campos, como en una universidad donde las matrículas de los alumnos comienzan por 1 en cada carrera. En esta situación, la matrícula es insuficiente para identificar a un alumno, haciéndose necesario usar también como clave la carrera. Las claves sirven para construir índices que no admiten valores repetidos.

Al procesar los movimientos hay que hacer un control básico con las claves. Las altas exigen que las claves de los registros a incorporar no existan previamente en el archivo. Las bajas y cambios, por el contrario, exigen que existan. Una alta con una clave ya existente es un error. Una baja o un cambio para una clave inexistente también es un error.

Los movimientos pueden ser procesados en tiempo real o por lotes. En el primer caso, los controles mencionados en el párrafo anterior son evidentes para quien captura el movimiento. Las fuentes de las novedades pueden ser variadas, como datos brindados por el interesado, documentos probatorios que nos presenta o formularios llenados al efecto. Para la captura se usan formularios en pantalla.

En el procesamiento por lotes, más formalizado, se usan formularios en papel con las novedades, que luego se transcriben a un archivo actualizador. Se puede usar un formulario para altas, otro para bajas y otro para cambios. También puede emplearse un solo formulario para los tres movimientos, donde hay un campo que indica para cada registro de qué movimiento se trata. Por años ha sido común usar cuadrículas como la que sigue.

Novedades de artículos
Período:

Código	Descripción	Precio	M

En cada casilla se escribe un solo carácter. Cada renglón se usa para un registro. Cada campo ocupa una cantidad constante de casillas contiguas. En las ilustra-

ciones, *Código* es el campo clave. *M* es el campo que indica el tipo de movimiento, mediante un código.

Por economía y precisión, conviene seguir las siguientes prescripciones:

- Para una alta, se llenan todos los campos.
- Para una baja, sólo se llena la clave y el tipo de movimiento.
- Para un cambio, se requiere la clave, los campos que han variado y el tipo de movimiento, dejando los otros en blanco. Llenar campos que no han cambiado es trabajo inútil y puede producir errores de transcripción.

Los campos del archivo actualizador son los mismos que los campos del archivo a actualizar, más el campo adicional *M*. Cuando se procesen las novedades, se usarán ambos archivos. El programa tomará cada novedad del archivo actualizador, les aplicará los controles básicos y, de ser correcta, afectará el archivo a actualizar. En el caso de un cambio, sólo modificará los campos con contenido, ignorando los que están en blanco.

Formularios

Las entradas pueden presentarse en distintos soportes, siendo los formularios la modalidad más común. Podemos clasificar los formularios en independientes del sistema, preparados para el sistema o preparados por el sistema.

Formularios independientes del sistema. Son totalmente independientes del sistema, pero contienen datos necesarios para él. Por ejemplo, las facturas o notas de débito de un proveedor que afecta nuestro sistema de compras, el documento de identidad del que se toman datos para el sistema de afiliados de una obra social, la escala de montos y tasas del impuesto a las ganancias que aplicamos a nuestros empleados en el sistema de liquidación de sueldos. Los datos que usa el sistema son una transcripción total o parcial de esos formularios, por lo que los errores pueden ser elevados. Hay que aceptar estos formularios tales como son, sin posibilidad de modificar su diseño.

Formularios preparados para el sistema. Son diseñados específicamente para el sistema, sea en papel, sea en pantalla. Hay libertad para diseñarlos como se crea conveniente. Cuando son formularios de papel, se llenan manualmente antes de transcribirlos a tablas, como el ejemplo dado anteriormente para altas, bajas y cambios. Puesto que las posibilidades de error son grandes, si la naturaleza del formulario lo permite, se pueden introducir datos para validación posterior.

Los formularios en pantalla preparados para el sistema tienen la apariencia de formularios en papel, pero pueden ser dotados de características que los hace mucho más potentes. Veremos este aspecto más adelante, cuando tratemos formularios inteligentes.

Formularios preparados por el sistema. Son una variedad de los formularios preparados para el sistema, con una característica especial. Son formularios de salida, impresos por un proceso, que sirven de entrada a otro proceso del mismo sistema (caso de los tres talones de la factura de pago) o de los que se obtienen datos que se almacenan en tablas que posteriormente sirven para rendir cuentas (caso de factura de un solo cuerpo). Hemos hablado de este tema en *Impresión con códigos de barra*. Estos formularios significan una reducción drástica de transcripción, con lo cual los errores disminuyen en el mismo grado.

¿Por qué son formularios, si todos sus datos son llenados automáticamente durante la impresión? Porque se les agrega información que indica que se ha realizado el pago, como sellos o firmas o talones complementarios.

Formularios convencionales e inteligentes

Al hablar de formularios, imaginamos una hoja de papel impreso donde hay que escribir datos. Consta de textos indicadores de los datos a escribir y espacios contenedores para ellos. Aplicando este concepto, podemos reconocer como formularios también un juego de varias hojas, un cuadernillo, una libreta, una cartulina, una o varias hojas plásticas, un sello, una carpeta, etc. Todas estas variedades son lo que llamamos formularios convencionales. Admiten cualquier dato, sin poder controlar automáticamente su corrección.

En computación, las pantallas pueden usarse igual que el papel para representar formularios convencionales. Los programas pueden desplegar los textos en determinadas zonas de las pantallas y pedir el ingreso de datos en otras. Hasta aquí las cosas serían iguales. Pero los programas pueden controlar los datos en el mismo acto de ingresarlos, rechazando los erróneos. El aprovechamiento de esta capacidad introduce una revolución: son los llamados formularios inteligentes, no porque sean inteligentes en sí mismos, sino por la inteligencia introducida en los programas que los manejan. Las posibilidades de estos nuevos formularios van más allá del control. Tan importantes son que ya no hay de un programa que dibuja un formulario, sino que el formulario es el programa.

Posibilidades de los formularios inteligentes

Consideremos algunas posibilidades que marcan una diferencia gigantesca con los formularios convencionales.

Objetos de formulario. Un formulario inteligente es una ventana, dentro del cual se pueden colocar otras ventanas predefinidas en cuanto a propiedades y funcionalidad. El formulario y sus ventanas contenidas se llaman objetos o controles. Como el formulario es un programa, se distinguen dos tiempos: el de diseño y el de ejecución. Los objetos pueden llevar programación asociada, no visible cuando se ejecuta el formulario, pero que se activa cuando ocurren determinados eventos, como dar clic, doble clic, arrastrar, escribir con el teclado, etc. Algunos objetos muestran el contenido de campos o variables y admiten modificar y validar el contenido mediante programación asociada. Vale la pena citar los siguientes.

ETIQUETAS. Sirven para introducir los textos fijos del formulario.

CUADROS DE TEXTO. Muestran el contenido de un campo de datos o de una variable, que puede cambiarse.

CUADRÍCULAS. Un formulario puede mostrar los campos de un solo registro. Para ver los campos de otro registro, se colocan botones de comando que llevan al registro siguiente, al anterior, al último, al primero o a uno cuyo número o valor clave se indica. Esto funciona bien cuando los registros son independientes uno de otro, como es el caso de registros maestros. Pero a veces varios registros están conceptualmente relacionados, como las líneas de una factura, las de notas de crédito de un proveedor, o las de materias cursadas por un alumno. En este caso, resulta más efectivo

presentar varios registros a la vez, porque se tiene la visión del conjunto. El objeto adecuado para visualizar varios registros a la vez son las cuadrículas, que muestran varios registros y campos de una tabla, o de varias tablas relacionadas, en un arreglo de filas y columnas. Entre otras posibilidades, permiten modificar los campos; dar altas y bajas de registros; cambiar el orden de presentación de los registros; mostrar únicamente los registros de una tabla secundaria correspondientes al registro seleccionado en una tabla primaria.

CUADROS DE LISTA Y CUADROS COMBINADOS. Ambos objetos manejan listas de filas (u opciones). La cantidad de filas no está limitada. Las filas pueden tener una o más columnas, cuyos valores se toman de una tabla, se escriben textualmente, etc. El cuadro de lista presenta la lista desplegada, mientras que el cuadro combinado necesita que se pulse un botón para desplegarla. Esta diferencia permite usar uno u otro objeto, según el espacio disponible en el formulario. Al elegir una fila, se puede guardar el valor de una de sus columnas en un campo o variable. Este funcionamiento es de gran importancia para introducir códigos a un campo, elegidos indirectamente por lo que representan. Por ejemplo, para introducir el código de cliente a una factura, se puede usar un cuadro de lista de dos columnas: la primera tiene los nombres de los clientes, ordenados alfabéticamente, y la segunda sus códigos. Ambos valores se toman de los registros de la tabla maestra de clientes. Al confeccionar una nueva factura, que genera un registro en la tabla de facturación, para llenar el código de cliente se puede usar el cuadro de lista o el cuadro combinado: el vendedor escribe unas pocas letras iniciales del nombre y la lista salta a los nombres que comienzan por esas letras, de modo que se puede ubicar el deseado. Elegido el nombre, el objeto guarda su código en el registro de factura. En la factura que se imprime, obviamente, se pueden escribir el código y el nombre del cliente.

Cuando hay tiempo para facturar, también se pueden usar un cuadro de lista o un cuadro combinado para los artículos que se facturan. El cajero escribe unas pocas letras iniciales del producto para que la lista se posicione en los nombres que comienzan con esas letras. Al elegir el nombre deseado, el objeto guarda el código y el precio unitario en la tabla de detalles de la factura.

