

**IMPLEMENTACIÓN DE COMPONENTE PARA  
ANÁLISIS DE TELEMETRÍAS EN CLUSTER DE  
ALTO DESEMPEÑO PARA EL PROYECTO  
AEROESPACIAL UNIANDINO – PUA**

DANIEL VALBUENA SOSA

UNIVERSIDAD DE LOS ANDES  
FACULTAD DE INGENIERÍA  
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN  
BOGOTÁ D.C.  
2013

**IMPLEMENTACIÓN DE COMPONENTE PARA  
ANÁLISIS DE TELEMETRÍAS EN CLUSTER DE  
ALTO DESEMPEÑO PARA EL PROYECTO  
AEROESPACIAL UNIANDINO – PUA**

**DANIEL VALBUENA SOSA**

Trabajo de grado presentado como requisito para optar  
por el título de Maestría en ingeniería de sistemas y  
computación

**DIRECTOR  
DARÍO CORREAL**

**UNIVERSIDAD DE LOS ANDES  
FACULTAD DE INGENIERÍA  
DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN  
BOGOTÁ D.C.**

**2013**

**IMPLEMENTACIÓN DE COMPONENTE PARA  
ANÁLISIS DE TELEMETRÍAS EN CLUSTER DE  
ALTO DESEMPEÑO PARA EL PROYECTO  
AEROESPACIAL UNIANDINO – PUA**

APROBADO POR

---

DARÍO CORREAL  
DIRECTOR

---

KELLY GARCES  
JURADO

---

FABIO ROJAS  
JURADO

---

FECHA APROBACIÓN

## **RESUMEN**

El Proyecto Uniandino Aeroespacial (PUA), tiene como objetivo principal la formación de ingenieros de diferentes áreas en proyectos de desarrollo aeroespacial, desde la perspectiva académica e investigativa como industrial.

El trabajo realizado en el proyecto PUA se encuentra apoyado por los departamentos de ingeniería mecánica, eléctrica – electrónica y de sistemas y computación. Cada departamento tiene asignado un conjunto de actividades específicas con el fin de alcanzar el objetivo propuesto del proyecto. Para el departamento de sistemas y computación las principales actividades a realizar son: A) Diseño e implementación de sistemas adaptativos para el manejo de la información de misiones. B) Diseño e implementación de sistemas de interpretación de información telemétrica y audiovisual de lanzamientos en tiempo real. C) Diseño e implementación de sistemas de manejo y control de información para misiones y lanzamientos aeroespaciales.

Durante los diferentes lanzamientos y pruebas de vehículos aeroespaciales se requiere poder realizar diferentes análisis sobre las telemetrías generadas en tiempo real para obtener información como: trayectoria actual, trayectoria esperada, coeficiente de arrastre entre otros, pero dada la cantidad de información obtenida y la complejidad matemática de los cálculos a realizar en cada tipo de análisis se ha detectado que la capacidad de procesamiento de un equipo de cómputo de escritorio o portátil no es suficiente para poder entregar los resultados de los diferentes análisis en el tiempo esperado, resultados que son vitales para el monitoreo de la misión y a futuro poder tomar decisiones durante el lanzamiento de los vehículos aeroespaciales. En este trabajo se hace énfasis en la

implementación del sistema “Componente para Análisis de Telemetrías – CAT” basado en una plataforma de computación de alto desempeño (HPC) para el análisis de las telemetrías generadas durante las diferentes pruebas y lanzamientos realizados en el proyecto PUA. Se demuestra que utilizando una arquitectura de solución fundamentada en el paradigma de programación paralela, es posible implementar el sistema CAT que responda a la necesidad de cómputo requerido para los análisis definidos y venideros en tiempo real para el proyecto PUA.

## **AGRADECIMIENTOS**

Realizo mis más sinceros agradecimientos a mi asesor Dario Correal por todas las enseñanzas, consejos y formación que me ha brindado durante todos estos años, por ser un referente en mi vida y trayectoria en la universidad durante más de 15 años. A mi familia por brindarme todo su apoyo incondicional, fe y energía en mi vida, a mis amigos por estar presentes en los momentos más indicados, a Ricardo Hernandez por su colaboración en la parte experimental, y finalmente a la Universidad de los Andes por brindarme el tiempo, espacio y recursos para desarrollar mi estudio y trabajo de grado.

## TABLA DE CONTENIDOS

Capítulo	Página
RESUMEN .....	iv
AGRADECIMIENTOS .....	vi
TABLA DE CONTENIDOS .....	vii
CAPITULO I: INTRODUCCIÓN .....	1
Organización del documento .....	1
Términos y convenciones .....	2
Antecedentes y contexto .....	5
Proyecto PUA .....	5
Sistema C3 .....	7
CAPITULO II: ASPECTOS CONCEPTUALES Y HERRAMIENTAS DE APOYO ...	10
Clúster HPC .....	10
Programación en paralelo .....	12
Redes Infiniband .....	13
MPI .....	15
Threads.....	17
Programación hibrida.....	18
Comunicación síncrona - asíncrona .....	20
Patrones de diseño en la programación paralela .....	22
Encontrar concurrencia .....	23
Patrones de descomposición .....	24
Patrones de análisis de dependencias.....	24
Patrones de evaluación del diseño .....	25
Estructura del algoritmo.....	26
Organización por tareas .....	26
Organización por descomposición de los datos .....	27
Organización por flujo de datos .....	27
Estructuras de soporte.....	27
Estructuras de programa .....	28
Estructuras de datos .....	29
Mecanismos de implementación.....	30
Gestión threads / Procesos .....	30
Atributos de calidad del sistema - Desempeño .....	31
Ley de Amdahl.....	31
Procesamiento de filtros digitales .....	35
Integración numérica .....	38
CAPITULO III: GENERALIDADES DEL PROYECTO .....	42
Problemática .....	42
Objetivo general.....	43
Objetivos específicos .....	43
Alcance .....	43

Contribuciones .....	44
<b>CAPITULO IV: ESTRATÉGIA DE SOLUCIÓN .....</b>	<b>45</b>
Aspectos críticos y estratégicos para las decisiones de diseño .....	45
Concurrencia .....	45
Desempeño.....	45
Extensible.....	46
Escalabilidad .....	46
Reglas indicadas por el proyecto PUA .....	46
Restricciones del proyecto PUA y tecnológicas .....	46
Requerimientos .....	47
Diagrama de contexto .....	48
Atributos de calidad .....	49
Diseño del sistema a través del uso patrones de diseño para software paralelo .....	49
Encontrar la concurrencia .....	50
Patrones de descomposición: .....	50
Agrupamientos de tareas identificadas .....	54
Orden de ejecución de las tareas en cada grupo .....	54
Estructura del algoritmo.....	55
Estructuras de soporte .....	57
Mecanismos de implementación.....	58
Gestión Procesos/Threads.....	58
Sincronización.....	59
Comunicación .....	60
Vistas arquitecturales de la solución.....	60
Vista funcional global .....	60
Vista componentes .....	61
Vista de despliegue .....	62
Vista de concurrencia.....	63
Flujo de información.....	64
Implementación de filtro LP - FIR.....	65
Implementación integración numérica.....	66
<b>CAPITULO V: EXPERIMENTACIÓN, VALIDACIÓN Y RESULTADOS .....</b>	<b>69</b>
Escenario 1 .....	69
Contexto.....	69
Descripción del experimento .....	70
Tipo de análisis a realizar .....	71
Experimentación .....	71
Resultados obtenidos .....	72
Comparación e interpretación de los resultados .....	73
Escenario 2.....	78
Contexto.....	78
Descripción del experimento .....	78
Tipo de análisis a realizar .....	80
Experimentación: .....	80
Resultados obtenidos: .....	81
Comparación e interpretación de los resultados: .....	81

Escenario 3.....	86
Contexto.....	86
Descripción del experimento .....	87
Tipo de análisis a realizar .....	90
Experimentación .....	90
Resultados obtenidos .....	91
Comparación e interpretación de los resultados .....	93
Análisis e interpretación general de los resultados .....	96
Análisis de Ley Amdahl.....	98
CAPITULO VI: CONCLUSIONES .....	100
TRABAJO FUTURO.....	101
TRABAJOS RELACIONADOS .....	102
BIBLIOGRAFIA .....	104

# CAPITULO I: INTRODUCCIÓN

## *Organización del documento*

El presente documento está dividido en seis capítulos, cada uno aborda un tema en particular de la tesis en la siguiente forma:

- Introducción: En este capítulo se da una definición de la terminología usada, se realiza una contextualización del proyecto PUA y sus antecedentes.
- Aspectos conceptuales y herramientas de apoyo: En este capítulo se da un acercamiento a las herramientas de apoyo y conceptos necesarios para la contextualización en el diseño e implementación del sistema de “Componente para Análisis de Telemetrías – CAT”.
- Generalidades del trabajo de tesis: Este capítulo se enfoca en la problemática sobre la cual se desarrolló la tesis, los principales objetivos y motivadores que respaldan el trabajo realizado, así como la justificación de este.
- Estrategia de solución: En este capítulo se hace énfasis en la estrategia utilizada para la implementación de la arquitectura de solución del sistema, así como de los componentes a nivel de diseño y análisis que estructuran la solución.
- Experimentación, validación y resultados: Este capítulo define los experimentos con los cuales se realiza la validación de la solución propuesta, además de presentar una comparación de los resultados.
- Conclusiones: En este capítulo se presentan las conclusiones de la tesis basándose en los resultados obtenidos de la experimentación. También ofrece síntesis del trabajo futuro a realizar y de los trabajos relacionados.

## **Términos y convenciones**

Los términos y convenciones usados en este documento son:

- PUA: Proyecto Uniandino Aeroespacial, es una plataforma de actividades tanto académicas, como de investigación y desarrollo, orientadas a la promoción de la ingeniería aeroespacial en la universidad de los Andes y en Colombia [1].
- C3: Sistema de Comando y Control, es la plataforma de software que contiene los componentes del sistema de monitoreo, transmisión y control remoto de misiones aeroespaciales no tripulada del Proyecto Uniandino Aeroespacial – PUA [2].
- CAT: Componente para Análisis de Telemetrías, corresponde al componente de procesamiento de información telemétrica en el sistema C3, éste se encarga de realizar todos los análisis de aceleración, velocidad, desplazamiento entre otros, sobre la información obtenida de los sensores y telemetrías desplegados en los vehículos aeroespaciales.
- Telemetría: Es un sistema de medición de magnitudes físicas con la capacidad de ser transmitidas a un observador lejano [3] a través de un sistema de telecomunicaciones [4].
- HPC: High Performance Computing – Computación de alto desempeño: Es un paradigma de computación que incluye la computación de cálculos complejos basándose en el procesamiento paralelo y/o distribuido, lo que permite una ejecución más rápida de las tareas intensivas requeridas, tales como la investigación sobre el clima, modelamiento molecular, simulaciones físicas, criptoanálisis, modelos geofísicos, diseño automotriz y aeroespacial, modelos financieros y la minería de datos entre otros. Estas simulaciones de alto

rendimiento requieren un tipo de plataforma de computación de altas prestaciones o clúster. [5]

- Clúster: Conjunto de nodos interconectados a través de una red de alto desempeño y baja latencia, tiene la característica de tener un nodo principal o maestro el cual tiene el rol de coordinar los diferentes trabajos a los demás nodos y de permitir el acceso a los usuarios a este sistema. [6].
- Nodo: Máquina o equipo de cómputo de altas prestaciones interconectado a otros nodos por una red de alto desempeño y baja latencia para formar así un clúster. El principal propósito es el de realizar procesamiento de computación de alto desempeño - HPC.
- Núcleo: Núcleo o unidad central de procesamiento existente en un chip o socket de un nodo de computo.
- MPI: Es un conjunto de librerías estándar que permiten la creación, administración de procesos y comunicación entre estos por medio de intercambios de mensajes (comúnmente en C, C++ y fortran) sobre un clúster [7].
- Proceso: Es la interacción entre el procesador y la memoria de un nodo, en la cual el procesador ejecuta una secuencia de instrucciones sobre un conjunto de datos alojados en memoria [8].
- Concurrency: Es la característica de un proceso, objeto o componente de un sistema para ejecutar un conjunto de operaciones lógicas de forma simultánea [9].
- Problema condición de carrera: Problema que se presenta cuando dos o más procesos o threads acceden de forma no controlada a un recurso compartido ocasionando una inconsistencia en la información [66].

- Thread: Un thread o hilo de ejecución hace referencia a la unidad de ejecución con un flujo de control independiente, en términos computacionales es la mínima unidad de trabajo, se encuentran asociados a un procesos y comparten el entorno de este [10].
- Latencia y ancho de banda: Un modelo que caracteriza el tiempo total para la transferencia de mensajes entre dos nodos de computo se define como la suma de un costo fijo más un costo variable que depende de la longitud del mensaje:

$$T_{transferencia\ mensaje} = \alpha + \frac{N}{\beta}$$

El  $\alpha$  costo fijo se llama latencia y es esencialmente el tiempo que se tarda en enviar un mensaje vacío usando el medio de comunicación, desde el momento en el cual la rutina de envío del mensaje es invocada hasta que los datos son recibidos por el destinatario. La latencia (en unidades de tiempo) incluye una sobrecarga debido al software y hardware que están presentes en el proceso, además del tiempo que toma el mensaje para atravesar el medio de comunicación. El  $\beta$  ancho de banda (dado en alguna medida de bytes por unidad de tiempo) es una medida de la capacidad del medio de comunicación. N es la longitud del mensaje [11].

- Infiniband: Es una tecnología de interconexión de baja latencia y de un gran ancho de banda con un bajo costo de procesamiento y es ideal para transportar múltiples tipos de tráfico (comunicación, clustering, almacenamiento) sobre una sola conexión [12].
- Filtro digital: Es un procedimiento numérico o algoritmo que transforma una secuencia de datos de entrada (input) en una secuencia de datos de salida (output)

la cual contiene un tipo deseable de características tales como menos ruido o interferencia en los datos. [13]

## **Antecedentes y contexto**

### **Proyecto PUA**

El Proyecto Uniandino Aeroespacial (PUA), se concibe como una plataforma de actividades tanto académicas, como de investigación y desarrollo, orientadas a la promoción de la ingeniería aeroespacial en la Universidad de los Andes y en Colombia. Su principal objetivo consiste en la formación de ingenieros de cualquier área con orientación a las actividades aeroespaciales tanto desde la perspectiva académica/investigativa, como industrial [14].

En el marco de sus actividades, el grupo PUA propone un objetivo en el mediano plazo, el cual es el desarrollo sistemático de vehículos aeroespaciales con una meta clara, la de llevar cargas útiles cada vez más alto dentro de la atmósfera colombiana hasta cruzar la frontera del espacio exterior [15].

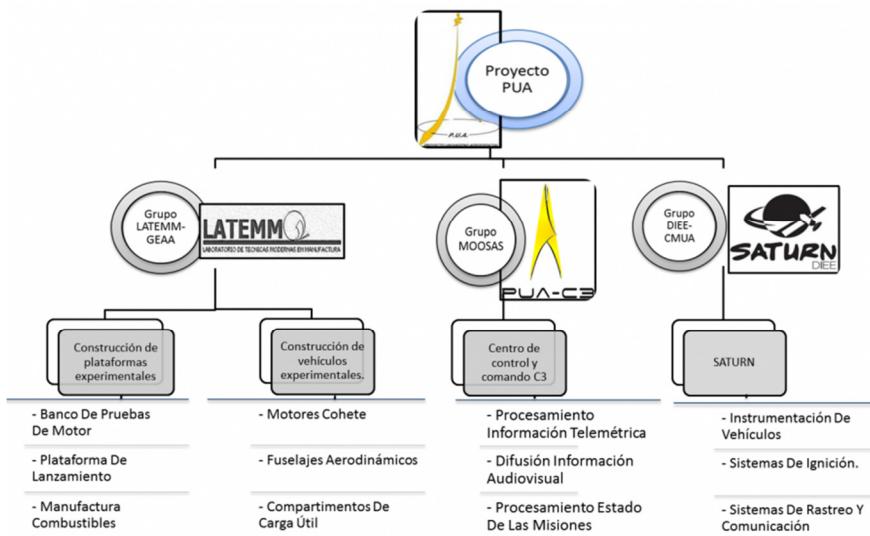


Fig. 1. Organigrama proyecto PUA, tomado del sitio PUA[14].

El departamento de ingeniería de sistemas y computación representado por el grupo de investigación MOOSAS tiene asignado un conjunto de actividades con el fin de proveer [14]:

- Sistemas adaptativos para el manejo de la información en las misiones.
- Diseño e implementación de sistemas de interpretación de información telemétrica y audiovisual de lanzamientos en tiempo real.
- Diseño e implementación de sistemas de manejo y control de información para misiones y lanzamientos espaciales.

Actualmente, el proyecto PUA ha realizado diferentes misiones para afianzar los conocimientos y capacidades tecnologías necesarias para lograr el objetivo principal, en este momento se encuentra en el desarrollo de la misión “SÉNECA VIII” la cual tiene como objetivo realizar un lanzamiento de un vehículo de desplazamiento vertical propulsado por un motor cohete tipo Candy con un sistema de transmisión de datos e imágenes en tiempo real, además deberá lograr una altura máxima de 2 kilómetros [16].

Teniendo en cuenta que en esta misión se contará con un sistema de transmisión de telemetrías generadas por el lanzamiento y dado el requerimiento de diseñar e implementar un sistema de interpretación y análisis de información telemétrica en tiempo casi instantáneo se decidió realizar una primer acercamiento a la implementación de este sistema de análisis en un computador portátil en el sitio de lanzamiento, sin embargo, se detectó que dado el volumen de datos a manejar (más de tres mil tramas de información por segundo) y la complejidad matemática en los análisis de esta información, no es posible entregar esta información interpretada en un tiempo aceptable.

Debido a lo anterior, se ha identificado la necesidad de realizar la implementación del sistema para análisis de telemetrías en una plataforma computacional de alto desempeño que responda adecuadamente al volumen de datos y tenga la capacidad de procesamiento para ejecutar los diferentes tipos de análisis, además de entregar la información procesada en el tiempo esperado, de igual forma, se determinó que el sistema debe utilizar paradigmas de programación paralela híbrida entre MPI e hilos de ejecución ya que los diferentes análisis a realizar pueden ser particionados y resueltos de forma simultánea para así obtener el máximo de provecho de los procesadores multinúcleo que en la actualidad están cambiando sustancialmente a nivel de hardware y software los paradigmas de programación [17].

Para este caso de estudio se decidió utilizar la plataforma computacional de alto desempeño de la facultad de ingeniería en la Universidad de los Andes. Esta plataforma posee una alta capacidad de procesamiento e intercomunicación, además posee las herramientas de programación y ejecución necesarias para responder a las necesidades de implementación y uso del sistema de análisis de telemetrías.

## **Sistema C3**

Con base en la documentación existente del sistema C3 [2], el sistema de comando y control de misión del grupo PUA es un proyecto universitario que busca automatizar, sistematizar, mejorar el tiempo en la recopilación de información, el monitoreo, control y la toma de decisiones de críticas de procesos y protocolos experimentales en las misiones. El sistema se divide básicamente en cuatro secciones principales:

La primera sección comprende la instrumentación y las computadoras a bordo de los vehículos de prueba y misiones realizadas. Estos recopilarán y transmitirán información del ambiente (velocidad de viento, temperatura, nivel de ruido, etc.), monitorearán el estado de los componentes (válvulas, tubería, cableado, etc.) y controlarán los periféricos necesarios para el cumplimiento de los objetivos de la misión (ignición de motor, control de válvulas, corte de potencia a los componentes, etc.). Además de recopilar la información sustancial para los objetivos de los experimentos (presión de cámara, temperatura de combustión, velocidad de flujo, fuerza de empuje, altura de vuelo, etc.).

La segunda sección del sistema está instalada en el sitio de lanzamiento y es llamada Plataforma en Tierra, ésta es usada como una estación intermedia que comunica el Centro de Control y Comando con el vehículo. Su principal funcionalidad es la de monitorear, compactar, procesar y emitir al servidor C3 información recibida de los puntos de control y de las condiciones del área de lanzamiento. También provee al personal ubicado en la plataforma de lanzamiento la información necesaria para el correcto desarrollo de las misiones.

Como tercera sección se encuentran los Puntos de Control (PC). Los PC son módulos replicables que reciben de las antenas receptoras la información telemétrica del vehículo experimental y la transmiten a la Plataforma en Tierra quien la compacta con la información recibida de las otras antenas.

Por último está la sección del centro de comando, que recibe la información de la plataforma en tierra, del vehículo de pruebas y de otros lugares (lecturas de radar en tierra, informes meteorológicos, informes de vigías, etc.) y las despliega a los diferentes controladores y al jefe de misión, de tal manera que puedan utilizar ésta información para

seguir el protocolo experimental y cumplir con los objetivos de la misión. Por otra parte, el centro de comando sirve como centro de difusión de información a invitados y otras entidades tales como canales de televisión, estaciones de radio, portales de internet y otras instituciones universitarias que estén interesadas en seguir el desarrollo de las misiones.

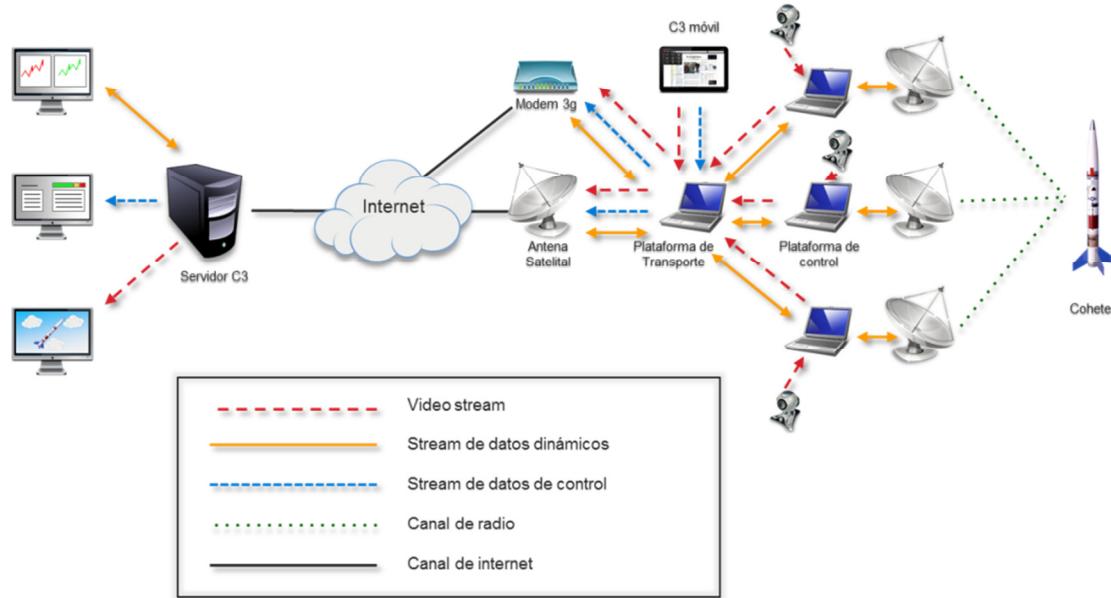


Fig. 2. Diagrama de contexto sistema PUA -C3, tomado de [2].

## **CAPITULO II: ASPECTOS CONCEPTUALES Y HERRAMIENTAS DE APOYO**

Este capítulo hace una introducción a todo el conocimiento necesario a nivel de programación y computación, además de aclarar los algoritmos de apoyo usados para la implementación del sistema de interpretación de información telemétrica con alta capacidad de procesamiento y así realizar el conjunto de análisis necesarios para adaptarlos al proyecto PUA logrando superar el inconveniente de procesar la información y entrega de los resultados esperados en tiempo real al líder de la misión.

### **Clúster HPC**

Un clúster de computación se puede definir como un conjunto de máquinas interconectadas a través de una red de baja latencia y alta velocidad y puede ser escalable en la medida que se adicionen más máquinas o nodos de cómputo al mismo. Están diseñados para proporcionar una mayor capacidad de procesamiento y cálculo que la que podría ofrecer un solo equipo de cómputo [18]. Existe la posibilidad de diseñar e implementar un clúster con cualquier tipo de ordenadores hasta poder implementar uno basado en las tecnologías de los proveedores de hardware líderes del mercado [19]. Por lo general los clústeres HPC se encuentran instalados con sistemas operativos de código abierto, como por ejemplo Linux o Sun Solaris OS [20]. La arquitectura general de un sistema de computación en paralelo se muestra en la siguiente figura:

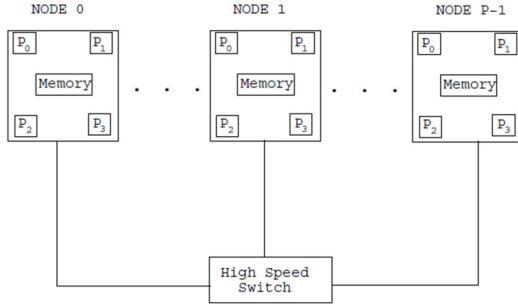


Fig. 3. Arquitectura general de un sistema en paralelo tomado de [6].

Actualmente, la facultad de ingeniería posee un clúster HCP con 10 nodos de cómputo, alojados en el centro de computación avanzada - MOX, las características del clúster de alto desempeño son las siguientes:

- 512 núcleos de procesamiento - AMD – Opteron - Processor 6282 SE
- 896 GB en RAM
- Capacidad de almacenamiento: 10TB sobre un sistema de archivos distribuido LUSTRE.
- Conectividad de alto desempeño Infiniband.
- Capacidad de procesamiento: 4TFlops – 4 billones de operaciones de números flotante por segundo (estimado teórico).
- Sistema de almacenamiento SAN sobre iSCSI

La arquitectura a nivel de hardware sobre la cual está desplegado el clúster es la siguiente:

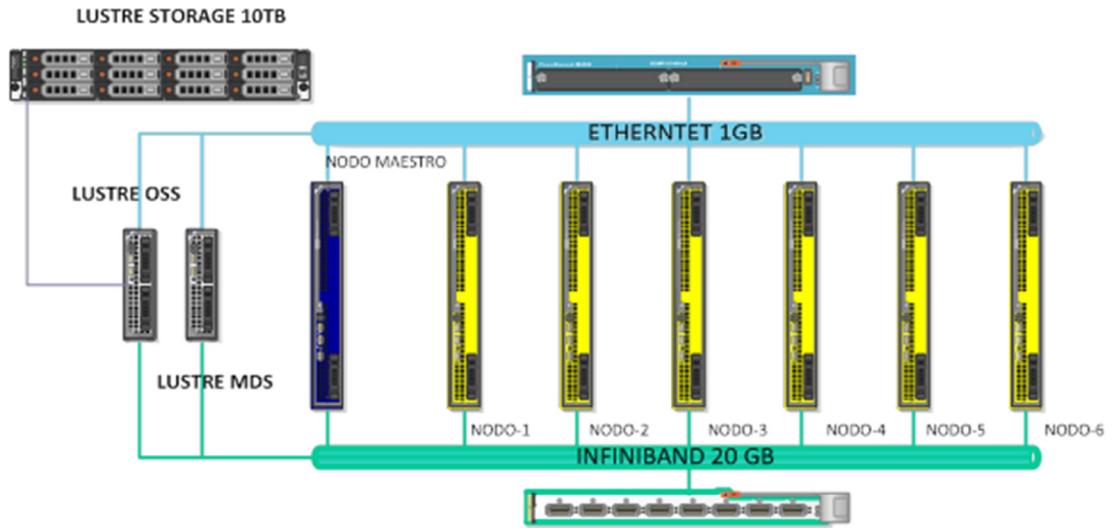


Fig. 4. Arquitectura general del clúster facultad de ingeniería uniandes.

## ***Programación en paralelo***

Tradicionalmente el software ha sido escrito enfocado en una programación serial o secuencial, esto quiere decir que solo una instrucción puede ser ejecutada a la vez en un núcleo de procesamiento, el software o problema a resolver se divide en una serie discreta de instrucciones donde cada instrucción solo puede ser ejecutada después de completar la anterior [28].

La programación o computación paralela permite utilizar de forma simultánea múltiples recursos computacionales para solucionar un problema, como característica de esto, permite ejecutar el software o programa utilizando múltiples procesadores y el problema se puede dividir en partes discretas que pueden ser solucionadas de forma concurrente, a su vez, se divide en un conjunto de instrucciones y estas se ejecutan de forma simultánea en diferentes procesadores, para evitar inconsistencias ej. Condiciones

de carrera diversos mecanismos de control y coordinación se usan en este tipo de programación [29].

Para poder implementar este tipo de programación, el problema computacional debe tener las siguientes cualidades: i) secciones de la lógica de programación se deben poder dividir discretamente para ser resueltas simultáneamente, ii) Debe ser posible la ejecución de múltiples instrucciones en cualquier momento, y iii) El problema debe poder ser resuelto en menor tiempo con múltiples recursos computacionales o clústeres que utilizando un único equipo de cómputo [29].

### ***Redes Infiniband***

Infiniband es un estándar de la industria, basado en canales de comunicación bidireccionales en una arquitectura de interconexión conmutada para comunicación entre equipos y también para almacenamiento [21].

Estas especificaciones definen un estándar abierto para una solución de interconexión escalable de baja latencia con una alta capacidad en ancho de banda. Actualmente se manejan velocidades de transferencia que van desde los 2.5Gbps hasta los 120Gbps dependiendo del tipo de la cantidad de pares o hilos utilizados en el enlace [22]. El hardware utilizado en las conexiones Infiniband tiene la capacidad de hacer accesos directos a memoria de forma remota (RDMA) sin tener que pasar por la CPU del nodo remitente o de los nodos receptores mientras se hacen las operaciones de lectura o escritura. El hardware también se encarga de que las conexiones establecidas sean fiables administrando la integridad de los datos. Esta combinación de características permite que las operaciones de E/S requieran un mínimo de uso en CPU.

