BREVE INTRODUCION A LA PROGRAMACION ESTRUCTURADA

ING. MOISES TACLE GALARRAGA

B. E. Instituto Tecnológico Stevens (EE. UU.)
M. E. Instituto Tecnológico Stevens (EE. UU.).
Profesor del Depto. de Ingeniería Eléctrica.
Director del Centro de Computación de la ESPOL (Escuela Superior Politécnica del Litoral).

El gran desarrollo tecnológico de los últimos años ha hecho posible la fabricación de computadores y, en general, de equipos electrónicos sofisticados de gran capacidad de acción y de costos cada vez menores. En el campo de la computación digital, es un hecho cierto que con los nuevos adelantos tecnológicos se estan produciendo equipos con capacidad de procesamiento notable, mayor velocidad, poco peso y volumen y costos que disminuyen dramáticamente. Paralelamente a este acelerado desarrollo del HARDWARE, los especialistas de alrededor del mundo concuerdan que, en el presente, el SOFTWARE domina la computación; es decir, los problemas a resolverse en los tiempos actuales y por lo menos en toda la década del 80 serán preferentemente en el desarrollo del SOFTWARE apropiado para las cada vez mayores y variadas aplicaciones de la computación digital. En un reciente estudio realizado por la corporación RAND se pueden encontrar los siguientes datos interesantes:

- Los costos anuales de desarrollo de SOFTWARE en los Estados Unidos exceden los 10 billones de dólares.
- Se estima que, en general, la relación actual de SOFTWARE/HARDWARE para un sistema de computación es mayor que 65/35 y se prevee una relación de 90/10 para 1985.
- Los programadores en proyectos de relativa magnitud invierten su tiempo de la siguiente manera:
  45-50 por ciento en chequeo de los programas, 35 por ciento en el diseño de los programas y solamente entre el 25-30 por ciento en la codificación correspondiente.

En la comunidad que labora en procesamiento de datos existe la creencia bastante fundamentada que los sistemas modernos de computación han alcanzado tal grado de complejidad que practicamente escapan del control y dominio adecuado del hombre. Para muchos programadores escribir SOFTWARE verdaderamente confiable, es decir libre de errores y fácil de mantener, es casi un imposible en los momentos actuales. La tendencia actual en el desarrollo del SOFTWARE subordina las consideraciones clásicas de eficiencia a una programación clara y lógicamente estructurada. Se prefiere modularizar los programas de tal manera que un módulo (segmento de programa) pueda ser reemplazado por un módulo funcionalmente equivalente pero con más eficiencia. Una investigación de DATAPRO RESEARCH CORPORATION revela que, en un típico programa escrito en FORTRAN, "solamente el 3 por ciento de las instrucciones consumen aproximadamente el 50 por ciento del tiempo de ejecución del programa". Esta situación nos sugiere la siguiente estrategia de codificación: primero escribamos el programa en forma directa teniendo en cuenta los criterios convencionales de claridad y confiabilidad; luego, después de que el programa esté trabajando, lo volvemos a escribir esta vez optimizando la codificación que consume mayor tiempo de ejecución. Este enfoque debería ayudar a un programador a invertir su tiempo de una manera más eficiente.

Consideremos ahora un proyecto de programación relativamente grande. Las estadísticas nos muestran que en esta situación un programador típico promedia solamente 10 instrucciones fuente depuradas por día. Como sabe mos que la codificación de 10 instrucciones solamente toma pocos minutos en una jornada de trabajo de ocho horas, cabe preguntarnos qué ocurre el resto del día.

Los programadores en un proyecto no solamente tienen la responsabilidad de codificar y diseñar un programa, sino que realizan actividades tales como: