TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

MÓDULO DE CAPTURA Y ALMACENAMIENTO EN TIEMPO REAL DE PARÁMETROS ASOCIADOS AL FUNCIONAMIENTO DE NODOS DE CÓMPUTO

Daniel Rubén García Avila [1], Omar Antonio Hernández Duany [2]

[1,2]Universidad Tecnológica de la Habana, Ave. 114 #11901 / Ciclovía y Rotonda. Marianao, La Habana, Cuba
[2] e-mail:omar.hd@tele.cujae.edu.cu

RESUMEN

En el campo de las Telecomunicaciones se requiere realizar el procesamiento en tiempo real de flujos de información multipropósito, como puede ser el procesamiento de video. Para lograr la efectividad de estos procesos complejos es necesario una gestión efectiva de la infraestructura computacional empleada con fines de lograr la monitorización y análisis en tiempo real de datos resultantes de los nodos que intervienen en el procesamiento. Existen muchas herramientas tanto de pago, como libres de costo desarrolladas para realizar el monitoreo; pero se requiere de un diseño más simple, genérico, escalable y rápido que resulte de utilidad para garantizar el adecuado funcionamiento de estas aplicaciones de gran complejidad. En este trabajo se ha desarrollado un sistema de monitoreo en tiempo real basado en la integración de *scripts Shell* y módulos programados en el lenguaje de programación C++ sobre sistemas operativos Linux. Esta solución realiza la captura, almacenamiento y análisis de datos asociados a los nodos de cómputo en aplicaciones de alto rendimiento que son ejecutadas en nodos de cómputo paralelos o distribuidos. Se realizó la validación de la solución del funcionamiento del sistema a partir del análisis de los datos adquiridos, los tiempos de ejecución de cada uno de los módulos y las secuencias de ejecución. Esta solución es extensible a una amplia gama de soluciones informáticas que deben funcionar de forma ininterrumpida.

PALABRAS CLAVES: Grandes volúmenes de información, procesamiento, tiempo real, monitoreo, procesos.

MODULE FOR REAL-TIME CAPTURE AND STORAGE OF PARAMETERS ASSOCIATED WITH THE OPERATION OF COMPUTE NODES

ABSTRACT

In the field of telecommunications, real-time processing of multipurpose information flows, such as video processing, is required. To achieve the effectiveness of these complex processes, effective management of the used computational infrastructure is necessary to conduct real-time monitoring and analysis of data resulting from the nodes involved in the processing. There are many tools; both paid and free of cost, developed to perform monitoring. Still, a more straightforward, generic, scalable, and fast design is required to ensure the proper functioning of these highly complex applications. In this work, we have developed a real-time monitoring system based on the integration of Shell scripts and modules programmed in the C++ programming language on Linux operating systems. This solution performs the capture, storage, and analysis of data associated with computing nodes in high-performance applications executed in parallel or distributed compute nodes. The validation of the system operation solution was performed based on the analysis of the acquired data, the execution times of each of the modules, and the execution sequences. This solution is extensible to a wide range of computing solutions that must operate uninterruptedly.

INDEX TERMS: Large volumes of information, processing, real time, monitoring, processes.

1. INTRODUCCIÓN

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

Impulsado por el rápido desarrollo de las tecnologías de la información y las comunicaciones, en la actualidad pueden apreciarse cambios significativos en el entorno académico y empresarial. Se ha producido un incremento significativo de soluciones computacionales que emplean la inteligencia artificial y el análisis de grandes volúmenes de datos minimizando los tiempos de ejecución, lo que ha comenzado a identificarse como Ciencias de Datos. Este enfoque tecnológico ha generado un amplio espectro de aplicaciones que integran soluciones tanto del dominio teórico, como el práctico. Las principales ventajas que aportan estas técnicas están vinculadas al aumento de la eficacia de los procesos de toma de decisiones, por lo que se requiere minimizar los tiempos de procesamientos asociados a los procesos con lo que es posible adoptar decisiones proactivas sin supervisión. Un modelo de proceso es esencial para el diseño de la monitorización y los sistemas de control. El proceso de adquisición de la información sobre las anomalías o fallas puede ser efectuado de forma eficiente a través de la extracción de los datos de los procesos disponibles del sistema operativo [1].

