Universidad De Costa Rica

Escuela de Ciencias de Computación e Informática

CI-1310: Sistemas Operativos

MAC OS X 10.9 Mavericks

Autor: José Daniel Picado Hidalgo

Carné: B04833

Fecha: 7 de Noviembre de 2013

Índice

1 Introducción

En este documento se va a tratar sobre el último lanzamiento del sistema operativo OS X de la compañía Apple®. Primero se tratará un poco de la historia de estos sistemas, luego se enfatizará en esta última versión lanzada en Octubre de 2013 por dicha empresa. El análisis sobre el sistema operativo estará enfocado sobre principalmente el manejo de recursos para el procesamiento de los trabajos generados por la interacción con el usuario.

1.1 Reseña Histórica

La compañía Apple® ha sido desde sus inicios una compañía enfocada a la producción de máquinas personales, para que usen personas con cada vez menos conocimientos técnicos. En 1976 se fundó la compañía en lo que fue un proyecto de un par de amigos (Steve Wozniak y Steve Jobs) de garaje en el que dieron vida a la Apple I [1]. Con el lanzamiento del Apple II se creó un sistema operativo que revolucionó la forma de usar computadoras con sistemas gráficos y el uso por primera vez del posiblemente dispositivo más ultimado en las computadoras personales para simplificar el uso de las mismas, el mouse[4].

Los años pasaron y Apple continuó enfocándose en computadoras personales pensadas tanto en software como en hardware para el uso de personas con menos conocimientos técnicos. En 1999 se anuncia la “muerte” del noveno sistema operativo de mac, el os 9, pero también se anuncia el nacimiento de la familia de sistemas operativos que revolucionan la forma de interactuar con una computadora personal macintosh, el OS X, que es lanzado en versión beta en 1999 de manera muy inconclusa, pero no es hasta el 2001 cuando se libera la versión final del OS X 10.0 o “Cheetah”, basado en UNIX [2].

El sistema operativo Cheetah se considera el padre de todos los sistemas operativos del nuevo milenio de la compañía Apple, entre los que se encuentran en orden cronológico: Cheetah, Jaguar, Panther, Tiger (que fue el sistema operativo más completo en su lanzamiento en 2004, con características que se preservan hoy 9 años después), en 2006 vino Leopard (sistema operativo que da el salto definitivo a los procesadores Intel, en detrimento de los Power PC), Snow Leopard (que se encargó de una reescritura de Leopard para optimizar el código heredado), finalmente vendrían los sistemas operativos que combinaron las mejores características de un sistema operativo de computadora personal y los dispositivos móviles: Lion, Mountain Lion y el más reciente Mavericks [2].

Como dato curioso Apple ha bautizado sus sistemas operativos OS X por 12 años con nombres de grandes felinos, pero han roto esta tradición en el lanzamiento del 10.9 al asignarle el nombre de Mavericks, esto se debe a que en las ramas de la biología se estaban acabando los nombres de grandes felinos y no sería bien visto eventualmente tener que disminuir el tamaño de sus “gatos” para poder nombrar sus sistemas operativos, es así como optan por buscar otro nombre para su nueva edición de sistema operativo y lo decidieron cambiar por los nombres de los lugares más “asombrosos” de California, sede de los cuarteles generales Apple[5].

2 Análisis de la tecnología interna de Mavericks

Éste sistema operativo se crea para enfrentar el gran desafío que plantean los sistemas computacionales modernos, a los que hay que tratar de sacarles el mayor provecho de la manera más óptima en todo aspecto. Mavericks brinda un medio potente y eficiente para la ejecución de todas las tareas necesarias, pero también piensa en la optimización de los recursos energéticos, esto para conseguir la “perpetuidad” de potencia en las computadoras personales móviles, como lo son la línea de Macbook de Apple [6]. A continuación se expondrá de manera primero resumida y luego técnica de las nuevas características de Mavericks para optimizar el uso de la computadora.

2.1 Kernel OSX

Como se expuso previamente los sistemas operativos son herederos de UNIX, por lo que la mayoría de sus controles son similares a este, en realidad OS X es el heredero de Unix con mayor propagación a nivel mundial [7]. El kernel de OS X Mavericks está montado sobre FreeBSD que es un sistema operativo que hereda de AT & T Unix mediante su ancestro UNIX BSD. Además gran parte de su kernel está montado sobre el tradicional sistema Mach, en su versión 3.0 [3]. Además de estar sobre estos clásicos sistemas implementa una Interfaz de programación de Kernel para su funcionamiento. El kernel de OS X es elprimer kernell que fue diseñado para dos tipos diferentes de arquitecturas de computadoras, dentro de un mismo instalador se podía correr y montar tanto en arquitectura de 32 bits como en arquitectura de 64 bits, siendo esto una capacidad única en su momento que le permitió una escalabilidad a casi cualquier computadora moderna, a pesar de que ahora sólo se produce para 64 bits, aún es posible configurarlo y correrlo sobre máquinas de 32 bits sin mayores complicaciones[3].