En el caso de los supermercados, donde se necesita velocidad para facturar las compras de los clientes, se usa un procedimiento diferente, que prescinde de estos objetos. Cada producto trae impreso de fábrica datos escritos en códigos de barra. El cajero pasa el producto por un scanner, el programa toma el código, busca ese código en la tabla de artículos y obtiene el nombre y el precio, que escribe en el ticket. Nótese que el código no depende del supermercado, sino que es otorgado por una institución global, con sede en Bélgica.

GRUPOS DE OPCIONES. Son un conjunto de valores excluyentes entre sí, representados como botones y texto aclaratorio. Al elegir una opción, se puede guardar el texto o número del botón en un campo o variable. Si bien tiene un comportamiento semejante a un cuadro combinado o a un cuadro de lista, el grupo de opciones es más limitado. Primero, porque guarda el texto o el número del botón, no otro valor. Segundo, porque las opciones deben ser pocas. No podría usarse para codificar el cliente en una factura, salvo que fueran muy pocos clientes, del 1 al 10, por ejemplo.

MARCO DE PÁGINAS. A veces, las dimensiones del formulario pueden resultar insuficientes para los objetos que debe contener. El marco de páginas permite definir varias páginas, distribuyendo en ellas los objetos. Cuando se activa una página, sola-

mente se ven los objetos que ella incluye.

BOTONES DE COMANDO. Esconden programación, que se ejecuta al dar clic sobre ellos. Por ejemplo, un botón *Calcular* puede iniciar un proceso complejo de facturación del servicio telefónico prestado por la empresa en febrero a treinta mil usuarios; un botón *Imprimir* puede iniciar la impresión de tales facturas, etc.

CONJUNTO DE FORMULARIOS. Muchas veces puede convenir trabajar con varios formularios relacionados, en vez de a uno por vez. Puede ser que, de un formulario principal, mediante botones se abran otros formularios y que no se pueda volver al principal hasta que se cierre. También puede ser que estén abiertos simultáneamente y se pueda pasar de uno a otro según se necesiten, sin necesidad de cerrarlos.

Control. El control de la exactitud o admisibilidad (validación) de los datos que entran es fundamental y debe practicarse toda vez que sea posible. Si entran datos erróneos, las salidas que de ellos deriven serán erróneas. Los formularios pueden controlar la validez de los datos de entrada, denunciando los errores que se produzcan a través de mensajes adecuados, para corregirlos en el momento.

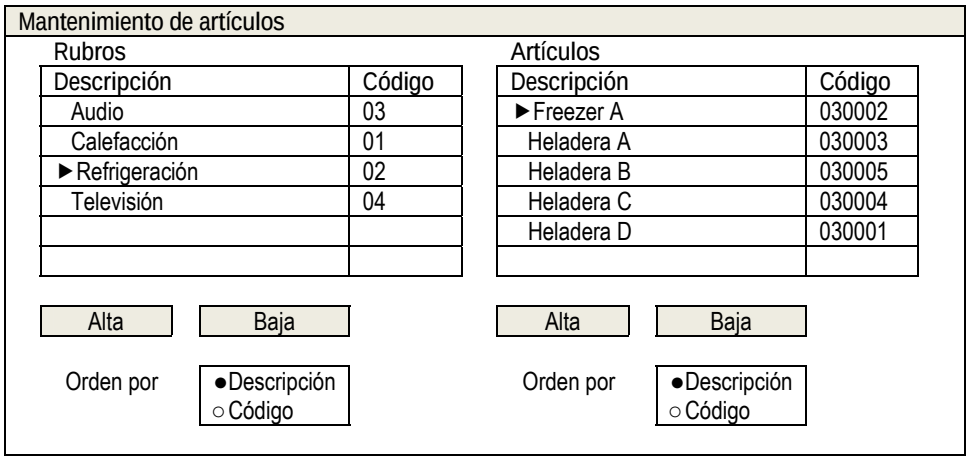
Codificación automática. Codificar es útil, pero la tarea de introducir códigos es complicada y está sujeta a errores. Un buen formulario puede facilitar esta tarea de dos formas. La primera consiste en usar códigos existentes. Así, si al ingresar una factura hace falta llenar el código de cliente con el valor correspondiente a un cliente existente, el formulario puede presentar un cuadro combinado con los nombres y los códigos de los clientes, tomados de la tabla maestra correspondiente. Mientras es indiferente que los códigos sean visibles, los nombres deben serlo, y conviene que estén ordenados alfabéticamente para facilitar su búsqueda. Cuando el operador selecciona un nombre, el formulario toma el código asociado y lo transcribe al campo respectivo de la factura. En este caso, el operador codifica indirectamente usando nombres, tarea más significativa que introducir códigos.

La segunda forma consiste en generar nuevos códigos automáticamente. Sería el caso de ingresar un nuevo cliente al archivo maestro, debiendo darle un código. Supongamos que el código de cliente sea un número correlativo. Si el código mayor existente es 537, al dar la alta el formulario busca ese número, le suma 1, y el resultado, 538, será el que corresponde al nuevo cliente. Los demás campos del registro, como el nombre del cliente, domicilio, teléfono, CUIT, etc., deben ser llenados por teclado. Esta forma de codificar es aplicable siempre que los códigos sigan una regla de formación, como es el caso de números o letras crecientes.

Aun cuando el código tenga varias porciones, si cada porción sigue una regla, puede aplicarse la generación automática. Supongamos que los códigos de artículo de una empresa tienen seis posiciones: las dos primeras para el rubro, las cuatro restantes para el artículo. Ambas partes se codifican con números, en forma creciente. El formulario de mantenimiento de artículos muestra los rubros en una cuadrícula y los artículos en otra, como indica la ilustración que sigue.

Si se da de alta un nuevo rubro, el formulario busca el rubro mayor y le suma 1. Este valor será el código del rubro de alta, cuyo nombre se debe ingresar por teclado. Por ejemplo, para dar de alta el nuevo rubro *Cocinas*, se pulsa el botón *Alta* (de *Rubros*). Esto inserta un registro con el código *05*. Habrá que escribir la descripción *Cocinas* para completarlo. Para dar de alta un nuevo artículo, se debe elegir previamente un rubro. El formulario busca entre los registros que comienzan con ese rubro

el que tiene número de artículo mayor, le suma 1 y da al rubro del nuevo registro el resultado de la suma. Por ejemplo, para dar de alta el artículo *Cocina XM*, hay que seleccionar el rubro *Cocinas* y pulsar el botón *Alta* (de *Artículos*). Esto inserta un registro con el código *050001*, que debe completarse con la descripción *Cocina XM*. Los rubros y los artículos aparecen ordenados alfabéticamente, para facilitar su búsqueda. Luego de dar las altas, ambas cuadrículas mostrarán los nuevos registros en el lugar que les corresponde. Opcionalmente, mediante botones de opción se puede cambiar el orden de *Rubros* o de *Artículos*, por *Descripción* o por *Código*.



Los puntos que siguen (ordenamiento, filtros y búsquedas), si bien están tratados acá, también son aplicables al diseño de salidas, porque son propios de formularios, y éstos pueden ser tanto de entrada como de salida.

Ordenamiento. Una tabla puede tener varios índices. Al activar uno de ellos, los registros se acceden en el orden que tal índice establece. En un formulario, mediante programación oculta, se puede cambiar el índice activo. Si se visualiza un registro por vez, el cambio de orden se verá al moverse al registro siguiente o anterior; si se visualizan varios registros a la vez, el cambio de índice se notará en el grupo de registros que se están viendo. El cambio de índice da mucha flexibilidad a los formularios, que permiten, por ejemplo, visualizar el archivo de clientes por nombre, código, CUIT, domicilio, provincia, etc. La programación oculta que cambia el índice activo se puede asociar a botones de comando, a un grupo de opciones, a un clic en el encabezado de la columna de una cuadrícula por donde se quiera ordenar, etc.

Filtros. Los formularios permiten filtrar los registros con los cuales se está trabajando. Por ejemplo, si quisiéramos ver los clientes de una determinada región geográfica, o solamente los mayoristas, o los responsables inscriptos ante IVA, etc., podríamos colocar los filtros necesarios programando grupos de opciones, cuadros de lista, botones de opción, etc. Una vez establecido un filtro, sólo veremos los registros que lo satisfacen.

Búsquedas. Si necesitáramos encontrar un registro con ciertas características, es posible programar el formulario para introducir estas características y hacer que las busque. Por ejemplo, si una cuadrícula muestra el contenido de la tabla de clientes donde hay miles de ellos, aunque cambiemos el ordenamiento la búsqueda manual línea a línea será lenta. Para agilizarla, podemos usar un cuadro de texto para escri-

bir el nombre del cliente a buscar. Al dar Enter, la programación asociada activa el índice de nombres, realiza la búsqueda y selecciona el registro del cliente requerido, dentro de la cuadrícula. Si no usáramos la cuadrícula, sino cuadros de texto para mostrar los campos de un solo cliente, el procedimiento de búsqueda descripto sería similar, presentando en el formulario los datos del registro hallado.

Otras formas de entradas

Además de formularios, las entradas provenientes del entorno pueden residir en otros soportes o usar otros medios. Veamos algunos.