En el procesamiento de flujos de información multipropósito dentro del ámbito de las telecomunicaciones como es el caso del procesamiento de video, se requiere la aplicación de herramientas para la gestión de los datos asociados a este proceso. Es necesario considerar que los volúmenes informativos asociados a los flujos de videos tienen una tendencia creciente y son capturados usualmente en tiempo real. En tales circunstancias resulta muy conveniente garantizar la gestión efectiva de la infraestructura a partir de la monitorización en tiempo real y el análisis de datos que se obtienen sobre el estado de los nodos en todo momento, en función de adoptar decisiones más efectivas que conduzcan a un comportamiento proactivo.

Para garantizar el proceso de análisis de la información relativa al funcionamiento de la infraestructura computacional, los nodos computacionales cooperan entre sí durante el procesamiento de los datos. Es necesario gestionar toda la información asociada a cada uno de los nodos de cómputo, los cuales pueden resultar dispositivos de cómputo heterogéneos. En ese caso será necesario garantizar el almacenamiento y recuperación en una base de datos, que debe realizarse con una alta efectividad para mejorar la eficiencia de la toma de decisiones en tiempo real. En esas dos circunstancias los métodos tradicionales de gestión de base de datos resultan insuficientes para lograr su funcionamiento en tiempo real.

Los métodos tradicionales de gestión de bases de datos, como las bases de datos relacionales, presentan grandes limitaciones en este caso de uso [2]. Las bases de datos relacionales son adecuadas para el almacenamiento y consulta de datos estructurados y predefinidos, pero no son tan eficientes en el manejo de grandes volúmenes de datos no estructurados y variables en tiempo real. Además, la recopilación y el análisis de datos en tiempo real pueden requerir una latencia extremadamente baja, lo que significa que los datos deben ser procesados y analizados en cuestión de segundos o incluso milisegundos. Las bases de datos tradicionales pueden no ser capaces de manejar este nivel de latencia, lo que puede resultar en un retraso significativo en la toma de decisiones basadas en los datos recopilados [13].

La solución desarrollada permite garantizar el procesamiento en tiempo real de los datos asociados al funcionamiento del sistema y crea las condiciones para la adopción de decisiones que permitan el adecuado funcionamiento de la infraestructura computacional de manera continua, preservando los datos del procesamiento, registros de estado, contadores de programa, entre otros. El método implementado permite garantizar el reinicio de cada uno de los procesos en ejecución cuando resulte conveniente, reiniciando a partir del punto en el que se encontraban ante la ocurrencia de alguna interrupción por causas externas como pueden ser las interrupciones del fluido eléctrico [3].

2. DESARROLLO DE LA SOLUCIÓN

Las aplicaciones utilizadas para el procesamiento digital de flujos de video necesitan una gran capacidad de cómputo y la realización de una gran cantidad de operaciones matemáticas, por tal razón demandan una gran cantidad de recursos de hardware para su ejecución. Cuando estas aplicaciones son ejecutadas, los volúmenes de información que deben gestionarse tienen una tendencia creciente y son capturados usualmente en tiempo real. En tales circunstancias es necesario garantizar la gestión en tiempo real de los parámetros asociados a los nodos que posibilite su funcionamiento continuo para evitar errores durante la ejecución de las aplicaciones y asegurar su efectividad y confiabilidad [4].

Se desarrolló un sistema de captura, almacenamiento y análisis de parámetros en tiempo real, para asegurar el funcionamiento estable de la plataforma las 24 horas del día, todas las semanas del año (24/7) pese a los inconvenientes

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

que puedan surgir. El sistema está dividido en tres procesos fundamentales: captura y almacenamiento de parámetros asociados al funcionamiento de los nodos de cómputo, análisis en tiempo real del funcionamiento del nodo de cómputo a partir del procesamiento de los parámetros capturados, y la toma de decisiones como resultado de la identificación de situaciones anómalas. La solución propuesta tiene una alta escalabilidad y su desarrollo se está llevando a cabo de una manera progresiva según se van desarrollando los diferentes módulos, a continuación de describe el funcionamiento de la primera etapa del proyecto referido a la captura y el almacenamiento de los datos de medición.

