Un modelo de diagramación estructurada para principiantes

ULISES AGÜERO*

RESUMEN

Este artículo presenta un modelo para la elaboración de diagramas de flujo simples y visibles, particularmente recomendable para principiantes.

El modelo elimina el uso de flechas y permite salidas anormales de ciclos. Los diagramas de flujo y los ciclos son considerados como sistemas, y la comunicación entre diagramas de flujo es gráficamente descrita para facilitar la comprensión.

INTRODUCCION

La construcción de un diagrama de flujo es un proceso de refinamiento que corresponde a la complejidad con que es expresada la solución a un problema determinado.

Mayer *(1)* ha indicado la importancia de los modelos para la enseñanza de programación a principiantes, con la finalidad de aumentar y facilitar la comprensión de los asuntos.

El diagrama de flujo es un modelo comúnmente usado en la enseñanza de los estatutos de acción y control de los lenguajes de programación. Sin embargo, los componentes usuales de los diagramas de flujo carecen de visibilidad y simplicidad, dos propiedades descritas por Du Boulay y compañeros *(2)* como importantes en el proceso de instrucción de lenguajes de computadoras a principiantes.

La visibilidad permite la caracterización de partes y procesos, pero el uso de flechas en los diagramas de flujo complica la visibilidad. La simplicidad proporciona una comprensión clara de un pequeño número de partes interactuantes. En los diagramas de flujo comunes, el número de entidades distintas es pequeño, pero la interacción entre ellas es a menudo confusa, especialmente para novatos. Estas inconveniencias pueden ser parcialmente evitadas si se enseña solo diagramación estructurada. No obstante, los elementos estructurados, como los bloques iterativos, son usualmente explicados en términos del elemento condicional y de fechas en un arreglo muy peculiar, y no son explicados como elementos con significado propio.

Aquí se propone la eliminación de las flechas en los diagramas de flujo, junto con un grupo de estructuras de diagramación que son fácilmente visualizadas, pero con suficiente poder como para permitir salidas abruptas desde bloques iterativos.

DIAGRAMAS DE FLUJO

Un diagrama de flujo es un símbolo (Figura No. 1), caracterizado por su símbolo de proceso asociado. El flujo de acción comienza en el indicador del inicio y sigue las líneas, siempre hacia abajo, hasta que alcanza el indicador del final.

SIMBOLOS BASICOS:

Los símbolos básicos se agrupan en las siguientes clases:

- Secuenciales: proceso, vacío.
- Condicionales: si, caso de.
- Iterativos: mientras, hasta que.

Cada símbolo básico tiene exactamente una línea de entrada y una línea de salida.

Sólo por la línea de entrada se puede ingresar a un símbolo básico.

* Profesor del Departamento de Computación Administrativa del Instituto Tecnológico de Costa Rica.

El autor da las gracias a M. Mata, C. Araya y R. Roel por colaborar, en una u otra forma, en el presente trabajo.

Símbolos secuenciales: el símbolo de proceso (Figura No. 2) puede representar una acción elemental, una refencia a un diagrama de flujo (esto permite recursividad), una tarea compleja o ninguna acción.

Cuando el símbolo de **proceso** no representa acción alguna, éste es usualmente sustituido por el **símbolo vacío** (Figura No.3). El símbolo de **proceso,** según su complejidad semántica, puede ser refinado para producir cualquier símbolo básico, o refinado por medio de concatenaciones de símbolos básicos.

Se dice que un símbolo básico es concatenado si su línea de entrada (línea de salida) es unida exactamente con una línea de salida (línea de entrada) de otro símbolo básico. La figura No.4 presenta ejemplos de concatenaciones válidas e inválidas.

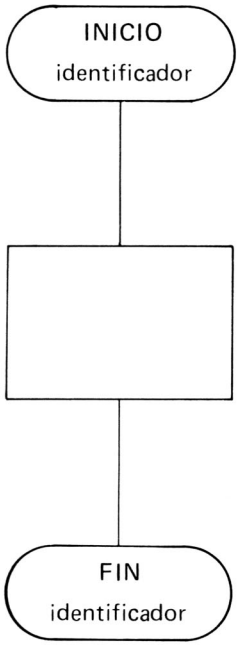


FIGURA No. 1. Esquema del diagrama de flujo.