Archivos en soportes magnéticos. Son el caso de CDs o pendrives. Puesto que ya están grabados, se ahorra tiempo de captura o transcripción. Si los datos ya están validados, se evitan procesos de control. Hay muchos ejemplos. Así, las sucursales de una casa matriz pueden remitir tablas en CD, pendrive o por Internet. El sistema central toma estos archivos y agrega a sus tablas los datos que necesita, evitando transcripción manual. Otro ejemplo es el préstamo de datos entre instituciones. Una obra social puede necesitar el padrón de todos los prestadores de salud de San Juan. En lugar de transcribirlos desde los padrones de las asociaciones médicas, odontológicas, etc., puede obtener de ellas el envío de sus archivos de profesionales, con los campos de datos que necesita. Quizás la estructura de los archivos recibidos no corresponda exactamente a la que usa la tabla de prestadores de la obra social, pero tiene datos elementales que pueden ser copiados a ella, automáticamente o con pocos pasos de transformación. Un tercer ejemplo son archivos producidos por un sistema y aprovechados por otro, como los totales de ventas mensuales por sucursal, que se exportan desde el sistema de facturación al sistema de estadísticas, o de una tabla a una planilla de cálculo.

Tarjetas magnéticas. El caso más conocido es el de las tarjetas de crédito y de débito, que tienen registrado el número de cliente, la clave, la fecha de vencimiento, etc. Para una compra a pagar con tarjeta, un lector toma datos de ésta y los transmite a la computadora central de la empresa que la emitió. Esta autoriza o deniega la operación, proporcionando datos del titular a la computadora del vendedor para completar la factura.

Etiquetas con códigos de barra. Son de uso corriente en los supermercados. Los productos llevan estas etiquetas impresas en los envases. Para facturar, el cajero pasa cada envase por un lector de etiquetas (scanner), donde está escrito el código del artículo. El scanner trasmite este dato al servidor central del supermercado, que proporciona el nombre y el precio del artículo a la caja registradora del empleado. Con estos datos se confecciona el ticket para el cliente y se registra la venta en un archivo, para descargar las existencias y contabilizar la operación. Esta tecnología ha producido una reducción sustancial del tiempo de facturación, con menos errores y menos exigencias para los cajeros.

Diseño de archivos

La palabra archivo tiene muchas acepciones. Por ejemplo, se aplica a una institución u oficina que se encarga de conservar documentos, como Archivo General de la Nación, Archivo histórico de San Juan, etc. También se aplica al conjunto de esos documentos, y, en ocasiones, hasta al mueble donde se guardan. Para los sistemas de información, los archivos son un *conjunto de datos relacionados que se almacenan como un todo, bajo un nombre único*. El soporte donde residen los datos es indiferente, siendo los discos magnéticos los más usados.

De acuerdo al concepto anterior, hay gran variedad de datos relacionados que se almacenan como un todo. Un programa es un conjunto de instrucciones relacionadas en función de un propósito, por lo cual es un archivo. Un texto, formado por palabras que tratan sobre un tema, es un archivo. Los datos que identifican a los proveedores son un archivo. Los datos que definen una imagen o una planilla de cálculo, también son archivos. En los sistemas de información, los archivos más importantes son las tablas. Tienen un aspecto como el que sigue:

Nombre	Domicilio	Teléfono	Ciudad	Provincia
Gutiérrez, Pedro	Salta 449	23-3345	San Salvador	Jujuy
Bello, Alicia	Luis M. Campos 4521, 7B	243-4487	La Plata	Buenos Aires
Santos, Mauricio	Quintana 673, 1ª	21-1289	San Luis	San Luis

Cada fila es un registro, formado por valores de atributos de una misma entidad (en este caso, las entidades pueden ser clientes, o suscriptores, o pacientes, etc.). Cada columna es un campo, que representa un mismo atributo. Cada intersección de una fila y una columna está ocupada por un dato, de modo que un dato es el valor que adquiere un atributo para una entidad. Un campo, entonces, es una variable. Nótese que todas las entidades están caracterizadas por los mismos atributos, es decir, tienen la misma estructura. En el archivo, se graba un registro tras otro, cada registro con sus campos componentes. Físicamente, entonces, se dice que el archivo está compuesto por registros y los registros están compuestos por campos. Al crear una tabla, primero se debe definir su estructura, tras lo cual se pueden agregar registros. La definición estructural es un componente físico de las tablas, que se almacena al comienzo de tales archivos.

La inclusión de la estructura en la tabla fue un adelanto trascendente en computación. Anteriormente, todos los programas que usaban una tabla dada debían incluir una descripción de su estructura. Si se cambiaba la estructura de la tabla, agregando, modificando o suprimiendo campos, todos los programas que la usaban quedaban afectados y debían ser modificados. Hoy, una modificación estructural no afecta indiscriminadamente a todos los programas que emplean la tabla: sólo aquéllos que hacen referencia a campos eliminados o modificados sufren el impacto. Esto es posible porque los programas de bases de datos no describen las estructuras, sino que hacen referencias a nombres de campos que existen explícitamente definidos en la estructura de la tabla, con sus tipos, tamaños, decimales, etc.

Un archivo tiene valor no solamente por los datos que contiene, sino por la facilidad que ofrece para recuperar los datos necesitados en un momento. De poco serviría un archivo muy completo si no tuviera una manera sistemática de encontrar lo que se necesita. El tema de la recuperación de datos es crítico, sean los archivos manuales o computarizados. En los primeros, implica un gran trabajo de organización. Mientras más organizado esté un archivo, más fácil será la recuperación, o como también se expresa, menor será el *tiempo de acceso*. Tratándose de archivos

computarizados, este aspecto está altamente automatizado: la organización se logra mediante índices y el acceso es prácticamente instantáneo.

Tipos de archivos

Distinguiremos distintos tipos de archivos, lo cual nos ayudará en el diseño. La mayoría de estos tipos se refieren a tablas. En un sistema de bases de datos, además de tablas, hay otros archivos.

Entidad almacenada

Según este criterio, las tablas se distinguen en archivos maestros y de transacciones.

Tablas maestras. Son tablas que contienen entidades maestras. Las entidades maestras son como los actores de una obra teatral. A lo largo de la obra pueden intervenir en distintas oportunidades, relacionándose entre sí por distintos intereses. Por ejemplo, clientes, proveedores, artículos, bienes de uso, socios, accionistas, empleados. Las entidades de una misma naturaleza, como clientes, tienen atributos comunes que las definen. Para ellas se crea una tabla de estructura constante. Cada entidad necesita un único registro, que debe ser identificado por un campo, llamado *clave*, como el número de cliente, de proveedor, de empleado.

En ocasiones, para obtener una clave para cada registro no alcanza un campo, sino que debe recurrirse a varios que, considerados como un todo, identifican cada registro. Sea una tabla que contenga un registro para cada combinación provincia – departamento de Argentina. El campo *Provincia* por sí solo no puede identificar a un registro, porque aparece varias veces en el archivo. Tampoco el campo *Departamento*, porque en distintas provincias hay departamentos que se repiten, como San Martín, que existe en San Juan, Mendoza, Buenos Aires. En esta circunstancia, hay que usar ambos campos para lograr la identificación de cada registro.

Tablas transaccionales. Son tablas que contienen transacciones. Las entidades maestras se relacionan de diversas formas, en distintas condiciones, con distintos propósitos. En una compra, por ejemplo, los artículos se relacionan con los proveedores: un mismo artículo puede ser suministrado por un mismo proveedor, pero en fecha, cantidad y precio diferentes. En una venta, los artículos se relacionan con los clientes: el mismo artículo puede ser vendido al mismo cliente, pero en fechas, cantidades y precios distintos. Cada uno de estos encuentros es una transacción. Los actores pueden ser los mismos, pero hay circunstancias que diferencian un encuentro de otro. Cada transacción requiere un registro, y a veces necesita varios, como veremos. Se represente en uno o varios registros, la transacción siempre debe tener una forma de ser identificada, por uno o varios campos. En los ejemplos anteriores, el identificador es el número de factura del proveedor o el número de factura por la venta a un cliente.

Es fundamental diferenciar estos dos tipos de tablas. Mezclar en un archivo elementos maestros y transaccionales complica la lógica y la programación. Veamos un ejemplo donde se mezclan elementos de ambos tipos, con las consecuencias no deseadas que puede traer.

Factura	Fecha	Cliente	Artículo	Cant.	Precio	Artículo
654213	15-08-01	González, Luis A.	Juego sábanas 1 p	5	18	Medias

654214	15-08-01	Pedro García	Par de medias	10	7	Funda t. chico
654215	15-08-01	Luis González	Funda chica	8	11	Camisa m. cortas
654216	15-08-01	Martín, José	J. sábanas 1 plaza	10	81	Pañuelo

Podemos observar varias desventajas:

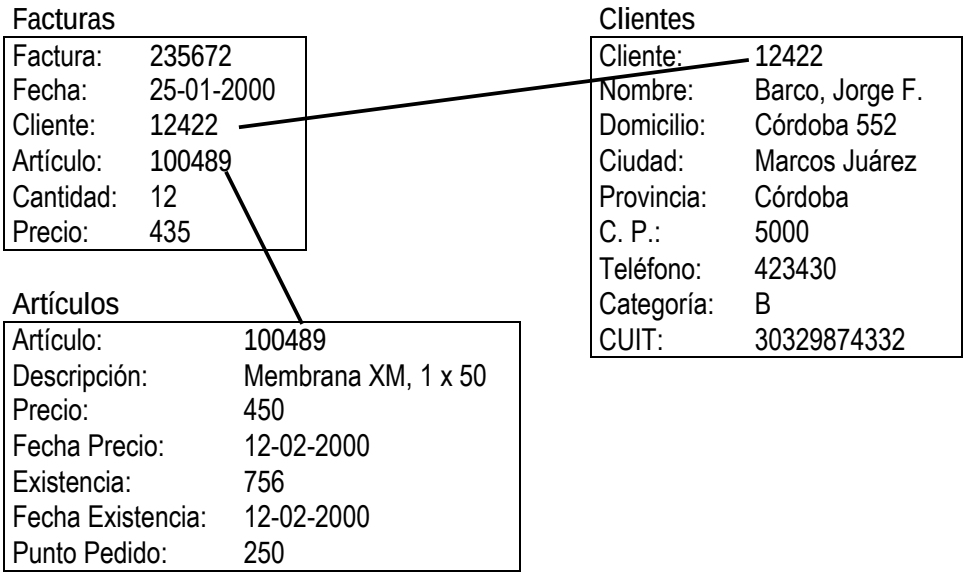
- El cliente *González, Luis A.* es el mismo que *Luis González*. Si ordenamos el archivo por el campo *Cliente*, ambos registros no aparecerán contiguos. Lo mismo sucede con *Juego sábanas 1 p* y *J. sábanas 1 plaza*, *Funda chica* y *Funda t. chico*, *Medias y Par de medias*. En los dos últimos casos hay otro inconveniente, que veremos al estudiar registros unitarios.