Las diferentes especificaciones de Infiniband definen un conjunto de verbos o instrucciones para acceder al hardware, en cada conexión un par de colas de envío y recepción son definidas entre los nodos sobre los cuales se está realizando la comunicación, permitiendo que los diferentes tipos de eventos sean adicionados en una cola especial [23]

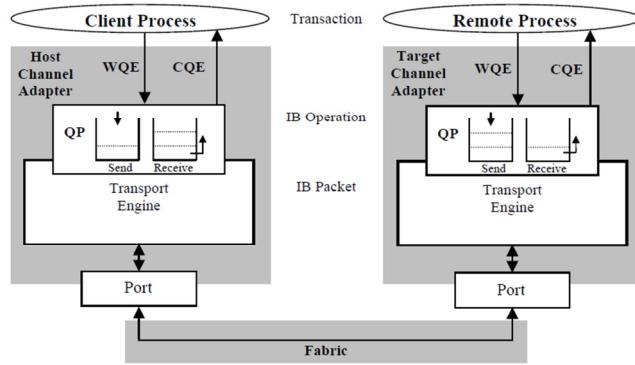


Fig. 5. Stack de comunicación Infiniband tomado de [32].

En el sistema de interconectividad Infiniband que posee la facultad de ingeniería está basado en hardware Mellanox 4X a una velocidad de 20Gbps o DDR, estos dispositivos de red se encuentran conectados entre sí a través de un switch Mellanox M2401G con una capacidad de 960Gbps. A continuación se presenta una comparación del tiempo de cómputo para resolver un problema utilizando la transformada de Fourier entre una red Ethernet y una red Infiniband:

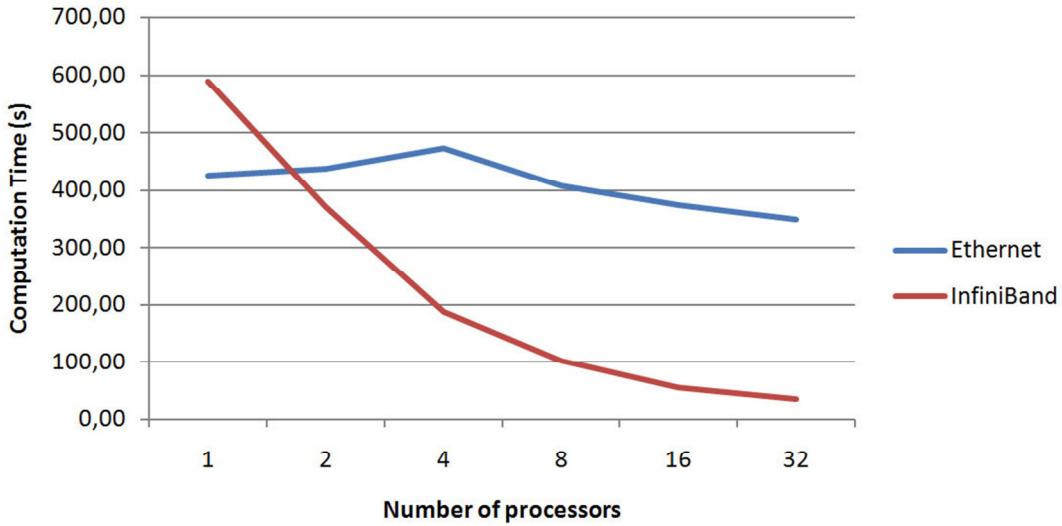


Fig. 6. Comparación de tiempo de computo para resolver una transformada de fourier en un cluster con interconectividad ethernet vs. Infiniband [67].

## **MPI**

MPI (Message-Passing Interface) es la especificación de un conjunto de librerías para el intercambio de mensajes entre procesos [24]. La función principal es la de realizar un intercambio de mensajes en el que los datos se mueven desde un espacio de direcciones o memoria de un proceso ubicado en un nodo a otro proceso localizado en un nodo distinto utilizando instrucciones en cada proceso a través de un canal de comunicación, a este tipo de programación se le llama programación distribuida.

En estas instrucciones existen extensiones que permiten operaciones de comunicación punto a punto, comunicaciones colectivas, acceso a memoria remota, creación dinámica de procesos y operaciones I/O en paralelo entre otros. De esta forma se puede lograr una intercomunicación entre los diferentes nodos de un clúster que estén ejecutando diferentes procesos para una misma aplicación [25] [26].

En el siguiente gráfico presenta una ilustración de intercambio de mensajes entre procesos distribuidos donde el valor de la variable “A” del proceso 1 que está siendo ejecutado en el nodo A es enviada al nodo B en el proceso 2 y almacenada en un espacio de direcciones local de este último en la variable “B”, todo esto utilizando las instrucciones de MPI: MPI\_Isend y MPI\_Irecv:

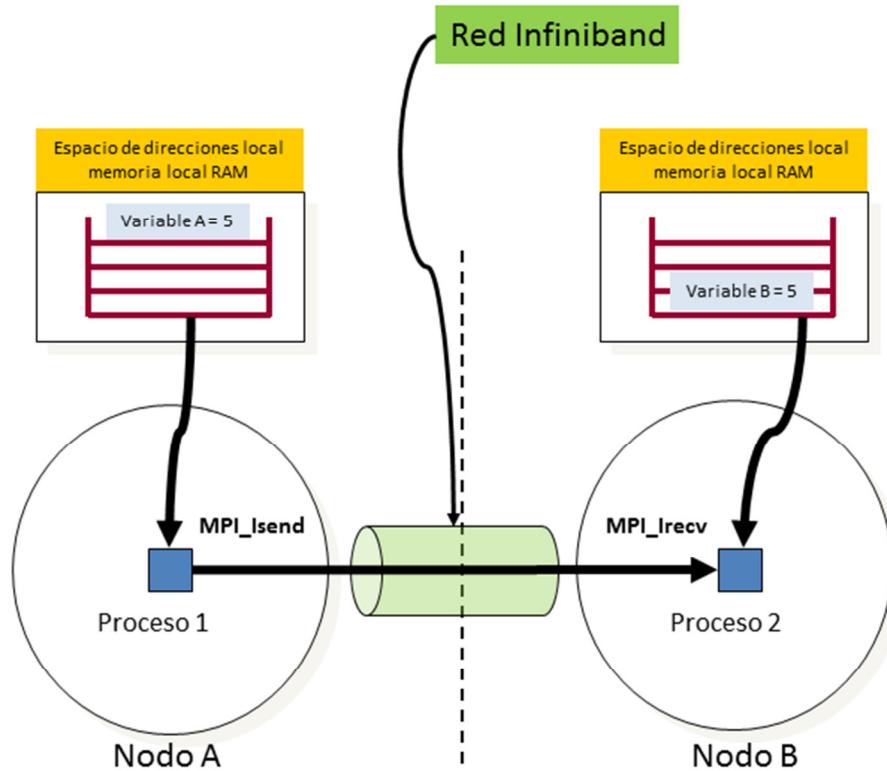


Fig. 7. Esquema general de comunicación MPI entre dos procesos de diferentes nodos.

Cabe resaltar que MPI es la especificación de un conjunto de librerías interface mas no una implementación, además, todas las operaciones MPI están expresadas como funciones, subrutinas o métodos de acuerdo al lenguaje. En la actualidad existen diversas implementaciones de MPI [27].

El clúster de la facultad de ingeniería ofrece diferentes versiones de implementaciones MPI: mvapich, open-mpi y mpich2 e Intel cluster studio sobre lenguajes de programación C, C++ y fortran.

## ***Threads***

Los threads o hilos de ejecución representan la unidad fundamental de ejecución en un sistema computacional y representan un conjunto secuencial de instrucciones y/u operaciones [41]. La implementación de threads a nivel de programación puede denominarse como programación en memoria compartida, v.g. en el contexto de los sistemas UNIX-LINUX los threads están asociados a un proceso y comparten todo el entorno de éste (acceso al mismo espacio de direcciones en memoria), esto conlleva a que los threads ocupen menos recursos que un proceso [40], lo cual en términos prácticos significa que varios threads puedan instanciarse de forma más rápida y ser ejecutados de forma concurrente en un mismo proceso y a su vez puedan usar el espacio de direcciones compartido por el proceso, esto quiere decir que cuando un thread almacena información en este espacio de direcciones cualquier otro thread dentro del mismo proceso la podrá acceder, sin embargo, cada thread también tiene su propio espacio de memoria para así mantener un control sobre las funciones y almacenamiento de datos en variables locales [42]. En la actualidad con el desarrollo tecnológico enfocado hacia los procesadores multinúcleo se da la posibilidad de tener múltiples threads ejecutándose de forma simultánea dentro de un mismo procesador utilizando los núcleos de este, permitiendo el paralelismo en un proceso con la ventaja de que la comunicación entre los threads es más rápida que comparada por ejemplo con la comunicación vía sockets [43].

La siguiente ilustración ilustra un ejemplo de cómo están relacionados los threads con los procesos y memoria. En este caso el nodo cuenta con un procesador de cuatro núcleos, en los que está instanciado un proceso que a su vez tiene creados cuatro threads, uno en cada núcleo. Como se puede evidenciar en la gráfica, cada thread tiene un segmento de memoria propio y ejecutan una operación propia utilizando dos variables locales: ‘X’ e ‘Y’, además se utiliza la variable global “T” que se encuentra en el espacio de direcciones común al proceso y por ende a los thread que se encuentran creados en él:

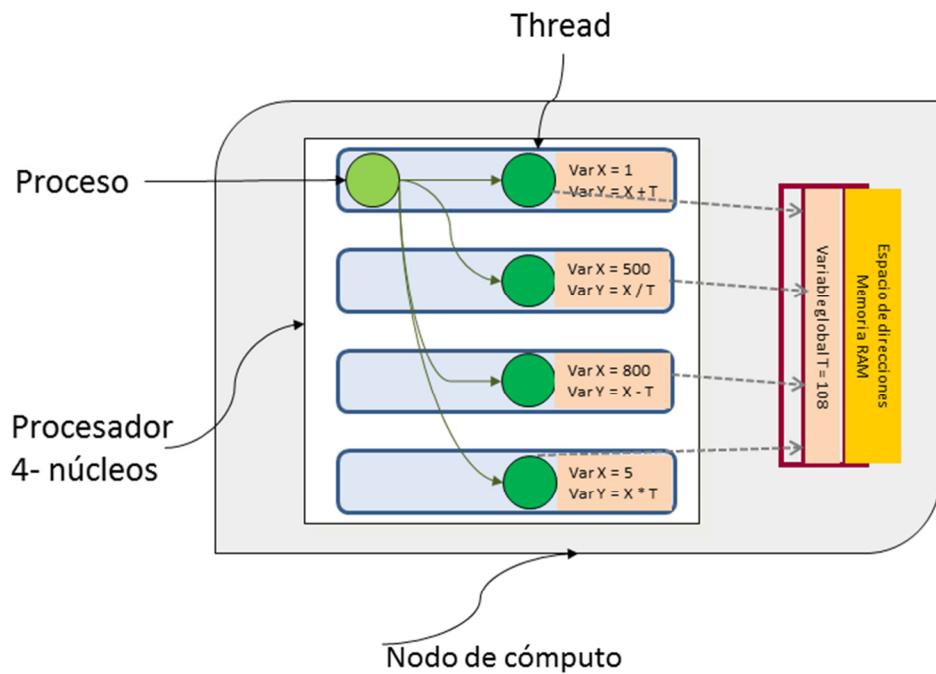


Fig. 8. Esquema general de implementación de los threads en un procesador multinúcleo.

## **Programación hibrida**

Para maximizar el desempeño de una aplicación utilizando un clúster HPC se suele implementar programas que combinen las implementaciones de multithreads y MPI [46]. La combinación de los modelos de programación en memoria compartida (threads) y programación distribuida (MPI) es una idea ya bastante desarrollada [44]. Esta

combinación se denota como modelo de programación hibrida, el cual trata de maximizar las fortalezas de los dos tipos de programación: La eficiencia, almacenamiento en memoria y facilidad de programación a través del modelo de programación en memoria compartida (threads) y la capacidad de escalabilidad que ofrece el modelo de programación distribuida (MPI) [45].

Para representar mejor lo que representa el modelo de programación hibrida se presenta a continuación un ejemplo basado en las ilustraciones de threads y MPI anteriormente elaboradas y donde se combinan los dos modelos de programación:

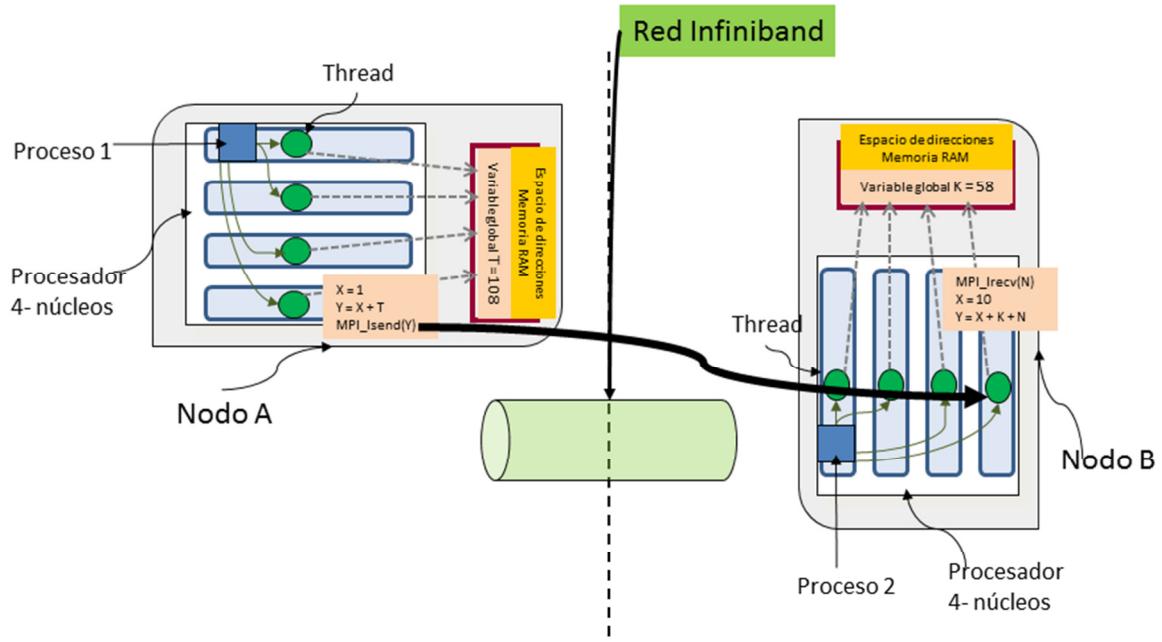


Fig. 9. Esquema general de implementación de programación hibrida.

En este ejemplo se puede ver que la interoperabilidad entre los threads de un proceso se puede complementar con el uso de instrucciones MPI\_Isend a procesos de otro nodo para enviar el valor de la variable “Y”, estos llamados son atendidos por un thread del proceso 2 a través de la función MPI\_Irecv, el cuál almacena el valor en la variable “N”. Según el estándar de MPI existen múltiples niveles de soporte para los threads, estos son:

- **MPI\_THREAD\_SINGLE**: Solo un thread será ejecutado.
- **MPI\_THREAD\_TUNNELED**: Los procesos podrán tener múltiples threads, pero solo el thread principal (el del proceso) podrá realizar llamados a funciones MPI.
- **MPI\_THREAD\_SERIALIZED**: El proceso puede tener múltiples threads, y múltiples threads podrán hacer llamados a funciones MPI pero solo uno a la vez.
- **MPI\_THREAD\_MULTIPLE**: Múltiples threads de un proceso podrán realizar llamados a funciones MPI sin ninguna restricción.

### ***Comunicación síncrona - asíncrona***

Para el intercambio de mensajes entre dos procesos existen dos estrategias generales que prevalecen: modo síncrono y modo asíncrono. Cada uno de estos tiene sus ventajas y desventajas [47].

Las comunicaciones síncronas también son llamadas comunicaciones bloqueantes ya que el proceso o thread que llama a la función de comunicación debe esperar hasta que esta se haya completado [48]. En el caso de las comunicaciones MPI síncronas la función llamada (**MPI\_Send**, **MPI\_Recv**) solo retorna el control al proceso o thread después de haber completado la operación de envío o recepción del mensaje [49].

Por otra parte, las comunicaciones asíncronas, también son llamadas comunicaciones no bloqueantes ya que el proceso o el thread pueden seguir ejecutando otras instrucciones mientras la comunicación se completa [48]. Para el caso de las comunicaciones MPI asíncronas la función llamada (**MPI\_Isend**, **MPI\_Irecv**) retorna de forma inmediata el control al proceso o thread aún sin haber completado la operación. Para garantizar que se ha completado la operación el programa podrá revisar el estado de la operación realizada [50].

Para visualizar de forma práctica las dos estrategias de comunicación, se retoma el ejemplo anterior de la programación híbrida (ver figura 8) y complementándolo utilizando un mecanismo de comunicación síncrona y también de forma asíncrona (Para facilidad en la visualización se restringe el ejemplo a un procesador de un solo núcleo):

En el caso de la comunicación síncrona las funciones son bloqueantes, para la función MPI\_Send, esta no retorna el control al proceso hasta que la función MPI\_Recv haya confirmado la recepción del mensaje, una vez completada la comunicación la función MPI\_Send retorna el control al proceso:

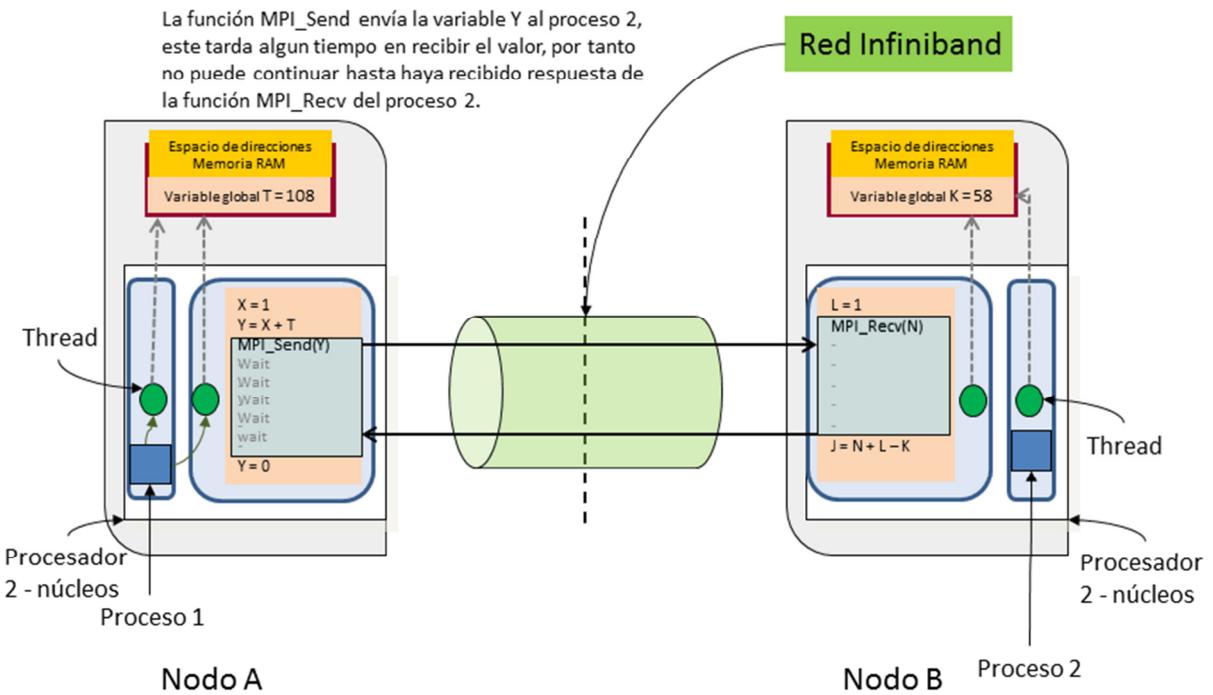


Fig. 10. Esquema general de implementación de programación híbrida.

En el caso de la comunicación asíncrona, las funciones son no bloqueantes, esto quiere decir que las funciones MPI\_Isend y MPI\_Irecv retornan el control al proceso de forma inmediata y el estado de la operación de envío y recepción se controlan con el uso de la función MPI\_Wait:

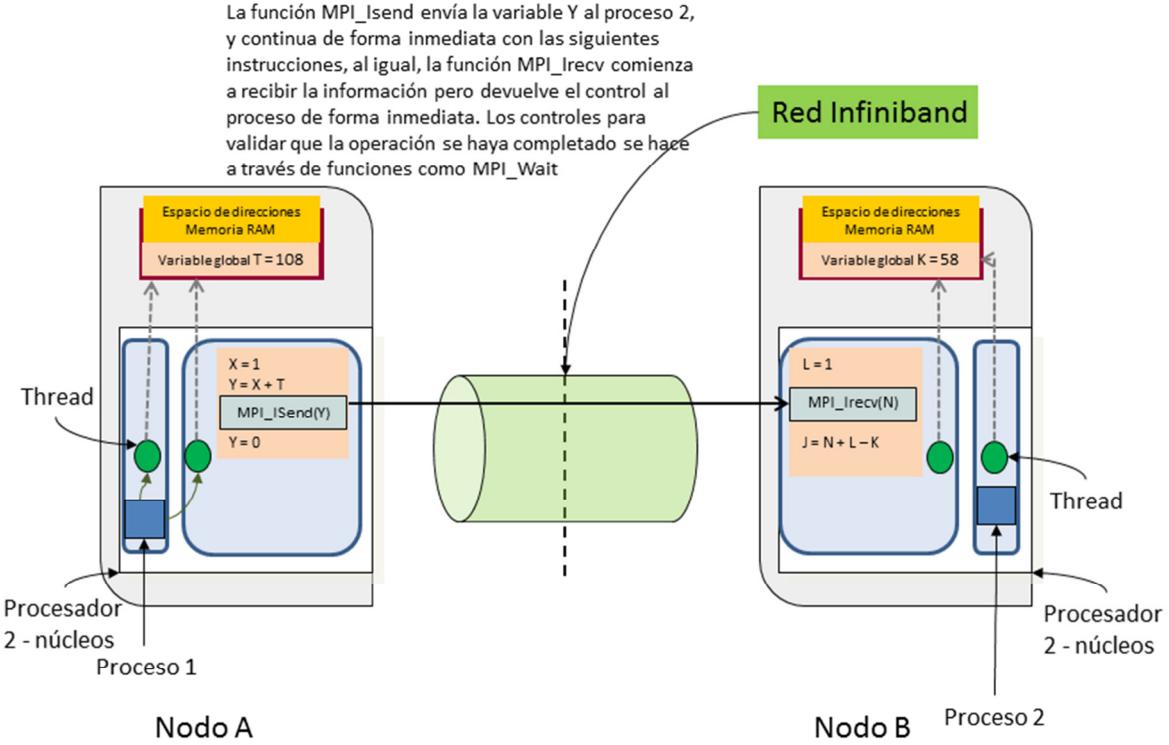


Fig. 11. Esquema general de implementación de programación híbrida.

### **Patrones de diseño en la programación paralela**

Para el desarrollo de un programa, sistema o componente en un ambiente de computación paralela de forma adecuada se deben implementar patrones de diseño con énfasis en programación paralela e hibrida. A continuación se explican un conjunto de patrones de diseño tomando como referencia la literatura existente en este tema [8] [9] [10] [11] [12].

Un patrón de diseño describe una buena solución a un problema recurrente en un contexto en particular, a su vez, se maneja el concepto de un lenguaje de patrones que permite estructurar los patrones de una forma en la cual se puede realizar el diseño de un sistema utilizando este lenguaje. Para el caso de la programación paralela o distribuida el lenguaje de patrones se organiza en cuatro espacios de diseño:

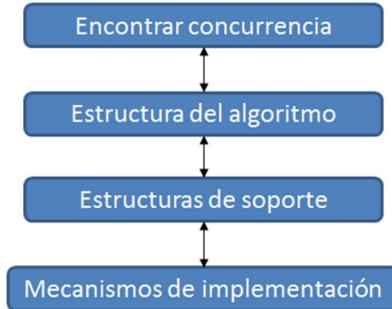


Fig. 12. Espacios de diseño utilizados en programación paralela.

A continuación se da una descripción de cada espacio de diseño:

- Encontrar concurrencia: Trabajar en el dominio del problema para identificar las posibles concurrencias y exponerlas para su uso en el diseño del algoritmo.
- Estructura del algoritmo: El trabajo se centra en estructuras de alto nivel para organizar el algoritmo paralelo.
- Estructura de soporte: Es la transformación de los algoritmos a un pseudocódigo de alto nivel para considerar cómo el programa será organizado y las técnicas a usar para manejar la información compartida.
- Mecanismos de implementación: Pasos finales para implementar el programa en paralelo.

## **Encontrar concurrencia**

En primer lugar se debe considerar el problema a ser resuelto y garantizar que la paralelización de un programa ayudará en su solución: ¿El problema es lo suficientemente complejo?, ¿posee una cantidad considerable de información y la necesidad de que la solución sea realmente rápida?. Entender cuáles partes del problema son computacionalmente intensivas es la clave pues ya que el esfuerzo de paralelizar el

problema se centrará en esas áreas. Para aclarar estos interrogantes se establecen un conjunto de patrones en este espacio agrupados según sus características:

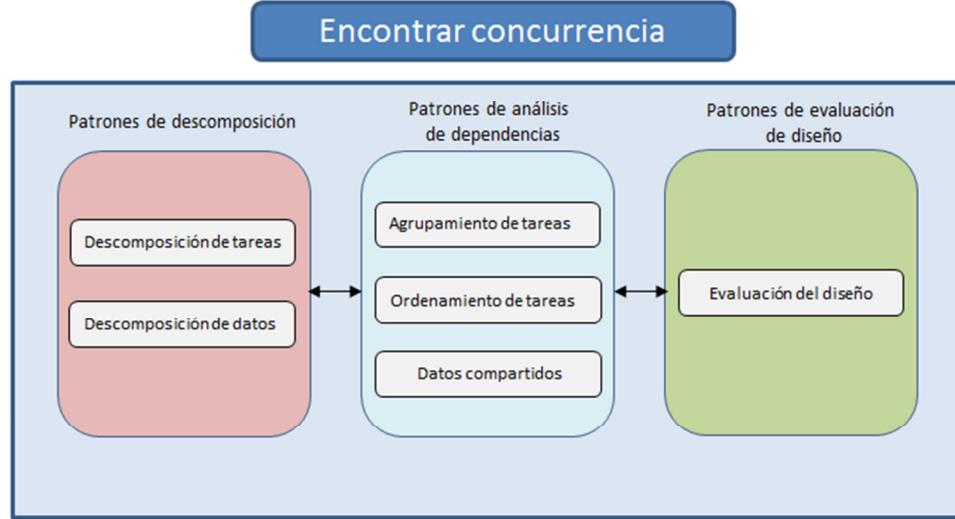


Fig. 13. Espacio de diseño: Encontrar concurrencia.

#### Patrones de descomposición:

El primer paso en el diseño de un algoritmo paralelo es la descomposición del problema en elementos que puedan ser ejecutados de forma concurrente. Los más importantes son:

- Descomposición en tareas: Se ve el problema como un conjunto de instrucciones que pueden ser divididas en tareas que pueden ejecutarse de forma simultánea.
- Descomposición de datos: Hace énfasis en los datos requeridos por las tareas y de cómo estos se pueden dividir en segmentos en tareas concurrentes.

#### Patrones de análisis de dependencias:

Permite realizar una organización de las tareas identificadas en grupos, generar un ordenamiento lógico de las mismas y determinar qué tipo de datos son compartidos entre las tareas:

- Agrupamiento de tareas: Se establece qué tareas se pueden agrupar para así simplificar el manejo de dependencias.
- Ordenamiento de tareas: Se Identifica el orden lógico de ejecución de las tareas dentro de un grupo.
- Datos compartidos: Se determina como la información común es compartida entre los diferentes tipos de tareas agrupadas.

**Patrones de evaluación del diseño:**

Permite establecer si el análisis de descomposición y dependencias es lo suficientemente adecuado para moverse al siguiente espacio de diseño. La evaluación se realiza desde diferentes perspectivas:

- Idoneidad para la plataforma de destino: Determinar la cantidad de núcleos, acceso a memoria e interconectividad entre los nodos que alojarán los diferentes grupos de tareas.
- Calidad del diseño: Los principales características que se evalúan son la simplicidad, flexibilidad y eficiencia, desempeño entre otras.
- Preparación para la siguiente fase de diseño: ¿Las tareas y sus dependencias son regulares o irregulares en cuanto al tamaño?, ¿la interacción entre las tareas es síncrona o asíncrona?, ¿las tareas están siendo agregadas en forma efectiva? Son algunos de los interrogantes a tener en cuenta en esta evaluación.

La salida de este espacio de diseño es la descomposición del problema en los siguientes elementos de diseño: descomposición en tareas, descomposición de datos, agrupamiento y ordenamiento de tareas y análisis de dependencias entre las tareas.

## Estructura del algoritmo

La meta en este espacio es la de refinar el diseño y estar más cerca de un programa que pueda ejecutar las tareas de forma concurrente y realizar un enlace entre estas y las diferentes unidades de ejecución como threads o procesos. De la gran diversidad de formas para definir una estructura de algoritmo por lo menos se debe seguir alguno de los siguientes patrones:

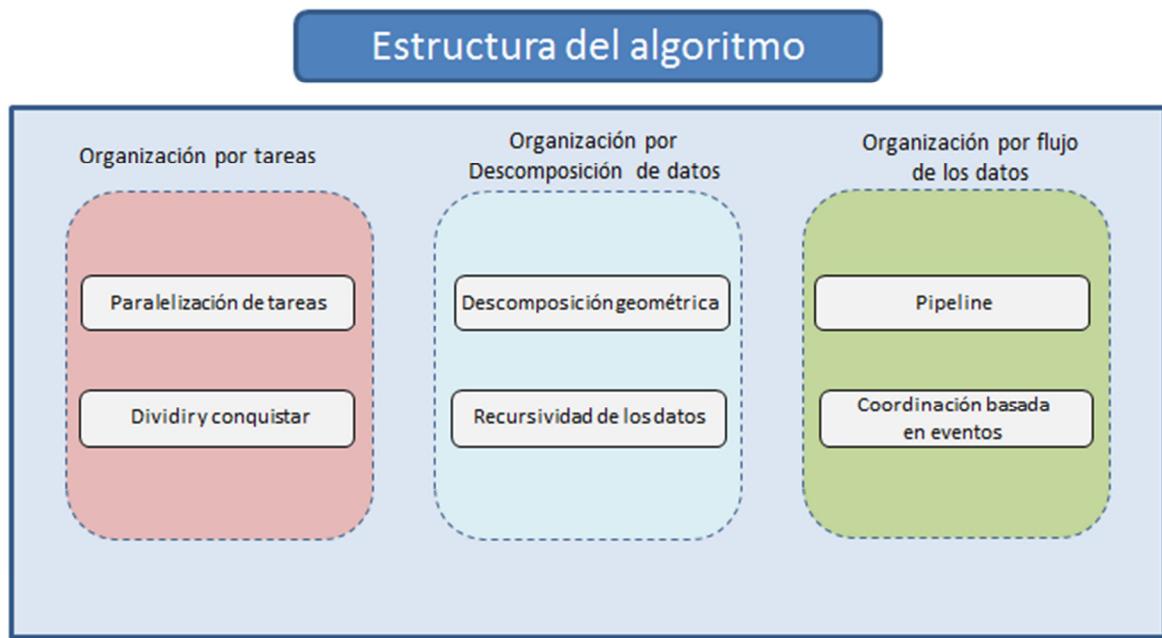


Fig. 14. Espacio de diseño: Estructura del algoritmo.