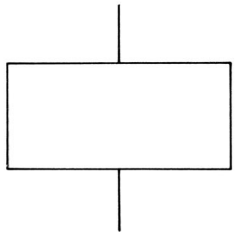
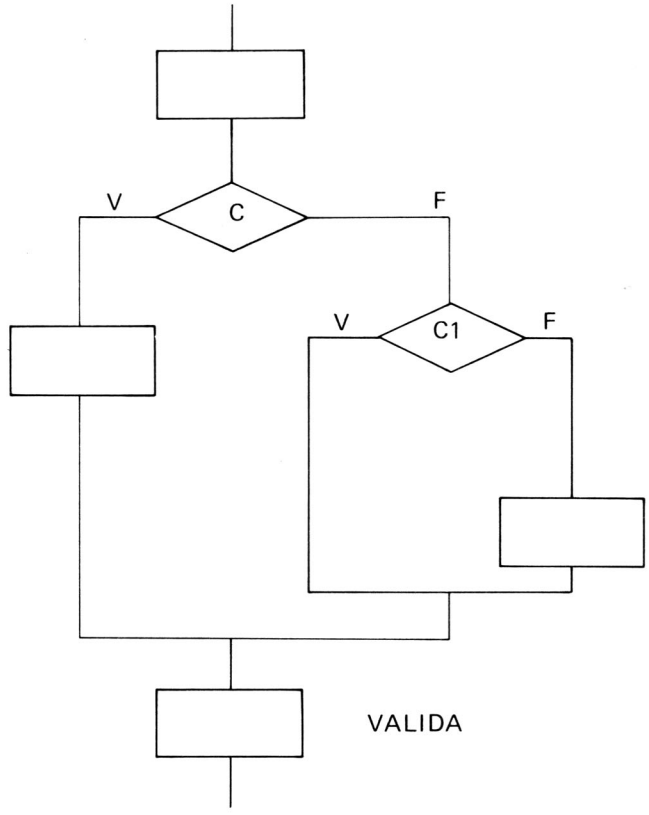


FIGURA No. 2. El símbolo de proceso.

FIGURA No. 3. El símbolo vacio.



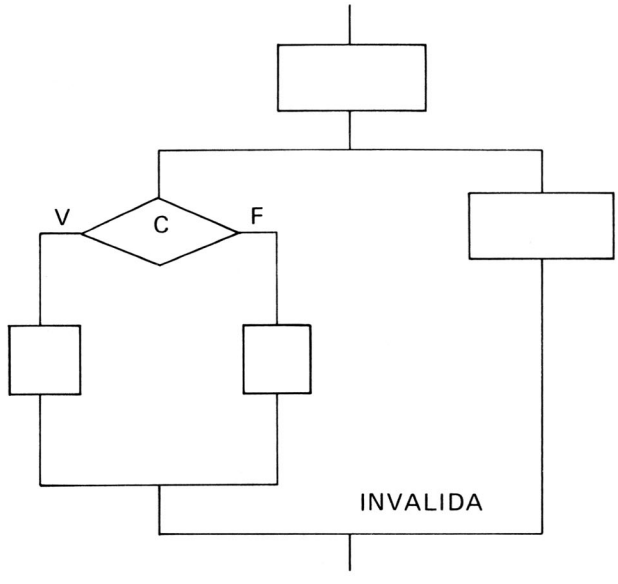


FIGURA No. 4. Ejemplos de concatenaciones.

Símbolos condicionales: el símbolo si (Figura No. 5) consta de una condición y dos símbolos de proceso. La línea de entrada del símbolo **si** es la que llega a la condición. La evaluación de la condición determina a cuál de los dos procesos se entra durante el flujo de acción.

El **símbolo caso de** (Figura No.6) consta de una expresión de control y N símbolos de proceso. El valor de la expresión de control determina a cuál proceso se entra durante el flujo de acción.

Los valores asociados con el símbolo deben ser mutuamente diferentes.

Uno y sólo uno de los símbolos de proceso puede ser asignado a la palabra "otros", con el significado de que el valor de la expresión no está en el conjunto de valores del símbolo (Figura No. 7).

El **símbolo mientras** (Figura No.8) consta de una condición lógica y un símbolo de proceso.