- El precio de las sábanas figura en una factura por $18 y en otra por $81. Como ambas son del mismo día, hay un error de transcripción en alguna.

- Escribir repetidamente nombres y precios es agotador y produce errores.

- Si queremos saber el domicilio o teléfono de cliente, no tenemos forma de obtener estos datos, pues todo hace sospechar que no hay un archivo de clientes.

Podemos mejorar el ejemplo creando dos tablas maestras, *Clientes* y *Artículos*, y uno transaccional, *Facturas*. Cada entidad maestra se define una vez, con mayor riqueza de datos, y se identifica con una clave interna (cliente 12422 y artículo 100489). Cada transacción se identifica con una clave interna (factura 235672) y necesita una referencia a las entidades maestras mediante claves externas (cliente 12422 y artículo 100489). En la ilustración siguiente, a partir de una transacción se pueden obtener todos los datos de las entidades maestras, con economía de almacenamiento y redundancia mínima de datos.



El campo *Precio* aparece en las tablas *Facturas* y *Artículos*. Esto no es una redundancia. La factura se confeccionó el 25-01-2000, cuando el artículo valía $435. El 12-02-2000 el precio varió a $450, como indica el registro del artículo. La factura

debe llevar el precio al momento de la venta, no el actual. La idea queda más clara si se dice que el campo *Precio* de la tabla *Facturas* es el precio facturado, que inicialmente se toma del campo *Precio* de la tabla *Artículos* y que queda fijo para siempre. El campo *Precio* de la tabla *Artículos*, por su parte, es el precio vigente del producto, que puede variar a través del tiempo.

La disquisición anterior sobre el precio haría que en un diccionario de datos deberían definirse dos entradas: *Precio facturado* y *Precio de venta*.

Procesos de actualización

Toda tabla sufre alteraciones, sea porque sus datos varían a lo largo del tiempo, sea porque se detectan errores. Si se producen estos cambios, una tabla debe ser actualizada lo antes posible. Las variaciones que sufren las tablas se conocen como *altas* o *bajas*, según se incorporen registros nuevos o se eliminen registros existentes; las variaciones que sufren los campos de un registro, sin que se elimine éste, se conocen como *cambios* o *modificaciones*. De acuerdo a estas variaciones, podemos distinguir dos tipos de tablas: actualizables y actualizadoras.

Tablas actualizables. Tanto las tablas maestras como las transaccionales sufren transformaciones. En las primeras, se producen normalmente por el sólo transcurso del tiempo. Vendemos nuevos artículos que debemos incorporar; dejamos de vender artículos que debemos eliminar para reducir nuestro almacenamiento; cambian los precios, o las existencias a una fecha. Otras veces las transformaciones se originan en errores que necesitamos corregir, como una descripción incorrecta o con errores ortográficos. En las transacciones, normalmente son debidas a errores. Sabemos de la importancia de los controles en la entrada de datos para reducir estos errores. Pero siempre existe la posibilidad de ellos. Sea por cualquiera de las causas indicadas, toda tabla maestra o transaccional debe poder actualizarse. La actualización necesita de procesos que cumplen específicamente esa función.

Los efectos de las altas, bajas y cambios pueden reducirse a la tabla que se actualiza, sin afectar a otras. Sin embargo, cuando hay tablas relacionadas, la actualización de una puede afectar a los demás. Por ejemplo, una tabla transaccional está relacionada con una o más tablas maestras por campos clave. En el diagrama anterior, *Clientes* se relaciona con *Facturas* por el campo *Cliente* y *Artículos* con *Facturas* por el campo *Artículo*. Las situaciones a considerar son las siguientes:

- Baja de un registro de *Clientes* que tiene registros en *Facturas* (eliminación).

- Cambio de un número de cliente en *Clientes* que tiene registros en *Facturas* (actualización).

- Alta o cambio de un registro en *Facturas* con un número de cliente inexistente en *Clientes* (inserción).

Las mismas situaciones se dan entre *Artículos* y *Facturas*. Lo que importa es decidir si se van a ignorar, restringir o reflejar en cascada los movimientos introducidos en una tabla en la otra tabla de la relación. Las tablas de bases de datos permiten poner en vigor estas decisiones fácilmente, sin necesidad de desarrollar programación específica, a través de reglas de integridad referencial.

Tablas actualizadoras. Contienen altas, bajas y cambios que se procesan por lotes. Las novedades se acumulan durante un período, se graban en una tabla y se procesan en un momento. Los campos de estas tablas son los mismos que los campos de las tablas a actualizar, más un campo que indique de qué novedad se trata. Si la actualización es en tiempo real, esta tablas son innecesarias.

Aunque sea preferible trabajar en tiempo real, en ciertos casos hay que procesar por lotes. Por ejemplo, si el directorio de una institución admite nuevos afiliados en su reunión mensual, la actualización de la tabla maestra de afiliados se hará una vez al mes. Podría ser que la actualización se hiciera tomando las solicitudes aprobadas e introduciéndolas por teclado, sin necesitar un archivo actualizador. Pero si la cantidad de altas es elevada y requiere validaciones complejas, puede convenir el uso de una tabla actualizadora.

Cuando las tablas actualizadoras son sometidas a procesos de validación, es posible encontrar errores que deben corregirse. En este caso, estos archivos actualizadores son a su vez archivos actualizables.

Vida útil

No todas los archivos tienen la misma duración: los hay que se mantienen por años; otros, en cambio, tienen una existencia efímera. Las precauciones a tomar con respecto a los archivos dependen en parte de su vida útil.

Permanentes. Son archivos que duran tanto como el sistema. Sufren variaciones a través de altas, bajas y cambios, sin dejar de ser conceptualmente los mismos. En raras oportunidades no sufren modificaciones, como podría ser un archivo que contenga los códigos y nombres de los elementos químicos.

Temporales. Son archivos de vida corta, como los que crea internamente un programa para obtener un resultado; los que crea un programa que usa y destruye otro; los que genera una consulta; los que son creados por aplicaciones como Access, VFP, Excel, Word, etc. Es recomendable que los archivos temporales sean destruidos por los programas automáticamente, para que no ocupen espacio.

Contenido

Las tablas son los archivos típicos de un sistema de base de datos, pero no los únicos. Hay de muchos tipos. Nótese que en VFP son archivos reales, mientras que en Access son virtuales, porque se guardan en la base de datos. Los principales son:

Tablas. Contienen registros de entidades maestras o transaccionales. Poseen dos partes: la definición de la estructura y los registros de datos. Todos los registros tienen la misma estructura, por lo que todos sus campos son de longitud fija. Las tablas pueden incluir características adicionales, como validación de campos, validación entre campos, máscaras, relaciones permanentes con otras tablas, reglas de integridad referencial, etc.

Archivos memo. Contienen texto sin límites de extensión que desarrollan un o varios campos distintos de datos de una tabla. Superan la limitación de los campos de longitud fija de las tablas. Pueden guardar imágenes, sonidos, hojas de cálculo, documentos de texto, etc.

Índices. El ordenamiento de los registros de una tabla responde a numerosas necesidades, dependiendo de los procesos a que serán sometidas. La tabla de facturas, por ejemplo, puede ser ordenada por un campo, como número de factura, cliente, fecha; por dos campos, como cliente y número de factura, vendedor y cliente; por tres campos, como vendedor, artículo y cliente; por cuatro, cinco, etc. El o los campos por donde se ordena se llaman *expresión* del índice, al que se le agrega el *sentido*, ascendente o descendente. La expresión puede ser un campo, una concatenación de campos o funciones sobre ellos.

Los índices pueden ordenar todos los registros o solamente los que cumplan una condición dada, que actúa como filtro. En VFP los archivos de índice pueden contener un solo índice, llamado *índice simple*, o varios, llamado *índice compuesto*. A su vez, una tabla puede contener un índice compuesto asociado a ella, de modo que sea abierto cada vez que se abra la tabla y sea actualizado automáticamente cada vez que se produzcan altas y bajas definitivas. Este tipo es el más ventajoso y se llama *índice compuesto estructural*.

Consultas. Contienen instrucciones en lenguaje SQL, de alto rendimiento. A partir de tablas u otras consultas (vistas en VFP) producen salidas de cualquier complejidad. Las salidas son tablas temporales, destruidas al dejar de usarlas, pero que pueden guardarse como tablas permanentes. Las consultas se construyen mediante un diseñador sencillo, fácil de aprender.

Programas. Contienen comandos de programación que realizan uno o varios procesos. Dentro de esta categoría hay que incluir formularios, informes y menús.

Bases de datos. Son tablas especializadas, que registran tablas, relaciones, conexiones, etc. Las bases de datos permiten definir reglas de validación interna de una tabla y reglas de integridad referencial (actualización, eliminación e inserción de registros entre tablas relacionadas).

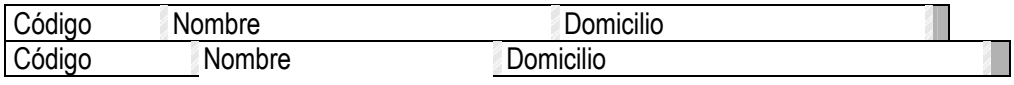
Tipos de registros

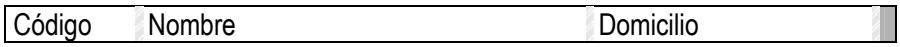
Los datos de entidades similares se guardan en registros. Estos pueden adquirir diversidad de formas, que dependen de las capacidades del lenguaje usado. Veamos algunas variedades.

Registros uniformes. Todos los registros contienen los mismos campos. La longitud de éstos puede ser fija o variable. Si la longitud es variable, se coloca una señal que indica el fin de cada campo y otra que indica el fin del registro. Estas señales son puntos y comas o caracteres especiales que no se pueden lograr con el teclado. Los gráficos siguientes ilustran lo dicho.

Código	Nombre	Domicilio

Registros uniformes de longitud fija



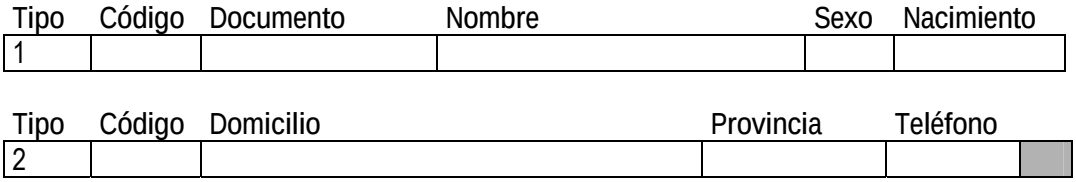


Registros uniformes de longitud variable

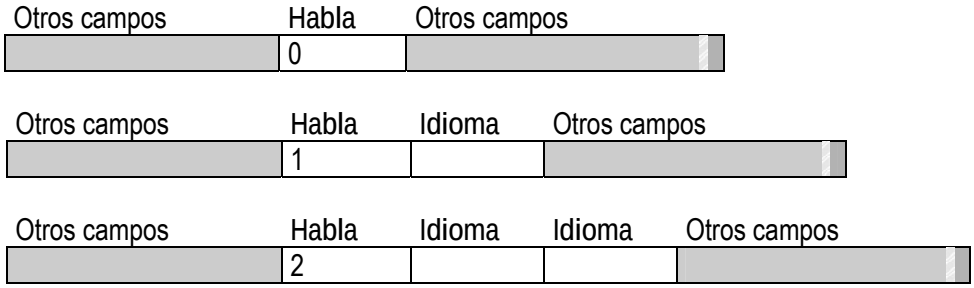
Las tablas de VFP y Access contienen registros uniformes de longitud fija. Si se necesita uno o más campos de longitud variable, su tipo debe ser memo. Un campo memo tiene longitud fija en la tabla, donde se guarda un número de referencia. La referencia se desarrolla en un archivo asociado, donde no hay limitación de tamaño.

Registros multiformes. Los registros contienen campos distintos, dependiendo de un campo que identifica el tipo de registro o de un campo que indica la cantidad de veces que se repite otro campo. La longitud de los campos puede ser fija o variable.

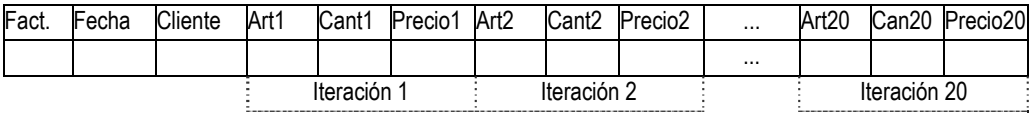
Como ejemplo del primer caso, sea que un campo indica el tipo de registro. Si contiene un 1, los campos siguientes son para documento de identidad, nombre, sexo y fecha de nacimiento. Si contiene un 2, los campos siguientes son para domicilio, provincia y teléfono.



Para el segundo caso, sea un registro de aspirantes a una beca con los campos nombre, especialidad, domicilio y cantidad de idiomas que conoce. Si este último tiene un 0, no se usa ningún campo adicional; si tiene un 1, hay un campo adicional para indicar cuál es el idioma; si tiene un 2, hay dos campos adicionales, uno para cada idioma, etc.



Registros no unitarios. Son registros que contienen un campo o un grupo de ellos que siempre se iteran, es decir, se repiten varias veces. Sea una factura que contiene los campos únicos *Factura*, *Fecha* y *Cliente*. Pero los campos *Artículo*, *Cantidad* y *Precio* son una subestructura que se itera, digamos 20 veces. Un registro no unitario para una factura contendría los datos únicos y los iterados.



Como los nombres de los campos no se pueden repetir en un registro, se han añado a los nombres los números 1, 2, …,20, en correspondencia con cada itera-

ción.

Los registros no unitarios tienen varias desventajas:

DESPERDICIO DE ESPACIO. Si una factura registra la venta de un artículo, se ocupará la primera iteración, quedando las diecinueve restantes vacías; si los artículos son dos, se desperdiciarán dieciocho, etc.

Sea que los campos del registro no unitario tienen los siguientes anchos:

Campo	Ancho
Fac	13
Fecha	8
Cliente	5
Art1	5
Cant1	3
Precio1	8
Art2	5
Etc.	

Para determinar el ancho de todo el registro, veamos que cada iteración ocupa en total 5 + 3 + 8 = 16 caracteres. Como son 20 iteraciones, el total ocupado por ellas es 20 x 16 = 320. Los campos que no se iteran (*Fac*, *Fecha* y *Cliente*) tienen un ancho de 13 + 8 + 5 = 26. El ancho total es, entonces, 26 +320 = 346.

Si el promedio de artículos vendidos en cada factura es de 5, se están desperdiciando en promedio 15 iteraciones por registro, es decir, 15 x 16 = 240 caracteres.

IMPOSIBILIDAD DE ORDENAR POR CAMPOS ITERADOS. Los registros pueden ordenarse sin inconvenientes por los campos que no se iteran; pero no por los campos iterados. Sea que el artículo *peine* fue comprado por 20 clientes. J veces fue al primer renglón de la factura, ocupando la primera iteración; K veces fue al segundo renglón, ocupando la segunda iteración, y así. Si se quiere ordenar el archivo por los artículos, como éstos están en 20 campos distintos y el ordenamiento debe hacerse por uno solo, será imposible. Si elegimos el primer campo *Artículo*, *peine* aparecerá J veces. Si elegimos el segundo, peine aparecerá K veces. Cosa similar sucederá si elegimos cualquiera de los restantes 18 campos.

DIFICULTADES PARA BUSCAR. La búsqueda de un artículo específico complica la programación. Si queremos detectar todos los clientes que hayan comprado el artículo *dedal*, por cada registro habrá que hacer hasta 20 preguntas para saber si el cliente lo compró.

Registros unitarios. Llamamos así a registros uniformes de longitud fija que no tienen iteraciones, evitando las desventajas del punto anterior. Toda iteración puede convertirse en registro unitario. Para ello, cada registro debe llevar los campos únicos y los iterados, pero estos últimos una sola vez. Siguiendo con el ejemplo anterior, el registro unitario de factura tendría los campos *Factura*, *Fecha*, *Cliente*, *Artículo*, *Cantidad* y *Precio*. Cada factura necesitará varios registros, tantos como artículos vendidos registre.

Factura	Fecha	Cliente	Artículo	Cantidad	Precio
1	10-10-01	25	170	8	24
1	10-10-01	25	809	5	63

1	10-10-01	25	556	4	7
2	10-10-01	731	029	6	21
2	10-10-01	731	672	5	38

La longitud de cada registro, ahora, siguiendo con los anchos de campo definidos anteriormente, es de 26 + 16 = 42. Si la cantidad de artículos promedio por factura sigue siendo 5, los cinco registros necesarios para una factura promedio ocuparán 5 * 42 = 210 caracteres, inferior a los 320 caracteres que mide el registro no unitario.

La siguiente tabla ilustra los anchos totales necesarios para una factura con 5, 6, 7, … artículos:

Artículos	**Ancho registro**	**Tamaño factura**
5	42	210
6	42	252
7	42	294
8	42	336
9	42	378
Etc.	Etc.	Etc.

Como vemos, cuando una factura incluya 8 o más artículos, se requerirá un total de caracteres mayor al que ocupa un registro no unitario.

El inconveniente ahora es que los campos únicos, *Factura*, *Fecha* y *Cliente*, que antes no se repetían en el registro no unitario, ahora se repiten en cada registro que desarrolla la factura. Si esto significa un desperdicio de almacenamiento, podemos partir la tabla en dos, digamos *Facturas* y *Detalles*, valiéndonos de un campo que podamos usar como clave, *Factura* en este caso. Un archivo tendrá los campos *Factura*, *Fecha* y *Cliente*; y el otro los campos *Factura*, *Artículo*, *Cantidad* y *Precio*. Ambos archivos se relacionan por el campo clave.

Facturas		
Factura	Fecha	Cliente
1	10-10-96	25
2	10-10-96	348

Detalles			
Factura	Artículo	Cantidad	Precio
1	170	8	24
1	809	5	63
1	556	4	7
2	029	6	21
2	672	5	38

El registro de *Facturas* ahora mide 26 caracteres y el de *Detalles* mide 29.

