# General Motors/North American Monitor For the IBM 704 Computer

CARLOS MARTÍN FLORES GONZÁLEZ, Carné: 2015183528

**Instituto Tecnológico de Costa Rica**
**Maestría en Computación**
**Sistemas Operativos Avanzados**
**Profesor: Francisco Torres Rojas, Ph.D**

En este artículo se hace una reseña del trabajo incipiente que se hizo a mediados de los años cincuenta en sistemas operativos. En particular, se da a conocer las contribuciones hechas por *General Motors* y *North American Aviation* en principio utilizando la computadora IBM 701.

En 1954 solamente existían 19 computadoras IBM 701. Esta computadora podía ejecuta 150 000 instrucciones por segundo y se alquilaba por $23,750 al mes. Ocupada una habitación de $40 \times 40$ pies. En cuanto a su arquitecura, era una máquina secuencial que solamente podía hacer una cosa a la vez. Era una máquina que trabajaba por lotes, tenía un lector de 150 tarjetas perforadas por minuto, una performadora que procesaba 100 tarjetas por minuto, una impresora de 48 caracteres y una memoria de 2000 palabras de 36 *bits*. El modo típico de uso era con el programador presente en la consola de operación. Cuando un programador quería ejecutar un programa debía de poner su nombre en una lista de espera y cuando se acercaba su momento de utilizar la computadora traía consigo un paquete (*deck*) de tarjetas perforadas. Cuando era su turno, se debía cerciorar que los tableros de control apropiados estaban instalados, colgar una cinta magnética, ir hacia la consola y mover algunos *switches*, cargar las tarjetas perforadas en el lector e iniciar la secuencia de inicio. Si las cosas no salían bien se perdía la oportunidad de utilizar la computadora y el próximo en la lista iba a tomar su turno. Usualmente la computadora pasaba más tiempo desocupada o inactiva que realizando algún trabajo.

Dado a que la computadora pasaba la mayoría de tiempo desocupada, se inician esfuerzos por mejorar la eficiencia en su uso. Algunas iniciativas se llevan a cabo para intentar sacar más provecho del equipo pero aun así la productivdad que se obtuvo con ellas siguió siendo pobre. En los años ciencuenta la programación era descrita como una especie de "caos organizado". El programador típico usaba lenguaje ensamblador, escogía sus rutinas favoritas para conversiones de decimal a binario, intentar saber cómo ensamblar y *debuggear* sus programas con las herramientas disponibles. Al final obtenía un programa pero generalmente más tarde de lo acordado. Se introduce el *IBM 701 Speedcoding System*. Se le considera el primero lenguaje de alto nivel y estaba destinado para el cálculo de números de punto flotante. En este contexto el "alto nivel" se refiere a que apuntaba a ser un lenguaje más expresivo y natural en contraste a la codificación en lenguaje máquina. Los beneficios de usar *Speedcode* eran: sacar al programador de las consolas y los cuartos de máquina, estandarizar configuraciones (evitar pasos innecesarios), evitar recargar de los trabajos cada vez que se usaba la computadora y el uso del paquete integrado de programas que traía producía resultados más rápidos que cuando se hacían desde cero.

En 1955 se forma el grupo de usuarios de IBM llamado SHARE el cual permitía a sus miembros intercambiar progras e ideas para beneficio mutuo. Antes de la tercera reunión de SHARE, *General Motors* y *North American Aviation* tenían propuestas para un sistema operativo para la IBM 704. Ambas compañias decidieron unirse y

como resultado se obtuvo un sistema operativo de muy alta calidad gracia al talento disponible de ambos equipos. El nuevo sistema operativo se instaló en 20 computadoras IBM 704. **Sobre la IBM 704:** sucesora de la IBM 701. Era más rápida (más del doble) y utilizada memoria de núcleos magnéticos esto incrementó el tiempo medio entre fallos a 8 horas. Alquiler: $35,550 al mes. Tenía mayor memoria, mayor velocidad pero el mismo equipo para entrada/salida. IBM se plantea cambios en este modo de operación para tener mayor beneficio de la 704. Primero ofrece tres nuevos equipos para convertir de tarjetas a cintas, para tomar imágenes de cintas e imprimirlas y para tomar imágenes de cintas y perforar tarjetas. Las tres eran independientes y no inteferían con el procesamiento de la computadora.