Su línea de entrada es la que está más cerca de la palabra MIENTRAS. Antes de ingresar, la condición es evaluada. Si resulta cierta, el símbolo de **proceso** asociado es ejecutado iterativamente mientras la condición se cumpla. Cuando la condición se torna falsa, la acción continúa a partir de la línea de salida del símbolo **mientras.** La condición es evaluada cada vez que el proceso es completamente ejecutado.

Si la condición es falsa antes de entrar al símbolo **mientras,** el flujo continúa a partir de su línea de salida.

El símbolo **hasta que** (Figura No.9), al igual que el símbolo **mientras,** consta de una condición lógica y un símbolo de proceso. Su línea de salida es la que está más cerca de las palabras HASTA QUE.

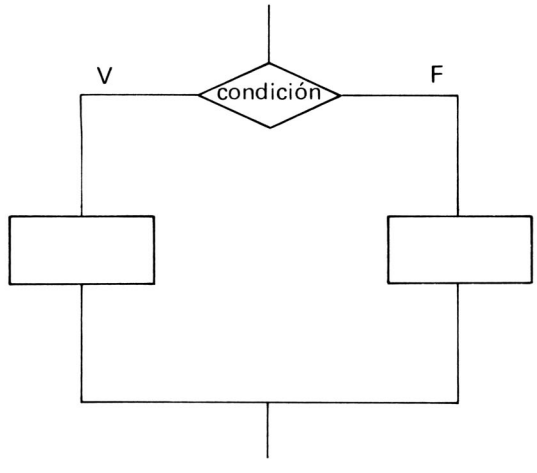


FIGURA No. 5. El símbolo SI.

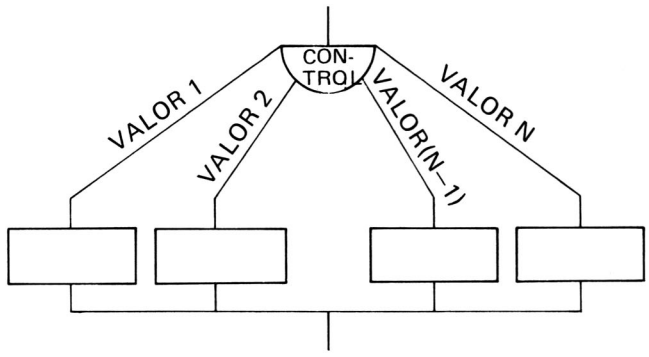


FIGURA No. 6. El símbolo caso de

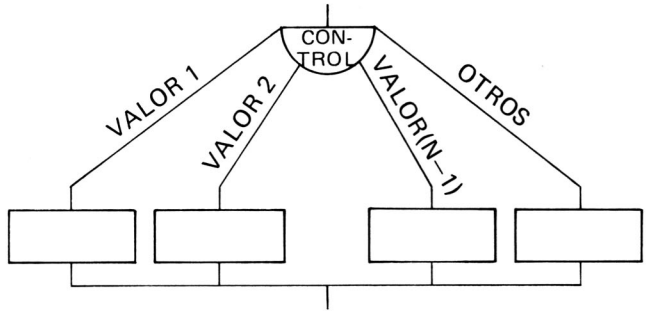


FIGURA No. 7. El símbolo caso de con "otros".

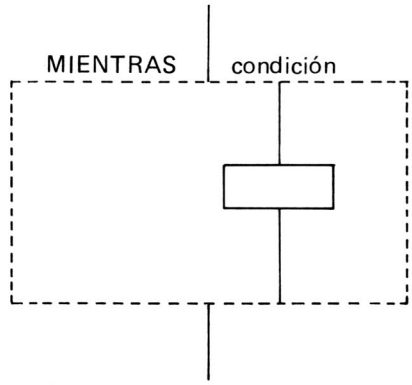


FIGURA No. 8. El símbolo mientras.

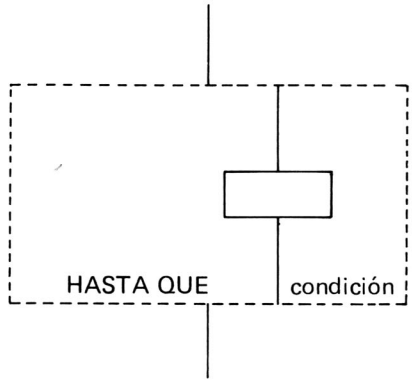


FIGURA No. 9. El símbolo hasta que.

La condición es primeramente evaluada después de que el símbolo de **proceso** concerniente es ejecutado una vez. Si la condición es verdadera, el flujo continúa desde la línea de salida del símbolo **hasta que**; si es falsa, el proceso es repetido continuamente hasta que la condición se torna verdadera. Esta condición es revisada después de cada ejecución del proceso.

Un aspecto importante de los símbolos **mientras** y **hasta que** es la sangría que se deja entre sus procesos y sus líneas de entrada y salida. De esta manera, el uso de sangría en la codificación está explícita en el diagrama.

DIAGRAMAS DE FLUJO PROPIOS Y BIEN FORMADOS

El problema de las salidas anormales de ciclos ha sido estudiado por Peterson y otros *(3)* y por Jensen *(4)*.

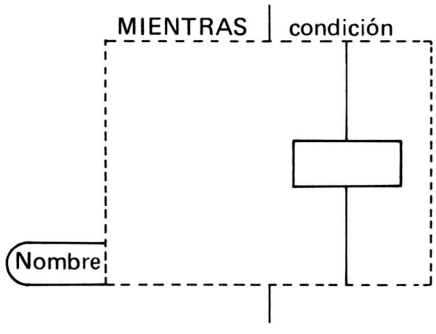
En relación con su trabajo, se dan las siguientes definiciones:

Un **diagrama de flujo propio** es el que no tiene etiquetas de salida.

Un **diagrama de flujo bien formado** es el que tiene etiquetas de salida válidas.

Etiquetas de ciclo y salidas: una pequeña modificación a los símbolos **mientras** y **hasta que** es la añadidura de una **etiqueta de ciclo**, según se presenta en la figura No. 10. En un diagrama de flujo particular, todas las etiquetas de ciclo deben ser diferentes.

Digamos que un símbolo básico está contenido en un símbolo iterativo si y solo si el símbolo básico forma parte de cualquier nivel de refinamiento del símbolo de proceso del símbolo iterativo. De esta manera, una **etiqueta de salida** (Figura No. 11) puede ser añadida a la línea de entrada o salida de cualquier símbolo básico de acuerdo con las siguientes reglas:



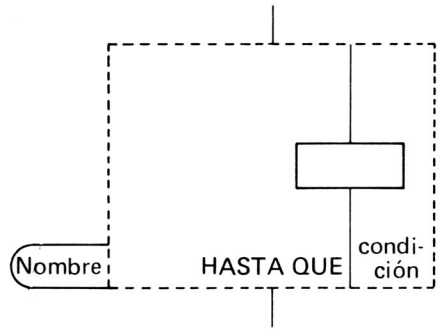


FIGURA No. 10. Símbolos iterativos con etiquetas.

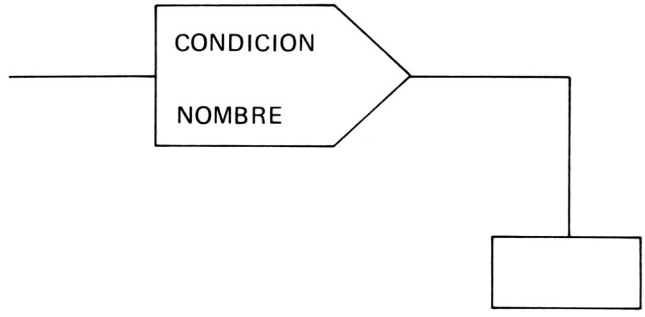


FIGURA No. 11. La etiqueta de salida.

— Debe existir exactamente una etiqueta de ciclo para cada etiqueta de salida.

— Un símbolo con una etiqueta de salida debe estar contenido en el bloque iterativo al que está añadida la etiqueta de ciclo correspondiente.

Todas las salidas que se muestran en el diagrama de flujo de la figura No. 12 son inválidas, y todas las salidas de la figura No. 13 son válidas.

Cuando una etiqueta de salida es encontrada durante la ejecución, la acción continúa desde la línea de salida del símbolo iterativo al que está añadida la etiqueta de ciclo correspondiente, previa ejecución del símbolo de proceso (o su refinamiento) correspondiente.