Es importante tener en cuenta que diferentes aspectos del análisis pueden llevar al diseño en distintas direcciones, para esto se realizará un breve análisis por cada grupo:

### Organización por tareas

Seleccionar esta opción es la mejor cuando se establece que las tareas definen la solución al problema y son el principio predominante además de determinar cómo las tareas están organizadas. Este patrón incluye la situación en la cual existe una dependencia entre tareas o estas son totalmente independientes ('embarrassingly parallel algorithms'),

además en los casos cuando existe una dependencia de tareas en la forma como se comparten los datos.

### *Organización por descomposición de los datos*

Se selecciona cuando la descomposición de los datos es el mayor principio que determina la concurrencia, existen dos patrones en este grupo:

- Descomposición geométrica: Donde el espacio del problema es dividido en subespacios.
- Recursividad de los datos: Cuando el problema se divide en términos de una estructura recursiva de datos, v.g.: un árbol binario.

### *Organización por flujo de datos*

Indica que los datos imponen un orden sobre los grupos de las tareas identificadas, según el orden que puede ser estático y regular se selecciona el patrón de pipeline o por el contrario si es dinámica e irregular se selecciona la opción de coordinación basada en eventos.

## **Estructuras de soporte**

El objetivo de este espacio de patrones es el de servir como etapa intermedia entre el espacio de la estructura algorítmica y de los mecanismos específicos de programación. Se les llama estructuras de soporte debido a que describen las estructuras que soportarán la lógica de los algoritmos paralelos. Estos se encuentran divididos en dos grupos:

## Estructuras de soporte

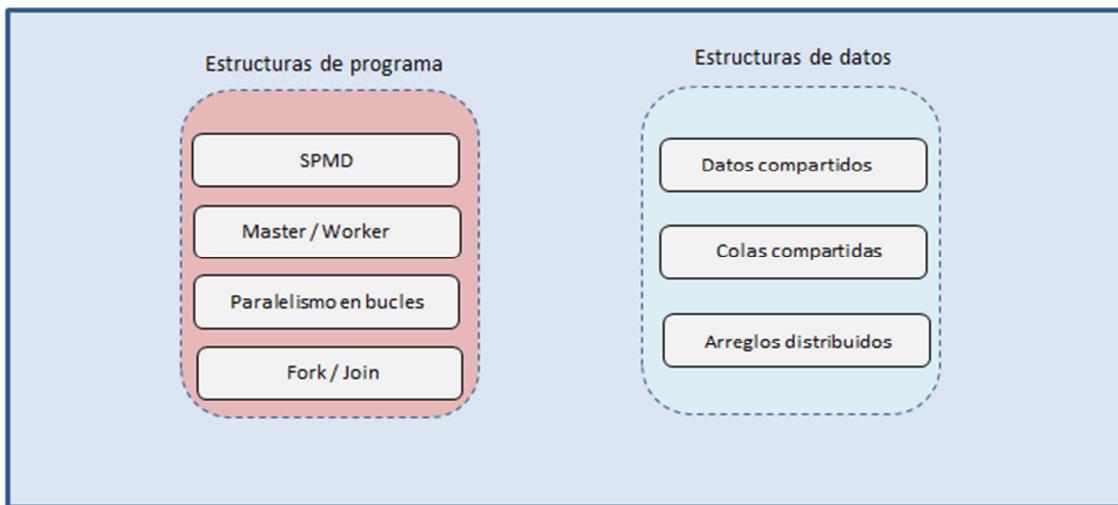


Fig. 15. Espacio de diseño: Estructuras de soporte.

### Estructuras de programa

Estos patrones definen los tipos de mecanismos para estructurar un código fuente en paralelo, estos incluyen:

- SPMD (single program, multiple data): Significa que todos los threads o procesos ejecutan un único programa o conjunto de tareas en paralelo, pero cada uno tiene su propio conjunto de datos.
- Master/Worker: Un proceso o thread maestro invoca a un pool de threads y un conjunto de tareas a ejecutar. De forma concurrente cada thread toma una tarea del conjunto de tareas y se encarga de ejecutarla hasta que todas las tareas del conjunto de tareas se hayan procesado.
- Paralelismo en bucles: Se encarga de transformar un programa serial en el cual las rutinas de bucles predominan.

- Fork/Join: El proceso principal se encarga de crear algunos threads hijos (Fork) para que ejecuten alguna porción de las instrucciones y por lo general espera a que se complete el trabajo de los threads hijos para luego hacer la reincorporación (Join) hacia el proceso principal.

Este tipo de patrones no son del todo exclusivos ni excluyentes entre sí, una combinación de ellos es totalmente válida y depende del criterio del diseñador en cómo aplicarlos.

### Estructuras de datos

Los patrones de este grupo hacen referencia a la relación del manejo de la dependencia de los datos entre procesos y tareas.

- Datos compartidos: Este patrón permite manejar el problema del manejo de los datos entre varios threads o procesos, teniendo en cuenta la integridad de los datos y el desempeño.
- Colas compartidas: Este patrón representa la implementación segura de una cola que mantiene la integridad de la información cuando es manipulada por varios threads.
- Arreglos distribuidos: Este patrón representa las estructuras de datos que se encuentran frecuentemente en la computación científica, se basa en el criterio de descomponer un arreglo de varias dimensiones en varios subarreglos y de cómo son distribuidos entre los diferentes threads o procesos.

## Mecanismos de implementación

Los mecanismos de implementación más que patrones proveen una descripción a alto nivel de los mecanismos en los diferentes ambientes y entornos de programación tales como OpenMP y MPI.

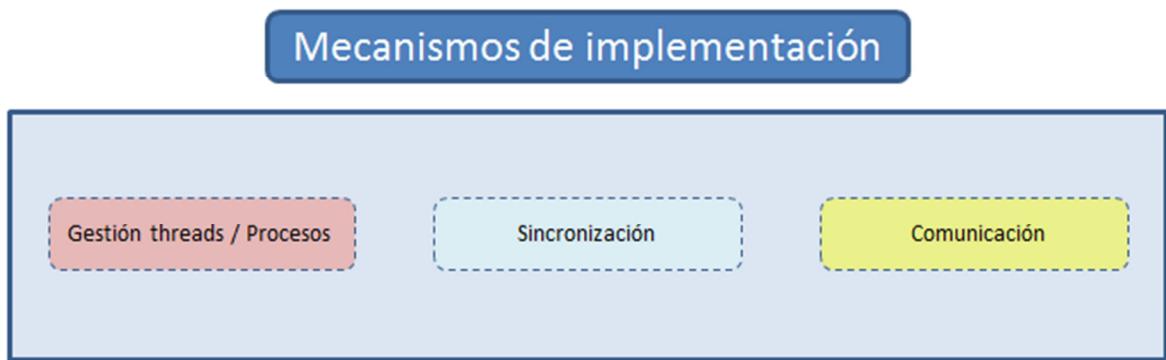


Fig. 16. Espacio de diseño: Mecanismos de implementación.

### Gestión threads / Procesos

En este punto se describe el control de cómo se crean, terminan y gestionan los diferentes procesos y threads usados en la computación paralela.

- **Sincronización:** Se establecen las restricciones y el orden de los eventos que ocurren dentro de los diferentes threads o procesos con el fin de controlar el acceso a los recursos compartidos.
- **Comunicación:** Se establecen los mecanismos de comunicación entre los diferentes threads y procesos teniendo en cuenta los diferentes atributos de calidad en el programa.

Para el sistema CAT se aplicó el uso de los diferentes espacios de diseño con el fin de tener una aplicación que respondiera adecuadamente a las necesidades.

## ***Atributos de calidad del sistema - Desempeño***

El desempeño y escalabilidad son atributos de calidad para un sistema en general. Para el caso de la computación estos pueden ser vistos como la habilidad del sistema de funcionar dentro de sus parámetros de desempeño y tener la capacidad de manejar incrementos en el volumen de procesamiento de la información [30].

El desempeño también puede ser visto como el grado en el cual un sistema o componente cumple sus funciones dentro de las restricciones dadas tales como velocidad, exactitud, uso de memoria, tiempo requerido para responder a un evento específico y el número de eventos procesados en un intervalo dado de tiempo [31].

Para la implementación del sistema CAT se espera que el clúster HPC tenga la capacidad de responder a los diferentes tipos de análisis y procesamientos en tiempo casi real, con el fin de presentar información de interés para los grupos de investigación de otros departamentos involucrados y que permitan realizar una evaluación de los diferentes componentes desarrollados.

## ***Ley de Amdahl***

Las principales razones para la implementación de un programa en paralelo están dadas por la necesidad de obtener mejor desempeño y/o resolver grandes problemas. El desempeño de un programa puede ser modelado y estimado, para esto se realiza un conjunto de análisis basados en el tiempo de ejecución y el conjunto de tareas que se pueden parallelizar [52]. Por ejemplo, la composición de un programa se puede dividir en tres secciones: Inicio o configuración, una sección de computación o análisis lógico y una sección de finalización o entrega de resultados, por tanto, si se ejecuta este programa en

un solo proceso se puede inferir que el tiempo de ejecución total estará compuesto por la suma de los tiempos de las secciones que lo componen [53]:

$$T_{total}(1) = T_{inicio} + T_{computación} + T_{finalización}$$

Fig. 17. Tiempo de ejecución total de un programa [53].

Ahora, se puede suponer que la sección de inicio y configuración no se pueden parallelizar, mientras que la sección de computo sí puede ser parallelizable y segmentada en  $P$  partes para sean ejecutadas en un mismo número de tareas independientes de forma concurrente en  $P$  procesos o threads (uno por cada parte segmentada), por ende el tiempo total de ejecución disminuye de la siguiente forma:

$$T_{total}(P) = T_{inicio} + \frac{T_{computación}(1)}{P} + T_{finalización}$$

Fig. 18. Tiempo de ejecución total de un programa paralelizable [53].

Una importante medida de ayuda es la velocidad relativa, la cual indica el factor de rapidez con la que se ejecuta el mismo programa de forma Paralela vs. Serial:

$$S(P) = \frac{T_{total}(1)}{T_{total}(P)}$$

Fig. 19. Factor de rapidez programa serial vs. Programa paralelo [53].

De igual forma, Se puede medir la eficiencia  $E$ , basándose en la rapidez  $S$  y el número de  $P$  partes o procesos:

$$E(P) = \frac{S(P)}{P}$$

Fig. 20. Eficiencia de un programa paralelo [53].

Reescribiendo la fórmula de la eficiencia en términos del tiempo, queda de la siguiente fórmula:

$$E(P) = \frac{T_{total}(1)}{P \times T_{total}(P)}$$

Fig. 21. Eficiencia de un programa paralelo en términos de tiempo ejecución serial vs. Tiempo ejecución paralelo [53].

En el caso ideal se quiere que la rapidez  $S$  sea igual a  $P$  (número de segmentos, threads o procesos en los que se ha paralelizado la sección de cómputo), esto llevaría a una eficiencia de 1, sin embargo en muy raras ocasiones esto es alcanzado debido a que los tiempos en las secciones de inicio y finalización no mejorarán adicionando más procesos o threads en estas, por tal razón, los tiempos de ejecución de estas secciones son una fracción del tiempo total, dicha fracción se llama fracción serial y es denotada por  $\gamma$  [54]:

$$\gamma = \frac{T_{inicio} + T_{finalización}}{T_{total}(1)}$$

Fig. 22. Fracción serial de un programa [53].

Por ende la fracción del tiempo empleado en la sección paralelizable es  $(1 - \gamma)$ . Así se puede reescribir la expresión para el tiempo total de computación como:

$$T_{total}(P) = \gamma \times T_{total}(1) + \frac{(1 - \gamma) \times T_{total}(1)}{P}$$

Fig. 23. Tiempo de ejecución programa paralelo en términos de fracción serial [53].

Y reescribiendo el factor de rapidez  $S$  en términos de esta nueva ecuación se obtiene la ecuación de la Ley de Amdahl [51] en términos de la fracción serial y de la cantidad de  $P$  (número de segmentos, threads o procesos en los que se ha paralelizado la sección de cómputo):

$$S(P) = \frac{T_{total}(1)}{\left(\gamma + \frac{1-\gamma}{P}\right) \times T_{total}(1)}$$

Fig. 24. Rapidez en términos de fracción serial y tiempo de ejecución serial [53].

Para finalmente obtener:

$$S(P) = \frac{1}{\left(\gamma + \frac{1-\gamma}{P}\right)}$$

Fig. 25. Rapidez de un programa en términos de fracción serial y cantidad de núcleos [53].

Un ejemplo del factor de rapidez según la cantidad de núcleos disponibles y según la porción de código que se puede paralelizar es el siguiente:

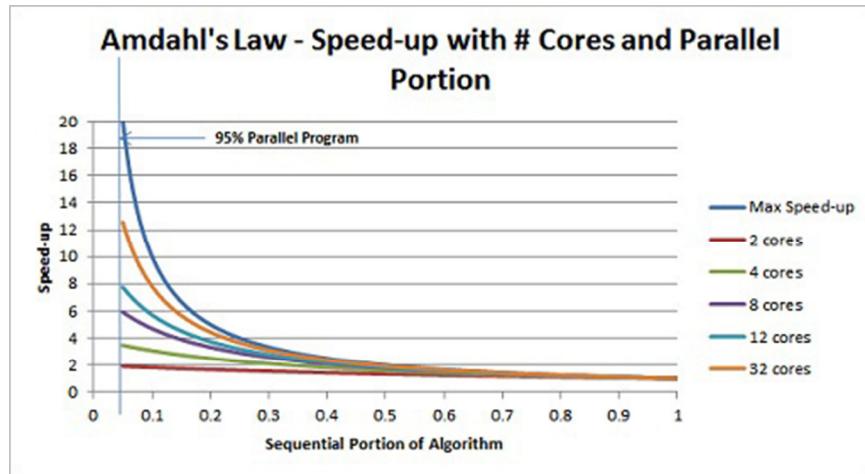


Fig. 26. Rapidez de un programa en términos de porcentaje serial del algoritmo [56].

Para determinar que realmente se dé una ganancia en desempeño al paralelizar las diferentes tareas de cómputo en el problema a resolver se hará un seccionamiento del programa CAT para determinar qué tipo de tareas son susceptibles de paralelizar y/o distribuir.

## **Procesamiento de filtros digitales**

En su definición más general, un filtro digital es un proceso computacional o algoritmo mediante el cual una señal digital de entrada (secuencia de muestras) es transformada en una señal digital de salida [57]. Los filtros tienen dos usos principales: ya sea para una separación o restauración de una señal. El caso de separación de una señal se presenta cuando esta ha sido contaminada con alguna interferencia, ruido u otra señal, en el caso de la restauración de una señal se da cuando esta ha sido distorsionada de alguna forma [58].



Fig. 27. Representación general de un filtro.

Los filtros digitales son muy importantes en el análisis de procesamiento de señales (DSP). El análisis de procesamiento de señales digitales (DSP) puede ser considerado como la captura, análisis y manipulación de una señal análoga por un computador o dispositivo digital. En el desarrollo de la teoría de filtros, los filtros digitales han tenido un gran impacto en el procesamiento de las señales digitales dado que se pueden emplear para realizar los mismos análisis de los filtros análogos pero con mayor desempeño y posibilidad de implementar una vasta cantidad de algoritmos para procesar la señal [59].

Dentro de las distintas clasificaciones de los filtros digitales, tenemos los filtros lineales y los no lineales. En los filtros lineales, existe una relación lineal entre los datos de entrada y los datos de salida:

$$\begin{aligned}
x_1 &\Rightarrow y_1 \\
x_2 &\Rightarrow y_2 \\
\text{Entonces } \alpha x_1 + \beta x_2 &\Rightarrow \alpha y_1 + \beta y_2
\end{aligned}$$

Fig. 28. Representación de un filtro digital lineal[60].

En el caso de los filtros no lineales, existe una relación no lineal entre la entrada y la salida del filtro:

$$\begin{aligned}
x_1 &\Rightarrow y_1 = x_1^2 \\
x_2 &\Rightarrow y_2 = x_2^2 \\
\text{Entonces } x_1 + x_2 &\Rightarrow (x_1 + x_2)^2
\end{aligned}$$

Fig. 29. Representación de un filtro digital no lineal[60].

Por otra parte, existen filtros recursivos y no recursivos. Los filtros no recursivos (filtro FIR – Finite Impulse Response) son aquellos cuya salida está calculada basándose exclusivamente con los valores de entrada:

$$y[n] = a_0x[n] + a_1x[n - 1] + \dots + a_{M-1}x[n - (M - 1)]$$

Fig. 30. Representación de un filtro FIR[61].

Mientras que los filtros recursivos (Filtro IIR – Infinite Impulse Response), son aquellos que además de utilizar los valores de entrada utilizan también los valores de salida previos:

$$y[n] = a_0x[n] + a_1x[n - 1] + \dots + a_{M-1}x[n - (M - 1)] - b_1y[n - 1] - b_2y[n - 2] - \dots - b_Ny[n - N]$$

Fig. 31. Representación de un filtro IIR[61].

Adicionalmente, dentro de los tipos de filtros, tenemos otro tipo de clasificación:

Los filtros paso bajo – LP: Los filtros LP dejan pasar las frecuencias que están por debajo de una determinada frecuencia de corte [62]. Un filtro paso bajo permite que las

'bajas frecuencias' pasen el filtro, este tipo de filtros tienen una frecuencia límite de corte que define el valor máximo que puede tener la frecuencia de una señal para pasarlo. Por ejemplo, si una señal posee dos tipos de frecuencia: 10 Hz y 20Hz y a esta se le aplica un filtro de paso bajo con una frecuencia de corte de 15Hz, la señal resultante solo estará compuesta por frecuencias de 10Hz ya que la frecuencia de 20Hz ha sido rechazada por el filtro [63].

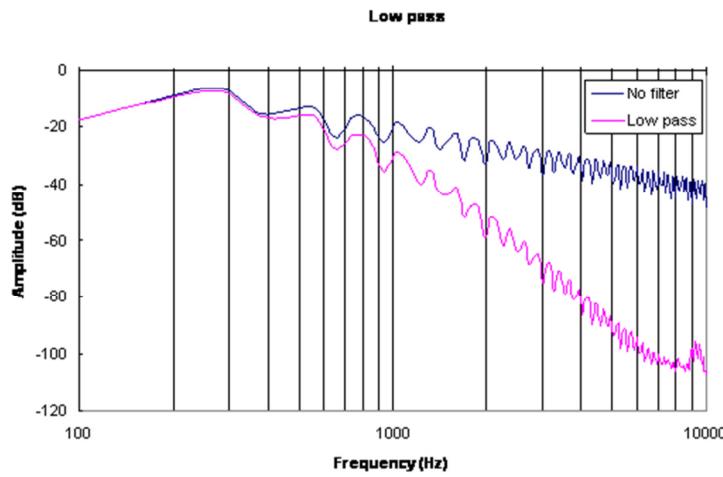


Fig. 32. Aplicación de un filtro LP a una señal con una frecuencia de corte de 400MHZ[64].

En este grafico se aprecia la aplicación de un filtro paso bajo sobre una señal, todas las frecuencias de más de 400 Hz son atenuadas.

Los filtros paso alto (HP) dejan pasar las frecuencias de una señal que están por encima de una determinada frecuencia de corte [62]. Estos tipos de filtros permiten que las frecuencias más altas que una frecuencia de corte pasen de forma intacta, mientras que las frecuencias más bajas son atenuadas por el filtro [63]

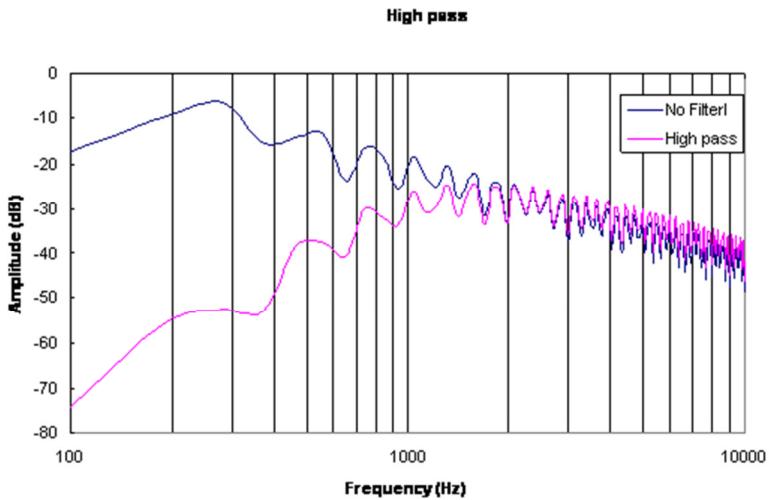


Fig. 33. Aplicación de un filtro HP a una señal con una frecuencia de corte de 1000MHZ[64].

En el anterior gráfico se observa la misma señal y se le aplica un filtro HP donde las frecuencias menores a 1000Hz son atenuadas, y las frecuencias más altas permanecen intactas.

Para efectos del análisis del problema se realizará la implementación de un filtro LP – FIR que permita remover los posibles ‘ruidos’ generados causados en las telemetrías.

### **Integración numérica**

Para revisar el tema de la integración numérica se tomará el contexto de poder determinar el desplazamiento de un vehículo aeroespacial con base en la telemetría obtenida de la aceleración.

En primer lugar tenemos que la aceleración es la tasa de cambio en la velocidad de un objeto en un tiempo dado, al mismo tiempo, la velocidad es la tasa de cambio de la posición del mismo objeto, en términos matemáticos, la velocidad es la derivada de la posición y a su vez la aceleración es la derivada de la velocidad [65]:

$$\vec{a} = \frac{d\vec{v}}{dt} \therefore \vec{v} = \frac{d\vec{s}}{dt} \therefore \vec{a} = \frac{d(d\vec{s})}{dt^2}$$

Fig. 34. Representación de la aceleración como la doble derivada de la posición[65].

Para el caso en el cual se desee obtener la posición del vehículo a partir de la aceleración se tendrá que aplicar la antiderivada o integración al conjunto de datos, con esto al aplicar una doble integración sobre la aceleración se podrá obtener la posición:

$$\vec{v} = \int \vec{a} dt$$

$$\text{y la posición } \vec{s} = \int \vec{v} dt \therefore \int \left( \int \vec{a} dt \right) dt$$

Fig. 35. Representación de la posición como la doble integral de la aceleración[65].

Una mejor forma de entender esto, es definir la integral como el área bajo la curva, donde la integral es la suma de pequeñas áreas con un ancho cercano a cero. Aplicando esto al fenómeno físico se entiende que la suma de la integración representa la magnitud de una variable física [65]:

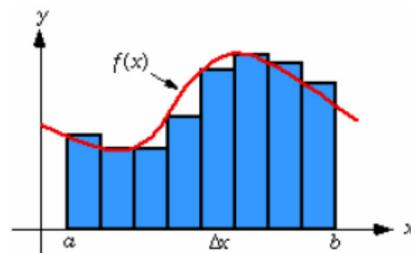


Fig. 36. Ejemplo de una señal de aceleración[65].

Con el concepto de área bajo la curva se puede realizar la siguiente deducción: La señal que se esté recibiendo nos brinda de forma instantánea el valor de su magnitud, entonces pequeñas áreas pueden ser creadas entre dos señales de muestra. Para realizar la aplicación del método de integración se tomará el tiempo de muestreo como la base del

área mientras que los valores de muestra representan la altura. Sin embargo, en una situación real al realizar los cálculos del área en esos segmentos se obtendrá un error:

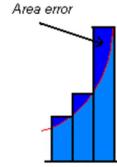


Fig. 37. Error al realizar una integración numérica[65].

Este error se irá acumulando en el tiempo durante el proceso de integración, que comúnmente se llama perdida de muestreo. Para reducir este error, se hará la siguiente deducción: El área resultante se verá como la suma de dos pequeñas áreas: el valor del área cuadrada más el valor del área del triángulo:

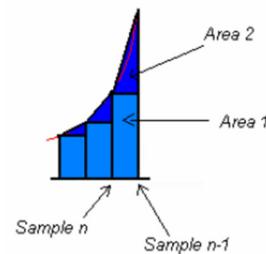


Fig. 38. Método de integración numérica trapezoidal para reducir el error en el proceso de integración[65].

A este método se le llama el método de integración trapezoidal, de este modo se podrá realizar una aproximación de primer orden de la señal. Como resultado de esto, ahora se podrá realizar una integración de la aceleración, para así obtener la velocidad y por ende la posición del vehículo aeroespacial:

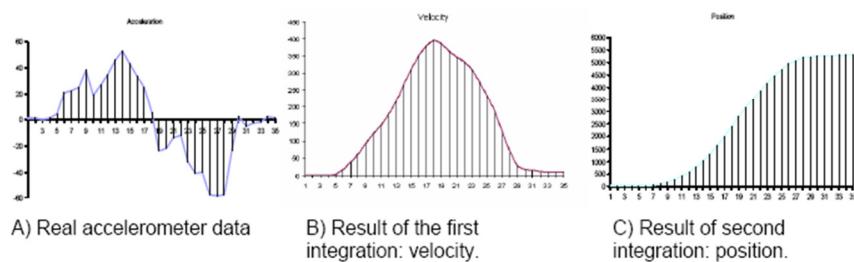


Fig. 39. Resultado del proceso de integración de una aceleración para obtener la velocidad y la posición[65].

Para el caso de estudio, se realizará primero un filtro de la señal para eliminar las singularidades o ruidos que eviten realizar la integración numérica, posteriormente, se realizará la integración de la aceleración en los ejes: X, Y, Z, para obtener la velocidad en el punto dado, y se aplicará una segunda integración numérica para obtener el desplazamiento del vehículo aeroespacial en un instante de tiempo.

## CAPITULO III: GENERALIDADES DEL PROYECTO

### ***Problemática***

En la actualidad el proyecto PUA ha avanzado de forma sustancial en el centro de comando y control C3, específicamente en la capacidad de adquirir, procesar y centralizar la información resultante de las telemetrías ubicadas en los vehículos de lanzamiento utilizados en las diferentes misiones del proyecto, sin embargo, el poder recolectar y recibir la información en el servidor C3 no es suficiente ya que en cualquier tipo de misión aeroespacial un requerimiento esencial es la capacidad de poder realizar análisis en tiempo real sobre la información recolectada de las telemetrías y así realizar la toma de decisiones en tiempo real con base a la información analizada, v.g.: trayectoria actual, trayectoria esperada, presión en la cámara de combustión, altura entre otros.

El problema principal que tiene el sistema C3 es que no posee un componente de alto desempeño para realizar el análisis de las telemetrías y procesamiento de los resultados. Dado el volumen de información entregado por los sensores (más de 3000 tramas de información por segundo) y la cantidad y complejidad de análisis a ser realizados: filtro pasa bajo, suavizado e integración (aplica para el caso de los sensores de aceleración) un equipo de cómputo estándar o PC no está en la capacidad de realizar el análisis de la información en tiempo real, por tanto, se hace necesario la implementación del componente de análisis en un sistema computación de alto desempeño clúster HPC.

Dado que existe una limitante tecnológica en los equipos de cómputo estándar para realizar el análisis de la información requerida en tiempo real para el proyecto PUA en el sistema C3, la principal pregunta que se desea resolver en esta tesis es:

¿Cómo puede un sistema de altas prestaciones computacionales ayudar a solucionar el problema de desempeño y análisis de telemetrías en tiempo real en el sistema C3 del proyecto PUA?

### ***Objetivo general***

Especificar, diseñar e implementar el componente de procesamiento y análisis de telemetrías – CAT, para el sistema C3 del proyecto PUA utilizando el clúster de alto desempeño de la facultad de ingeniería.

### ***Objetivos específicos***

- Integrar el paradigma de programación en paralelo, complementado con el uso de patrones de diseño en programación en paralelo para el CAT.
- Realizar el diseño y la implementación del sistema CAT utilizando: MPI y threads en el clúster de alto desempeño.
- Realizar experimentos de validación sobre las telemetrías para garantizar la funcionalidad del sistema CAT de forma conjunta con el sistema C3 en general.
- Validar la necesidad de implementar el componente de análisis de telemetrías CAT en el clúster de alto desempeño de la facultad de ingeniería por medio de la ley de Amdahl.

### ***Alcance***

- Se realizará la implementación de la solución basándose en la implementación de MPI para el lenguaje de programación C.
- Los análisis y algoritmos implementados serán los de aplicación de filtro paso bajo e integración numérica para el caso de las aceleraciones.

- Realizar el diseño y la implementación del sistema CAT utilizando: MPI y threads en el clúster de alto desempeño.
- Se ejecutará la validación de la solución con 2 tipos de experimentos utilizando sensores de aceleración.

## ***Contribuciones***

La principal contribución al proyecto PUA es el de entregar un componente de análisis de telemetrías en tiempo real para el sistema C3, con la capacidad de incrementar el poder de computo, adicionalmente, realizar una evaluación completa del sistema C3 para determinar qué puntos se deben mejorar, validar el funcionamiento del sistema con un algoritmo de filtro LP – FIR e integración numérica, finalmente, y la posibilidad de incrementar de adicionar nuevos análisis y tipos de filtros que puedan ser requeridos a futuro en el proyecto PUA.

## **CAPITULO IV: ESTRATÉGIA DE SOLUCIÓN**

En este capítulo se define la estrategia de solución utilizada para el diseño e implementación del sistema CAT. Esta estrategia está basada en conceptos de análisis y diseño de software y aplicación de patrones de diseño con énfasis en programación paralela teniendo en cuenta el tipo de tareas a realizar, la lógica que se desea parallelizar o distribuir así como el mecanismo de comunicación en el sistema para garantizar las necesidades planteadas, adicionalmente, por último, se realiza una descripción de los análisis implementados a nivel de: tipo de filtro de paso bajo usado y del tipo de integración numérica utilizada para el caso de las aceleraciones.

### ***Aspectos críticos y estratégicos para las decisiones de diseño***

Para realizar un diseño adecuado del sistema CAT primero se debe realizar un análisis con el fin de identificar los aspectos críticos de este y que tienen impacto en la etapa de diseño. A continuación se presentan los aspectos críticos identificados:

#### **Concurrencia:**

El sistema CAT debe estar en capacidad de realizar los diferentes tipos de análisis requeridos por cada sensor de telemetría de forma concurrente y distribuida en el clúster HPC.

#### **Desempeño:**

El sistema debe poder procesar las tramas y realizar los análisis indicados en el menor tiempo posible, en tiempo real.

**Extensible:**

El sistema CAT debe estar en capacidad de extender su funcionalidad a nuevos tipos de análisis v.g.: cálculo de altura basándose en la ecuación de atmósfera estándar, con un mínimo impacto en el código ya existente.

**Escalabilidad:**

El sistema CAT debe poder escalar a nuevos nodos de cómputo y aumentar la concurrencia con mínimo impacto en el código ya existente.

***Reglas indicadas por el proyecto PUA***

Para realizar una integración exitosa de este sistema CAT con el sistema C3 se establecieron un conjunto de reglas que deben ser cumplidas por el sistema, estas reglas son:

- Se requiere como mínimo la implementación de un filtro de paso bajo LP para todos los datos obtenidos.
- Para la información filtrada se requiere realizar una suavización a través de la fórmula de los mínimos cuadrados.
- El cálculo de la velocidad estará basado en la integral de la aceleración, mientras que el cálculo de la altura estará basado en la integral acumulada de la velocidad.
- Durante la ejecución del programa toda la información de los análisis debe estar almacenada temporalmente en memoria que a futuro se puedan realizar análisis que requieran toda la información v.g. Análisis de Fourier.

***Restricciones del proyecto PUA y tecnológicas***

- El sistema estará basado en la implementación MPI de OpenMP 1.6.5 y utilizará threads nativos POSIX.

- El lenguaje de programación usado será C, dado que el clúster posee ya las librerías MPI compiladas para este lenguaje.
- Los datos resultantes de los análisis deben entregarse al servidor C3 con el mismo valor de tiempo recibido en la trama de información original.
- La paralelización de la solución estará basada en tecnología CPU presente en el clúster de la facultad de ingeniería de la Universidad de los Andes.
- Las conexiones entrantes desde el servidor C3 se recibirán a través del puerto UDP 5000.
- Los datos resultantes se entregarán al servidor C3 a través del puerto UDP 6000.
- El proceso principal del componente CAT se deberá ejecutar siempre en el nodo principal del clúster.

## **Requerimientos**

- Acceso a clúster de alto de alto desempeño de la facultad de ingeniería.
- Puertos de conexión en el servidor C3 y clúster disponibles.
- Red Infiniband habilitada en el clúster.
- Datos de validación y experimentación disponibles y cargados en el sistema C3.
- Manejo de información entrante desde el servidor C3: La información es recibida a través de tramas o cadenas de caracteres de la siguiente forma:

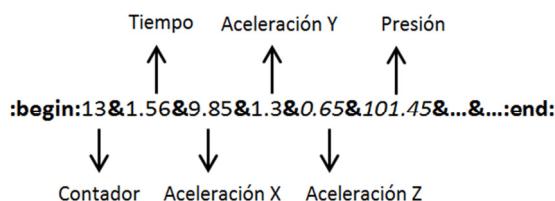


Fig. 40. Tramas de información recibida sin procesar.

- Manejo de información analizada hacia el servidor C3: La información es enviada a través de tramas o cadenas de caracteres de la siguiente forma:

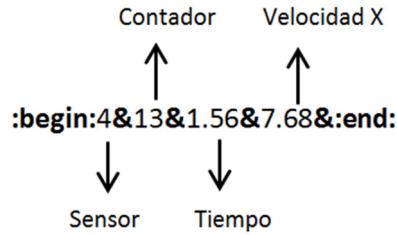


Fig. 41. Tramas de información procesadas y enviadas a servidor C3.

### **Diagrama de contexto**

En este diagrama se realiza una contextualización de cómo el sistema CAT se incorpora en el sistema C3. Básicamente, el componente CAT se encuentra desplegado en el clúster HPC y este recibe toda la información de las telemetrías a través de un socket UDP por el puerto 5000, de igual forma, retorna la información analizada y procesada al servidor C3 por un socket UDP por el puerto 6000.

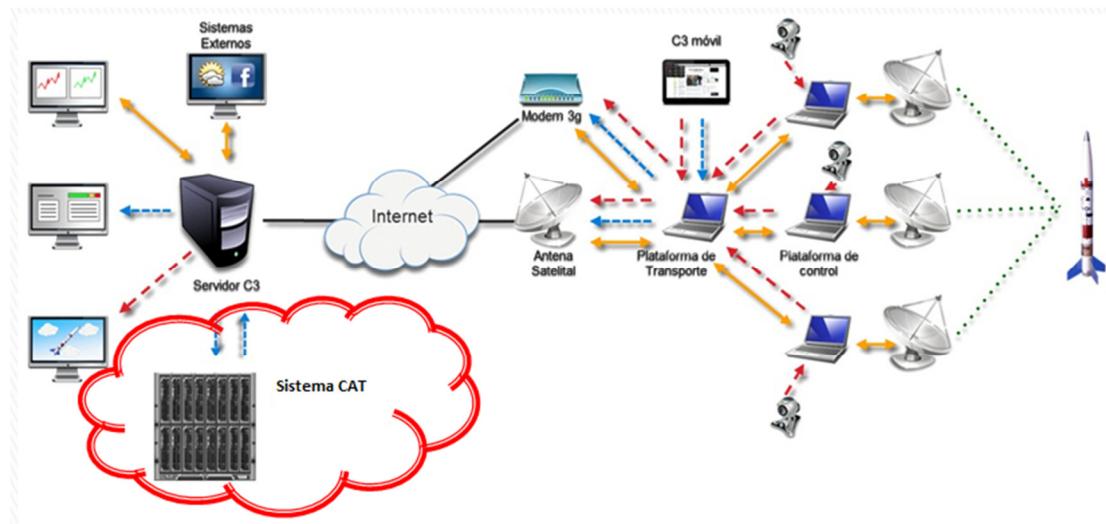


Fig. 42. Diagrama de contexto sistema C3. Tomado de [2].

## **Atributos de calidad**

Dadas las reglas del negocio, restricciones, factores críticos de diseño y teniendo un contexto del sistema, los principales atributos de calidad identificados en el sistema CAT son:

- Desempeño: El sistema CAT debe recibir, procesar y entregar los resultados de análisis de cada trama de datos al servidor C3 en tiempo casi real.
- Concurrencia: El sistema CAT tendrá la capacidad para analizar de forma simultánea la información de los siguientes sensores: aceleración X, aceleración Y, aceleración Z, Presión, Temperatura y Fuerza, adicionalmente, para cada uno de estos sensores se realizarán los análisis de: filtro paso bajo, suavizado y para el caso de las aceleraciones el cálculo de velocidad y altura.
- Extensibilidad: El sistema CAT debe poder adicionar nuevos tipos de análisis v.g.: serie de Fourier, filtros adaptativos, cálculos de: trayectoria actual, estimación de trayectoria esperada, coeficiente de arrastre entre otros con solo adicionar o reemplazar alguna función ya existente.
- Escalabilidad: El sistema CAT debe poder aumentar la capacidad de procesamiento adicionando nuevos nodos reiniciando el proceso principal y con ajustes mínimos al código existente.

## **Diseño del sistema a través del uso patrones de diseño para software paralelo**

Los requerimientos, restricciones y atributos de calidad son las características que rigen el diseño de un sistema, una vez identificados es posible realizar un diseño del

sistema CAT que cumpla las diferentes características expuestas, para esto, se implementaron los patrones de diseño para software paralelo en sus cuatro espacios.

## Encontrar la concurrencia

Para encontrar la concurrencia en el sistema CAT se tuvo que realizar un buen entendimiento del problema a ser resuelto dadas las características necesarias para la solución, además de eso, se identificaron las partes del programa que son más intensas en cómputo, las estructuras de datos claves en la solución y el manejo de estas en cada tarea.

### Patrones de descomposición:

Se encontró que dependiendo del tipo de telemetría, se realizarán diferentes tipos de análisis, por tanto la descomposición más adecuada de tareas es la siguiente:

Tarea	Descripción
T1	Recibir tramas de información desde el servidor C3.
T2	Enviar tramas de información a tareas de análisis específico.
T3	Recibir la trama de información.
T4	Aplicar análisis de filtro sobre los datos.
T5	Realizar análisis de integración instantánea.
T6	Realizar análisis de integración acumulada.
T7	Enviar los datos procesados al servidor C3.

Fig. 43. Tareas identificadas en el sistema CAT.

A continuación se describe cada tarea:

T1. Recibir tramas de información desde el servidor C3: Se debe contar con una tarea especializada que sea capaz de recibir un gran volumen de tramas de información y almacenarlas temporalmente en una estructura de datos compartida para que una tarea que se encargue de transmitir esta información a las tareas de análisis la pueda acceder.

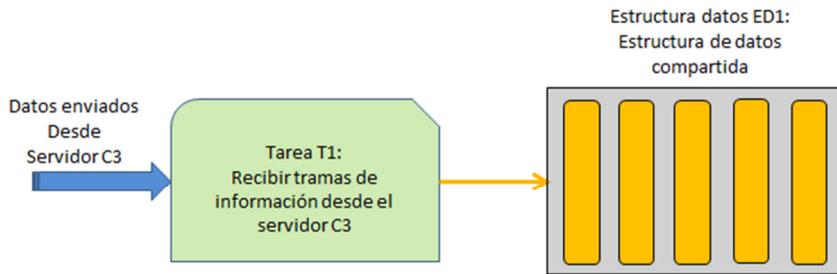


Fig. 44. Detalle tarea T1.

T2. Enviar tramas de información a tareas de análisis específico: Se debe contar con un conjunto de tareas que sean capaces de extraer la trama de información almacenada en la estructura compartida y enviarla a las diferentes tareas que realizan los análisis específicos.

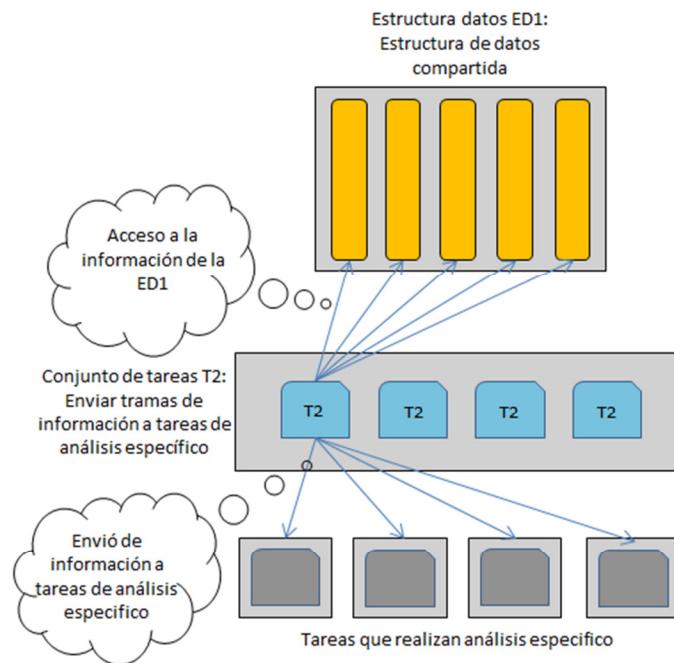


Fig. 45. Detalle tarea T2.

T3. Recibir la trama de información: De la trama de información que se recibe de T2 se debe seleccionar el dato correspondiente al tipo de análisis y posteriormente se debe

guardar de forma permanente en una estructura de datos compartida para realizar el análisis específico.

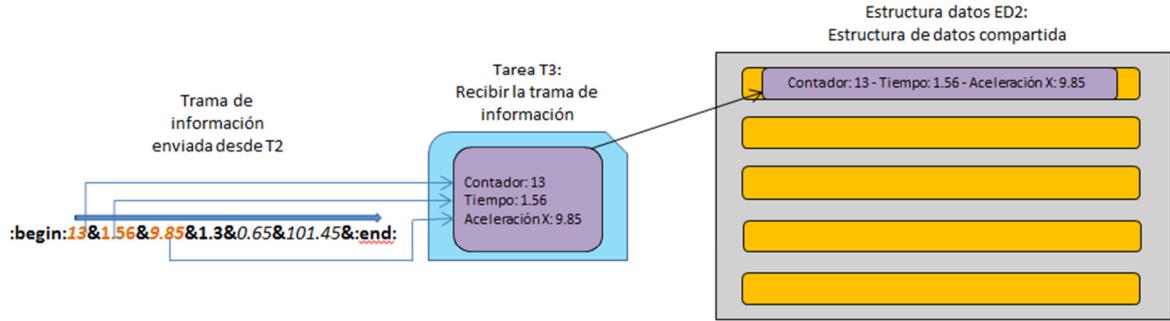


Fig. 46. Detalle tarea T3.

T4. Aplicar análisis de filtro sobre los datos: Para todas las telemetrías recibidas se debe aplicar un filtro LP - FIR que permita eliminar y suavizar los datos que hayan sido afectados por algún ruido o perturbación. Una vez realizado el cálculo los datos resultantes serán almacenados en la estructura de datos compartida.

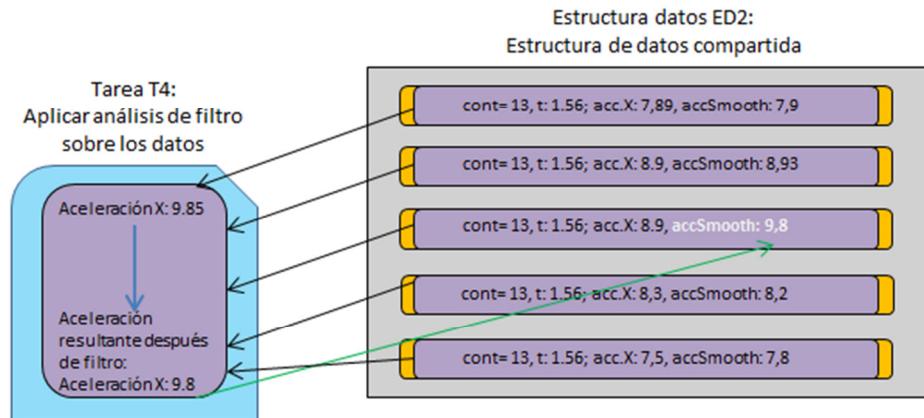


Fig. 47. Detalle tarea T4.

T5. Realizar análisis de integración instantánea: Para el caso de las telemetrías correspondientes a la aceleración en sus diferentes ejes, se requiere realizar un proceso de integración instantánea por el método trapezoidal para calcular la velocidad instantánea

del vehículo aeroespacial, una vez realizado el cálculo los datos resultantes serán almacenados en la estructura de datos compartida.

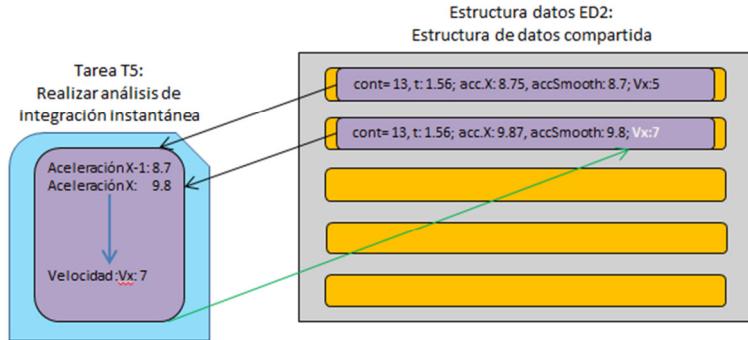


Fig. 48. Detalle tarea T5.

T6. Realizar análisis de integración acumulada: Para el caso de las telemetrías correspondientes a la aceleración en sus diferentes grados de libertad, se requiere realizar un proceso de integración acumulada basándose en velocidad obtenida también usando el método trapezoidal para calcular la posición del vehículo aeroespacial en un momento dado, una vez realizado el cálculo los datos resultantes serán almacenados en la estructura de datos compartida.

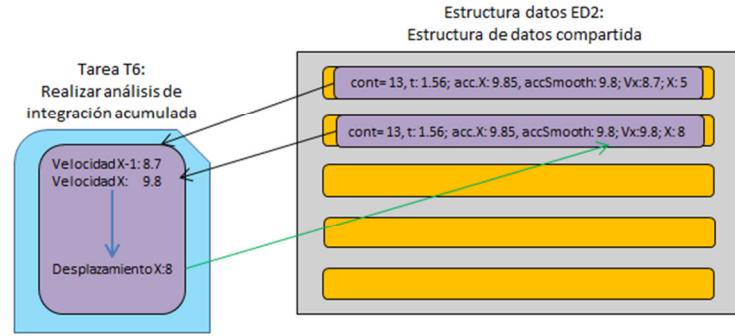


Fig. 49. Detalle tarea T6.

T7. Enviar los datos procesados al servidor C3: Una vez se hayan realizado los diferentes análisis sobre los datos, estos serán enviados en una trama de respuesta con el formato indicado al servidor C3 para su despliegue en el tablero de control.

Por otra parte, las estructuras de datos identificadas son:

Estructura datos	Descripción
ED1	Un buffer de almacenamiento para almacenar temporalmente las tramas recibidas desde el servidor C3.
ED2	Una estructura de datos compartida para almacenar los datos originales y procesados correspondientes a cada tipo de análisis.

Fig. 50. Estructuras de datos identificadas.

Agrupamientos de tareas identificadas:

Grupo1: Agrupamiento de tareas T1 y T2. Las tareas agrupadas acá se encargan de recibir y distribuir las tramas de información hacia las diferentes tareas que realizan en análisis.

Grupo2: En este agrupamiento se encuentra la tarea T3. Esta tarea se encarga de recibir y almacenar en una estructura de datos compartida para los otros grupos de tareas asociados al tipo de telemetría realicen los análisis respectivos.

Grupo3: Telemetría aceleración, agrupamiento de tareas T4, T5, T6 y T7. Las tareas de este grupo ejecutan los análisis de filtro y suavización incluyendo los procesos de integración numérica para obtener la velocidad instantánea y desplazamiento, y finalmente enviar los resultados al servidor C3.

Grupo4: Telemetría presión, agrupamiento de tareas T4 y T7. Las tareas de este grupo ejecutan los análisis de filtro y suavización para luego enviar los resultados al servidor C3.

Grupo5: Telemetría temperatura agrupamiento de tareas T4 y T7. Las tareas de este grupo ejecutan los análisis de filtro y suavización para luego enviar los resultados al servidor C3.

Orden de ejecución de las tareas en cada grupo

Grupo1: Las tareas T1, T2 se ejecutan de forma concurrente de forma independiente.

Grupo2: La tarea T3 se ejecuta de forma independiente.

Grupo3: Para las tareas T4, T5, T6 el orden de ejecución es: T4 ->T7 -> T5 -> T7 -> T6 ->T7.

Grupo4: Para las tareas T4, T7 el orden de ejecución es: T4 ->T7.

Grupo5: Para las tareas T4, T7 el orden de ejecución es: T4 ->T7.

Una representación general de agrupamiento del sistema por tareas es:

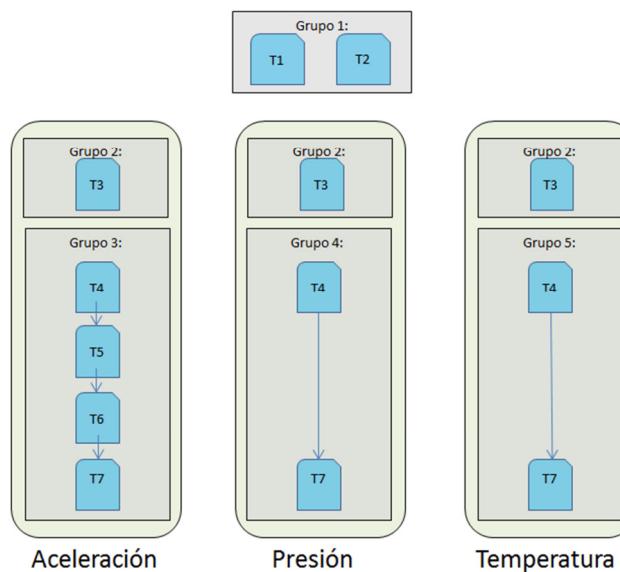


Fig. 51. Agrupamiento del sistema CAT por tareas según el tipo de telemetría.

## Estructura del algoritmo

Una vez encontrados los elementos de diseño en la solución planteada y además de identificar que el patrón por descomposición por tareas es el predominante, se debe refinar más el diseño por medio del espacio de estructura del algoritmo para determinar qué tipo de patrón es más adecuado frente al problema presentado, teniendo en cuenta que las tareas definidas tienen un orden de ejecución establecido y no se presenta ninguna

recursividad, el patrón de paralelización de tareas es el patrón que predomina en este espacio.

En este patrón intervienen tres elementos que se tuvieron en cuenta: las tareas, dependencias entre ellas y la asignación de estas en los diferentes threads o procesos.

- Tareas: Las tareas se encuentran agrupadas con la posibilidad de que cada grupo de tareas se pueda ejecutar en un procesador del nodo y a su vez, cada tarea se pueda ejecutar en un núcleo para un procesamiento concurrente.
- Dependencias: Las dependencias entre tareas se encuentran por medio del orden de ejecución establecido, v.g. no se puede ejecutar el cálculo de la posición del vehículo sin haber calculado la velocidad instantánea en primer lugar.

Las dependencias a nivel de datos compartidos aparecen entre las diferentes tareas de un mismo grupo, ya que estas deben leer y escribir en la misma estructura de datos, por tanto el manejo de la concurrencia es un punto a tener en cuenta.

- El principal aspecto que se tuvo en la asignación de las tareas en los diferentes procesos y threads fue la de mantener un buen balance en los recursos, se desean evitar casos de crear múltiples procesos o threads que realizan pocas o ninguna operación de computo, o por el contrario tener pocos recursos para ejecutar todas las tareas requeridas. Finalmente, otro punto a tener en cuenta es el de mantener una baja latencia de comunicación entre las tareas de un mismo grupo por tanto se debe evitar distribuir un grupo de tareas en diferentes nodos.

El resultado de la aplicación de este patrón es un algoritmo más estructurado que puede ser ejecutado de forma concurrente realizando un mapeo de las tareas entre los diferentes nodos y procesadores del clúster.

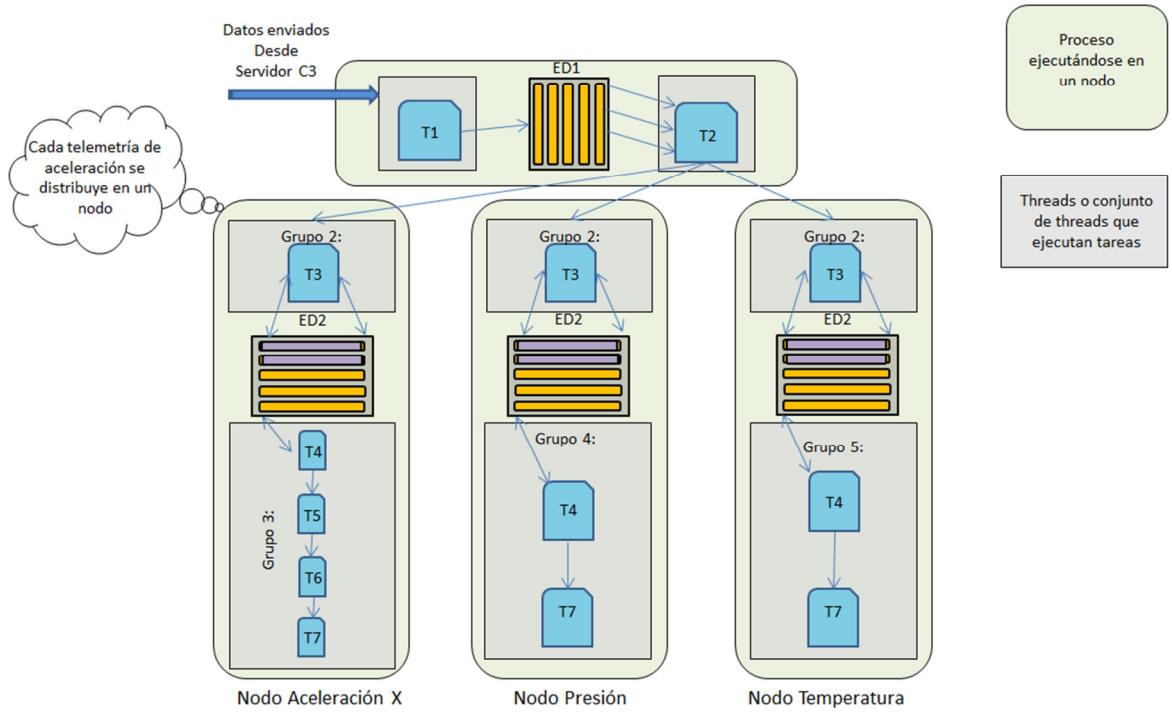


Fig. 52. Estructura general del algoritmo distribuido en procesos y threads

## Estructuras de soporte

Hasta el momento ya se tiene un diseño a alto nivel del algoritmo paralelo que responde al problema establecido, en esta etapa se pasó del diseño a alto nivel a un diseño más detallado donde se especifican las estructuras a nivel de programación que soportan el algoritmo de solución planteado. El diseño se hizo tomando como base los dos grupos de patrones presentes en este espacio.

A nivel de estructuras de programa se detectó que dos tipos de patrones son necesarios para la solución, el patrón SPMD y Master/Worker son necesarios.

El patrón SPMD es necesario para el grupo de tareas Grupo1, ya que son rutinas que se ejecutan únicamente en un proceso principal, se encargan de recibir y distribuir las tramas de información hacia los diferentes procesos que ejecutan los análisis por cada tipo de telemetrías. También se evidencia el uso de este patrón en el grupo de tareas

Grupo2 ya que es una rutina que se ejecuta en todos los nodos que realizan los análisis por cada tipo de telemetría.

El patrón Master/Worker es necesario aplicarlo para los grupos de tareas Grupo3, Grupo4 y Grupo5, ya que estos recibirán un conjunto de tareas para que sean ejecutadas por diferentes workers de forma concurrente.

En cuanto al grupo de patrones de estructuras de datos, también es necesario implementar dos tipos de patrones:

El patrón de colas compartidas es necesario aplicarlo al grupo de tareas Grupo1 ya que la cola a implementar actúa como un almacenamiento temporal de los datos recibidos por la tarea T1 y a su vez, es usada por el conjunto de tareas T2 para extraer la información y entregarla de forma asíncrona a los diferentes grupos de tareas para que realicen los análisis y procesamiento de los datos.

El patrón de datos compartidos también es importante ya que todos los datos de los análisis realizados por cada tipo de telemetría deben ser almacenados y accedidos por las diferentes tareas de forma concurrente.

## **Mecanismos de implementación**

En este espacio, más que identificar qué patrones aplicar se provee una descripción a alto nivel de los mecanismos implementados según el diseño obtenido y que se ajusten al entorno del clúster. Los mecanismos de implementación identificados según cada clasificación son:

### Gestión Procesos/Threads

La decisión de en qué momento usar procesos o threads en un programa paralelo es una de las más importantes, puesto que una mala implementación podría llevar un bajo

desempeño en la ejecución del programa, una comunicación con una alta latencia hacia el sistema C3 e incluso la incapacidad de sincronizar adecuadamente la información y estructuras de datos compartidas identificadas entre las tareas identificadas.

Para tener un adecuado balance en el uso de recursos en el clúster se decide que para cada tipo de telemetrías, el conjunto de tareas que intervienen en el análisis respectivo se ejecutarán en un nodo distinto. Para lograr esto, se necesita que a nivel del sistema operativo en cada nodo se realice una asignación de recursos en memoria, contadores, registros y descriptores a archivos abiertos entre otros, para que exista por lo menos un proceso ejecutándose en un procesador de cada nodo.

Al tener un proceso instanciado en cada nodo, los diferentes conjuntos de tareas asignadas pueden ejecutarse en un pool de threads creado dentro del proceso con la ventaja de compartir los mismos recursos asignados al proceso para garantizar una mejor sincronización, comunicación y utilización de recursos.

### Sincronización

Llegado a un mayor nivel de detalle en el diseño de la solución del sistema CAT por medio de una aplicación paralela y habiendo definido cómo es la distribución de tareas en threads por cada proceso, es importante especificar el mecanismo de cómo las diferentes instrucciones en cada tarea realizan operaciones de lectura o escritura en las estructuras de datos e información compartida en memoria sin presentar problemas.

Para lograr evitar problemas de condición de carrera y para evitar inconsistencias en la integridad de la información se utilizaron mecanismos de exclusión mutua a través de instrucciones mutex, con esto se logra proteger los recursos compartidos de los posibles accesos concurrentes a estos.

## Comunicación

En la implementación de la solución se ha detallado cómo se realiza un intercambio de la información entre las tareas de un mismo grupo a través de la memoria compartida en un proceso, sin embargo, aún no es evidente cómo se realiza el intercambio de información entre el proceso que recibe la información del servidor C3 y los diferentes procesos distribuidos en los nodos. Para solucionar esto en la comunicación entre los diferentes nodos se utilizará el mecanismo de intercambio de mensajes MPI (Message Passing Interface) por medio de instrucciones asíncronas punto a punto desde el proceso que contiene al pool de threads y el cual ejecuta la tarea T2, encargada de extraer los mensajes de la cola compartida y de distribuirlos a los distintos nodos para que estos puedan realizar el procesamiento de las telemetrías.