La siguiente tabla ilustra los anchos totales necesarios para una factura con 5, 6, 7, … artículos:

Facturas	Detalles	Artículos	Detalles x Artículos	Total
26	29	5	145	171
26	29	6	174	200
26	29	7	203	229
26	29	8	232	258
26	29	9	261	287
26	29	10	290	316
26	29	11	319	345
Etc.	Etc.	Etc.	Etc.	Etc.

Ahora, cuando una factura incluya 11 o más artículos, se requerirá un total de caracteres mayor al que ocupa un registro no unitario.

Seguridad

Dado que los datos son un recurso valioso, deben ser protegidos. Un incendio, una inundación o las alimañas pueden destruir archivos en papel. En el caso de archivos computarizados también hay riesgos: pueden ser dañados por el campo magnético que produce una máquina, los cortes de energía, un intruso, etc. La seguridad se aplica a los archivos permanentes, sean originales o copias.

Originales. Son los archivos sobre los que se trabaja cotidianamente. En ellos reside la totalidad del sistema. Hay varias formas de protegerlos:

DEFINIR USUARIOS Y PERMISOS. Se establece lo que cada usuario puede hacer sobre los archivos, como leerlos, modificarlos, destruirlos, copiarlos, cambiarlos de carpeta. Al ingresar al sistema, el usuario debe dar una contraseña. Una vez identificado, se activan los permisos que le corresponden. Estas medidas de seguridad son propias de sistemas en redes, pero pueden ser imitadas en computadoras autónomas.

RESTRINGIR EL USO. Se puede exigir como norma de programación que los programas usen los archivos solamente para leerlos, salvo que deban actualizarlos.

USAR INSTALACIONES SEGURAS. Malos cableados, conexiones flojas, tensión inestable, cortes de energía, pueden provocar severos daños. La solución de estos problemas y el uso de aparatos adecuados, como UPS, reducen el riesgo.

OBTENER COPIAS DE SEGURIDAD. Es una prácticas muy sana, muchas veces descuidada. Hay que tener copias de todos los archivos del sistema, cualquiera sea su contenido. Los que sufren cambios frecuentes deben ser copiados con mayor asiduidad. Las copias pueden ser en discos, CDs, pendrives, en una nube, etc. Se puede automatizar un calendario de copiado, de modo que, llegado el momento de generar copias, el sistema solicite lo necesario.

Copias. Son las indicadas en el párrafo anterior. Si un archivo original sufre daños irreparables, se lo puede reponer a partir de la copia. Mientras más actualizada sea ésta, menos datos habrá que rehacer en el original. La importancia de las copias hace necesario protegerlas. Las medidas de seguridad consisten en mantenerlas en gabinetes a prueba de fuego y de campos magnéticos, en otro sector del edificio o en otro edificio. Si los datos contenidos son confidenciales, se pueden depositar en cajas de seguridad de instituciones especializadas.

Especificaciones de archivos

Las especificaciones de archivos son descripciones detalladas de lo que van a contener, destinadas al programador. Aquí hay un importante punto de contacto entre analista, programador y soft usado. De la clasificación que hicimos según contenido, no todos los archivos necesitan especificaciones. Veamos los que sí las requieren.

Tablas. Las especificaciones pueden darse en concordancia con el software a emplear o de un modo genérico. En el primer caso, el analista conoce las convenciones del soft para definir las estructuras de estos archivos. Por ejemplo, en VFP, las especificaciones de una tabla *Encfac* pueden ser como en el ejemplo siguiente.

Tabla: Encfac **Base de datos: Fac**

Campo	Nombre	Tipo	Ancho	Dec
1	FAC	C	13	
2	FEC	D	8	
3	CLI	N	5	0
4	VEN	N	3	0
5	ANU	L	1	

Campo	Título	Formato	Máscara	Validación de campo	Mensaje
1	Factura	!	A999999999999	No vacío	"Factura errónea"
2	Fecha			No vacío	"Fecha errónea"
3	Cliente		99999	Existente en tabla CLI	"Cliente inexistente"
4	Vendedor		999	Existente en tabla VEN	"Vendedor inexistente"
5	Anulada				

Validación de registro	Mensaje
La letra del campo FAC debe ser igual a la letra con que está definido el cliente en la tabla *Cli*.	"Letra de factura incorrecta"

El segundo caso se emplea cuando no se conoce el soft a emplear. La AFIP, por ejemplo, exige la presentación de declaraciones juradas del impuesto a las ganancias en disquetes. No le importa el soft que usan los contribuyentes, pero sí el formato en que deben presentar los datos. La tabla anterior, usando una notación genérica, puede tomar la forma que sigue:

Archivo: Encfac

Campo	Contenido	De	A	Tamaño	Comentarios
Factura	Caracteres	1	13	13	Primera posición: A, B o C. Demás posiciones: números
Fecha	Numérico	14	19	6	En formato aammdd
Cliente	Numérico	20	24	5	Ceros a la izquierda si hace falta
Vendedor	Numérico	25	27	3	Ceros a la izquierda si hace falta
Anulada	Caracteres	28	28	1	'S' cuando esté anulada

Índices. Los índices también dependen del soft empleado, por lo que sus especificaciones pueden ser de acuerdo a él o en términos genéricos. En el primer caso, los archivos necesarios para la tabla *Encfac* pueden establecerse de la siguiente forma:

Índice estructural compuesto de la tabla Encfac

Nombre	Tipo	Expresión	Sentido	Filtro
FAC	Principal	fac	Ascendente	
CLI	Normal	str(cli,5)	Ascendente	
VEN	Normal	str(ven,3)	Ascendente	
CLIFAC	Normal	str(cli,5)+fac	Ascendente	

Sin hacer referencia al soft, las especificaciones pueden ser:

Índices del archivo ENCFAC

Índice	Criterio
1	Factura
2	Cliente
3	Vendedor
4	Cliente y Factura

Integridad referencial. Cuando se trabaja con tablas de bases de datos, el analista puede establecer cuáles serán las tablas a unir con relaciones permanentes y qué reglas aplicar para los cambios de clave y las bajas y altas de registros. La integridad referencial depende del soft usado. Por ejemplo, para las tablas del sistema de facturación, trabajando en VFP, se puede especificar lo siguiente:

Tabla madre	Tabla hija	Cambios	Bajas	Altas	Índice madre	Índice hija
ART	LINFAC	Cascada	Restringir	Restringir	ART	ART
CLI	ENCFAC	Restringir	Restringir	Restringir	CLI	CLI
VEN	ENCFAC	Restringir	Restringir	Restringir	VEN	VEN
ENCFAC	LINFAC	Cascada	Cascada	Restringir	FAC	FAC

Nombres de archivos

Los nombres de archivos en disco siguen reglas de formación fijadas por el sistema operativo usado. Windows acepta nombres de hasta 255 caracteres. Prohíbe algunos caracteres, pero acepta espacios intermedios. Dentro de los límites de tales reglas, el analista debe dar nombres significativos a sus archivos.

Hay archivos que contienen todos los posibles registros de igual formato, por lo que el nombre es uno solo. El archivo de artículos que vende la empresa puede llamarse *Art*, con lo se define bien lo que contiene y podrá contener. Otras veces, aunque los formatos de registro sean iguales, se usan varios archivos, por una circunstancia que los diferencia. Así, los archivos de contabilidad se diferencian por los ejercicios que cubren o los archivos de sueldos por los meses que contienen. Si bien se podrían guardar en un solo archivo los registros de diferentes períodos, con el tiempo se iría haciendo enorme y poco eficiente. Como lo que interesa es manejar un período u otro, conviene que estén en archivos específicos.

Cuando hay varios archivos de contenido similar, deben tener nombres distintos. Para formarlos, conviene que sigan una regla. Los archivos contables podrían llamarse *Cont1998*, *Cont1999*, *Cont2000*, etc., donde las cuatro letras indican que se trata de contabilidad y los cuatro números indican el ejercicio. Los archivos de sueldos, podrían llamarse *S012001*, *S022001*, *S032001*, *A012001*, *A022001*, etc., donde las letras indican si se trata de sueldo o aguinaldo, los dos primeros números el mes o cuota del aguinaldo y los cuatro últimos el año. La regla de formación puede ser manejada por el programa que genera estos archivos, usando los datos del período, tipo de liquidación, etc., para formar automáticamente los nombres.

Diseño de procesos

En sistemas manuales, el diseño de procesos se materializa en procedimientos. Estos se pueden desarrollar en forma narrativa o gráfica (cursogramas), incluyendo tablas de decisión, lenguaje estructurado y otras herramientas. Todo este material, más los formularios y los tipos de salidas a producir, se documenta en manuales de procedimientos.

En sistemas computarizados, el diseño de procesos se materializa en programas. El material auxiliar, como formularios de entrada, diseños de salidas, diseño de archivos, codificación, lenguaje estructurado, etc., se distribuye o en la documentación específica de cada programa, o en la documentación del sistema. La documentación servirá a los programadores, operadores, auditores, directivos y gerentes.

No hay formas preestablecidas de cómo documentar los sistemas y los programas. Aunque se ha intentado alguna normalización, presentada a modo de sugerencias, el análisis de distintos sistemas y programas hace concluir que es muy amplia y libre. Depende de cada analista, de lo que exige cada institución y de las facilidades documentales del software que se emplea. Pero puede afirmarse con seguridad que es más ventajoso contar con documentación que carecer de ella, porque da continuidad a las tareas de mantenimiento, especialmente cuando se recurre a nuevos analistas y programadores para continuar lo que empezaron otros.