Módulo de captura

Para la obtención de la información referente a los parámetros de estado asociados al sistema se ha empleado la integración entre *scripts Shell* de Linux y módulos programados en lenguaje de programación C++. Los scripts utilizan comandos del sistema operativo Linux cuyas funcionalidades resultan de utilidad para la obtención de información de *hardware* y permiten obtener datos sobre los sensores y sus mediciones, las capacidades de almacenamiento, además de parámetros de estado del nodo como el uso de memoria RAM (*Random Access Memory*), CPU (*Central Processing Unit*), almacenamiento, el Uptime (Tiempo activo), así como las revoluciones del fan de los procesadores [5].

La información que se extrae en tiempo real acerca del funcionamiento de los procesos en ejecución se obtiene del directorio virtual "/proc", en este directorio del sistema operativo Linux se almacena información muy detallada acerca de todos los procesos activos. Se realiza un pre-procesamiento de los datos que son obtenidos en los ficheros existentes en este directorio para lograr una mejor visualización, disminuyendo la complejidad en el procesamiento y análisis posterior [6].

Herramientas utilizadas

Se desarrolló un *script Shell* de Linux para iniciar el servicio máster, que se encarga de la gestión central del proceso de captura y almacenamiento. En programación se usa el término *script* para referirse al código fuente escrito en algún tipo de lenguaje interpretado. Existen diferencias entre los lenguajes de programación compilados y los lenguajes de programación tradicionales. En los tradicionales el código fuente no se compila una sola vez y se transforma en un binario, sino que se necesita de un intérprete que haga de intermediario, y cada vez que se quiera ejecutar el programa, el intérprete debe traducir el código para que la máquina lo "entienda". Existen muchos lenguajes interpretados, como pueden ser el propio usado en el intérprete Bash, que es el más usado en GNU/Linux y otros Unix, además de lenguajes tan conocidos como Perl, Python, Ruby, JavaScript, entre otros. Con estos lenguajes se puede escribir el código del *script*, que no es más que un fichero de órdenes o procesamiento por lotes [7].

Se utilizó el directorio virtual "/proc" como referencia para adquirir la información de los procesos del sistema operativo. Este directorio realmente no existe, pero se puede explorar. Sus archivos de longitud cero, no son binarios ni de texto, pero se pueden examinar y mostrar. Este directorio especial contiene todos los detalles sobre su sistema Linux, incluido su *kernel*, procesos y parámetros de configuración. Es necesario considerar que, en Linux, todo se gestiona como un archivo; incluso se accede a los dispositivos como archivos (en el directorio "/dev"). Aunque se podría pensar que los archivos "normales" son de texto o binarios (o posiblemente archivos de dispositivo o canalización), el directorio /proc contiene un tipo extraño: "archivos virtuales". Estos archivos se enumeran, pero en realidad no existen en el disco; el sistema operativo los crea sobre la marcha si intenta leerlos. La mayoría de los archivos virtuales siempre tienen una marca de tiempo actual, lo que indica que se mantienen actualizados constantemente [8].

El directorio /proc en sí se crea cada vez que arranca el sistema. Se necesita trabajar como *root* (Administrador) para poder examinar todo el directorio; algunos de los archivos (como los relacionados con el proceso) son propiedad del usuario que lo inició. Aunque casi todos los archivos son de solo lectura, algunos de los que se pueden escribir (especialmente en "/proc/sys") permiten cambiar los parámetros del *kernel*. El directorio "/proc" está organizado en directorios y subdirectorios virtuales, y agrupa los archivos por temas similares [9]. Trabajando como *root*, el comando "#: ls /proc" muestra una lista con la estructura de la Fig. 1 donde se pueden observar los PID (Process Identifier) de los procesos activos:

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

```
1     2432  3340  3715  3762  5441  815       devices      modules
129   2474  3358  3716  3764  5445  acpi      diskstats    mounts
1290  248   3413  3717  3812  5459  asound    dma          mtrr
133   2486  3435  3718  3813  5479  bus       execdomains  partitions
1420  2489  3439  3728  3814  557   dri       fb           self
165   276   3450  3731  39    5842  driver    filesystems  slabinfo
166   280   36    3733  3973  5854  fs        interrupts   splash
2     2812  3602  3734  4     6     ide       iomem        stat
2267  3     3603  3735  40    6381  irq       ioports      swaps
2268  326   3614  3737  4083  6558  net       kallsyms     sysrq-trigger
2282  327   3696  3739  4868  6561  scsi      kcore        timer_list
2285  3284  3697  3742  4873  6961  sys       keys         timer_stats
2295  329   3700  3744  4878  7206  sysvipc   key-users    uptime
2335  3295  3701  3745  5     7207  tty       kmsg         version
2400  330   3706  3747  5109  7222  buddyinfo loadavg      vmcore
2401  3318  3709  3749  5112  7225  cmdline   locks        vmstat
2427  3329  3710  3751  541   7244  config.gz meminfo      zoneinfo
2428  3336  3714  3753  5440  752   cpuinfo   misc
```

Figura 1: Listado de directorios y ficheros del directorio virtual "/proc".

Captura y almacenamiento de datos

Los datos que se almacena contienen información referente a los procesos que están siendo ejecutados en tiempo real obtenidos del directorio virtual "/proc", temperaturas de CPU, uso de memoria RAM y porcentaje de uso con respecto los siguientes aspectos: memoria RAM total, almacenamiento, además las revoluciones por minuto del fan del CPU y el *Uptime* de los nodos. Para la validación de la solución propuesta se realizaron experimentos en el nodo máster del clúster que se encuentra en el laboratorio de Cómputo de Alto Rendimiento para las Telecomunicaciones y Electrónica de la Universidad Tecnológica de la Habana. En este laboratorio se realiza el procesamiento de disimiles aplicaciones con énfasis en las de visión computacional. Se instalaron las aplicaciones y librerías necesarias para el funcionamiento adecuado de los *scripts* y programas creados. [10]

Se creó un directorio llamado "pmonitor" dentro del directorio "/etc/" (contiene los ficheros de configuración y utilidades para la administración.), para ubicar los programas ejecutables implementados con C++ encargados de la captura de los datos, y también para los ficheros encargados del almacenamiento de los datos obtenidos, de una manera organizada. Se creó un *script Shell* de Linux llamado "pmonitor.sh", ubicado en el directorio "/etc/init.d/" (se encuentran los *scripts* que son ejecutados al inicio del sistema), este servicio asegura que el *script* máster se ejecute al inicio para mantener un control constante desde el arranque brindando robustez a la solución.

El *script* máster se encarga de invocar continuamente los programas de captura y almacenamiento, estos son:

- "process_capture.cpp": Encargado de recopilar, procesar y almacenar toda la información referente a los procesos activos.
- "general_data.cpp": Orientado a la captura y almacenamiento de los datos provenientes de los sensores, así como los parámetros de estado que contienen información sobre la memoria RAM, el almacenamiento y el *Uptime*, estos datos son guardados en dos ficheros de almacenamiento llamados "general" y "procesos". Por lo tanto, a través del máster se establece una gestión central del proceso de monitoreo del nodo de cómputo [11]. En la Fig. 2 se puede observar una descripción gráfica del proceso de captura y almacenamiento.