## **Vistas arquitecturales de la solución**

A continuación se presentan algunos modelos y vistas para un mejor entendimiento del sistema:

### **Vista funcional global**

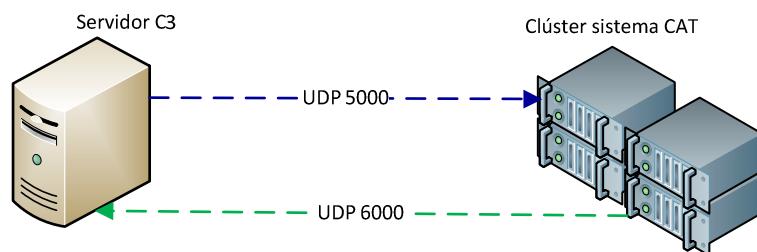


Fig. 53. Vista funcional global CAT

Este diagrama presenta un contexto general de la interacción entre el sistema C3 y el sistema CAT, la comunicación entre los dos sistemas se realiza a través del protocolo UDP, las tramas de información que envía el servidor C3 hacia el sistema CAT

instanciado en el clúster se establecen en el puerto 5000. Las tramas de información de respuesta se envían a través del puerto 6000 utilizando el mismo protocolo.

## Vista componentes

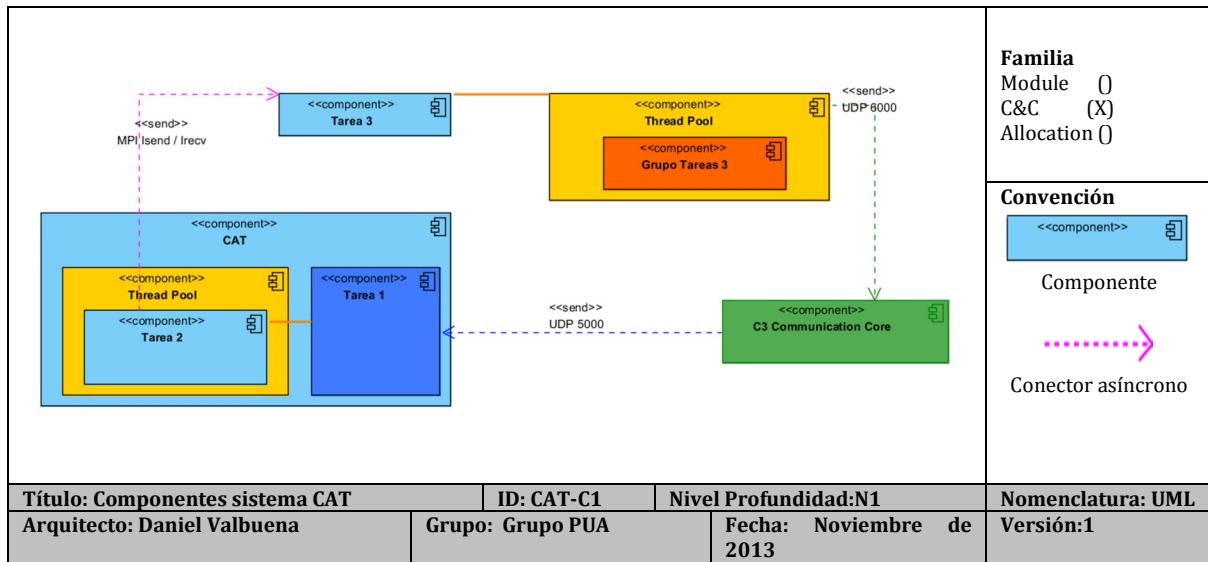


Fig. 54. Vista de componentes sistema CAT

Este diagrama presenta los principales componentes del sistema CAT identificados por tareas y grupos de tareas, el componente Tarea 1 recibe la información del sistema C3 y se encarga de almacenarlo en memoria compartida, existe un componente thread pool que se encarga de ejecutar diversos tipos de tareas, para el caso del proceso maestro, este extrae la información de la memoria compartida y la envía a los diferentes nodos donde se realizan los análisis. El componente Tarea 3 se encarga de seleccionar los datos relevantes para el análisis y de guardarlos en memoria compartida, finalmente, el grupo de tareas 3 es ejecutado dentro del componente thread pool y los resultados obtenidos son enviados al sistema C3 específicamente al componente C3 ‘Communication Core’.

## Vista de despliegue

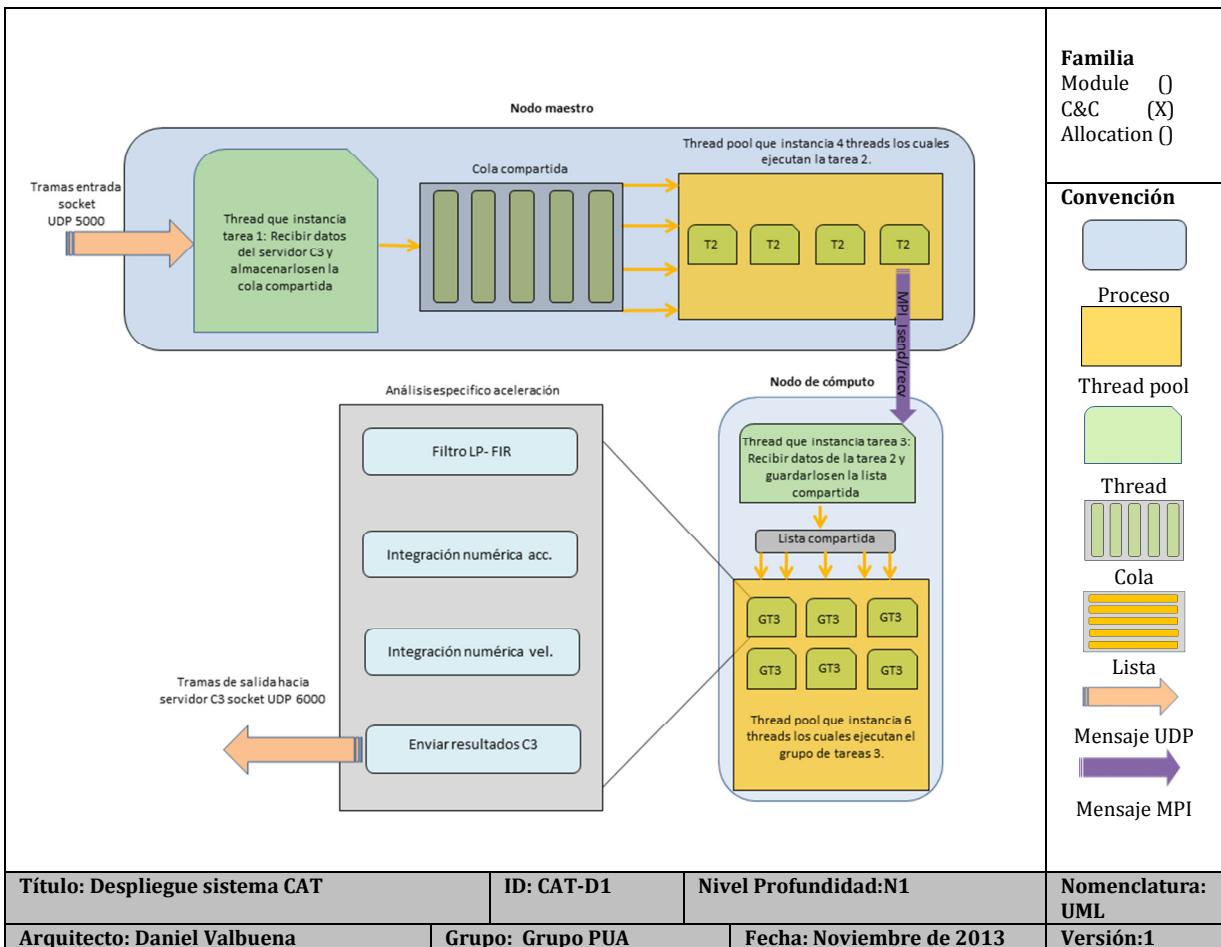


Fig. 55. Vista de despliegue sistema CAT

Esta vista presenta el despliegue general del sistema CAT sobre el clúster HPC de la facultad de ingeniería, a modo de ejemplo se presenta el despliegue del nodo maestro y la interacción de este con un nodo de cálculo, cabe decir que por cada tipo de análisis se despliega un nuevo nodo, sin embargo todos presentan la misma estructura de despliegue: Un thread que recibe la información del nodo maestro y la almacena en una lista compartida, y un conjunto de threads que ejecutan el análisis indicado, para este caso se detalla el análisis específico de la aceleración.

## Vista de concurrencia

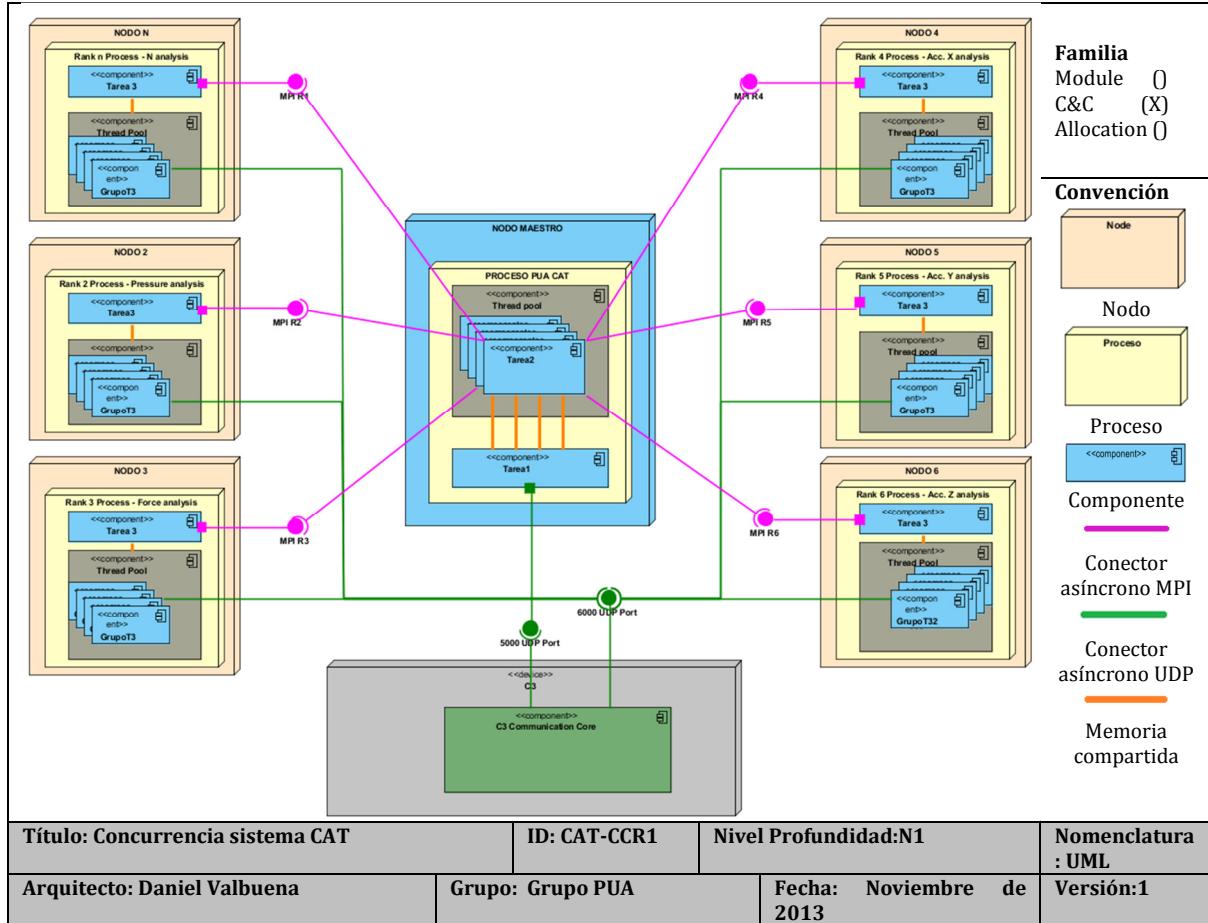


Fig. 56. Vista de despliegue sistema CAT

Esta vista presenta el esquema general de los procesos, threads, pool de threads que debe tener el sistema C3 para su operación. Existe el proceso maestro el cual se encarga de recibir todas las tramas de información del sistema C3 y utilizando un thread pool toma la información almacenada en memoria compartida y la reenvía a los diferentes procesos distribuidos en los nodos de cómputo del clúster HPC. Allí la información es almacenada en una estructura de datos compartida, para que un pool de threads pueda ejecutar las tareas de análisis y entregar los resultados al sistema C3 para la visualización.

## Flujo de información

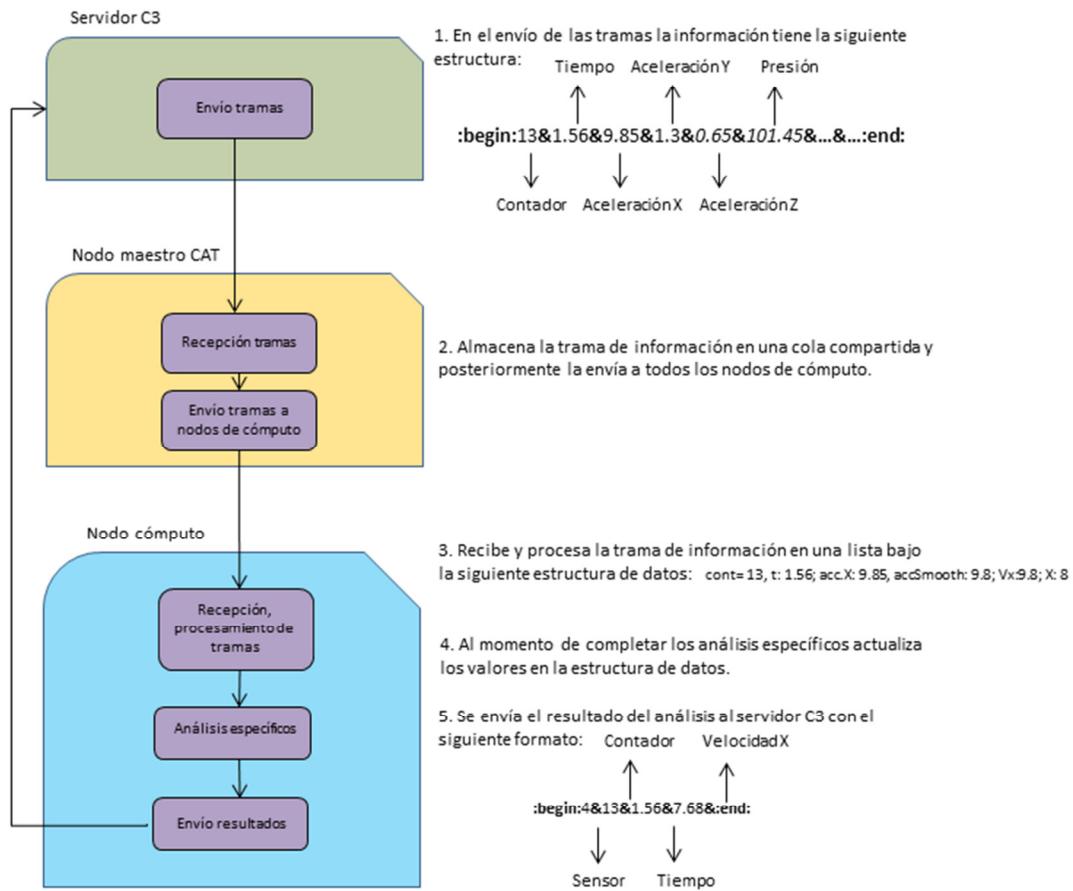


Fig. 57. Flujo de información sistema CAT

El modelo de flujo de información indica cómo y en qué formato la información es recibida desde el servidor C3 y es gestionada en el nodo maestro, el cuál a través de una cola compartida la almacena de forma temporal, para que luego esta sea distribuida hacia los diferentes nodos de cómputo. Cuando la información es recibida allí se almacena en una lista compartida sobre una estructura de datos que contiene los datos importantes para realizar los diferentes análisis, en la medida que estos se vayan ejecutando dicha estructura seguirá actualizándose y los resultados serán reenviados al servidor C3 en un formato específico para el despliegue visual.

## **Implementación de filtro LP - FIR**

Tomando como referencia diferentes documentaciones en este tema [61] [64] [65] [66], para realizar un filtro de los datos de la señal recibida y así poder eliminar los ruidos de los sensores se utiliza un filtro baso bajo en su forma más sencilla que consiste en realizar un promedio sobre un conjunto de datos y a cada uno de estos valores multiplicarlos por un coeficiente. A este tipo de algoritmo se le llama suavización triangular, la cual que le da un peso a través de cada coeficiente al conjunto de datos. Para realizar el cálculo de los valores se tuvieron en cuenta diferentes casos debido a las condiciones de extremos o de borde en un instante de tiempo:

- Cuando la señal está ubicada en el extremo de la lista de valores, no se realiza ningún suavizado puesto que no se tienen valores adyacentes que puedan atenuar la señal.
- Cuando la señal posee por lo menos un valor adyacente:

$$y[n] = \frac{x[n-1] + 2x[n] + x[n+1]}{4}$$

Fig. 58. Suavización señal con un dato adyacente.

- Cuando la señal posee dos valores adyacentes:

$$y[n] = \frac{x[n-2] + 2x[n-1] + 3x[n] + 2x[n+1] + x[n+2]}{9}$$

Fig. 59. Suavización señal con dos datos adyacentes.

- Cuando la señal posee tres valores adyacentes:

$$y[n] = \frac{x[n-3] + 2x[n-2] + 3x[n-1] + 4x[n] + 3x[n+1] + 2x[n+2] + x[n+3]}{16}$$

Fig. 60. Suavización señal con tres datos adyacentes.

Con este tipo de algoritmo se garantiza que los valores de la señal extremadamente superiores a sus valores adyacentes sean atenuados, y a su vez, los valores que sean extremadamente inferiores a los valores adyacentes sean incrementados. Esto conlleva a un suavizado de los valores, atenuando el ruido pero sin distorsionar la señal. Se entiende que cada valor “ $x$ ” corresponde a un registro de la estructura de datos en la lista ordenada ubicada en cada nodo de cómputo, como ilustración de esto se presenta el caso general:

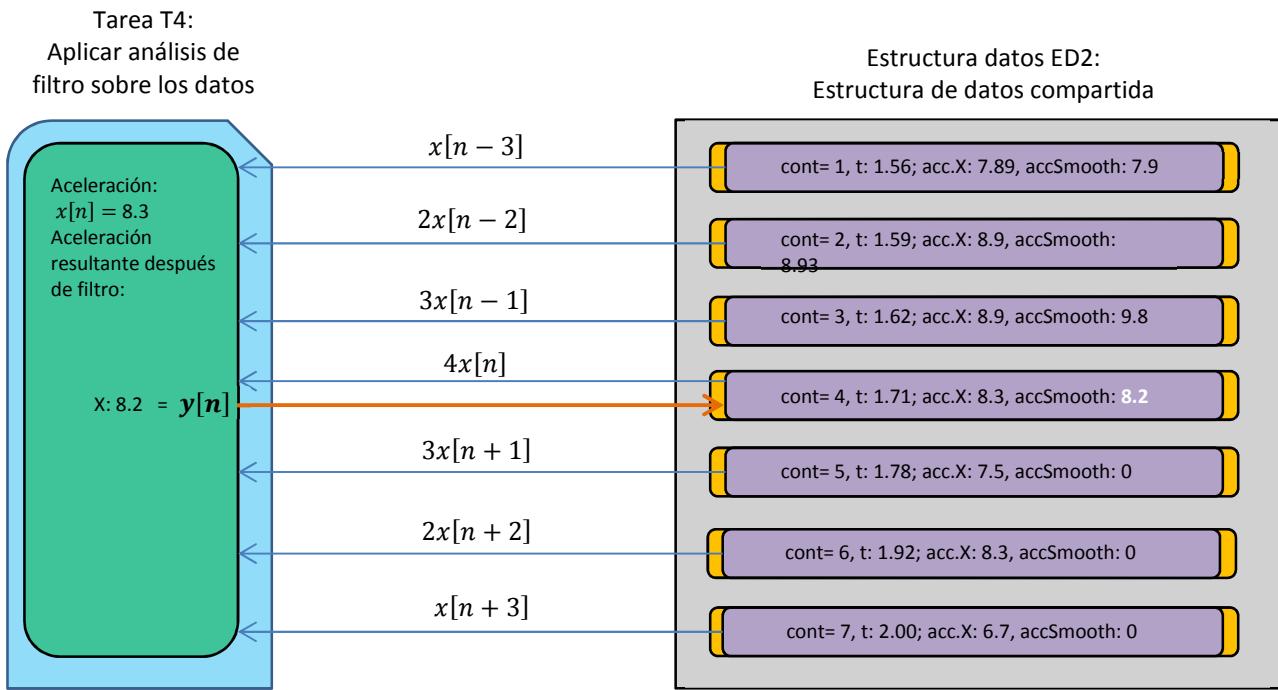


Fig. 61. Interacción de tarea análisis de filtro con estructura de datos compartida.

## Implementación integración numérica

Se usará el método trapezoidal de integración numérica para calcular la velocidad y posición del vehículo aeroespacial. En el caso de la velocidad la ecuación de integración está definida de la siguiente forma:

$$V = (t[n] - t[n-1]) \times \frac{(a[n] + a[n-1])}{2}$$

Fig. 62. Método de integración trapezoidal para calcular la velocidad.

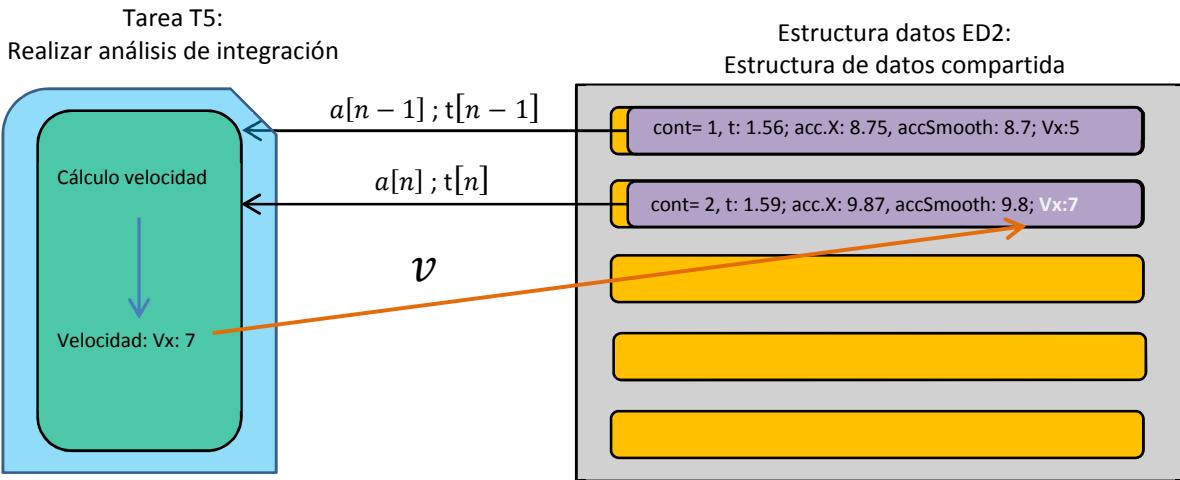


Fig. 63. Interacción de tarea análisis de integración velocidad con estructura de datos compartida.

Para el cálculo de esta integral de velocidad es necesario conocer el valor de la aceleración actual y previa sobre el segmento de la curva al que se va a calcular el área. Dado que el tiempo entre los intervalos de medición no es el mismo, se incorpora el valor actual y el valor previo en la ecuación para tener una aproximación un poco más precisa de la integral.

En el caso del cálculo del desplazamiento se utiliza el mismo método de integración pero de forma acumulada ya se desea obtener el desplazamiento del vehículo no solo en un instante de tiempo, sino el movimiento total que ha tenido desde el lanzamiento, para este caso la ecuación es:

$$P = (t[n] - t[n-1]) \times \frac{(V[n] + V[n-1])}{2} + P[n-1]$$

Fig. 64. Método de integración trapezoidal para calcular el desplazamiento.

Para el cálculo de esta integral acumulada de posición es necesario conocer el valor de la velocidad actual y previa sobre el segmento de la curva al que se va a calcular el área. Dado que el tiempo entre los intervalos de medición no es el mismo, se incorpora el valor

actual y el valor previo en la ecuación para tener una aproximación un poco más precisa de la integral, adicionalmente es necesario obtener el valor de la posición previa para así ir calculando el desplazamiento del vehículo durante todo el tiempo que dure la misión.

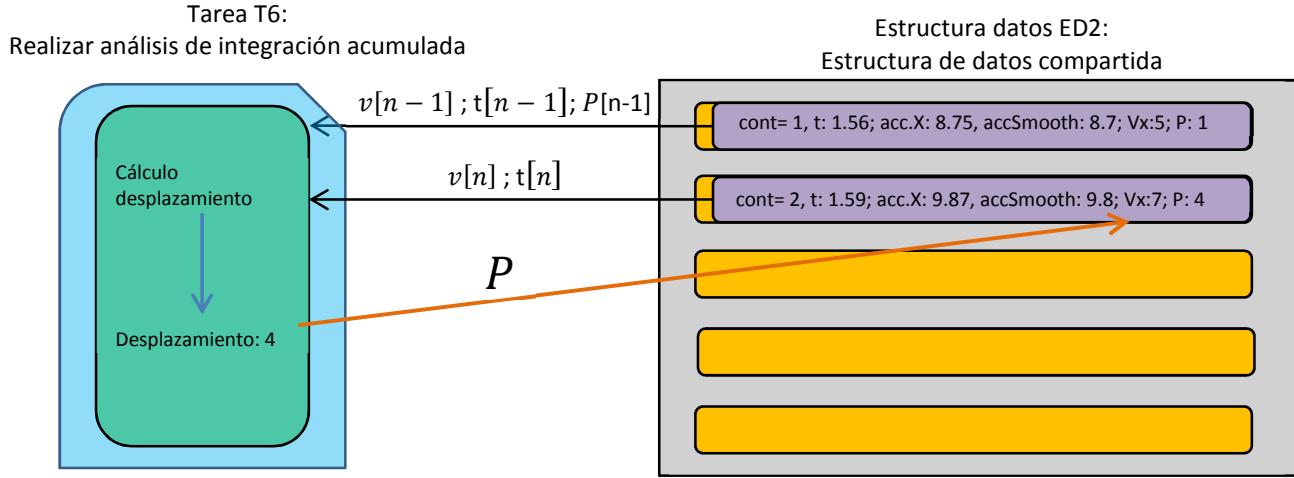


Fig. 65. Interacción de tarea análisis de integración acumulada con estructura de datos compartida.

## **CAPITULO V: EXPERIMENTACIÓN, VALIDACIÓN Y RESULTADOS**

En este capítulo se realiza el proceso de experimentación, validación y resultados de la implementación del sistema de análisis de telemetrías CAT. Para el proceso de experimentación se tomaron tres casos de estudio los cuales tratan de abarcar un amplio rango de posibilidades de aplicación del sistema CAT. La estructura de cada caso de estudio consiste en: contexto, tipo de análisis a realizar, resultados obtenidos, comparación e interpretación de los resultados.

Debido a problemas en la calidad de los datos obtenidos de los lanzamientos en las misiones Ainkaa 3, no fue posible utilizar esta información en los casos de estudio.

### ***Escenario 1***

#### **Contexto**

El primer escenario está basado en el estudio reportado en una publicación indexada ISI [67]: “*An experimental investigation of change detection in uncertain chain-like systems*” en el cual se implementan un conjunto de telemetrías de aceleración para un sistema de sistema de monitoreo, supervisión y estado de una estructura. El experimento consiste en la implementación de una red de sensores de aceleración con comunicación inalámbrica y capacidades computacionales para ser usados en la implementación de algoritmos distribuidos de monitoreo, supervisión y estado de una estructura. Este tipo de topología de implementación de sensores en cadena puede abarcar en varios tipos de aplicaciones prácticas, v.g.: rascacielos, torres de transmisión, plataformas de explotación, turbinas de aire y alas de aviones. Los datos experimentales fueron obtenidos de una estructura de cuatro plantas cada una con un sensor de aceleración en

cada piso, además en la base cuenta con un agitador que aplica una fuerza en el plano X desplegada en un banco de pruebas del Laboratorio Nacional de los Álamos (U.S.A).

## Descripción del experimento

La estructura de cuatro plantas consiste de 4 placas de aluminio ( $30.5 \times 30.5 \times 2.5$  cm) atornilladas a cuatro columnas de aluminio ( $17.7 \times 2.5 \times 0.6$  cm) en cada piso. Dos elementos adicionales en el tercer piso sirven para introducir una brecha de no linealidad en el sistema. Toda la estructura se encuentra montada sobre dos rieles que permiten el desplazamiento del sistema en una sola dirección, un agitador electro dinámico es usado para proveer una excitación aleatoria entre 20 y 150 Hz a la estructura de prueba. La red de sensores desplegados consiste de cuatro acelerómetros con una sensibilidad nominal de  $1000\text{mV/g}$ . Estos fueron adicionados en cada piso para medir la respuesta dinámica de la estructura en prueba. La prueba total duró 25 segundos con una frecuencia de 320Hz.

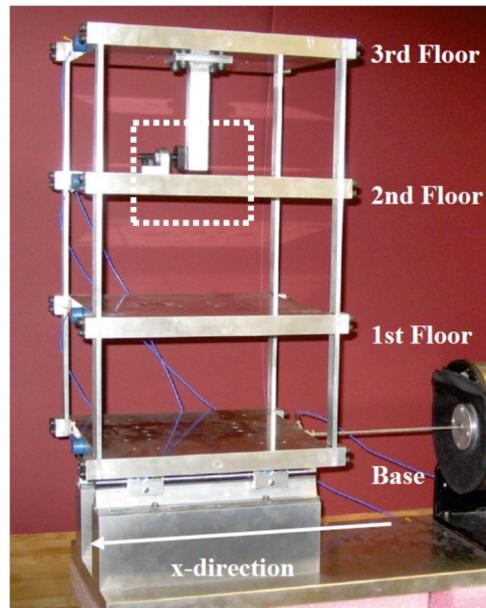


Fig. 66. Estructura de 4 plantas, tomado de [67].