Nos limitaremos al diseño de procesos de sistemas computarizados.

Componentes de un sistema

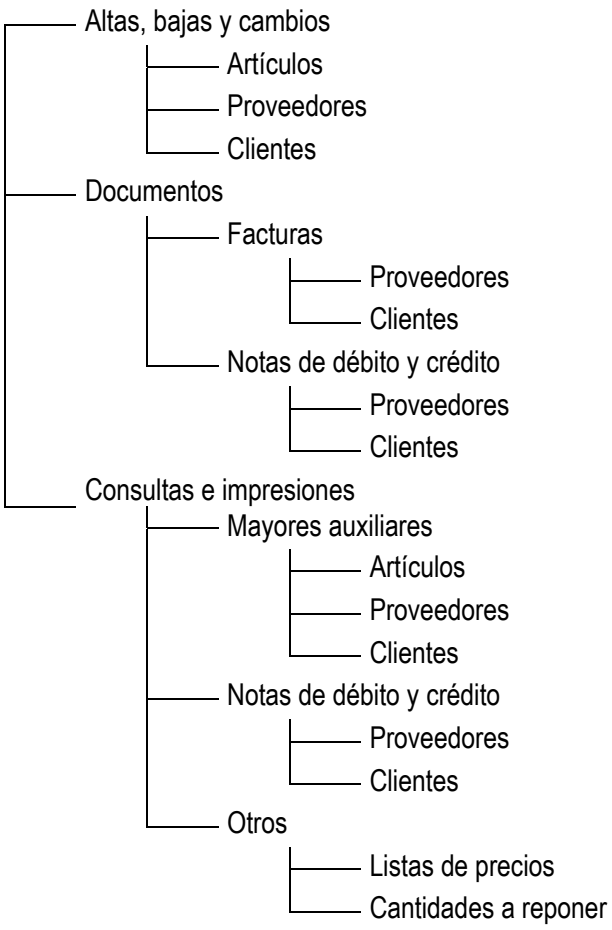
Un sistema está formado por subsistemas. Cada subsistema comprende una serie de opciones, cada una de las cuales ejecuta un programa específico. Cada programa realiza un requerimiento o un conjunto de ellos.

El lector puede reconocer estos componentes en una barra de menús: la barra representa el sistema, cada menú representa un subsistema y cada opción de un menú representa un programa a ejecutar. En ocasiones, una opción no lleva directamente a ejecutar un programa, sino que presenta una serie de sub opciones; pero éstas, finalmente, llevan a ejecutar un programa.

La división del sistema en subsistemas y la distribución de las opciones no es algo mecánico, sino que requiere de criterio, desarrollando un esquema que incluya los requerimientos de procesos. Un esquema lógico y lo más sencillo posible facilita establecer los programas necesarios. Por ejemplo, en un nuevo sistema de facturación, los requerimientos de procesos exigen que permita:

- Dar altas, bajas y cambios de clientes, proveedores y artículos
- Registrar facturas, notas de débito y crédito de proveedores y clientes
- Consultar e imprimir fichas de stock, mayores auxiliares de proveedores y clientes, precios, cantidades a reponer de los artículos, notas de débito y crédito a proveedores y clientes
- Facturar e imprimir ventas a clientes.

Un esquema de agrupación de lo anterior en subsistemas podría ser:



Otro esquema podría ser:

El primer esquema se basa en las operaciones que realiza el sistema. *Clientes*, por ejemplo, aparece disperso en varios subsistemas. El segundo esquema se basa en entidades maestras, por lo que resulta mejor. En el subsistema *Clientes*, por ejemplo, apare-

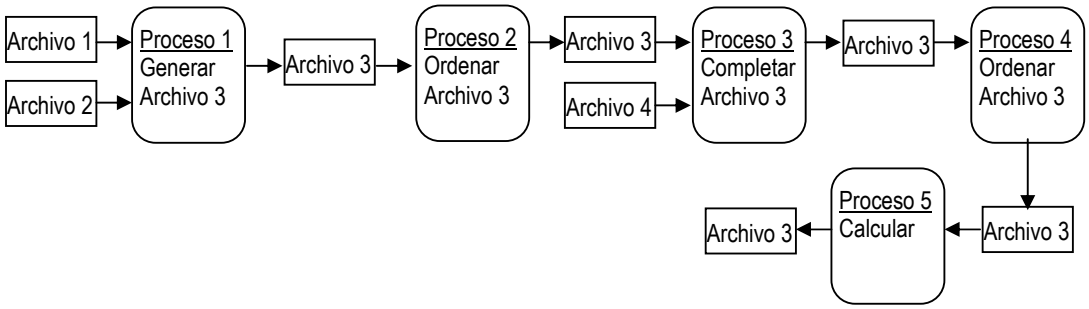
cen todas las operaciones que se pueden hacer sobre ellos.
xxxxx

Programas necesarios

Cualquiera de los esquemas anteriores orientan a establecer los programas. La totalidad del sistema se va a manejar mediante una barra de menús. En el segundo esquema, los menús serán *Artículos*, *Proveedores* y *Clientes*. Las opciones de cada menú serán las del esquema. Así, para el menú *Artículos*, las opciones serán *Altas, bajas y cambios*, *Mayores auxiliares (fichas de stock)*, *Listas de precios* y *Cantidades a reponer*. Al elegir una opción, el menú llevará a la ejecución de un programa. Esto evidencia que los programas necesarios están determinados por el esquema elegido.

Los programas deben incorporar el total de requerimientos de procesos. Note que nunca hay una correspondencia uno a uno entre un proceso y un programa. El caso normal es que un programa ejecute varios procesos. Otro caso poco frecuente es que un proceso pueda ser desarrollado en varios programas.

Veamos el primer caso. Si para producir un resultado se necesita una serie de procesos encadenados, uno tras otro, en estricta secuencia, sin omitir ninguno y sin intervención humana entre ellos, conviene que sean desarrollados como partes de un único programa. Su desarrollo como programas separados puede hacer que los operadores olviden ejecutar alguno, obteniendo seguramente resultados finales erróneos. Por ejemplo, una serie de procesos que generen un archivo X a partir de otros dos, ordenen ese archivo X, lo completen con datos de un cuarto archivo, ordenen nuevamente el archivo X por otro concepto y finalmente hagan cálculos sobre éste y los guarden en él, muy posiblemente conviene que sean desarrollados por un solo programa, en lugar de varios. El programa podría esquematizarse como sigue.



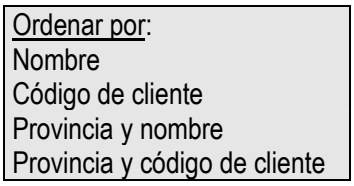
Veamos el segundo caso. Si un proceso como "imprimir intereses por mora de pagos incumplidos a una fecha", desarrollado por un solo programa puede crear trastornos, conviene que sea dividido en diferentes programas. Por ejemplo, si para imprimir los intereses hace falta calcularlos, y si el cálculo es largo y complejo, puede que el proceso retarde una impresora. Más todavía, los impresos tienen sus complicaciones: es común que un impreso largo deba rehacerse, porque el papel se atasque en la impresora, porque la tinta comience a agotarse y el listado se torne poco legible, porque algunas hojas se ensucien, etc. En estas circunstancias, deberá rehacerse parte del listado. Para ello sería necesario empezar el cálculo desde el principio, aunque el intervalo de hojas a imprimir sea cualquiera. Los inconvenientes se solucionarían usando dos programas: uno que haga los cálculos y los coloque en un archivo temporal y otro que los imprima. De esta forma, la lentitud del cálculo se dará una sola vez, pero se tendrá la posibilidad de impresión cuantas veces se desee, con independencia del cálculo. Si se elige esta solución, en el menú correspondiente deberá haber dos opciones: una para realizar el cálculo y otra para imprimir.

Recursos generales utilizables por los sistemas y los programas

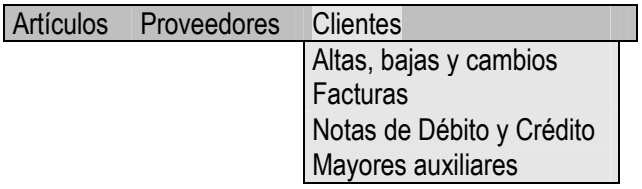
Veamos algunos recursos generales para aumentar la calidad de un sistema.

Menús. Los menús permiten elegir una opción entre varias. Un menú se puede usar dentro de un programa o un menú puede llevar a ejecutar distintos programas.

El primer caso es un menú *emergente* de opciones verticales. Estos menús, muy usados en las aplicaciones de Windows, pero poco comunes en aplicaciones comerciales, son presentados por el programa al cumplirse ciertas condiciones o al ser invocados a voluntad del usuario dando clic secundario sobre un objeto.



El otro caso es presentar la totalidad de un sistema en una barra de menús, como venimos viendo. Al elegir un menú, se despliega una lista de opciones. Al elegir una opción, se ejecuta el programa que ella invoca. Una opción, si hace falta, puede desplegar un conjunto de opciones verticales; éstas, a su vez, sus propias opciones verticales, y así.



Existe la posibilidad de asociar comentarios a cada opción horizontal o vertical del menú. Estos comentarios permiten describir brevemente qué produce el menú u opción en cuestión, siendo una forma rudimentaria de ayuda.