DIAGRAMAS DE FLUJO Y SIMBOLOS ITERATIVOS COMO SISTEMAS

Para facilitar la visibilidad, los diagramas de flujo y los símbolos iterativos deberían ser considerados como sistemas con salidas, un proceso, entradas y restricciones (Figura No. 14).

Las entradas son aquellos parámetros usados por el proceso del diagrama de flujo o del símbolo iterativo.

Las salidas son aquellos parámetros que representan los resultados del proceso.

El proceso es el definido por el símbolo de proceso del diagrama de flujo o del símbolo iterativo, o por cualquier refinamiento.

LA COMUNICACION ENTRE DIAGRAMAS DE FLUJO

Cuando un símbolo de proceso se refiere a un diagrama de flujo, la comunicación debe establecerse entre el diagrama de flujo que refiere y el diagrama de flujo referido.

Si usamos una terminología como la de Pascal *(5)*, los diagramas de flujo se comunican por medio de parámetros. Los parámetros del diagrama de flujo que refiere son llamados **parámetros efectivos**, y los parámetros del diagrama de flujo referido son llamados **parámetros formales.**

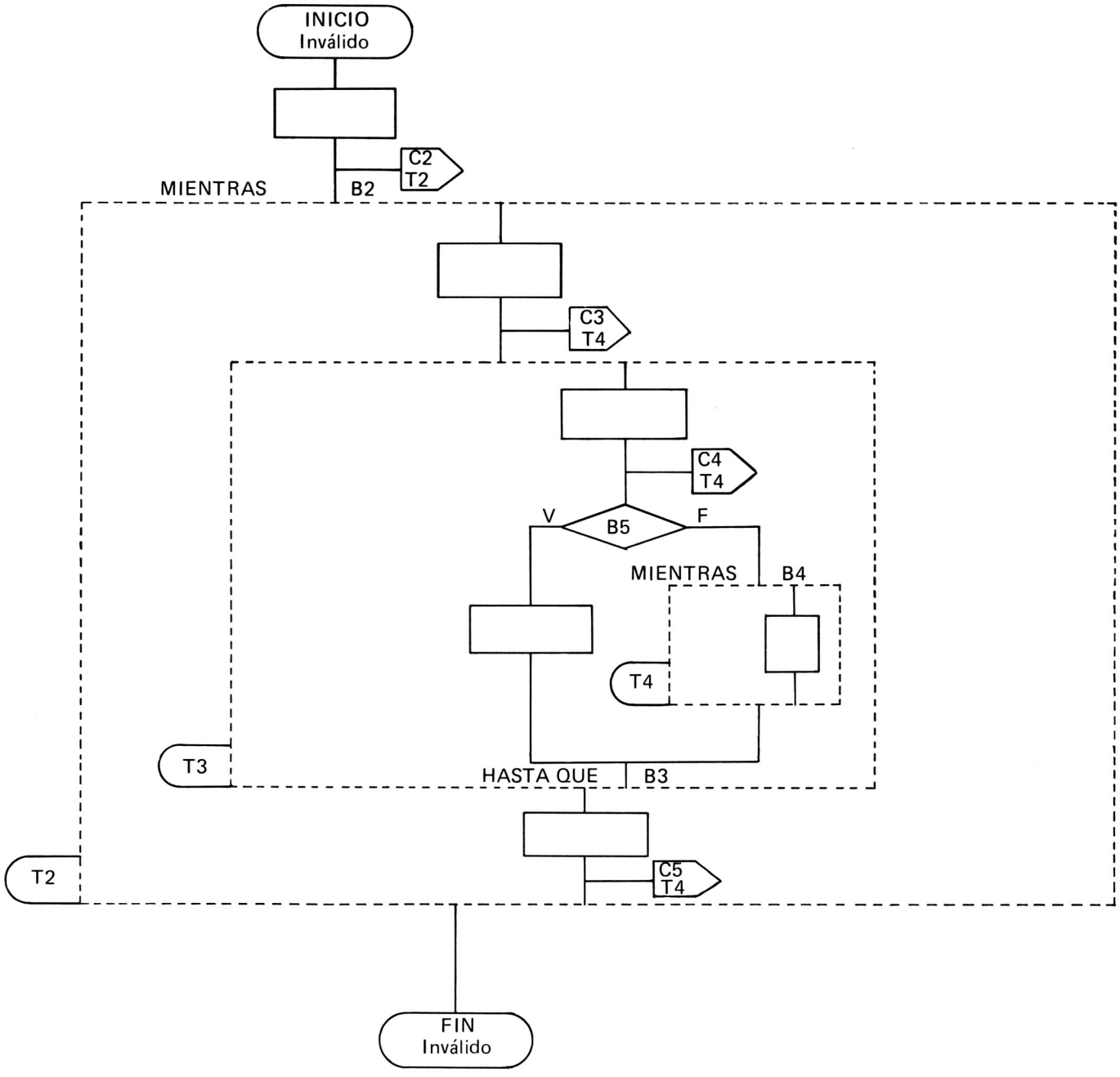


FIGURA No. 12. Salidas invalidadas.

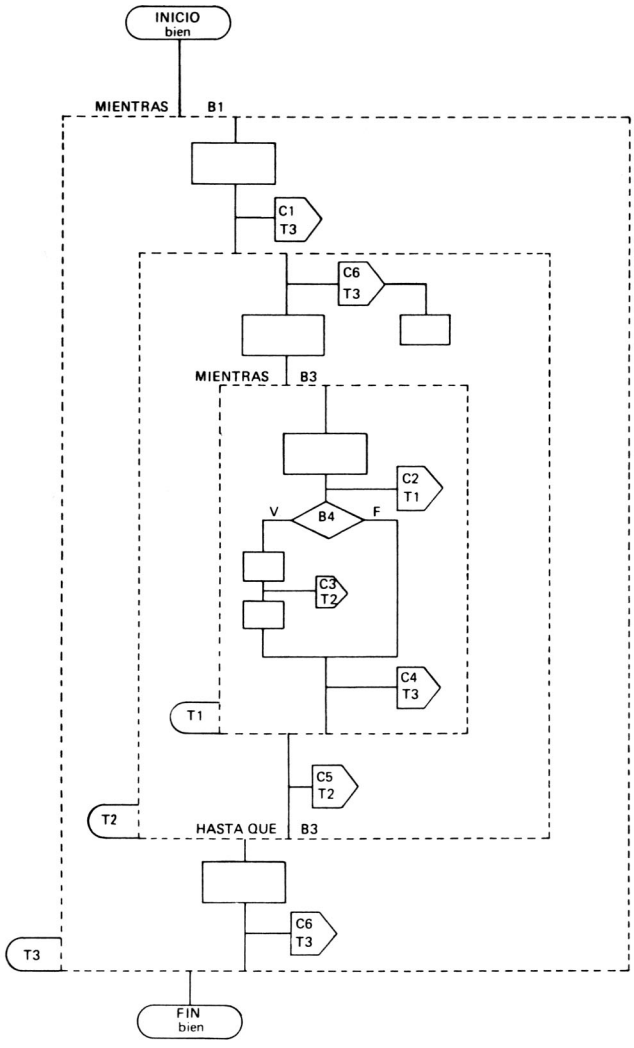


FIGURA No. 13. Un diagrama de flujo bien formado.

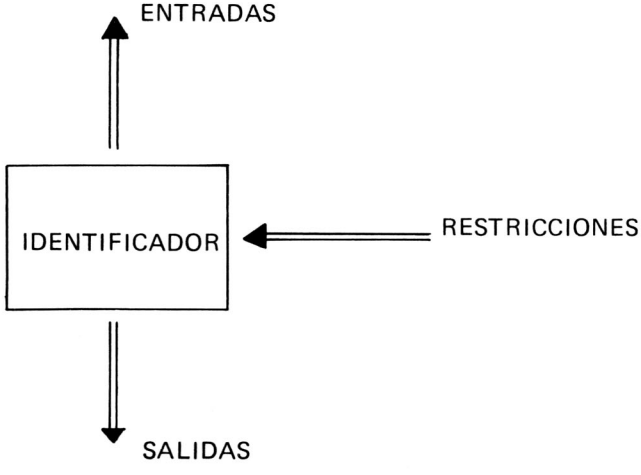


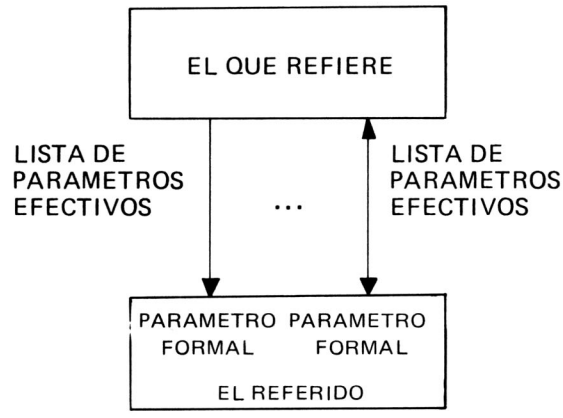
FIGURA No. 14. El símbolo de sistema.