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

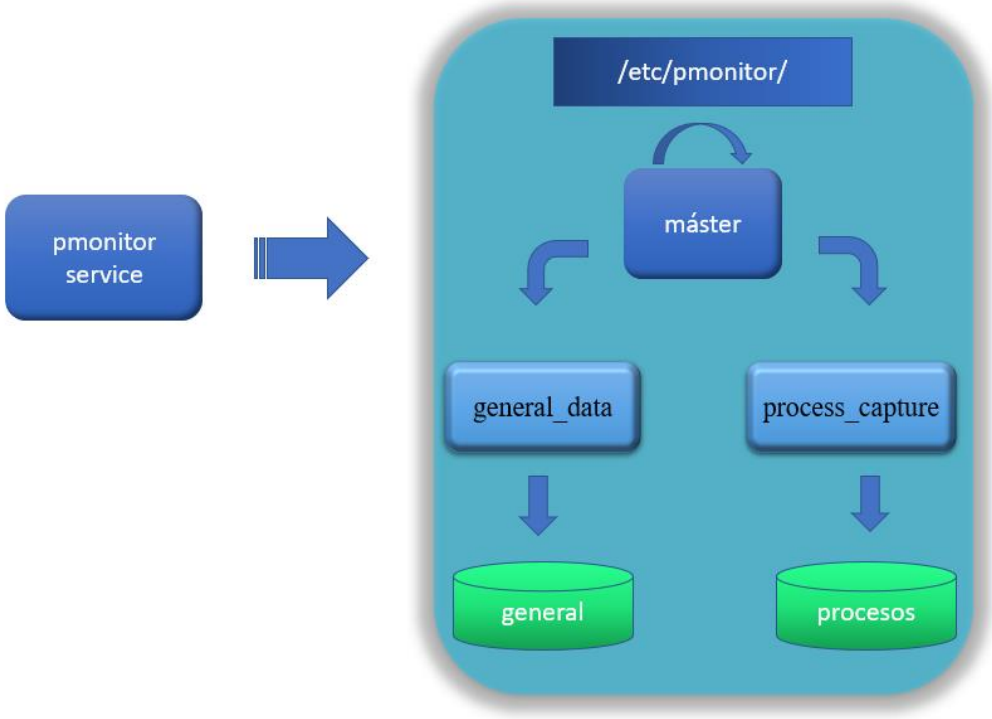

Figura 2: Estructura del proceso de captura y almacenamiento.

El entorno de desarrollo empleado para la implementación de los módulos programados en lenguaje C++ utilizados en la solución propuesta fue *Visual Studio Code*. Su funcionamiento se basa en la invocación del script de captura de los parámetros y el almacenamiento de forma continua para una mejor visualización de los resultados correspondientes a cada nodo de cómputo. Cada nodo constituye la plataforma computacional disponible y en este caso se validó en el Nodo Máster del clúster. Las características técnicas principales de los nodos son: Procesador: Intel Core i7-5500 a 2.4 GHz de 2 núcleos. Tarjetas de Memoria RAM: DDR3 8 GB. Disco Duro: Disco SATA 1 TB con velocidades de lectura/escritura de 6MB/s. Tarjeta gráfica: No. Sistema Operativo: Ubuntu 20.04.

Esta solución fue diseñada para ser ejecutada en todos los nodos computacionales que integran un clúster de alto rendimiento. También se diseñó para una infraestructura distribuida en la que se requiere integrar la potencia computacional en función de solucionar problemas complejos desde el punto de vista computacional. En este caso pueden emplearse nodos heterogéneos que demandan la estimación de los rendimientos computacionales para la asignación balanceadas de los procesos [12].

3. RESULTADOS OBTENIDOS

Se realizaron las pruebas en el nodo descrito, donde se configuró el directorio huésped del programa y se ubicaron los módulos de captura. Se comprobó que al iniciar el sistema operativo en la computadora se estuviese ejecutando el servicio “pmonitor.sh”, encargado de dar inicio al proceso “. /master”. A través de la ejecución del comando “#: ps –aux | grep master” en la consola del sistema se observó que el proceso máster inició y se mantiene activo continuamente ejecutando su función de control sobre los programas encargados de la captura y el almacenamiento de los parámetros de estado e información de los sensores.

Durante la ejecución activa del proceso máster se lleva a cabo una constante escritura en los ficheros destinados al de almacenamiento de los datos. Los dos ficheros de almacenamiento con nombres “general” y “procesos” están destinados al almacenamiento de la información del nodo en tiempo real. El primero guarda la información enviada por el programa de C++ (“general_data.cpp”), referente a la temperatura de CPU, revoluciones del fan y espacios en memoria RAM y disco. El segundo fichero de almacenamiento obtiene los datos sobre los procesos activos a través del programa “process_capture.cpp”, el cual utiliza las funcionalidades de la POO (Programación orientada a objetos).