Ayudas. Amén de los comentarios asociados a opciones de un menú, a veces hace falta una descripción detallada de lo que hace un subsistema, un proceso, los datos necesarios en una entrada, la información adicional que requiere un proceso, etc. Esto puede desarrollarse en un archivo de ayuda asociado al sistema, donde mediante textos y gráficos se aclaren las dudas del usuario. Un buen sistema de ayuda puede reemplazar con ventajas el uso de manuales impresos. Hay dos formas de ayuda.

CONTEXTUALES. También llamadas ayudas sensibles al contexto, tienen la virtud de mostrar ayuda específica sobre el sector del sistema donde se está trabajando, cuando son invocadas (es tradicional invocarlas con la tecla F1).

NO CONTEXTUALES. Al ser llamadas presentan un índice de tópicos sobre los que brinda ayuda. Para llegar al tópico requerido, hay que dar uno o varios pasos intermedios para ubicarlo.

Mensajes. Los mensajes son una forma de poner en conocimiento del usuario directo lo que está sucediendo durante la ejecución de un programa. No dependen de la voluntad del usuario, sino que son dados automáticamente por el lenguaje de programación o por el sistema operativo. Por ejemplo, si se quiere hacer una división por 0, seguramente habrá un mensaje de error; o si se pretende copiar a un pendrive y la unidad no está preparada, tendremos un mensaje de advertencia. Hay otros mensajes que sí pueden ser

creados en los programas del nuevo sistema. Se usan con distintos propósitos, como se indica a continuación.

MENSAJES DE ERROR. Son los que se dan cuando se ha producido un error. Cada mensaje debe indicar claramente cuál es el error. Por ejemplo, si se están capturando datos de clientes y hay que ingresar el código de provincia al campo correspondiente, si el valor ingresado es incorrecto se puede dar un mensaje como el que sigue:

```
Código de provincia incorrecto. Debe variar entre 01 y 24.
```

Un mensaje de error puede obedecer a situaciones más complicadas. Así, si se registra el pedido de un alumno para cursar una asignatura, el programa debe verificar si tiene aprobadas o regulares las correlativas previas. Si no se satisface ese requisito, el mensaje puede ser como el siguiente:

```
Imposible cursar Estadísticas II, porque
    No tiene aprobadas
        Matemáticas I
        Matemáticas II
    No tiene regularizadas
        Estadísticas I
```

MENSAJES DE ADVERTENCIA. Advierten sobre las consecuencias de una acción a tomar. Por ejemplo, suponga que se quiere liquidar los sueldos de abril 2014. El programa liquidador generará un archivo de nombre S042014. Una liquidación puede rehacerse varias veces, por errores de conceptos liquidados o incorporación de nuevos conceptos, hasta que se dé por buena. Al comenzar la ejecución, el programa pide que se indique el período a liquidar. Luego, como es posible que la liquidación ya exista, es decir que ya haya sido generado el archivo S042014, el programa debe comprobar si existe. Si no lo halla, será porque es la primera vez que se liquida tal período y generará tal archivo. Si lo halla, conviene que dé un mensaje tal como

```
El archivo indicado ya existe. Si prosigue, será destruido.
¿Quiere proseguir? (S / N)
```

¿Por qué tal advertencia? Pues porque el operador puede equivocarse al indicar el período, de modo que si el programa no hiciera la advertencia, volvería a liquidarse algo que ya está listo.

MENSAJES DE AVANCE. Se dan para indicar lo que está haciendo un programa. Tienen un efecto psicológico positivo, porque el operador conoce que el programa está trabajando. Piense el lector en un programa que entra a ordenar un archivo muy grande y luego hace cálculos sobre él. Si esto dura, digamos, 45 minutos, el operador puede desesperarse si no tiene mensajes que le indiquen que el programa funciona. Para este caso, mensajes como los que siguen pueden calmar los nervios:

```
Ordenando S042014 por sector y categoría . . .
```

```
Calculando bonificación por antigüedad . . .
```

El mensaje podría consistir en ir dando la cantidad de registros que se llevan procesados, por ejemplo, 1000, 2000, 3000...; o indicar que se está trabajando sobre tal sector

y categoría; o cualquier forma apropiada que indique el avance.

MENSAJES DE RESULTADOS. Indican qué resultó de un proceso, para fines de control. Por ejemplo, si una acción se procesó con éxito o no; cuántos registros se procesaron en total; cuánto es la suma de uno o varios campos numéricos; cuántos registros se han grabado; etc.

Documentación de cada programa

Establecidos los programas que realizarán todos los procesos del sistema, es necesario documentar cada uno de ellos, de modo que sean la guía para que el programador puede crear el programa específico y una guía para entender lo que hace y modificarlo en el futuro. En sustancia, la documentación debe incluir lo siguiente.

Nombre. Todo programa se graba como un archivo, por lo cual debe tener un nombre, que sirve para invocarlo cuando se lo quiere ejecutar o modificar. Es deseable usar una codificación para los nombres. Por ejemplo, que represente el sistema o subsistema y un número distintivo. Para ello se debe estipular qué representa cada posición del nombre.

Subsistema y sistema. Referencia al subsistema y sistema a que pertenece el programa.

Objetivo. Descripción concisa y clara de los propósitos del programa.

Almacenes y flujos de datos. Referencia a las entradas y salidas finales e intermedias. Se debe mencionar los datos elementales a introducir por teclado, los formularios y archivos de entrada, los textos, tabulados, gráficos y archivos a obtener. Se puede hacer una representación esquemática mediante un diagrama de flujo de datos u otra técnica apropiada.

Procesos. Descripción breve y precisa de las actividades que debe realizar el programa, tales como validaciones, mensajes, cálculos, ordenamientos, actualizaciones, etc. Se debe hacer una referencia a tablas de decisión, instrucciones en lenguaje estructurado, sistemas de codificación, menús, ayudas, mensajes, formularios y objetos de que se va a valer el programa.

Formas de operación. Explicación detallada para los usuarios directos (operadores) de cuándo y cómo se debe ejecutar el programa, cuáles son los datos que deben introducir, qué tipo de papel se necesita, etc. Esta información puede ser escrita como parte de las especificaciones del programa o se puede colocar en un manual de operación. También puede escribirse en un archivo de ayuda, que el operador puede llamar a voluntad.

Anexos. Incluye los diseños de pantalla, impresos, formularios convencionales e inteligentes, sistemas de codificación, tablas de decisión, etc. Como este material puede ser exclusivo del programa o de tipo general, usado por varios programas, hay que incluir solamente el del primer tipo. El material general, como el diseño de archivos, debe ser incluido como documentación del sistema.

Documentación del sistema

Consiste en documentar las características generales del sistema. Junto con la documentación de cada uno de los programas, constituye el material que explica en qué consiste el sistema y cómo ha sido subdividido en subsistemas. Sus partes principales son las que siguen.

Descripción. Exposición de cuáles son las características y ventajas principales del sistema, los subsistemas componentes y los programas que integran cada uno. Basta con el empleo de palabras, pero suele incluir una descripción general en forma gráfica, mediante diagramas de sistema, herramientas que no desarrollamos.

Documentación general del sistema. Puede incluir varios capítulos. Así, es obligado que conste de una parte donde se presente el diseño de todos los tipos de archivos. Otro capítulo hace referencia a los sistemas de codificación general. Otro tiene que ver con la normativa general del sistema. Esta contempla las reglas a que debe atenerse toda la programación, como formas, tamaños y colores de objetos, informes, mensajes y ayudas. También las convenciones sobre teclas de acceso rápido, nombres, claves de seguridad, uso de archivos y producción de copias de seguridad. Un capítulo especial debe hacer referencia a los componentes no computarizados del sistema, que, por su importancia, merecen un comentario especial.

Componentes no computarizados. La tecnología ha reducido exitosamente la intervención humana, encargándose de actividades rutinarias que realiza con más precisión y rapidez. Pero, si bien la proporción de trabajo manual es notoriamente menor, es imposible eliminarlo totalmente, porque es el ser humano el que hace y usa los sistemas. Cuando decimos componentes no computarizados queremos significar formularios de papel, instrucciones sobre cómo llenarlos, procedimientos de oficina representados gráficamente a través de cursogramas o descriptos verbalmente, tablas de decisión que simplifican procedimientos con gran cantidad de acciones y condiciones, etc. De hecho, antes de la automatización y durante siglos, todo lo que existía eran sistemas de información totalmente manuales. Incluso hoy los encontramos en abundancia.

Un sistema de información, entonces, tiene dos componentes: uno no computarizado y otro computarizado. La tarea del analista, en consecuencia, debe cubrir ambas dimensiones. Desgraciadamente, es muy frecuente que un analista apunte únicamente a las computadoras, porque ignora el valor y la influencia de la porción no computarizada. Esto lleva a la producción de sistemas imperfectos, que no provocan sino impactos parciales. Se ha llegado a decir con ironía que un sistema informático dentro de un caos administrativo no hace sino aumentar el caos. El lector puede encontrar en nuestra historia nacional ejemplos que confirman tal aserción.

Los usuarios decisores suelen tener este mismo sesgo: piensan que un sistema de información es exclusivamente un sistema computarizado. A veces desestiman el desperdicio de recursos que supone la porción no computarizada. Con la seguridad que da el desconocimiento, suelen creerse capacitados para encarar reformas en ese terreno, produciendo irracionalidad altamente costosa. La solución adecuada de las porciones no computarizadas supone el conocimiento de herramientas analíticas específicas, experiencia y sentido común. Esto debería ser una capacitación indispensable del analista, que no siempre poseen.

Dado que todo sistema de información incluye este componente, por lo que se espera que el analista lo estudie y optimice al desarrollar un nuevo sistema, es indispensable que sea especificado en la documentación, como se ha indicado.