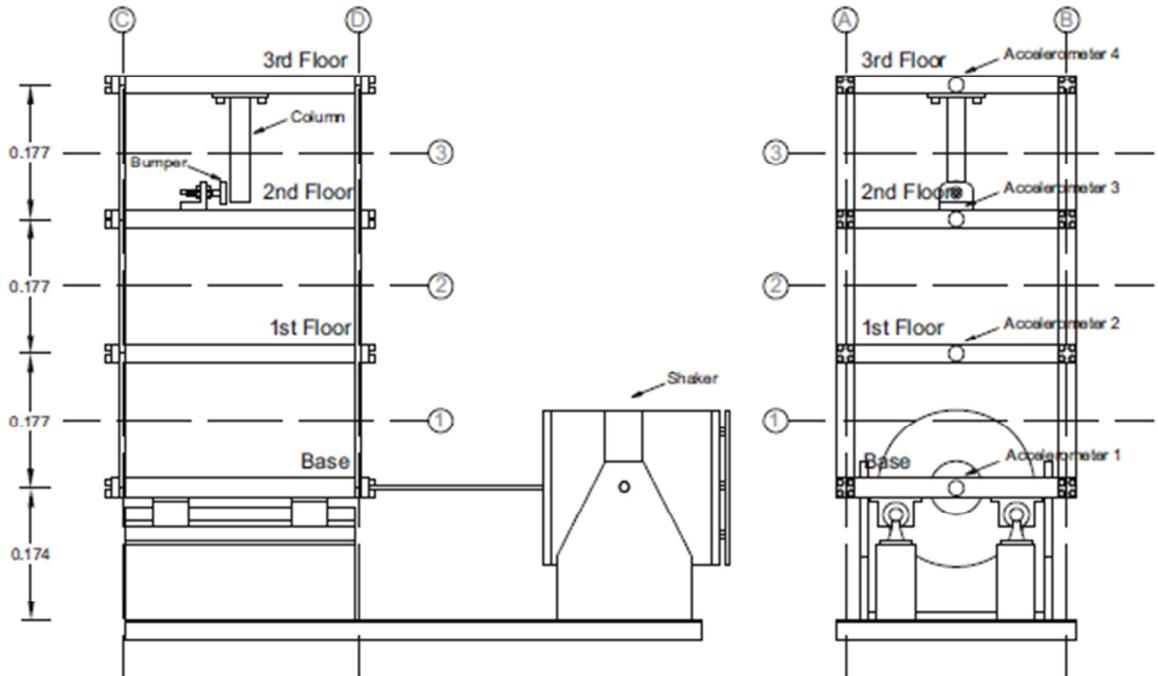


Fig. 67. Diseño de estructura: Vista lateral y frontal, tomado de [67].

### Tipo de análisis a realizar

Análisis de aceleración sobre el eje X, cálculo de velocidad y desplazamiento para los tres primeros sensores.

### Experimentación

Para cada uno de los sensores de aceleración se logró obtener los datos arrojados de la experimentación en el laboratorio, estos datos fueron usados en la experimentación adicionándolos al sistema de plataforma en tierra y así ser enviados al sistema C3.

El servidor C3 se encargó de recibir la información y generó las tramas de información que fueron enviadas al sistema CAT, inmediatamente, este se encargó de recibir y distribuir las tramas en los diferentes nodos. Cada nodo involucrado realizó el filtro paso bajo para la aceleración, cálculo de velocidad y desplazamiento para los tres primeros sensores y reenvió la información procesada al servidor C3 para la visualización. El

servidor C3 envió la información a graficar al navegador de los clientes para que finalmente sea desplegada a los usuarios.

## Resultados obtenidos

En el navegador del cliente, para las aceleraciones en tres de los sensores después de aplicar el filtro los resultados obtenidos fueron los siguientes:

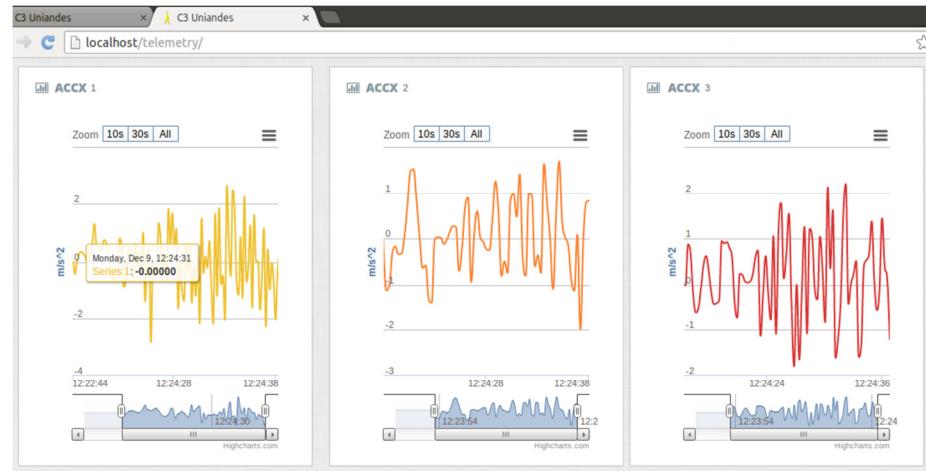


Fig. 68. Resultados filtro LP sobre telemetrías de aceleración 1, 2, 3

Una vez realizado el cálculo de la velocidad en cada sensor los resultados obtenidos fueron los siguientes:

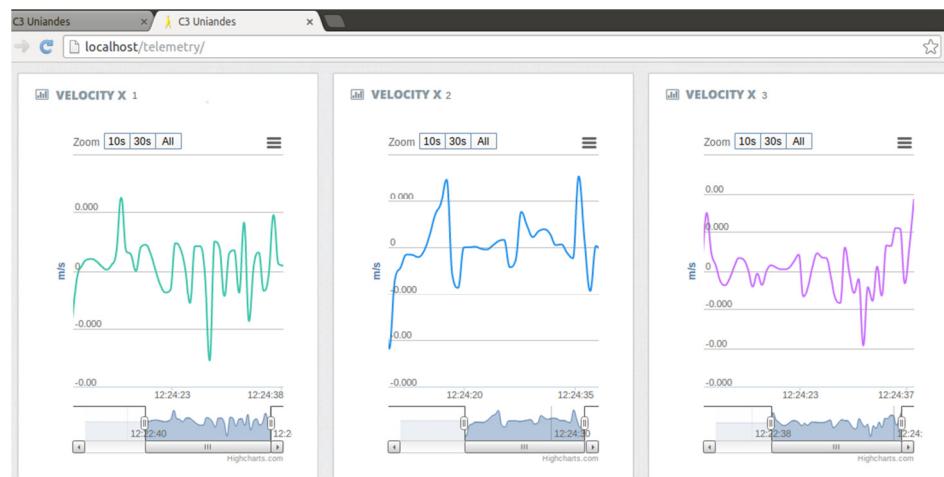


Fig. 69. Resultados de cálculo velocidad 1,2,3

Los resultados del cálculo del desplazamiento fueron los siguientes:

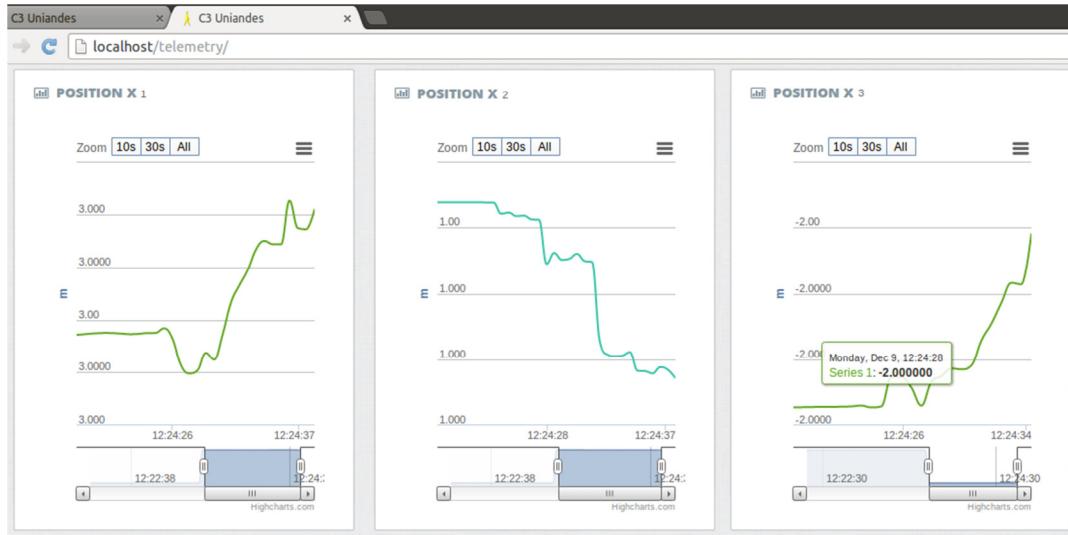


Fig. 70. Resultados de cálculo desplazamiento 1,2,3

## Comparación e interpretación de los resultados

Para realizar la comparación e interpretación de los resultados, se tomaron los datos de entrada y se analizaron usando el programa MATLAB, para así determinar qué tan precisos son los resultados que arroja el sistema CAT y qué tan fiable son los resultados a nivel gráfico. Para este caso se realizó el análisis sobre la aceleración del primer sensor.

Aceleración filtrada en MATLAB:

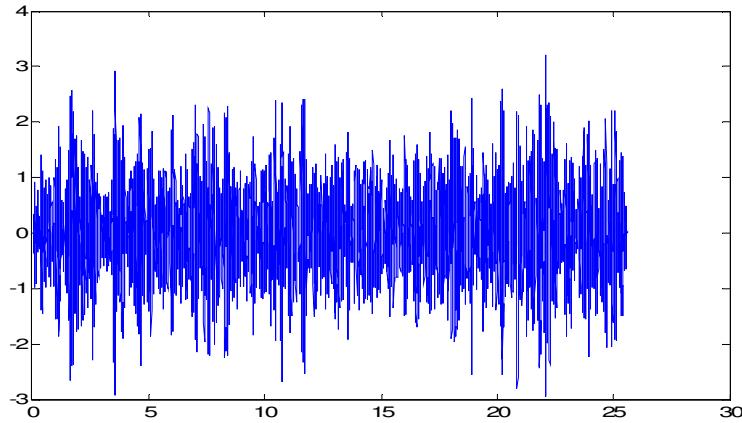


Fig. 71. Aceleración filtrada utilizando MATLAB.

Aceleración filtrada en sistema CAT:

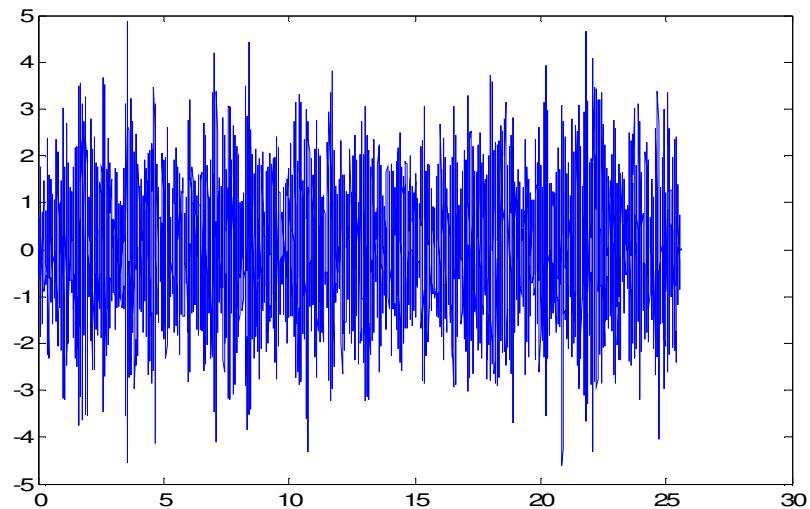


Fig. 72. Aceleración filtrada utilizando el sistema CAT.

Aceleración presentada en el sistema C3:

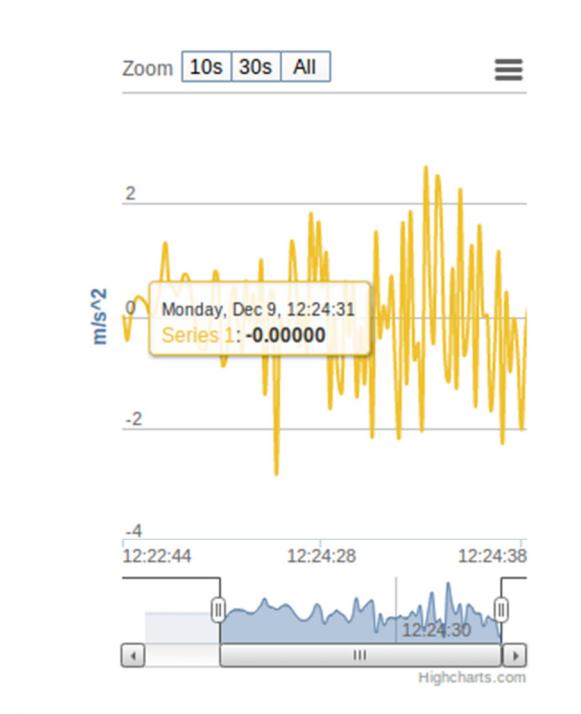


Fig. 73. Aceleración graficada en el sistema C3.

En primer lugar se evidencia que aplicando el filtro en el sistema MATLAB atenúa más la señal que frente al sistema CAT, por otra parte es claro que el componente de graficación C3 atenúa aún más la señal, e incluso despliega un comportamiento de la curva no esperado.

Velocidad en MATLAB:

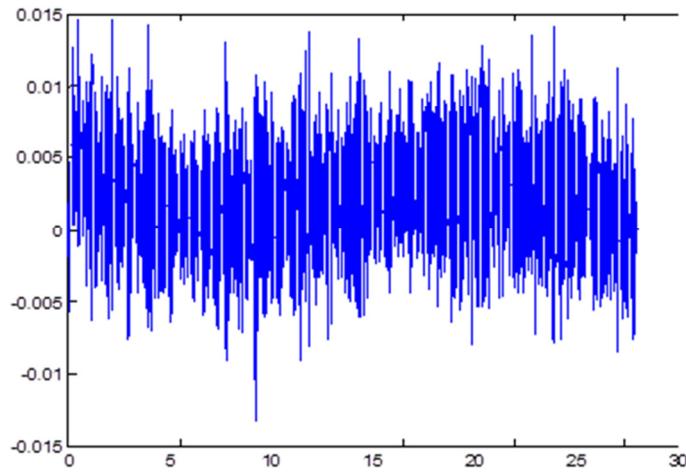


Fig. 74. Velocidad calculada utilizando MATLAB.

Velocidad en sistema CAT:

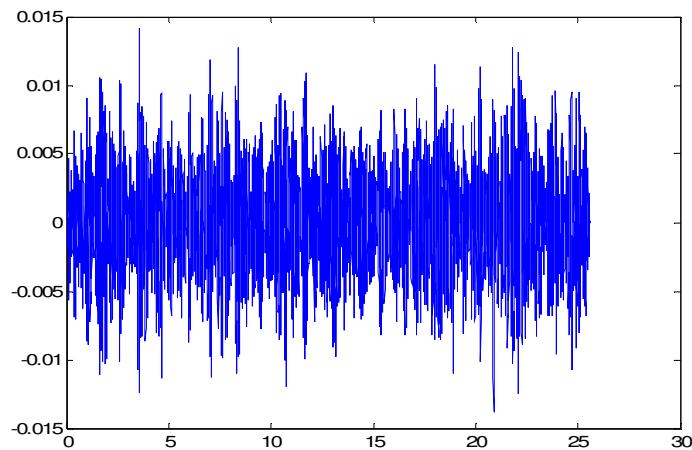


Fig. 75. Velocidad calculada utilizando el sistema CAT.

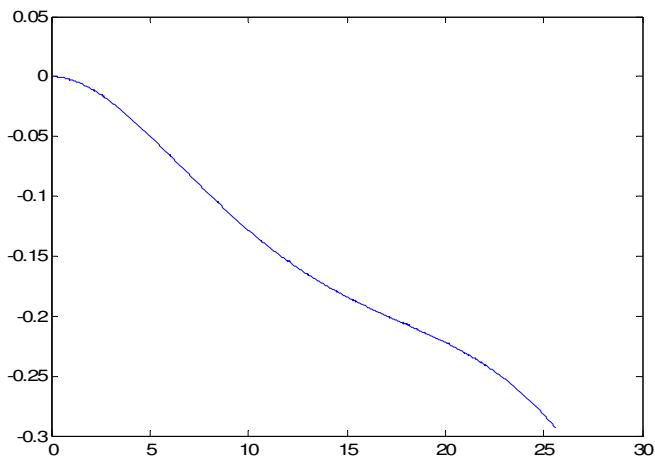
Velocidad presentada en sistema C3:



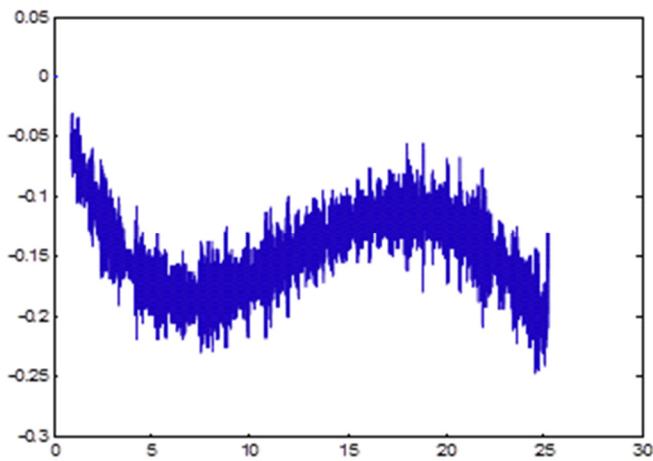
Fig. 76. Desplazamiento graficado en el sistema C3.

Nuevamente en esta comparación se evidencia que no se la atenuación en MATLAB es ligeramente distinta frente al sistema CAT, sin embargo, nuevamente se puede observar que la presentación de la información por parte del sistema C3 no arroja resultados congruentes con la información que se le envía.

Desplazamiento calculado en MATLAB:



Desplazamiento calculado en sistema CAT:



Desplazamiento presentado en sistema C3:



Fig. 77. Desplazamiento graficado en el sistema C3.

Como resultado de esta comparación se evidencia que la información desplegada por el sistema C3 no refleja información similar a la información que le es entregada por el sistema CAT. Por otra parte es necesario revisar qué otros métodos de suavización e integración pueden implementarse para un sistema análisis en tiempo real.

## ***Escenario 2***

### **Contexto**

El tercer caso de estudio utilizó información telemétrica del lanzamiento de un modelo a escala de 4" de diámetro del misil Erint/PAC3 (Extended Range INTerceptor (ERINT)/Patriot Advanced Capability - 3 (PAC3)) [68]. El misil PAC3 es un misil operativo desde 1997 en diferentes fuerzas militares del mundo. David Schultz, vicepresidente del Dallas Area Rocket Society (DARS) realizó el diseño, ensamblaje y lanzamiento del modelo como requisito para obtener la certificación de nivel 3 de la National Association of Rocketry (NAR level III certification).



Fig. 78. Lanzamiento cohete Erint/PAC3 [68]

### **Descripción del experimento**

Como primer paso David se realizó el diseño del misil a escala utilizando la herramienta RockSim:

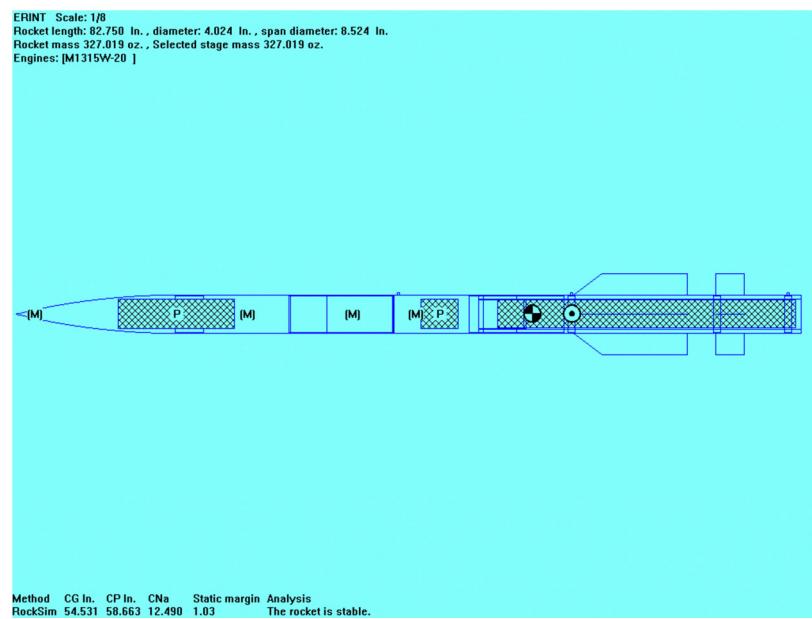


Fig. 79. Diseño cohete Erint/PAC3 [69]

Posteriormente comenzó con el ensamblaje del prototipo según el diseño realizado:

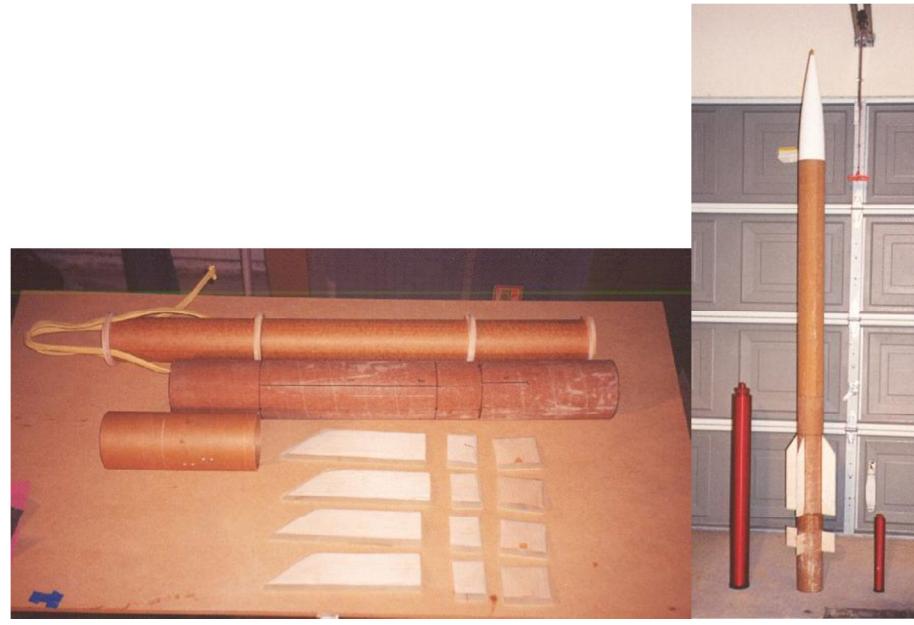


Fig. 80. Proceso de ensamble cohete Erint/PAC3 [69]

Para finalmente realizar el lanzamiento y despliegue del paracaídas para así cumplir uno de los requisitos de la certificación.



Fig. 81. Despliegue paracaídas cohete Erint/PAC3 [69]

Durante todo el lanzamiento se realizaron análisis de telemetrías sobre un sensor de aceleración y presión, para el caso de estudio se utilizará la información obtenida del sensor de aceleración para aplicar los análisis de filtros, cálculo de velocidad y desplazamiento.

### **Tipo de análisis a realizar**

Análisis de aceleración, cálculo de velocidad y altura sobre el sensor desplegado en el cohete con base en la telemetría obtenida del experimento.

### **Experimentación:**

Utilizando la herramienta de software RDAS [70] se logró obtener los valores de tiempo y aceleración de la experimentación realizada en campo, estos datos fueron adicionados al sistema de plataforma en tierra para que este lo enviara al sistema C3.

El servidor C3 se encargó de recibir la información y generó las tramas de información que fueron enviadas al sistema CAT. En este caso, al realizar el análisis de una sola telemetría solo se realizó la distribución de las tareas de análisis en un solo nodo, en este se realizó el cálculo del filtro LP – FIR para la aceleración, cálculo de velocidad y desplazamiento, y se reenvió la información procesada al servidor C3 para la visualización. El servidor C3 envió la información a graficar al navegador de los clientes y finalmente desplegada a los usuarios.

## Resultados obtenidos:

En el navegador del cliente los resultados obtenidos fueron:

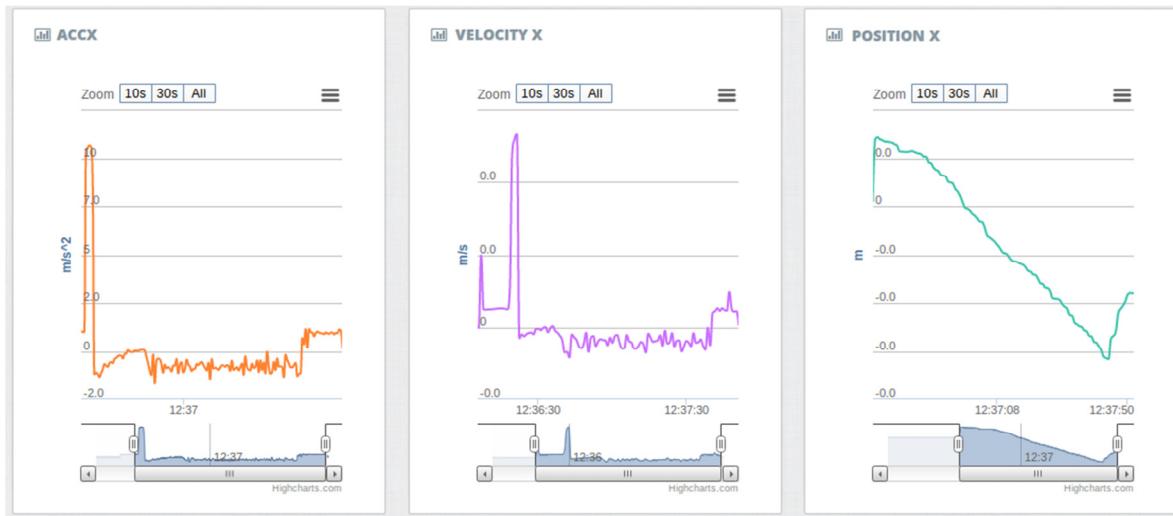


Fig. 82. Resultados filtro LP sobre telemetría de aceleración, calculo de velocidad y desplazamiento

## Comparación e interpretación de los resultados:

Resultados aceleración sistema RDAS:

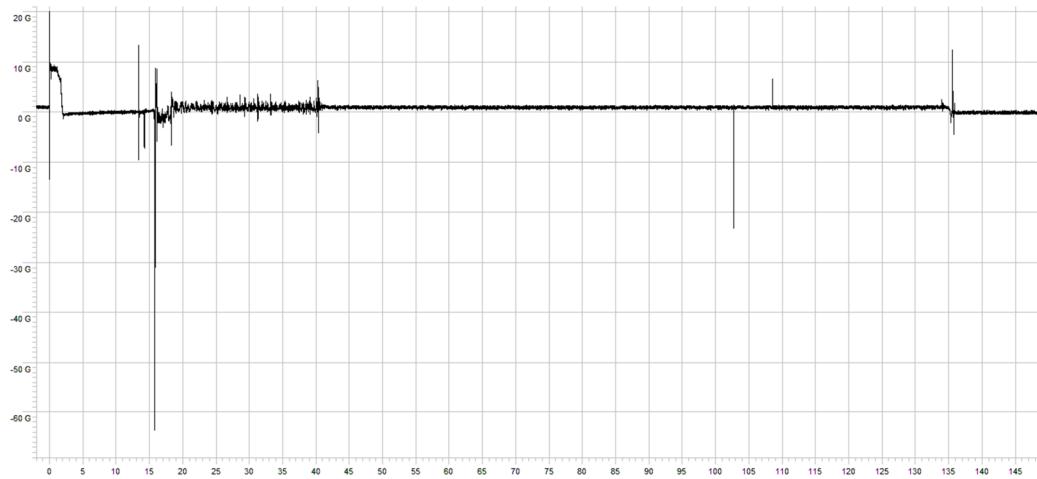


Fig. 83. Aceleración registrada en el sistema RDAS.

#### Aceleración calculada en sistema CAT

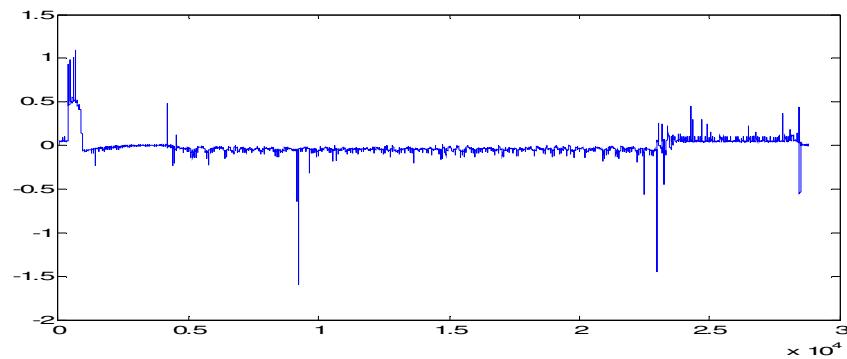


Fig. 84. Aceleración filtrada utilizando el sistema CAT.