Notas del sistema operativo propuesto en 1955 por SHARE:

1. Toda entrada y salida se hacía por medio de los equipos de conversión independientes. Producían y recibían archivos ahora conocidos como `SYSIN` y `SYSOUT`.
2. `SYSIN` contenía un lote de trabajos independientes. Cada trabajo se identificaba con tarjetas de control `JCL`, la secuencia iniciaba con una tarjeta que tenía la información de la cuenta y el nombre del programador.
3. Toda la organización de los datos se hacía por medio de programación y no por alambrado. Redujo el tiempo de minutos a milisegundos.
4. Se sacaron a los programadores de los cuartos de máquinas. Se contratan operadores y se estandariza la comunicación entre ellos. Esta forma de trabajo se adopta rápidamente.
5. Se gana eficiencia en la programación al reutilizar rutinas de conversión decimal a binario y al proveer un ambiente de ejecución que tenía entrada en binario, computación en binario y salida en binario.
6. Se aumenta el *hardware* añadiendo un reloj personalizado. El sistema toma muestras del relog cuando se leía un nuevo trabajo y guardaba los registros que se usaron durante la ejecución para efectos contables y de mantenimiento.
7. Era posible con el archivo de `SYSIN` ensamblar un programa, cargarlo en la memoria, presentarle datos de entrada y guardar los resultados del ensamblaje y de ejecución en una sola pasada.
8. Se diseña un sistema que podía ser extendido, se le podía agregar otros traductores al menú de rutinas de conversión, por ejemplo Fortran.

La implementación del sistema fue dictado por la poca disponibilidad de núcleos de memoria y por la gran cantidad de rutinas de conversion decimal–binario y viceversa. El sistema operativo resultante tenía una fase de traducción de la entrada, el cual convertía datos de decimal a binario, y los programas de lenguaje fuente a lenguaje objeto. La fase de ejecución que esta casi exclusivamente a cargo del programador. La fase de traducción de la salida que procesaba salidas en la impresora, tarjeta perforada y registros contables. Ventajas de este modelo:

1. Proveía la mayor cantidad de memoria al programador de la aplicación durante la fase de ejecución.
2. Era más eficiente:
   - Las rutinas de conversión eran más grandes que los sub-archivos de los trabajos individuales.
   - Los programas que requerían de una mayor conversión se cargaban solo una vez en memoria por lote de trabajo, lo que daba como resultado un menor número en el total de ciclos en lugar de que las rutinas grandes de traducción fueran cargadas y ejecutadas al principio y al final de cada trabajo.
   - Al darle a los traductores toda la máquina durante el proceso de conversión, los traductores se podían hacer todo lo grande que se necesitara para ganar eficiencia en la ejecución y por lo tanto estaban optimizados para ser más rápidos para la conversión de tareas que tenían que hacer.

Cuando el código fue escrito, 90 % del trabajo fue dedicado a los traductores de entrada y salida y sólo un 10 % del esfuerzo fue dedicado a ayudar a los programadores durante la fase de ejecución.

## 1. ¿CUÁL ES EL PROBLEMA QUE PLANTEA EL *PAPER*?

El uso eficiente de la computadora y sus recursos. Es claro que conforme los ingenieros, programadores y operadores se familiarizaban cada vez más con estos equipos, iban proponiendo, probando e implementando nuevos mecanismos para sacar mejor partido del equipo. Los enfoques con los que venían trabajando resultaban lentos y tediosos. Las motivaciones financieras también jugaban un rol importante porque el tener un proceso de desarrollo lento por lo general no se podían cumplir con plazos de entrega con los clientes.

