Venus Rover Framework

La NASA ha decidido abrir una oficina en Ecuador y está reclutando las mejores mentes de ESPOL. Para esto ha creado un “challenge” que consiste en diseñar e implementar el framework de procesamiento de datos de su futura misión al planeta Venus. El framework va a ser evaluado por expertos y la persona que obtenga el mejor resultado será contratada por NASA.

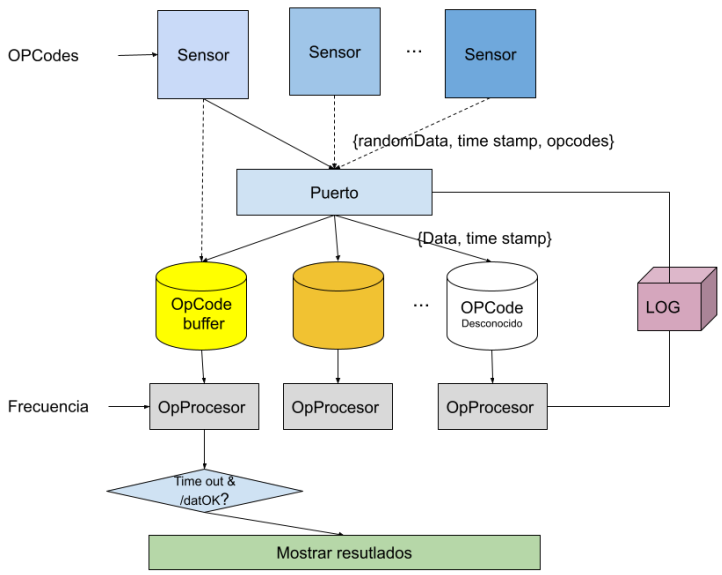
Las metas de diseño que la NASA ha establecido para el framework son las siguientes:

Autocontenido, así un solo ejecutable que pueda transmitirse desde la tierra
Espacio de almacenamiento mínimo, mientras menos bytes en el disco duro mejor.
Alto rendimiento, el framework debe aprovechar los procesadores existentes en el rover
Depurable, el framework debe ser capaz de almacenar datos para revisión en caso de falla.
Resiliente, el framework debe considerar los posibles problemas y fallas del resto del hardware del rover.

Dado que el hardware del rover todavía no existe, el framework debe ser capaz de ser simulado en un computador. Los sensores se simularán con un programa que utilice protocolos de internet (IP) empleando conexiones de red locales.

1. Requerimientos

El esquema general de funcionamiento del rover es el siguiente:



Sensores

Uno o varios sensores generan paquetes de datos, que contiene la siguiente trama de datos “[codigos_de_operaciones], dato, marca de tiempo;”. Donde los codigos_de_operaciones son una colección de números enteros que indican en qué operaciones se utiliza el dato, la marca de tiempo es el tiempo en unix time en el que se generó el dato y el dato es un valor entero positivo entre 0 y 255. La trama se debe representar como un string. Ej:

mensaje= [2,4,9],99,1095379198.75;

Significa que llegó el dato (99) se utiliza en las operaciones 2,4,9 y que fue generado en 1095379198.75

La conexión con los sensores se debe manejar de forma que si un sensor se desconecta, o deja de transmitir datos por un periodo *Y* de tiempo la falla quede registrada en el LOG del sistema. Para simular estos casos, debe existir un mecanismo para generar la pausa, reactivación, terminación correcta y terminación súbita de un sensor.

OPCodeBuffers

Representan el almacenamiento temporal de los datos a ser procesados en cada operación, del ejemplo anterior, una copia del dato 99 va a permanecer almacenado en los buffers correspondientes a las operaciones 2,4 y 9 hasta que sean leídos por el OPProcesor. Como no hay control sobre el volumen de datos a recibir el buffer va a tener un tamaño limitado de 20 mensajes. Si el buffer está lleno el último mensaje reemplaza al mensaje más antiguo.

OPProcesors

Son rutinas que extraen datos del buffer y ejecutan operaciones matemáticas sobre los datos a una frecuencia *Fp*. La frecuencia de ejecución de la rutina es un parámetro de entrada para cada OPProcesor. Las operaciones se identifican con un número único, si llega un mensaje y la operación solicitada no existe se debe crear un OPProcesor de forma dinámica.

Si le dejan de llegar datos por *X* minutos el OPProcesor debe terminar y liberar recursos. Para que el cálculo se pueda realizar los datos deben ser válidos. Los datos son válidos si el tiempo entre los mensajes a procesar es menor que *X/20.* Si se excede este tiempo los datos no son válidos y se descartan.

El OPProcesor debe notificarse cuando se inicializa, se realiza un cálculo exitoso y cuando ocurren errores, las notificaciones se deben guardar en el log del SO. Para simplificar el problema en todos los OPProcesor el cálculo que se realiza es el promedio de los datos.

LOG

Para efectos de depuración el framework debe trabajar en modo real-time y modo-debug, en el primer modo solo se visualizan los resultados de los cálculos en la consola donde se ejecuta el framework.. En el segundo, se agregan las notificaciones y se guardan en el LOG. Se debe usar el syslog para manejar el envío de mensajes al log.

2. Entregables (mínimos)

1. Programa en c, sensor.
2. Programa(s) en c, framework.
3. Documento PDF con diseño
4. Scripts

3. Rúbrica

Caracteristica	**Puntaje**
¿Se diseñó el framework de forma que se minimice el tiempo de procesamiento de datos?	25
¿El framework genera resultados adecuados incluso en caso de fallas?	20
¿Los procesos/hilos cumplen con los requerimientos funcionales utilizando técnicas de manejo de procesos/hilos?	20
¿Los datos se transfieren, almacenan y procesan correctamente?	15
¿El código está bien estructurado y escrito correctamente?	10
¿Se explota el existente HW en un computador	10
Se agregaron scripts de compilación/prueba	0
Se incluyó diagrama de despliegue y explicación del diseño (2 paginas max)	0
TOTAL	100
Los parámetros están “quemados en el código”	-25
Se usan librerias mágicas (incluye wrappers, etc) que no viene en un sistema POSIX compatible.	-40
No entrega documento de diseño	-50
No entrega scripts de instalacion y/o prueba	-30
Es un programa secuencial que no emplea las técnicas vistas en el curso	-100