#### Aceleración presentada en sistema C3:

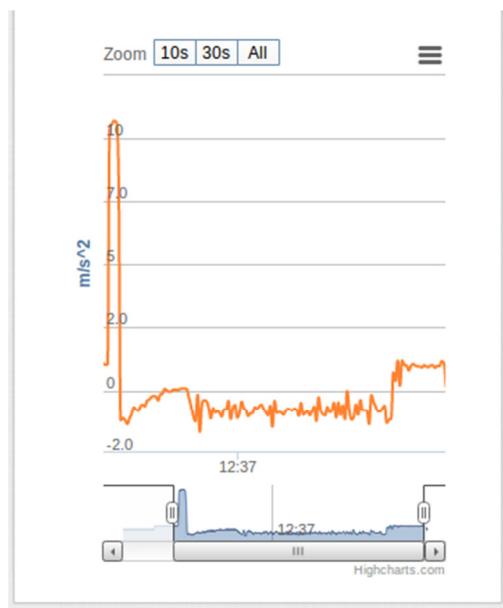


Fig. 85. Aceleración graficada en el sistema C3.

Velocidad obtenida del sistema RDAS:

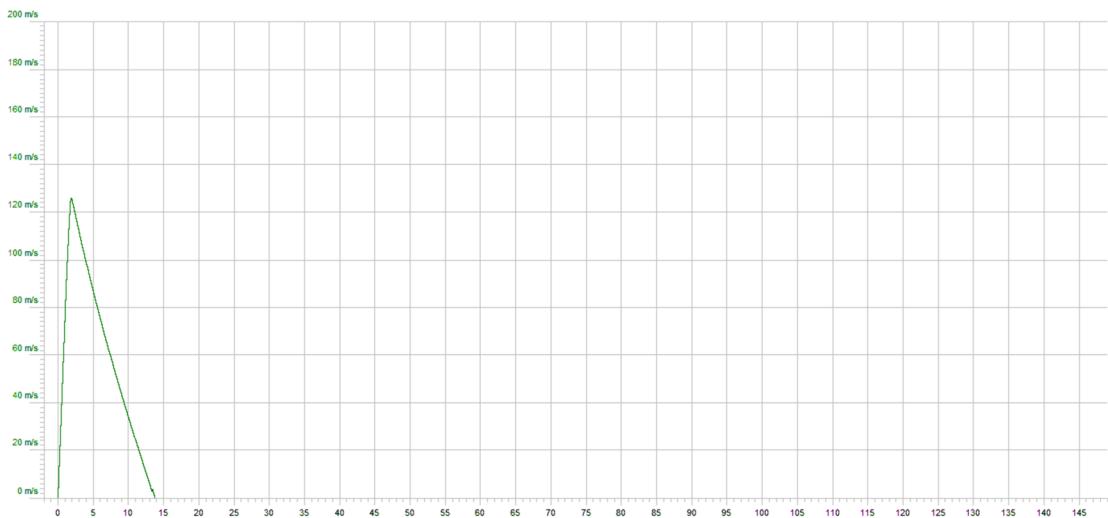


Fig. 86. Velocidad registrada en el sistema RDAS.

Velocidad calculada en sistema CAT:

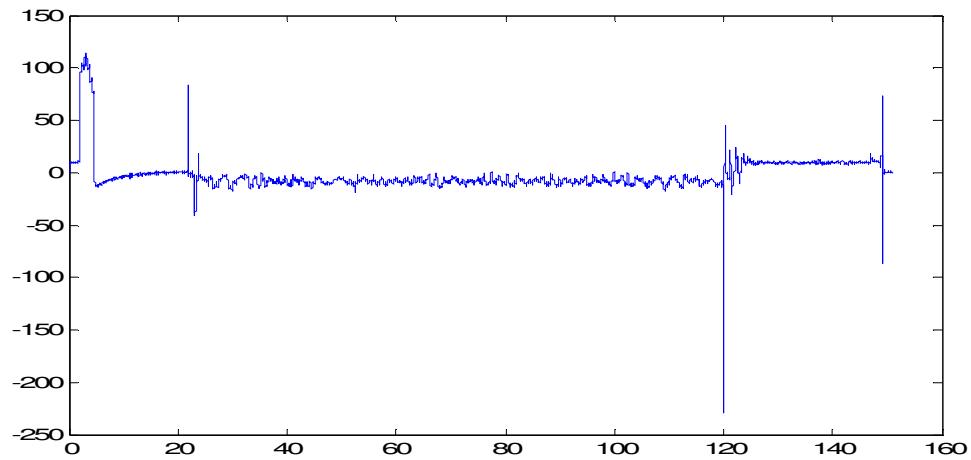


Fig. 87. Velocidad calculada utilizando el sistema CAT.

Velocidad presentada en sistema C3:

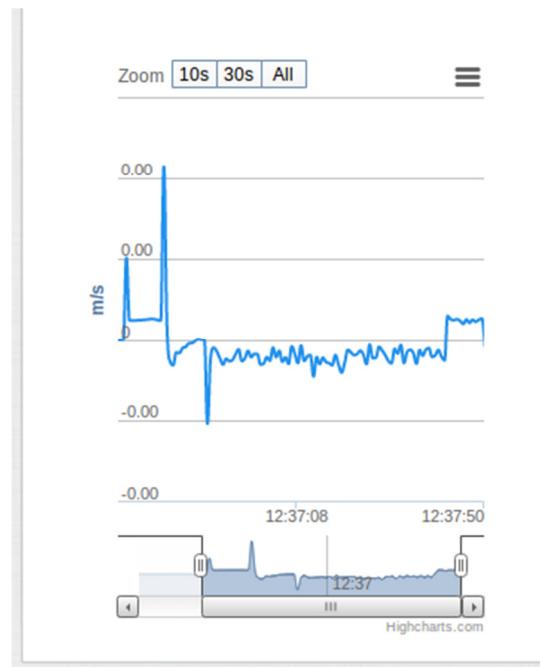


Fig. 88. Velocidad graficada en el sistema C3.

Desplazamiento obtenido del sistema RDAS:

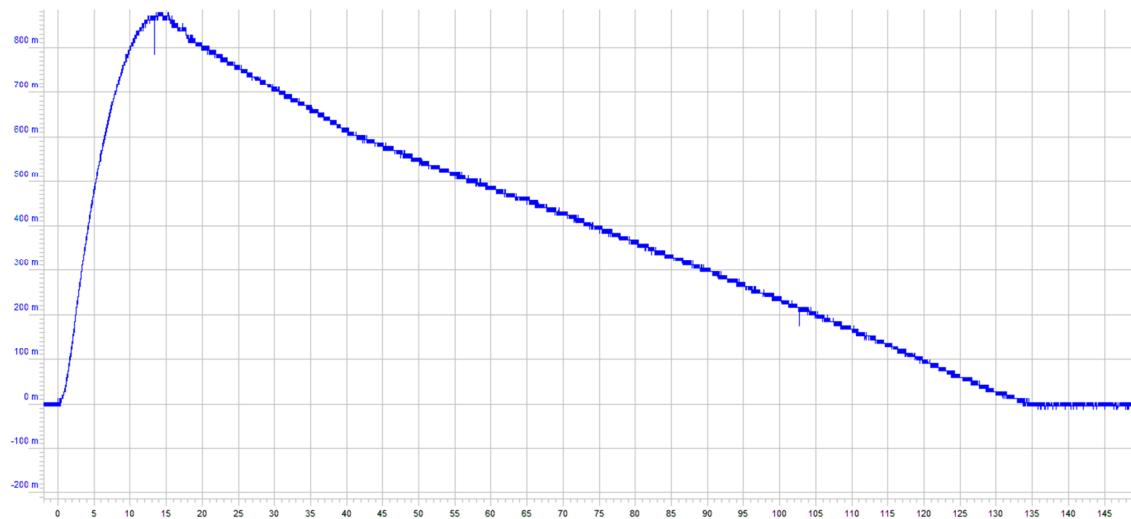


Fig. 89. Desplazamiento registrado en el sistema RDAS.

Desplazamiento calculado en sistema CAT:

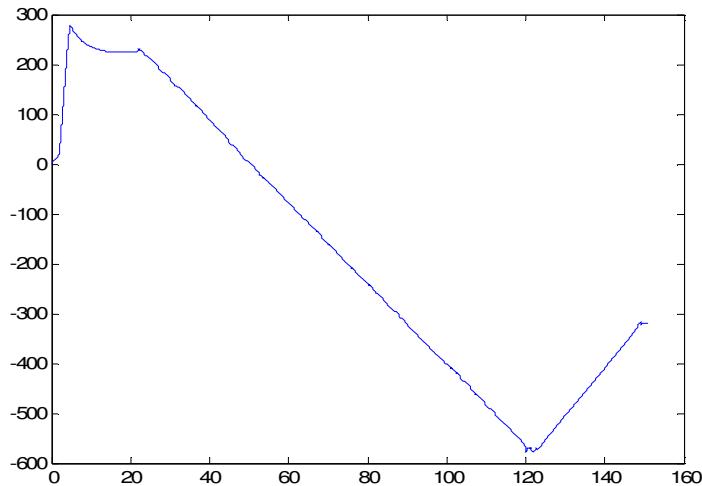


Fig. 90. Desplazamiento calculado utilizando el sistema CAT.

Desplazamiento presentado en sistema C3:

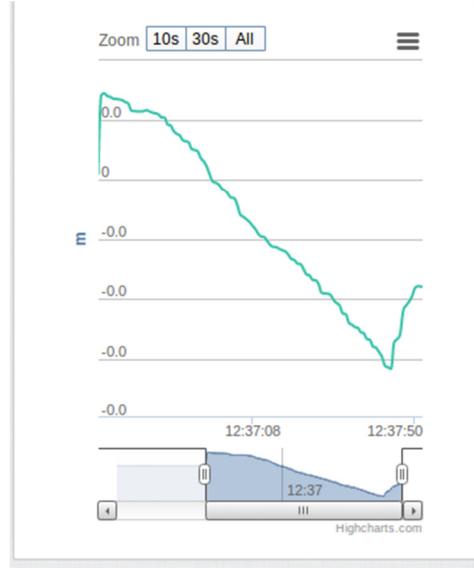


Fig. 91. Desplazamiento graficado en el sistema C3.

Como resultado de estas comparaciones, en este caso se encuentra una mejor tendencia del comportamiento de las diferentes curvas presentadas en los diferentes tipos de análisis, sin embargo se debe revisar el trabajo de las escalas en la presentación de la información y evaluar la posibilidad de realizar un filtro y suavizado para las curvas de velocidad y aceleración.

### ***Escenario 3***

#### **Contexto**

El tercer caso de estudio se desarrolló en los laboratorios del departamento de ingeniería mecánica sobre una mesa de vibraciones con el apoyo del estudiante Santiago Arteaga quien diseñó el experimento. El propósito es poder observar por el desempeño de las funcionalidades implementadas en el Clúster C3 y poder comprobar su precisión mediante la utilización de un sistema de oscilaciones sencillo que permita conocer la señal de entrada (mesa de vibraciones), un componente que permita alterar la salida de

las ondas (amortiguador, resorte o similar) y mediante el análisis y procesamiento de estas dos señales conocidas observar si el Clúster C3 es capaz de interpretar señales digitales provenientes de un sistema mecánico real.

## **Descripción del experimento**

Para realizar esta confirmación de manera sencilla se construye un sistema de segundo orden sub-amortiguado utilizando un marco hueco de lámina de aluminio en forma de tambor, tensando una tela de aluminio en la parte superior e instalando todo este sistema sobre la máquina de vibraciones provista por el laboratorio de mecánica. Después de esto se procede a instrumentar el sistema con dos acelerómetros, uno de ellos puesto en la base de la mesa de vibraciones al interior de tambor y el otro instalado sobre la membrana de nilón tensada sobre el marco de aluminio, al tener esto se procede a ejecutar pruebas de oscilación durante un periodo de 3 minutos pasando por el rango de 5.0 a 30.0 Hz (evitando la zona de los 8.0 Hz debido a la resonancia mecánica del sistema) para obtener datos de aceleración en el eje vertical en estado estable y de transición del sistema. Al final la información es transformada de formato análogo a digital y es enviada al sistema C3 mediante un programa intermedio para ser procesados por el Clúster C3.

Lista de equipos e instrumentos utilizados:

- VI de adquisición de datos implementando en LabVIEW 2010 32bits.
- Tarjeta National Instruments 9215 de 4 canales.
- Mesa de vibraciones con rango de operación de 5.0 Hz a 60.0 Hz
- Tambor hueco de aluminio, (12.0 cm de diámetro, 7.0 cm de alto y espesor de lámina 1.3mm).

- Membrana amortiguada (tela de Nilón).
- Acelerómetro KISTLER 8319B2 de  $\pm 2.0$  g
- Acelerómetro KISTLER 8319B10 de  $\pm 10.0$  g



Fig. 92. Mesa de vibraciones laboratorio Ing. Mecánica



Fig. 93. Control del generador de señales y moto reductor de la mesa de vibraciones

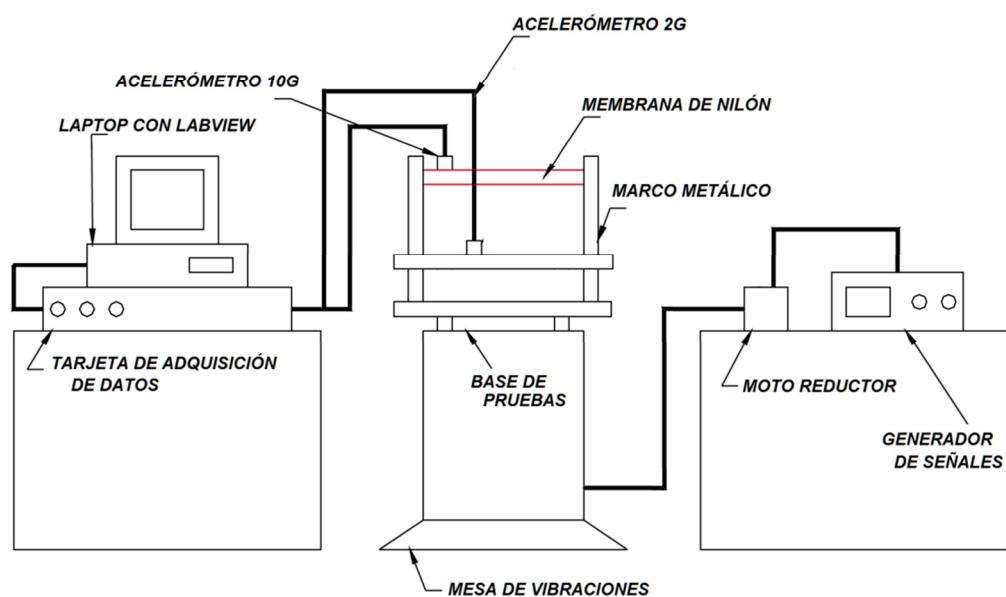


Fig. 94. Descripción general de equipos e instrumentos utilizados

## **Tipo de análisis a realizar**

Análisis de aceleración sobre el eje X, cálculo de velocidad y desplazamiento de los dos sensores desplegados en la mesa de vibraciones.

## **Experimentación**

1. El marco metálico es fijado a la base de pruebas de la mesa de vibraciones.
2. Se instala el acelerómetro de 2G en la base del marco metálico a escala de uso lineal de 995 mV/g.
3. Se tensa la membrana de nilón sobre el marco y se asegura a este.
4. Se instala el acelerómetro de 10G sobre la membrana de nilón a escala de uso lineal de 197 mV/g.
5. Se conectan los acelerómetros a la tarjeta de adquisición de datos NI9215.
6. Se inicializa y configura el VI de adquisición en el laptop por medio del programa Labview a una frecuencia de muestreo de 1000 Hz y una persistencia de archivos del VI en modo automático.
7. Se calibra el rango de prueba (5.0 Hz a 60.0) mediante el generador de señales y el moto reductor.
8. Se inicializa la prueba 10.0 Hz y se realiza una prueba preliminar para revisar todo el montaje.
9. Se realizan las diferentes pruebas en las escalas de 15.0Hz, 20.0 Hz y 25.0Hz para el sistema manteniendo cada escala por lo menos 30 segundos.
10. Se recuperan los archivos del VI de Labview.
11. Los archivos son formateados por medio de un programa intermediario para ser enviados al C3 y analizados por el clúster.

La experimentación se realiza en tres diferentes escalas en por lo menos 5 oportunidades durante 1 minuto cada escala iniciando desde el mínimo operacional de la mesa de vibraciones hasta llegar a los 25.0 Hz requeridos por el experimento. Al final la experimentación tendrá cinco archivos de por lo menos dos minutos de información recopilada por el sistema.

## Resultados obtenidos

En el navegador del cliente, para las aceleraciones en los dos sensores después de aplicar el filtro los resultados obtenidos fueron los siguientes:

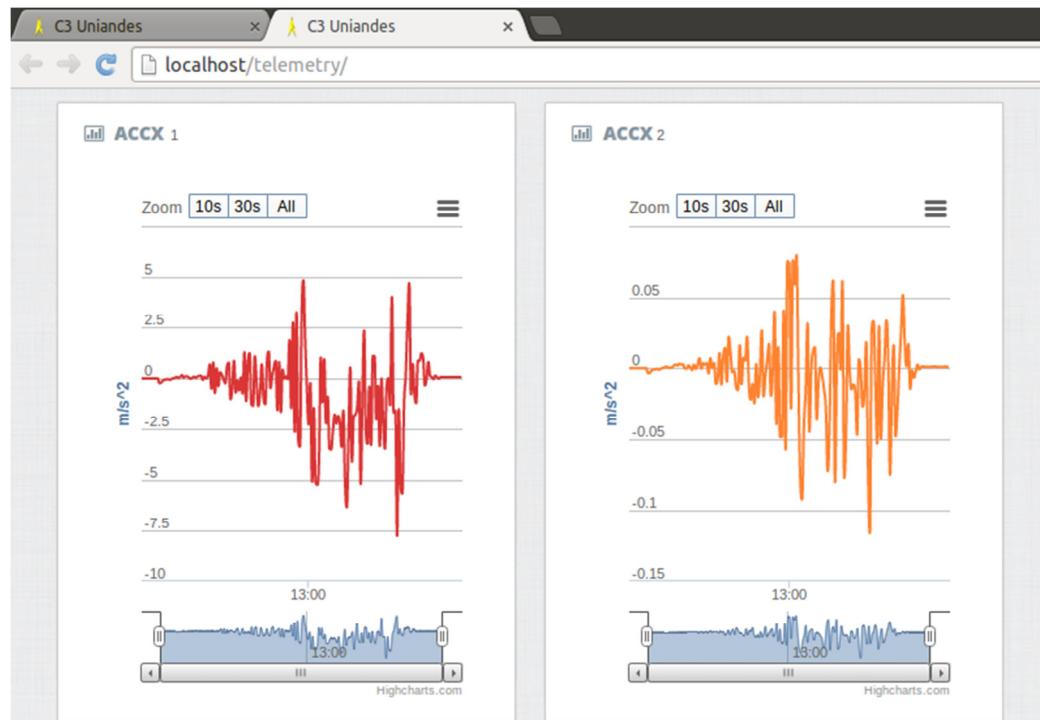


Fig. 95. Resultados filtro LP sobre telemetrías de aceleración 1, 2

Una vez realizado el cálculo de la velocidad en cada sensor los resultados obtenidos fueron los siguientes:

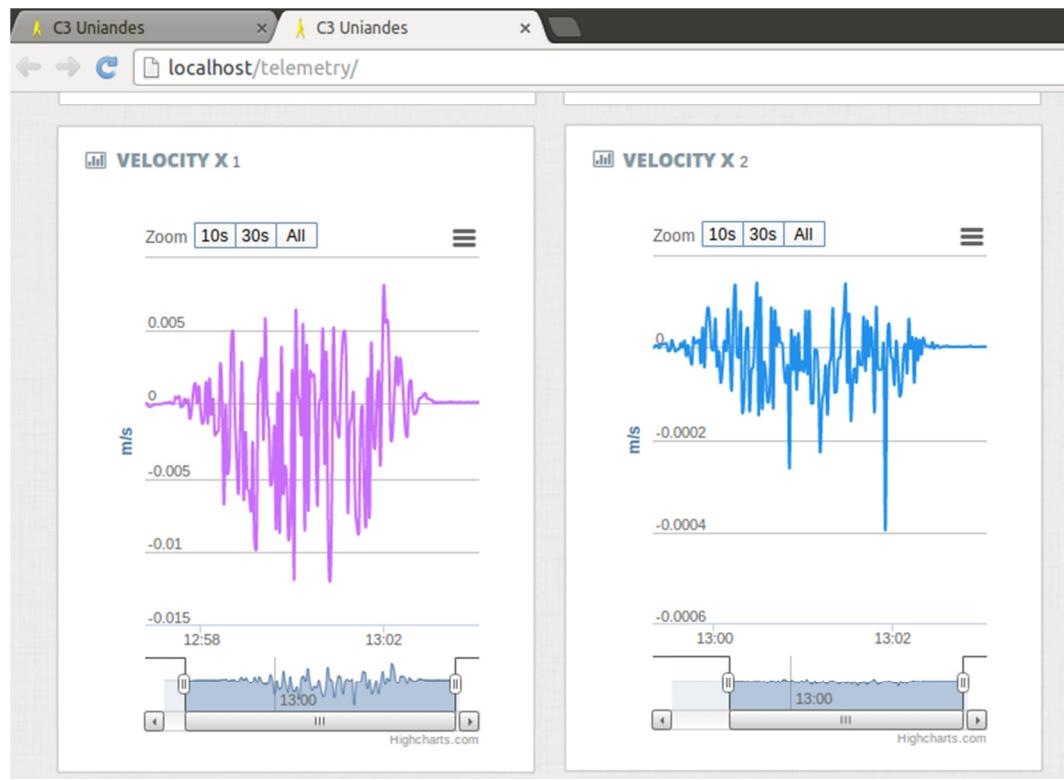


Fig. 96. Resultados de cálculo velocidad 1,2

Los resultados del cálculo del desplazamiento fueron los siguientes:

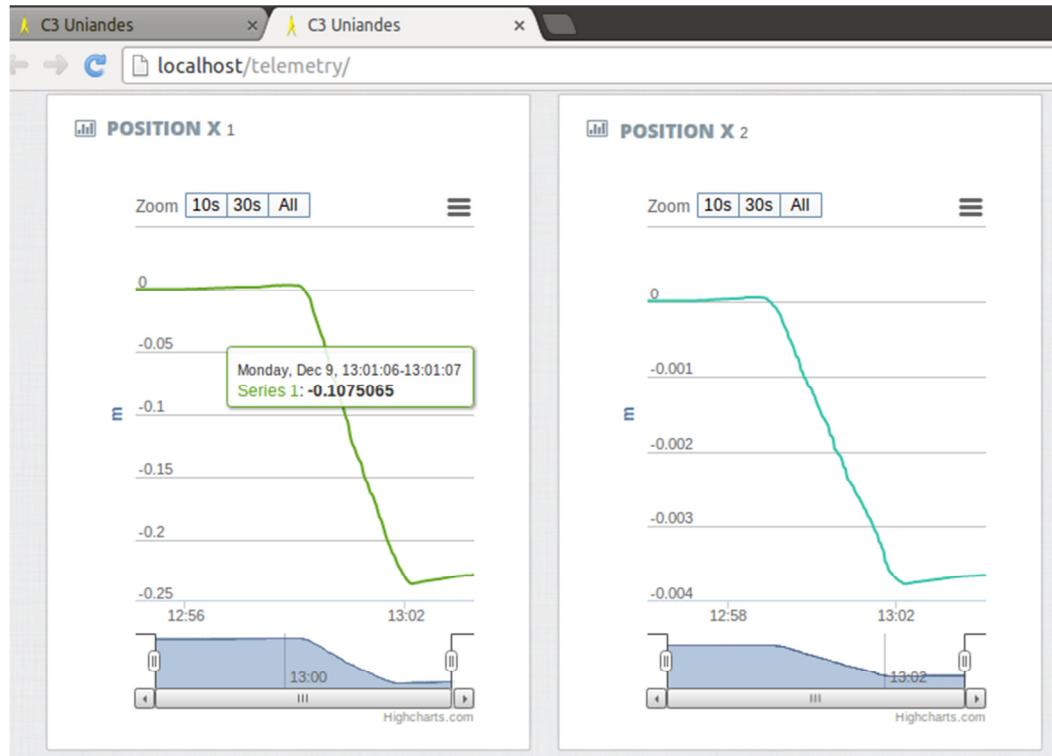


Fig. 97. Resultados de cálculo desplazamiento 1,2

## Comparación e interpretación de los resultados

Para este caso se realizó el análisis e interpretación de los datos para el primer sensor y se realizó la comparación entre los datos analizados por MATLAB y el sistema de graficación C3.

Aceleración filtrada en MATLAB:

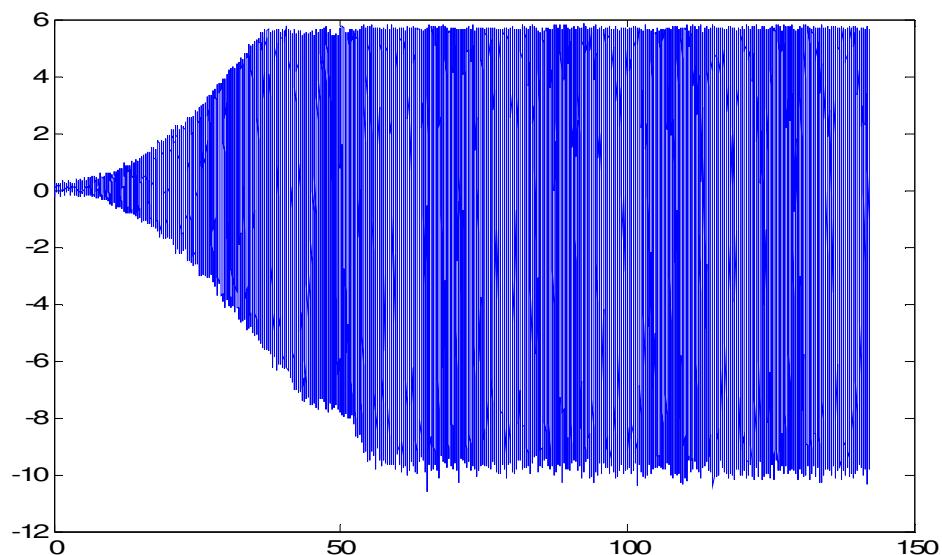


Fig. 98. Aceleración filtrada utilizando MATLAB.

Aceleración presentada en sistema C3:

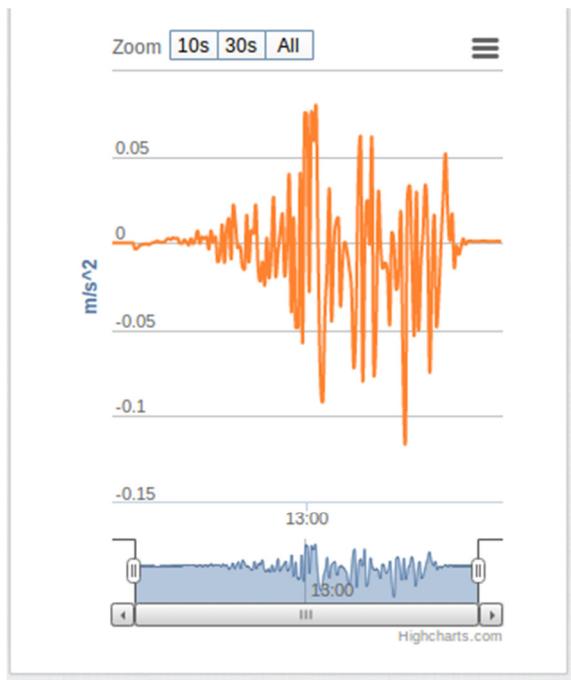


Fig. 99. Aceleración graficada en el sistema C3.

Velocidad obtenida de MATLAB:

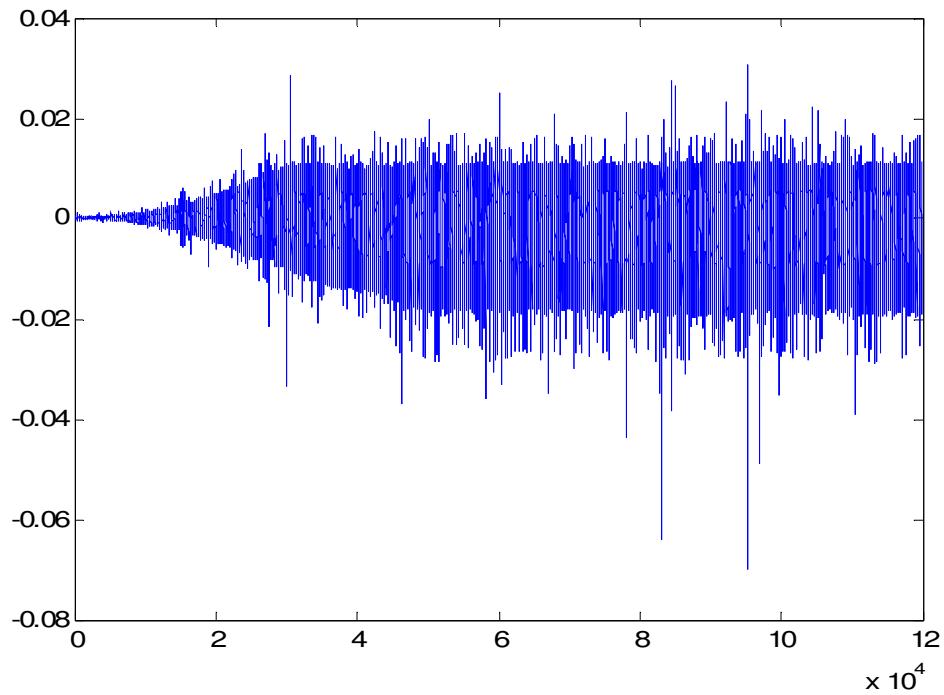


Fig. 100. Velocidad calculada utilizando MATLAB.