## 2. ¿POR QUÉ EL PROBLEMA ES INTERESANTE O IMPORTANTE?

Porque brinda una mirada de los primeros días de la computadora y de sus limitaciones, pero también de los aciertos y el trabajo tesonero que hubo detrás para impulsar esta disciplina. Cuando se piensa, por ejemplo, en soluciones previas para resolver los problemas del momento (próxima pregunta), llama la atención que las soluciones se iban dando sobre la marcha, ellos estaban creando con cada nueva mejora toda una nueva disciplina.

## 3. ¿QUÉ OTRAS SOLUCIONES SE HAN INTENTADO PARA RESOLVER ESTE PROBLEMA?

Durante el periódo al que nos remite el autor, los primeros intentos para resolver el problema estuvieron más del lado del proceso que acompañaba la ejecución de un programa que de la máquina. Por ejemplo, al principio el autor comenta que para hacer su trabajo más rápido evitaba cambios en el alamabrado, ejecutaba desde cinta magnética sin usar en lugar de tarjetas perforadas, esto reducía el tiempo de carga de tres minutos a diez segundos. Luego, cada registro de entrada se cargaba en una única dirección, de esta forma podía guardar el programa en cinta y recargar parches a través de la lectora de tarjetas. El podía ejecutar una prueba en tres minutos o menos.

Luego se introducen soluciones como *Speedcode* el cual llegó a ayudar a estandarizar configuraciones en la máquina y a brindar una serie de rutinas reutilizables que hicieron que el trabajo de programación se hiciera más rápido (ya no había que escribir la misma rutina para calcular un logaritmo una y otra vez).

Conforme el artículo avanza, se menciona la inclusión de nuevos equipos de conversión de entradas y salidas de decimal a binario.

## 4. ¿CUÁL ES LA SOLUCIÓN PROPUESTA POR LOS AUTORES?

La propuesta conjunta de *General Motors* y *North America Aviation* fue la creación de un sistema que estaba concebido e implementado en tres fases distintas de procesamiento:

1. **Fase 1**: datos de decimal a binario, y los programas de lenguaje fuente a lenguaje objeto
   *a*) Carga de traductor de entrada
   *b*) Traducción del lote de entrada
2. **Fase 2**: ejecución
   *a*) Tomar muestra del reloj del sistema
   *b*) Cargar un trabajo
   *c*) Ejecución del trabajo
3. **Fase 3**: salida
   *a*) Carga de traductor de salida
   *b*) Traducción del lote de salida

Esta solución supuso un cambio de paradigma con respecto al trabajo que se venia realizando en los *mainframes*, ya que además del sistema propuesto, hubo cambios en la forma en cómo el personal interactuaba, se introduce el personal de operadores de cómputo y se saca a los programadores del cuarto de máquinas. También se empiezan a adoptar una serie de convenciones en el manejo de las cintas magnéticas de entrada y salida (`SYSIN, SYSOUT`),

tarjetas de control y hasta en la configuración de los equipos periféricos como impresoras, perforadoras y lectoras con el fin de que esta configuración se lleve a cabo por medio de programación y no por alambrado.

## 5. ¿QUÉ TAN EXITOSA ES ESTA SOLUCIÓN?

Este fue un trabajo pionero. Esta solución representa el punto de partida para el trabajo que luego se llevó a cabo en otra serie de sistemas operativos para computadoras IBM: SOS, IBSYS y OS/360. A pesar de que se centra en el trabajo en sistemas operativos se pueda ver también los inicios de muchas otras prácticas y patrones que se siguen hoy día: colaboración, *Open Source*, creación de convenciones y estándares para codificación y operación, principios de auditoría informática (esto se puede ver en la forma en cómo la computadora era configurada para registrar los recursos asocidos a una cuenta o cliente), inicio de los lenguajes de programación y el desarrollo de nuevo *hardware*.