La figura No. 15 presenta dos símbolos posibles para representar la comunicación con el diagrama de flujo de un procedimiento en Pascal. Una flecha dirigida hacia el diagrama de flujo referido indica que es un **parámetro de valor** (el parámetro formal toma el valor del parámetro efectivo), y una flecha bidireccional indica que es un **parámetro de referencia** (la dirección en memoria de los parámetros formal y efectivo es la misma).

La figura No. 16 muestra la comunicación para el diagrama de flujo de una función. La flecha que parte desde el diagrama de flujo referido indica que un valor será devuelto con el nombre de la función.

Note que los parámetros de referencia no son permitidos porque se desean evitar los resultados inesperados.

La lista de parámetros efectivos contiene todos los parámetros efectivos que son usados por el diagrama de flujo referido y que corresponden a un parámetro formal dado.



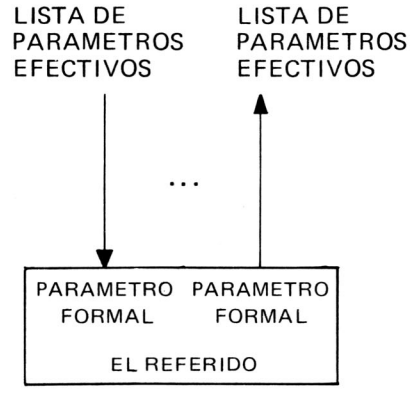


FIGURA No. 15. Comunicación con un procedimiento.

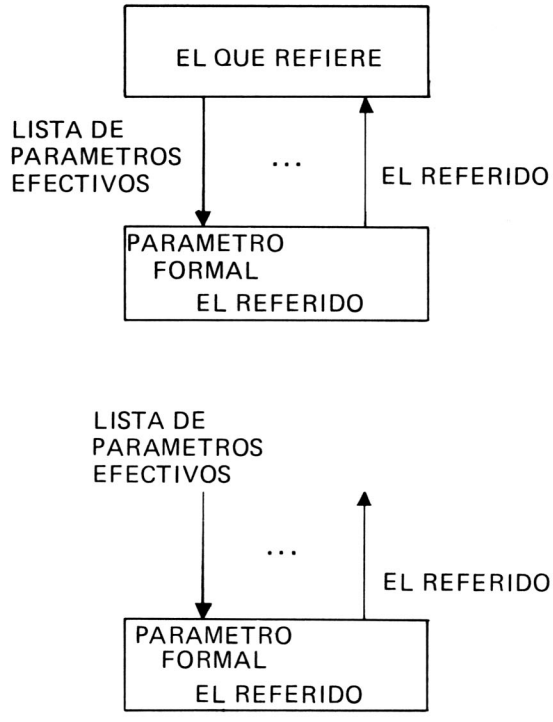


FIGURA No. 16. Comunicación con una función.

LITERATURA CONSULTADA

1. Mayer, Richard E. "The psychology of how novices learn computer programming". **Computing Surveys.** 12 (1): 121–141, marzo, 1981.

2. Du Boulay, B.; O'Shea, T. y Monk, J. "The black box inside the glass box: Presenting computering concepts to novices". Artículo No. 133, Dep. Artificial Intelligence, Univ. Edinburgh, Scotland. 1980.

3. Peterson, W.W.; Kasami, T. y Tokura, N., "On the capabilities of while, repeat, and exit statements". **Comm. ACM.** 16 (8): 503–512, agosto, 1973.

4. Jensen, Randall W. "Structured programing". **Computer.** 14 (3): 31–48, marzo, 1981.

5. Jensen, K. y Wirth, N. **Pascal user manual and report.** 2 ed. New York: Springer–Verlag, 1979.