Velocidad presentada en sistema C3:

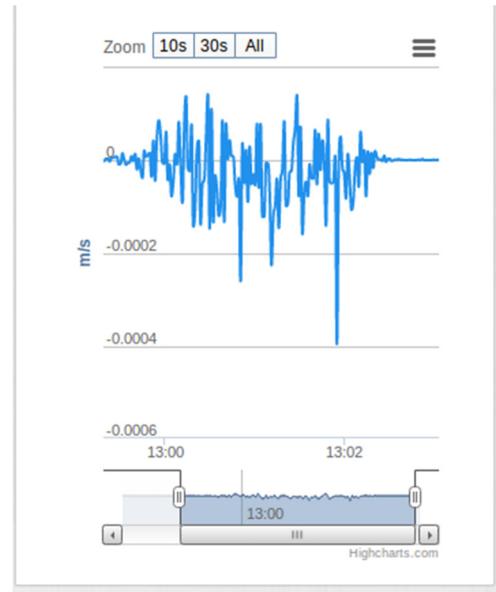


Fig. 101. Velocidad graficada en el sistema C3.

Desplazamiento obtenido de MATLAB:

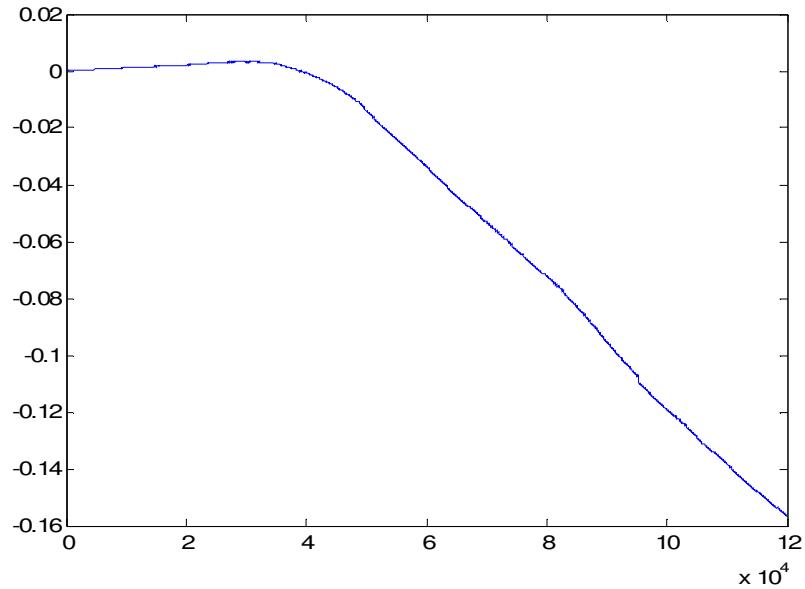


Fig. 102. Desplazamiento calculado utilizando MATLAB.

Desplazamiento presentado en sistema C3:

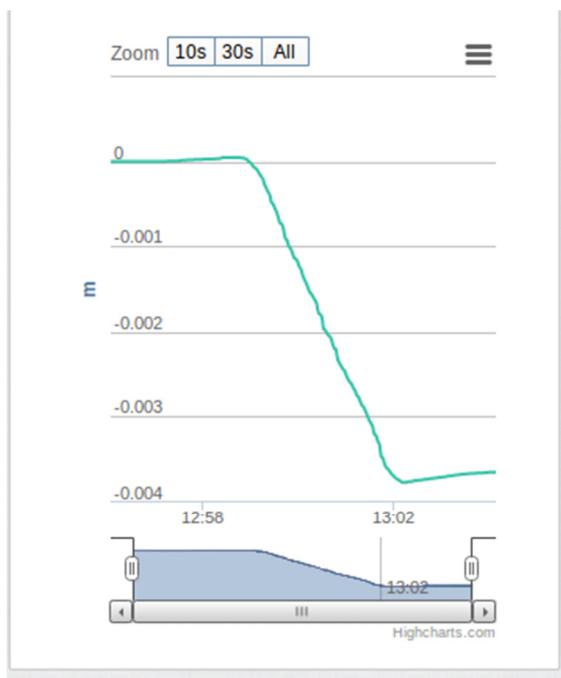


Fig. 103. Desplazamiento graficado en el sistema C3.

Como resultado de estas comparaciones, en este caso se encuentra que la información representada por el sistema C3 graficó resultados diferentes frente a los obtenidos experimentalmente en el laboratorio. En complemento a la comparación y conclusiones adicionales es necesario realizar más experimentación de una misma señal a diferentes frecuencias y determinar así el impacto de la misma en el proceso de análisis y despliegue de información.

### ***Análisis e interpretación general de los resultados***

Realizando un análisis en general de las diversas telemetrías procesadas se encontró que no es posible realizar una comparación analítica de las curvas resultantes debido a la falta de implementación de una metodología para esto, sin embargo, al realizar una

revisión a priori del comportamiento de las curvas entre los diferentes sistemas (resultados experimentales vs. resultados sistema CAT vs. resultados visualizados por sistema C3) se encontró lo siguiente:

Existe incongruencia en la escala dimensional en la que se están graficando los valores, en algunos casos se presentan diferencias de magnitud del orden 10, en algunos casos se detectó que el sistema de visualización C3 no presentó la información de escala dimensional lo que hace imposible realizar cualquier tipo de análisis y determinar la calidad de la información presentada.

Para el caso de las diferencias de magnitudes es necesario revalidar los resultados experimentales y los datos resultantes del sistema CAT para determinar si existe un problema en la lógica de programación en las funciones de filtro o análisis específico. De igual manera se debe realizar una comparación analítica de las dos curvas para garantizar la calidad de los datos, sin embargo se debe tener en cuenta que al realizar el análisis de los datos experimentales en un programa como matlab, este se realiza sobre todo el conjunto de datos a diferencia del análisis en el sistema CAT el cual los realiza frente a una ventana de datos en un tiempo determinado.

En el caso de la visualización de la información en el sistema web C3 se encontró que este componente de visualización no es capaz de graficar la información debido al alto volumen de los datos, en algunos experimentos el sistema colapsó lo que hizo imposible la posibilidad de tomar muestras de los resultados. Para esto se debe revisar la frecuencia con que el sistema CAT envía los datos al sistema C3 para enviar información a una menor tasa de frecuencia y a su vez validar que la escala dimensional con la que la información es representada es coherente y acorde con los resultados de los datos

experimentales y del sistema CAT. De igual forma es necesario encontrar el mecanismo de análisis analítico de la curva para determinar que la tendencia de la curva y los datos tienen una calidad aceptable.

### **Análisis de Ley Amdahl**

Para determinar la rapidez ganada del sistema CAT al ser implementado de forma paralela se debe determinar que porción del código no es susceptible de paralelización, para esto es necesario realizar una revisión de todas las tareas identificadas en el sistema:

Tarea	Descripción	Paralelizable?
T1	Recibir tramas de información desde el servidor C3.	NO
T2	Enviar tramas de información a tareas de análisis específico.	SÍ
T3	Recibir la trama de información.	SÍ
T4	Aplicar análisis de filtro sobre los datos.	SÍ
T5	Realizar análisis de integración instantánea.	SÍ
T6	Realizar análisis de integración acumulada.	SÍ
T7	Enviar los datos procesados al servidor C3.	SÍ

Fig. 104. Identificación de tareas paralelizables en el sistema CAT.

Encontrando la fracción de código no paralelizable se encontró que el 14.28% del código es secuencial y corresponde a la recepción de información desde el servidor C3, mientras que el 85.72% del código es paralelizable. De forma general se encuentra que el programa se ejecuta 7 veces más rápido que de forma secuencial.

$$7.002 = \frac{1}{(1 - 0.8572)}$$

Fig. 105. Cálculo de rapidez ganada en el sistema CAT según ley Amdahl.

Utilizando la fórmula general, se encuentra que el programa se ejecuta (teóricamente) 7 veces más rápido que de forma secuencial. Sin embargo, la rapidez también depende de qué tantos núcleos estén siendo usados para ejecutar las instrucciones del código paralelo.

Se realizó el mismo calculo, determinando la ganancia en rapidez para 1, 2, 4, 8, 16, 32, 64 y 128 núcleos.

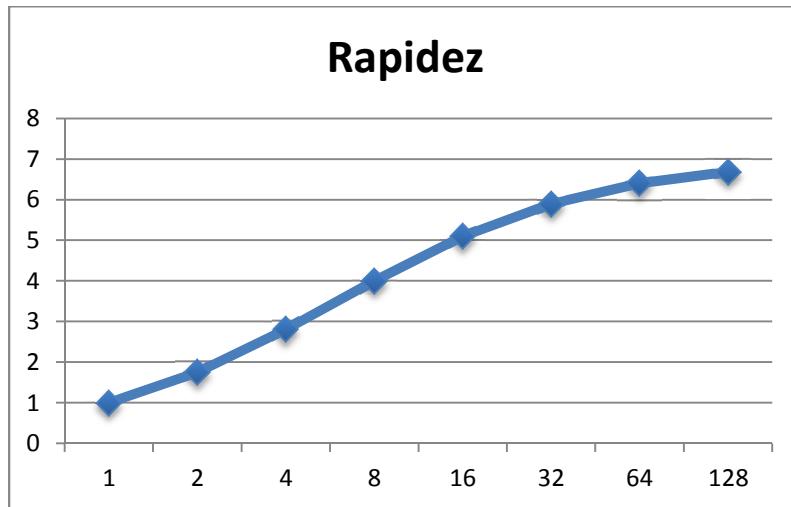


Fig. 106. Cálculo de rapidez ganada por cantidad de núcleos en el sistema CAT según ley Amdahl.

En esta gráfica se puede ver la tendencia que a mayor cantidad de núcleos se va obteniendo más rapidez, sin embargo, no se alcanzará a llegar al límite teórico pues para lograrlo se necesitaría una infinita cantidad de núcleos.

## CAPITULO VI: CONCLUSIONES

El desarrollo del trabajo se enfocó en dos áreas principales: Computación paralela y análisis de telemetrías: Para el caso de la computación paralela las conclusiones encontradas fueron:

El realizar un adecuado diseño de la solución fue determinante para lograr cumplir con las necesidades del proyecto PUA. Un primer acercamiento a la solución no fue el más adecuado puesto que no se implementó un diseño ni se tuvieron en cuenta los atributos de calidad detectados, además de no lograr una implementación de los diversos patrones de diseño en programación paralela. Al realizar un rediseño de la solución desde su inicio apoyándose en el uso de patrones de diseño, no solo se logró un mejor desempeño en la solución sino que también se logró la posibilidad de adicionar nuevos tipos de análisis, además de poder utilizar todo el clúster.

La validación de la ley Amdahl fue de gran utilidad puesto que puso en evidencia una ganancia en desempeño, por ende en tiempo y eficiencia al poder distribuir las secciones de código que eran susceptibles de paralelización.

El poder utilizar el clúster de alto desempeño para realizar este tipo de procesamiento, poder utilizar más de 8 procesadores, 128 núcleos, 256 GB en RAM y una conectividad de baja latencia da la abre las posibilidades de aumentar la complejidad de los análisis, por ejemplo: Calculo de trayectoria esperada, realizar el diseño, simulaciones de los componentes de los cohetes apoyándose en el uso de herramientas como Ansys, MATLAB, Comsol, entre otros.

Para el caso de análisis de telemetrías, las conclusiones fueron:

Es necesario realizar un trabajo de calibración y validación de los sensores a usar, además revisar la frecuencia de muestreo de la señal, debido a que los datos recolectados de los lanzamientos Ainkaa no fueron útiles, el obtener este tipo de información para realizar la validación del sistema no es fácil, además del valor de los datos como tal.

Se hizo la implementación sobre un filtro paso bajo sencillo suavizado de cuatro coeficientes, se deben realizar más pruebas diferentes tipos de filtros con más o menos coeficientes para determinar qué tanto afecta de forma positiva o negativa la señal de salida. A su vez realizar la implementación con otra clase de filtros, la literatura recomienda filtros adaptativos. Se hace en especial el uso del filtro Kalman, este filtro fue usado en el proyecto Apollo de la nasa y en la literatura se hace énfasis en el gran aporte de este en el éxito del proyecto.

Se sugiere considerar la Implementación de otro método de integración numérica que ofrezca un manejo más adecuado del error y así ofrecer un resultado más preciso en los cálculos.

Se debe realizar una revisión de integración con el sistema de visualización C3 puesto que no se logra realizar el despliegue de los gráficos con el desempeño adecuado, al comenzar a enviar más de 100 tramas por segundo al sistema de visualización este comienza a presentar problemas de congelamiento de imagen y problemas en el navegador.

## **TRABAJO FUTURO**

Realizar la implementación de un filtro adaptativo por ejemplo, Kalman. Puesto que este tipo de filtros son los más usados para sistemas GPS, sistemas aeroespaciales.

Investigar sobre la aplicación de otro algoritmo de suavizado, entre la literatura consultada se hace mucho énfasis en el método Levenberg–Marquardt.

Realizar una codificación de la solución en lenguaje de programación C++, ya existen una gran variedad de librerías y recursos muy útiles a nivel de filtros, métodos numéricos de integración entre otros.

Realizar el trabajo de obtención de la altura basandose en la presión y temperatura a través de la ecuación de atmosfera estándar.

Realizar la implementación del análisis de trayectoria esperada y realizar el recálculo de esta durante el vuelo del vehículo aeroespacial.

Adicionar los cálculos de fuerza de arrastre, coeficiente de arrastre y demás análisis que sean de utilidad para la misión y proyecto en general.

## **TRABAJOS RELACIONADOS**

Actualmente, en el área de la computación de alto desempeño y análisis de sistemas, se encuentran una gran cantidad de aplicaciones y usos, que van desde el análisis de sistemas en tiempo real, simulaciones de sistemas mecánicos, eléctricos, atmosféricos, procesamiento gráfico y análisis y optimización de sistemas de tráfico, logística, entre otros.

En todas las áreas donde tenga aplicación la implementación de aplicaciones paralelas, la distribución de los datos y el diseño es una cuestión clave que puede determinar el rendimiento y la escalabilidad. El problema de la distribución de datos suele ser manejado diferentes formas en función de la utilización de estructuras de datos [33].

Las aplicaciones en tiempo real de primera generación eran relativamente simples y no implicaban sofisticados algoritmos o una extensa y compleja capacidad funcional. En las

últimas décadas, sin embargo, ha habido una demanda mayor de recursos computacionales dado que las aplicaciones se han vuelto más complejas o de seguridad crítica, en temas como de navegación aeroespacial, seguimiento de fábricas y plantas de energía nuclear [34].

Un caso más específico en el área de aplicaciones en tiempo real y uso de sensores de telemetría se encuentran en aplicaciones militares, por ejemplo en el uso de radares, en estos sistemas los requisitos computacionales para el procesamiento de los datos obtenidos de los sensores en una red de radares surgen de la necesidad de cancelar la interferencia no deseada y mejorar la calidad de la señal. Por tal razón algoritmos digitales de procesamiento deben ser capaces de anular la interferencia y aumentar la intensidad de la señal para que el objetivo deseado en el radar pueda ser fácilmente identificado y rastreado. Gran cantidad de información obtenida de antenas adaptativas debe ser calculada en tiempo real, para cada cálculo adaptativo se requiere realizar un cálculo matemático para calcular la inversa de una matriz con una latencia de milisegundos [35].

## BIBLIOGRAFIA

- [1] Proyecto PUA. "Proyecto Uniandino Aeroespacial" pua.uniandes.edu.co. [Online]. Available: <https://pua.uniandes.edu.co> [Accessed: Oct. 18, 2013]
- [2] Documento "Centro comando y Control – PUA". 2012 – Dario Correal, Juan Sebastián Urrego, Santiago Arteaga
- [3] RAE. "Definición telemetría" rae.es. [Online]. Available: <http://lema.rae.es/drae/?val=telemetr%C3%ADA> [Accessed: Nov. 27, 2013]
- [4] Universidad Politecnica Salesiana. "Conceptos de telemetría, telemando y red GSM" ups.edu.ec. [Online]. Available: <http://www.dspace.ups.edu.ec/bitstream/123456789/418/Capitulo2.pdf> [Accessed: Nov. 16, 2013]
- [5] Mellanox. "High Performance Computing" mellanox.com. [Online]. Available: [http://www.mellanox.com/page/hpc\\_overview](http://www.mellanox.com/page/hpc_overview) [Accessed: Sept. 27, 2013]
- [6] Karniadakis G., and Kirby R. Parallel Scientific Computingin C++ and MPI. A seamless approach to parallel algorithms and their implementation. Cambridge University Press, 2003
- [7] Ortega-Arjona, Jorge Luis (2010-04-20). Patterns for Parallel Software Design (Wiley Software Patterns Series) (Kindle Locations 6224-6226). Wiley.
- [8] Rauber, T.; Rünger, G. ; Parallel Programming for Multicore and Clusters Systems; Springer, 2012; pp, 148
- [9] Ortega-Arjona, Jorge Luis (2010-04-20). Patterns for Parallel Software Design (Wiley Software Patterns Series) (Kindle Locations 6138-6140). Wiley. Kindle Edition.
- [10] McCool, Michael; Reinders, James; Robison, Arch (2012-07-31). Structured Parallel Programming: Patterns for Efficient Computation (Kindle Locations 9228-9229). Elsevier Science. Kindle Edition
- [11] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patrones para la programación paralela (Serie Patrones de Software) (Ubicaciones Kindle 692-699). Pearson Education. Versión Kindle
- [12] Infiniband Trade Association. "About Infiniband" infinibandta.org. [Online]. Available: [http://www.infinibandta.org/content/pages.php?pg=about\\_us\\_infiniband](http://www.infinibandta.org/content/pages.php?pg=about_us_infiniband) [Accessed: Sept. 16, 2013]
- [13] University of Cambridge. "Basics of Digital Filters - Signal Processing and Communications Laboratory" sigproc.eng.cam.ac.uk. [Online]. Available: [http://www-sigproc.eng.cam.ac.uk/~op205/3F3\\_4\\_Basics\\_of\\_Digital\\_Filters.pdf](http://www-sigproc.eng.cam.ac.uk/~op205/3F3_4_Basics_of_Digital_Filters.pdf) [Accessed: Oct. 7, 2013]
- [14] Proyecto PUA."Qué hacemos?" pua.uniandes.edu.co. [Online]. Available: <https://pua.uniandes.edu.co/doku.php?id=principal:hacemos> [Accessed:Oct. 4, 2013]
- [15] Proyecto PUA. "Quienes somos" pua.uniandes.edu.co. [Online]. Available: <https://pua.uniandes.edu.co/doku.php?id=principal:somos> [Accessed: Sept. 8, 2013]
- [16] Proyecto PUA. "MISIÓN SÉNECA VIII" pua.uniandes.edu.co. [Online]. Available: <https://pua.uniandes.edu.co/doku.php?id=misiones:mision15> [Accessed: Aug. 11, 2013]
- [17] Qiu, Judy and Bae, SH. Performance of Windows Multicore Systems on Threading and MPI Concurrency and Computation- Practice & Experience, 2012 Volume: 24 Issue: 1 Special Issue: SI Pages: 14-2
- [18] Ferreira, Kettmann, Thomasch, Silcoks, Chen, Daunois, Ihamo, Harada, Hill, Bernocchi and Ford. Linux HPC Clúster Installation, IBM. June 2011.
- [19] Jin H., Jespersen D., Mehrotra P., Biswas R., Huang L. and Chapman B. Michael and Potter, Kurt. High performance computing using MPI and OpenMP on multi-core. Emerging Programming Paradigms for Large-Scale Scientific Computing Volume 37, Issue 9: 562-575, September 2011.
- [20] Boukerche A., Al-Shaikh R., and Notare M. Towards highly available and scalable high performance clusters, Journal of Computer and System Sciences 73:1240–1251,2007

- [21] Infiniband Trade Association. "Resources" [infinibandta.org](http://www.infinibandta.org). [Online]. Available: <http://www.infinibandta.org/> [Accessed: Aug. 12, 2013]
- [22] Grun P., Introduction to InfiniBand for End Users. Infiniband trade association. 2010.
- [23] Schwickerath U. and Andreas H. InfiniBand-Experiences at the Forschungszentrum Karlsruhe, Nuclear Instruments and Methods in Physics Research A 559:81–84, 2006.
- [24] Fagg G., and Dongarra J., Building and Using a Fault-Tolerant MPI Implementation. International Journal of High Performance Computing Applications 18: 353, 2004.
- [25] Ron Brightwell R., Hemmert K., Murphy R., Rodrigues A, and Underwood K. A Hardware Acceleration Unit for MPI Queue
- [26] Gangadharappa T., Koop M., and Panda D.K. Designing and Evaluating MPI-2 Dynamic Process Management Support for InfiniBand. Parallel Processing Workshops, 2009. ICPPW '09. International Conference. Page(s):89 - 96, 2009
- [27] MPI Forum "Documentation" [mpi-forum.org](http://www.mpi-forum.org). [Online]. Available: <http://www.mpi-forum.org/docs/docs.html> [Accessed: Sept. 19, 2013]
- [28] Dongarra J., Kennedy K. and White A. The Sourcebook of Parallel Computing. Morgan Kaufmann, Chapter 1. 2012
- [29] Lawrence Livermore National Laboratory. "Introduction to Parallel Computing" [llnl.gov](https://computing.llnl.gov/tutorials/parallel_comp/#Whatis). [Online]. Available: [https://computing.llnl.gov/tutorials/parallel\\_comp/#Whatis](https://computing.llnl.gov/tutorials/parallel_comp/#Whatis) [Accessed: Sept. 19, 2013]
- [30] Correal D., and López N. Arquitectura de Software course. "Atributos de Calidad y Perspectivas" [sistemas.uniandes.edu.co](http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-atributosalidad.pdf). [Online]. Available: <http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-atributosalidad.pdf> [Accessed: Oct. 21, 2013]
- [31] Correal D., Arquitectura de Software course notes. "Atributos de Calidad" [sistemas.uniandes.edu.co](http://sistemas.uniandes.edu.co/~isis2503/dokuwiki/lib/exe/fetch.php?media=principal:modulo3-atributosdecalidad.pdf). [Online]. Available: <http://sistemas.uniandes.edu.co/~isis2503/dokuwiki/lib/exe/fetch.php?media=principal:modulo3-atributosdecalidad.pdf> [Accessed: Oct. 21, 2013]
- [32] Mellanox Technologies. Introduction to Infiniband. White Paper. 2003
- [33] Fresno J., Gonzalez-Escribano A., and Llanos D. Extending a hierarchical tiling arrays library to support sparse data partitioning. J Supercomput (2013) 64:59–68.
- [34] Adnan Sherif A., Cavalcanti A., Jifeng H., and Sampaio A. A process algebraic framework for specification and validation of real-time systems. Formal Aspects of Computing (2010) 22: 153–191
- [35] David R. Martinez D. Application of Parallel Processors to Real-Time Sensor Array Processing. Parallel Processing, 1999. 13th International and 10th Symposium on Parallel and Distributed Processing, Page(s):463 - 469, 1999
- [36] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patterns for Parallel Programming (Software Patterns Series) (Kindle Locations 6979-6981). Pearson Education. Kindle Edition
- [37] Ortega-Arjona, Jorge Luis (2010-04-20). Patterns for Parallel Software Design (Wiley Software Patterns Series) (Kindle Locations 6331-6333). Wiley. Kindle Edition.
- [38] T. Rauber and G. Rünger, *Parallel Programming*, 287 DOI: 10.1007/978-3-642-37801-0\_6, © Springer-Verlag Berlin Heidelberg 2013 288 6 Thread Programming
- [39] T. Rauber and G. Rünger, *Parallel Programming*, 287 - DOI: 10.1007/978-3-642-37801-0\_6, © Springer-Verlag Berlin Heidelberg 2013 - 288 6 Thread Programming
- [40] Subhash Saini, Robert Ciotti, Brian T.N. Gunney, Thomas E. Spelce, Alice Koniges, Don Dossa, Panagiotis Adamidis, Rolf Rabenseifner, Sunil R. Tiyyagura, Matthias Mueller, Performance evaluation of supercomputers using HPCC and IMB Benchmarks, Journal of Computer and System Sciences, Volume 74, Issue 6, September 2008, Pages 965-982
- [41] Rabenseifner R. ; “Hybrid Parallel Programming on HPC Platforms” in proceedings of the Fifth European Workshop on OpenMP, EWOMP '03, Aachen, Germany, Sept. 22-26, 2003

- [42] William Gropp W.; Rajeev Thakur R.; Issues in Developing a Thread-Safe MPI Implementation; Recent Advances in Parallel Virtual Machine and Message Passing Interface Lecture Notes in Computer Science Volume 4192, 2006, pp 12-21
- [43] University of Wisconsin Madison. "Asynchronous vs Synchronous Communication" wisc.edu. [Online]. Available: <http://academictech.doit.wisc.edu/blend/facilitate/communicate> [Accessed: Dec. 1, 2013]
- [44] Lawrence Livermore National Laboratory. llnl.gov. [Online]. Available: [https://computing.llnl.gov/tutorials/parallel\\_comp/#DesignCommunications](https://computing.llnl.gov/tutorials/parallel_comp/#DesignCommunications) [Accessed: Oct. 15, 2013]
- [45] National Computational Infrastructure. "Blocking Operations" nci.org.au. [Online]. Available: <http://nf.nci.org.au/training/MPIProg/slides/slides.060.html> [Accessed: Oct. 13, 2013]
- [46] National Computational Infrastructure. "Non Blocking Operations" nci.org.au. [Online]. Available: <http://nf.nci.org.au/training/MPIProg/slides/slides.061.html> [Accessed: Oct. 13, 2013]
- [47] Amdahl GM. Validity of the single-processor approach to achieving large scale computing capabilities. In: Proceedings of the American Federation of Information Processing Societies Spring Joint Computer Conference. Montvale, NJ: AFIPS Press; 1967;483– 485.
- [48] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patterns for Parallel Programming (Software Patterns Series) (Kindle Locations 616). Pearson Education. Kindle Edition
- [49] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patterns for Parallel Programming (Software Patterns Series) (Kindle Locations 617). Pearson Education. Kindle Edition
- [50] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patterns for Parallel Programming (Software Patterns Series) (Kindle Locations 618). Pearson Education. Kindle Edition
- [51] Mattson, Timothy G.; Sanders, Beverly A.; Massingill, Berna L. (2004-09-15). Patterns for Parallel Programming (Software Patterns Series) (Kindle Locations 619). Pearson Education. Kindle Edition
- [52] IBM. "HPC architecture and scaling" ibm.com.[Online]. Available: <http://www.ibm.com/developerworks/cloud/library/cl-ind-cloud-e-learning2/Ahmdahl-law.jpg> [Accessed: Dec. 1, 2013]
- [53] Universitat Pompeu Fabra. upf.edu.[Online]. Available: <http://www.dtic.upf.edu/~egomez/teaching/sintesi/SPS1/Tema7-FiltrosDigitales.pdf> [Accessed: Nov. 25, 2013]
- [54] The Scientist and Engineer's Guide to Digital Signal Processing. "Introduction to Digital Filters" dspguide.com. [Online]. Available: <http://www.dspguide.com/ch14/1.htm> [Accessed: Oct. 17, 2013]
- [55] Practical Digital Signal Processing for Engineers and Technicians Edmund Lai PhD, BEng; Lai and Associates, Singapore
- [56] National Chiao Tung University. "Adaptive Signal Processing". nctu.edu.tw. [Online]. Available: <http://cwww.ee.nctu.edu.tw/course/asp/ASP01.pdf> [Accessed: Nov. 25, 2013]
- [57] Purdue University. "Digital Filter Design" purdue.edu.[Online]. Available: [https://engineering.purdue.edu/~bouman/ece438/previous/Spring99/lecture/module\\_1/1.7\\_dig\\_filter\\_design/1.7.3\\_iir\\_filter\\_design.pdf](https://engineering.purdue.edu/~bouman/ece438/previous/Spring99/lecture/module_1/1.7_dig_filter_design/1.7.3_iir_filter_design.pdf) [Accessed: Aug. 8, 2013]
- [58] Universitat Pompeu Fabra. "Introducción al filtrado digital" upf.edu. [Online]. Available: <http://www.dtic.upf.edu/~egomez/teaching/sintesi/SPS1/Tema7-FiltrosDigitales.pdf> [Accessed: Nov. 5, 2013]
- [59] Lai E.; Practical Digital Signal Processing; Newnes; 2003
- [60] Karma-lab. wikidot.com. [Online]. Available: <http://karma-lab.wikidot.com/korg-m3:an-introduction-to-the-korg-m3-sound-engine> [Accessed: Nov. 9, 2013]
- [61] Freescale. freescale.com. [Online]. Available: [http://www.freescale.com/files/sensors/doc/app\\_note/AN3397.pdf](http://www.freescale.com/files/sensors/doc/app_note/AN3397.pdf) [Accessed: Nov. 24, 2013]
- [62] Microsoft. microsoft.com. [Online]. Available: <http://support.microsoft.com/kb/551080/es> [Accessed: Dec. 4, 2013]

- [63] Nicktips. nicktips.files.wordpress.com.[Online]. Available: <http://niktips.files.wordpress.com/2010/01/ethernet-vs-infiniband1.jpg> [Accessed: Nov. 29, 2013]
- [64] University of Maryland. umd.edu.[Online]. Available: <http://terpconnect.umd.edu/~toh/spectrum/Smoothing.html> [Accessed: Dec. 14, 2013]
- [65] International Society of Biomechanics. isbweb.org.[Online]. Available: <http://isbweb.org/software/sigproc/bogert/filter.pdf> [Accessed: Dec. 7, 2013]
- [66] The University of Arizona. arizona.edu.[Online]. Available: [http://www.ltrr.arizona.edu/~dmeko/notes\\_8.pdf](http://www.ltrr.arizona.edu/~dmeko/notes_8.pdf) [Accessed: Dec. 8, 2013]
- [67] Miguel R. Hernandez-Garcia, Sami F. Masri, Roger Ghanem, Eloi Figueiredo, Charles R. Farrar, An experimental investigation of change detection in uncertain chain-like systems, Journal of Sound and Vibration, Volume 329, Issue 12, 7 June 2010, Pages 2395-2409
- [68] David Schultz. earthlink.net. [Online]. Available: <http://home.earthlink.net/~david.schultz/> [Accessed: Nov. 29, 2013]
- [69] David Schultz. earthlink.net. [Online]. Available: <http://home.earthlink.net/~david.schultz/level3/erint/construction.html> [Accessed: Dec. 11, 2013]
- [70] RDAS aedelectronics.nl.[Online]. Available: <http://www.aedelectronics.nl/rdas/> [Accessed: Dec. 10, 2013]