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

Con ello cada proceso se representa como un objeto, facilitando así la estructuración de los datos obtenidos, su análisis y garantizando la organización en el proceso de escritura del fichero de almacenamiento.

Mientras se encontraba en ejecución el proceso máster, fueron almacenados en tiempo real, los datos capturados y preprocesados por los programas de C++, en los ficheros de almacenamiento. En la Fig. 3 se observa una pequeña muestra de los datos almacenados en el fichero de almacenamiento "general".

```
1   26985*43.000000*69.320000*15.200000*3.360000*29.000000*0
2   26999*41.000000*69.320000*15.200000*3.340000*28.900000*0
3   27001*41.000000*69.320000*15.200000*3.360000*29.100000*0
4   27003*41.000000*69.320000*15.200000*3.340000*28.900000*0
5   27005*44.000000*69.320000*15.200000*3.350000*29.000000*0
6   5171*40.000000*69.360001*15.200000*2.210000*19.100000*0
7   5197*41.000000*69.360001*15.200000*2.190000*18.900000*0
8   5665*39.000000*69.449997*15.200000*2.260000*19.500000*0
9   5719*44.000000*69.449997*15.200000*2.210000*19.100000*0
10  5930*40.000000*69.449997*15.200000*2.250000*19.500000*0
11  6073*44.000000*69.440002*15.200000*1.640000*14.200000*0
12  6398*39.000000*69.440002*15.200000*2.140000*18.500000*0
13  6543*42.000000*69.440002*15.200000*2.190000*19.000000*0
14  6588*39.000000*69.440002*15.200000*2.200000*19.000000*0
15  802*39.000000*69.349998*15.200000*2.320000*20.100000*0
16  803*39.000000*69.349998*15.200000*2.320000*20.100000*0
```

Figura 3: Fichero de almacenamiento "general".

En la Fig. 4 se observa la información obtenida y preprocesada de 10 procesos activos que fueron capturados durante la ejecución del proceso máster. Aquí se pueden observar los PID de cada uno de estos procesos con su nombre (o descripción), así como diferentes parámetros intrínsecos que son de gran a importancia en la etapa de análisis. Estos parámetros brindan la información necesaria para la detección de anomalías y el análisis estadístico y predictivo que se pretende realizar en las etapas posteriores de la solución propuesta.

```
1   1*systemd*S*0*0:0:1*0:0:1*1*0:13:23*14/01/2022 08:26:19*0.349938
2   2*kthreadd*S*0*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
3   3*rcu_gp*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
4   4*rcu_par_gp*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
5   6*kworker/0:0H-kblockd*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
6   7*kworker/0:1-events*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.026152
7   8*kworker/u8:0-i915*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.082192
8   9*mm_percpu_wq*I*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
9   10*rcu_tasks_rude_*S*2*0:0:0*0:0:0*1*0:13:23*14/01/2022 08:26:19*0.000000
```

Figura 4: Fichero de almacenamiento "procesos".

Comprobación de los tiempos de ejecución

Se realizaron pruebas de tiempo de ejecución de los programas de captura y almacenamiento, esta prueba consistió en calcular la demora del tiempo de ejecución de cada programa 10 veces, luego se calculó el tiempo promedio para estas

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

10 muestras. En la Tabla 1 se observan los tiempos referentes a los programas "process_capture.cpp" y "general_data.cpp", donde se refleja una gran rapidez en la captura y el almacenamiento de los datos.

Tabla 1: Tiempos de ejecución de los módulos programados en lenguaje C++.