2.2 Proceso de arranque de Mavericks.

A diferencia de muchos de los sistemas operativos que tienen segmentos de memoria, definidos desde que se desarrollan para que resida la partición de arranque, Mavericks utiliza segmentos al azar. Mientras se inicia el ordenador automáticamente se mueve al azar dentro de la memoria los espacios en los que residen el kernel y los segmentos de código del sistema, esto para prevenir ataques de malware durante los inicios del sistema dificultando la localización de estos segmentos, esta técnica se conoce como colocación del espacio de direcciones aleatorio o ASLR por sus siglas en inglés [3].

2.3 Optimización del almacenamiento en memoria.

Normalmente uno de los mayores problemas en especial con programación paralela es el espacio necesario de memoria para poder correr todas las aplicaciones y trabajos, almacenando todos sus datos y tratar de evitar escrituras en dispositivos de almacenamiento masivo. Mavericks presenta una solución a este problemas y es que implementa escaneos sobre la memoria para detectar segmentos de memoria que tienen mucho tiempo de no usarse y reducir el espacio en memoria ocupado por el mismo antes de tener que sacarlo de la memoria[8].

Existe una rutina llamada Compressed Memory que se dispara cuando se detecta que hay riesgo de desbordar el espacio físico en memoria, esta rutina se encarga de comprimir los

segmentos que tienen una antigüedad superior a un periodo definido por los programadores a través del algoritmo de compresión WKdm que es un algoritmo muy usado por su eficiencia con un rendimiento de tiempo. Es un algoritmo híbrido que utiliza métodos estáticos y de diccionario, logrando una excelente compresión y una cota de tiempo de N(d) (d = tamaño del diccionario)[9].

El uso de la compresión permite reducir páginas enteras de memoria a menos de la mitad del tamaño en nano segundos. Un beneficio que trae la compresión a parte de poder almacenar más elementos en memoria es que la descompresión es tan rápida como la compresión lo que facilita el manejo de los datos. Al poder almacenar más datos en memoria reduce la cantidad de swapps necesarios para la ejecución de los trabajos, permitiendo terminar antes los trabajos y optimizando el uso del CPU. Finalmente otro gran beneficio de la compresión es que en caso de ocupar hacer swaping con un dispositivo de almacenamiento masivo, se pueden enviar las páginas comprimidas disminuyendo la escritura y lectura [3].

2.4 App Nap

Mavericks tiene una gran particularidad y es que procesa en respuesta a la necesidad del usuario y no a la de los procesos como tal, se refleja fielmente en especial en los trabajos de las aplicaciones de usuario, pues si una aplicación está abierta, pero no está en pantalla, accediendo a internet, o reproduciendo música, esta aplicación entra en un estado de bajo consumo y le da paso a las aplicaciones que realmente el usuario ocupa[6].

El App nap regula la asignación de recursos para cada proceso, los procesos dormidos así como los de unix les baja la prioridad. Los programadores pueden usar estas ventajas en sus aplicaciones a través del API de IOKit y el IOMAssertion. Al reducir la prioridad de los procesos de Unix permite que se ejecuten con una mayor prioridad los procesos de usuario. Cuando una aplicación sale del estado de siesta, se le reasigna la prioridad correspondiente y se vuelve a ejecutar de manera regular. Los 3 principios básicos para el App nap son:
1 Regular y disminuir el tiempo en CPU de los procesos “dormidos”.
2 regulación de los dispositivos de entrada y salida (si no los ocupan en este momento se les debe expropiar para los procesos en los que el usuario los requiere)
3 Disminuir prioridades a los procesos de UNIX[3].