Prueba	Valor de tiempo ("process_capture.cpp")	Valor de tiempo ("general_data.cpp")	Unidad de medida
1	0.007871	0.001026	Segundos
2	0.009388	0.001212	Segundos
3	0.029612	0.001214	Segundos
4	0.013821	0.001116	Segundos
5	0.034627	0.001289	Segundos
6	0.033434	0.001213	Segundos
7	0.042467	0.001099	Segundos
8	0.034202	0.001078	Segundos
9	0.013541	0.001587	Segundos
10	0.031105	0.001082	Segundos
Promedio	**0.025007**	**0.001192**	**Segundos**

Análisis de los resultados

Como se puede observar en la Tabla 1 de los tiempos de ejecución presentada anteriormente, estos programas se ejecutan en el orden de los milisegundos lo cual es un requisito fundamental para las aplicaciones que trabajan en tiempo real. Ocurrieron algunas fluctuaciones en los valores de tiempo de ejecución, pero las variaciones fueron mínimas, y siempre oscilaron un rango de valores definido, lo que demuestra la estabilidad del programa en lo que se refiere al criterio de tiempo.

La figuras 3 y 4 permiten visualizar que el proceso de adquisición y almacenamiento de los datos se está llevando a cabo satisfactoriamente. La información que está siendo capturada en tiempo real está siendo almacenada de forma organizada en los ficheros de almacenamiento destinados para ese fin. No existe ningún error en la escritura de los ficheros de almacenamientos, y se encuentran en constante modificación durante la ejecución del proceso ". /master".

4. CONCLUSIONES

La solución desarrollada resulta de mucha utilidad para lograr el monitoreo y control de procesos complejos que deben funcionar de forma paralela todo el tiempo sin ningún tipo de interrupción, como es el caso del procesamiento digital de flujos de video en tiempo real. Resulta indispensable la preservación de la integridad de los nodos computacionales ante la ocurrencia de factores externos, como es el caso de las interrupciones del fluido eléctrico, ya que pueden causar grandes afectaciones. No puede soslayarse que los procesos del tipo mencionado deben procesar grandes volúmenes de información y necesitan una gran capacidad de cómputo y la ejecución de funciones matemáticas complejas que no deben ser interrumpidas arbitrariamente.

Para este tipo de aplicaciones es muy importante la disponibilidad de los recursos y de la infraestructura de hardware empleada para el procesamiento porque demandan mucho tiempo de ejecución y requieren seguridad y confiabilidad de la plataforma que lo soporta. De esta forma asegurar una efectividad en el proceso de análisis y en los resultados que se obtienen. Por estos motivos es necesario garantizar la gestión en tiempo real de los parámetros asociados a los nodos, de modo que posibilite su funcionamiento estable, para evitar errores durante la ejecución de las aplicaciones que se ejecutan sobre el sistema y asegurar su efectividad durante todo el tiempo de ejecución.

Con la realización de este trabajo se logró la implementación de un sistema de adquisición de datos asociados a

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

Revista Telemática. Vol. 22 No. 3, Octubre-Diciembre, 2022, p.24-32 ISSN 1729-3804

parámetros de funcionamiento de nodos computacionales heterogéneos a través de la integración de *scripts Shell* de Linux y módulos programados en lenguaje C++. El sistema que resulta sencillo, genérico y escalable; posibilita la realización de la captura de datos desde cada uno de los nodos que integran una infraestructura de cómputo distribuida.

La información correspondiente a los parámetros asociados al funcionamiento del sistema resulta de gran importancia para la ejecución del monitoreo de infraestructuras computacionales que requieren la realización de cálculos computacionales y no cuentan con sistemas protección y respaldo eléctrico. Se realizaron pruebas para demostrar el funcionamiento exitoso del sistema diseñado y se demostraron sus capacidades para el funcionamiento en tiempo real, satisfaciendo los tiempos de ejecución requeridos en rangos de tiempo en el orden de los milisegundos.

REFERENCIAS

[1] H. Luo, H. Zhao, y S. Yin, «Data-Driven Design of Fog-Computing-Aided Process Monitoring System for Large-Scale Industrial Processes», *IEEE Trans. Ind. Inform.*, vol. 14, n.º 10, pp. 4631-4641, oct. 2018, doi: 10.1109/TII.2018.2843124.

[2] L. Liu, S. Yang, L. Peng, y X. Li, «Hierarchical Hybrid Memory Management in OS for Tiered Memory Systems», *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, n.º 10, pp. 2223-2236, oct. 2019, doi: 10.1109/TPDS.2019.2908175.

[3] V. Bandura, A. Malitchuk, M. Pasieka, y R. Khrabatyn, «Evaluation of Quality of Backup Copy Systems Data in Telecommunication Systems», en *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, Kyiv, Ukraine, oct. 2019, pp. 329-335. doi: 10.1109/PICST47496.2019.9061379.

[4] M. Adnan, Y. Lu, A. Jones, F.-T. Cheng, y H. Yeung, «A New Architectural Approach to Monitoring and Controlling AM Processes», *Appl. Sci.*, vol. 10, n.º 18, p. 6616, sep. 2020, doi: 10.3390/app10186616.

[5] V. Gite, «Linux Read CPU Temperature Sensor Chip Data Including Voltage and Fan Speed With lm-sensors», nixCraft, 2021. https://www.cyberciti.biz/faq/howto-linux-get-sensors-information/ (accedido 16 de agosto de 2022).

[6] «Learn more about your Linux system with inxi | Enable Sysadmin», 2021. https://www.redhat.com/sysadmin/learn-more-inxi (accedido 16 de agosto de 2022).

[7] Isaac, «¿Qué es un script? Te lo explicamos de forma simple...», *Linux Adictos*, 23 de marzo de 2018. https://www.linuxadictos.com/que-es-script.html (accedido 16 de agosto de 2022).

[8] J. Kim, W. Choe, y J. Ahn, «Exploring the Design Space of Page Management for Multi-Tiered Memory Systems», p. 15, 2021.

[9] D. Both, «Using the Linux FHS», en *The Linux Philosophy for SysAdmins*, Raleigh, North Carolina, USA: Apress, Berkeley, CA, 2018, pp. 81-105. [En línea]. Disponible en: https://doi.org/10.1007/978-1-4842-3730-4

[10] A. Widjajarto, D. W. Jacob, y M. Lubis, «Live migration using checkpoint and restore in userspace (CRIU): Usage analysis of network, memory and CPU», *Bull. Electr. Eng. Inform.*, vol. 10, n.º 2, Art. n.º 2, abr. 2021, doi: 10.11591/eei.v10i2.2742.

[11] «Uso y control de servicios en Linux», 2020. https://linuxtotal.com.mx/index.php?cont=info_admon_003 (accedido 16 de septiembre de 2022).

[12] B. Gerofi *et al.*, «Performance and Scalability of Lightweight Multi-kernel Based Operating Systems», en *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Vancouver, BC, may 2018, pp. 116-125. doi: 10.1109/IPDPS.2018.00022.

[13] J. LU, I. HOLUBOVÁ, «Multi-model Databases: A New Journey to Handle the Variety of Data», ACM Comput. Surv, p. 38, Jun. 2019, doi: doi.org/10.1145/3323214.

SOBRE LOS AUTORES

Daniel Rubén García Avila. Ingeniero en Telecomunicaciones y Electrónica CUJAE. Profesor Adiestrado, Integrante del proyecto TeleHPC de la Facultad de Ingeniería en Telecomunicaciones y Electrónica de la CUJAE, ORCID: https://orcid.org/0000-0002-9615-8544.

Omar Antonio Hernández Duany. Licenciado en Análisis de Sistemas Especiales de Comunicaciones y Licenciado en Cibernética Matemática UH. Máster en Ciencias en procesamiento digital de señales CUJAE. Investigador Auxiliar, Profesor Auxiliar, Jefe de la disciplina de programación y de Laboratorio de Computo de Alto Rendimiento

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele

para las Telecomunicaciones de la Facultad de Ingeniería en Telecomunicaciones y Electrónica. ORCID: http://orcid.org/0000-0002-0073-1036.

CONFLICTO DE INTERESES

Los autores no manifiestan conflicto de intereses ni de las instituciones afiliadas en relación al contenido del artículo aquí reflejado.

CONTRIBUCIONES DE LOS AUTORES

Ambos autores realizaron la conceptualización, preparación, creación y desarrollo del artículo, revisión crítica de cada una de las versiones del borrador del artículo y aprobación de la versión final a publicar.

Manuscrito recibido: 11-10-2022, aceptado: 29-05-2023
Sitio web:http://revistatelematica.cujae.edu.cu/index.php/tele