2.5 Control de Procesos

Es preferible que la mayoría de los procesos del sistema operativo sólamente se ejecuten cuando son necesarios y no que estén viviendo en memoria siempre que se ejecuta el sistema, para esto se una un mecanismo de lanzamiento por demanda, que se basa en activar procesos en respuesta a acciones, como por ejemplo: cuando hay algún problema de entrada salida, cuando se agregan dispositivos, cuando se crean archivos, etc. También es posible utilizar una calendarización de ejecución a partir de transcursos de tiempo, por ejemplo es posible a través de la llamada al sistema startCalendarInterval o un intervalo genérico con la llamada startInterval, estas se usan para poder monitorear el sistema periódicamente sin necesidad de estar en memoria todo el tiempo, disminuyendo el trabajo del CPU.[3]

Gran central de despacho: es el planificador que usa OS X Mavericks, el cual da soporte a la programación concurrente y paralela. Este planificador está diseñado para simplificar la sincronización de procesos para los programadores, programado de manera eficiente utilizando llamados al sistema. Simplifica la cantidad de código para poder sincronizar y ejecutar de manera sincrónica y asincrónica los trabajos. Utiliza 3 grandes estructuras básicas: las colas, los bloques y receptáculos de hilos. Las bloques son extensiones a lenguajes como C, C++ u Objective C que le da formas similares a lenguajes interpretados con

las cuales permite sincronizar el uso de variables teniéndolas de manera global sin requerir que el programador lo hago de forma explícita. Los bloques son partes de código similar a los lenguajes interpretados, esto permite trabajar con variables compartidas entre procesos, son extensiones que funcionan con g++ para c, c++ y objective c. Las colas de despacho funcionan para añadir bloques para ser ejecutados, mientras las colas seriales permiten acceso con exclusión mutua a los diferentes hilos a las variables compartidas.[3]

2.6 Cajas de Arena

Las cajas de arena son una técnica para evitar que los procesos no hagan algo que no deben, por ejemplo que una aplicación de correo no escriba en disco, o que una aplicación de texto acceda a la red, al menos no sin consentimiento explícito del usuario. Estas restricciones pueden evitar daños mayores en el ordenador. Hay exclusiones que son accesos mandatorios para ciertos procesos de manera temporal, pero son la excepción a la regla, para lograr mantener el propósito de las cajas de arena.

3 Conclusiones.

1) El sistema operativo OS X Mavericks es la combinación ideal entre la capacidad de ejecución necesaria en sistemas modernos con la respuesta y eficiencia necesaria para que el procesador siempre esté disponible y el gasto de energía se mínimo.

2) Mavericks simplifica el uso completo de la computadora personal en todos los aspectos con una respuesta súper eficiente, pues simplifica desde el uso para usuarios inexpertos hasta usuarios experto como los programadores a los que les es cada vez más fácil optimizar procesos y sincronizarlos.

3) La compresión de memoria junto el app nap permiten sacar el máximo provecho de las computadoras personales pues hacen que los procesos ejecutándose sólo sean los necesarios y que el acceso a memoria sea lo más rápido posible.

4 Bibliografía

1 History of Apple Inc. Autor Desconocido, visitado el 31 de Octubre de 2013.
http://www.nostoptechnology.com/

2 La evolución de OS X a través de las conferencias para desarrolladores [Especial Historia WWDC], Santamaría, Pedro, para applesfera.com visitado el 31 de Octubre de 2013.
http://www.applesfera.com/eventos/la-evolucion-de-os-x-a-traves-de-las-conferencias-para-desarrolladores-especial-historia-wwdc

3 OSX Mavericks Core Technology Overview. Apple Inc, version Octubre de 2013.

4 Apple-History 1976-1981. Autor desconocido, visitado el 1 de Noviembre de 2013.
http://apple-history.com/h1

5 Porque OS X Mavericks no tiene nombre de gato, Sánchez Eduardo, visitado el 6 de noviembre de 2013
http://computadorasmac.about.com/od/mac-os-x/a/Por-Que-Os-X-Mavericks-No-Tiene-Nombre-De-Gato.htm

6 4 Claves para entender OS X Mavericks, la próxima revolución para Mac. Merino Iñigo, visitado 2 de Noviembre de 2013
http://appleweblog.com/2013/10/os-x-mavericks-claves

7 OS X for UNIX Users. Apple inc, vesión Julio 2011.

8 OS X Mavericks Advanced Technologies. Apple inc, visitado en 25 de Octubre de 2013
http://www.apple.com/osx/advanced-technologies/

9 Analysis of Comprension Algorithms for Program Data, Simpson Matthew, Dr. Burua, Rajeev y Biswas Surupa. Version: 12 de Agosto de 2013.
http://terpconnect.umd.edu/~barua/matt-compress-tr.pdf