Protocolo de información sobre el estado de carga computacional de clusters y/o host's debilmente acoplados

Ferrari, Alejandro Jesús
Taus, Santiago
Aranda, Jesús María
Campetella, Sebastián Edgardo

21 de febrero de 2012

Índice

1. Objetivo general

Administrar en alta disponibilidad la información concerniente de los recursos de cada host participante en un cluster y/o host debilmente acoplados; Con el fin de facilitar y optimizar la toma de decisión de los sistemas de software que requieran uno o más recursos en particular

1.1. Objectivos específicos

- Crear un protocolo específico de comunicación para la adquisición de información y datos de control
- Crear los clientes/servidores para el manejo del protocolo y prestación de la información

1.2. Justificación

Debido a la tendencia que marca la computación en la nube y el uso de cluster en contra partida de equipos stand alone más potentes; esta es un area de interes aún creciente. Una vez establecida la idea y analisis, se encontró en el estado del arte que existe en particular un protocolo llamado Ganglia, que resuelve de forma similar y en otros disímil y/o están ausentes parte de ellos que aquí se contemplan y viceversa.

1.3. Alcance

El alcance del objetivo es lograr implementar el protocolo de aplicación para:

- Informar la carga de CPU,RAM,SWAP y mantener la tabla de información del cluster
- Mantener el cluster operable para informar la disponibilidad

Este protocolo de aplicación se centra en el intercambio de mensajes entre un host servidor y uno o mas host/s cliente/s.

El funcionamiento basico es el siguiente:

El protocolo permite transmitir mensajes entre los extemos cliente y servidor. Estos mensajes están estructurados para que pueda ser facilmente escalable. Su propósito es mantener información de los recursos de cada host de un cluster en otro host servidor, sean cuales sean estos recursos (tráfico de red, HD's, CPU, RAM, SWAP, dispositivos acoplados, etc) su extensión en cuanto a recursos no es un limitante. Para esto se emplea un esquema de cliente-servidor (aunque como hemos dicho que es escalable, tambien se ha pensado para que otro host cliente por alguna eventualidad puede tomar el rol servidor como plan de contingencia, no contemplado en los

objectivos del alcance pero si de la idea original). En esta presentación solo se limitara a ocuparse de los objetivos antes mencionados(no necesariamente todos los host/s deben estar in situ). A los mensajes lo acompañan un codigo de operación y la longitud del mensaje, el motivo del código de operación es generar un comportamiento/interpretación (control o ejecución de un comando) en el tratamiento del mensaje. Este protocolo de aplicación utiliza como protocolo subyacente al protocolo UDP (User Data Protocol)

2. Arquitectura Cliente-Servidor

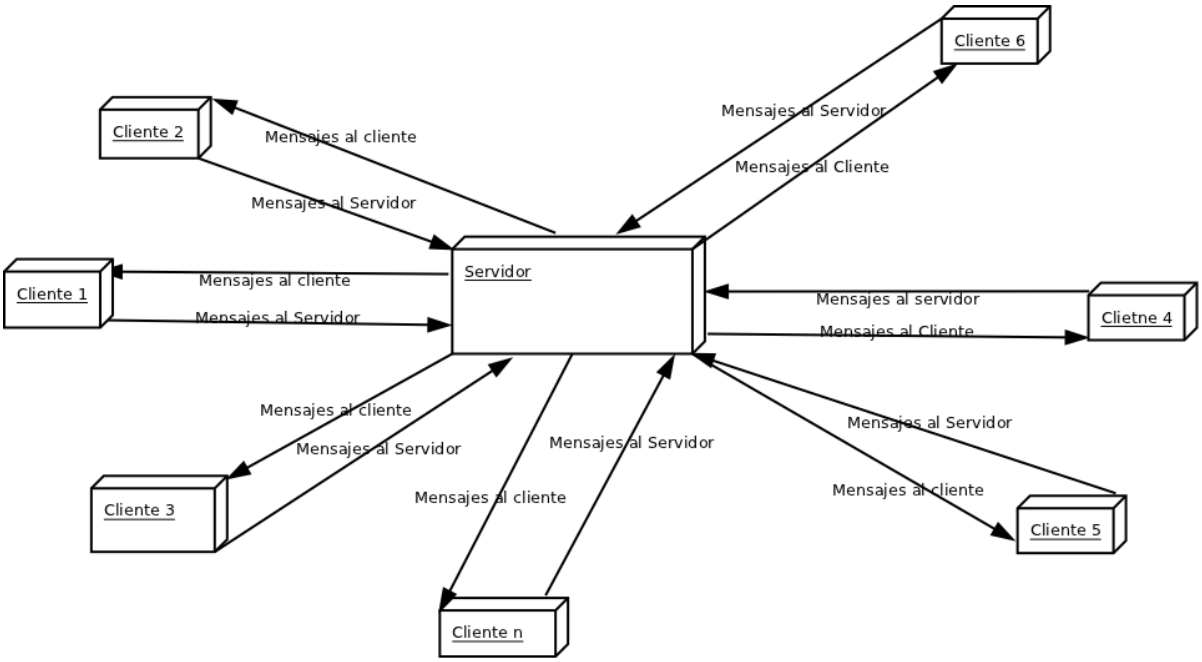

Figura 1: Arquitectura Cliente-Servidor

3. Protocolo de aplicación sobre UDP

Aclaracion:

- No se calcula el checksum dado que UDP ya calcula el checksum incluído los datos.
- 1 Byte de Operaciones
- 2 Bytes de longitud del mensaje. Esta longitud la usamos como delimitador de mensaje.

Mensaje

OPERACIONES	LONGITUD DEL MENSAJE
DATOS	

Cuadro 1: Estructura del protocolo

Estructura del protocolo

Nombre = 'protocolo de informacion de carga de recursos de un host' en ingles 'protocol information resource load of a host'. siglas en ingles pirlh asi llamaremos la estructura

```
struct pirlh {
  char operation; // 1 byte para las operaciones
  int16 leng; // 2 bytes para la longitud del mensaje
  char msg[2000]; // cadena de caracteres
}
```

3.1. Código de operaciones del protocolo

Campo operación (comando) del protocolo de longitud de 8 bits = 1 Bytes

20 = Servidor responde al nodo activo con la configuracion
21 = Servidor envia un eco
22 = Servidor envia señal que el numero unico del host cliente no corresponde y le pide desconectarse

01 = Cliente envia señal de conexión por multicast
02= Cliente responde 'eco' al servidor
03 = Cliente envia recursos al servidor

4. Servidor

Funciones del servidor

- Recibe por medio de multicast solicitud de nodos activos. el cliente almacena la IP junto con demas datos del servidor
- Enviar tiempo de recepcion de datos configurable en segundos y/o milisegudos (cuando conecta)
- Enviar un eco como pregunta de si esta activo
- Sincronizar tiempo en 5,10,etc milisegundos o segundos el envío de información de los nodos con un Δt de tolerancia
- controla que los host tengan un ID_HOST_CLIENTE_UNICO para una ip, se usa la dirección MAC
- Mantiene una lista de nodos activos.

Servicio a las aplicaciones

- Listar nodos segun un orden de preferencia (cpu,ram,disco) $->$ archivos xml separados por recurso y en forma descendente.
- Verificar si un nodo esta activo (para que la aplicación pueda enviarle al nodo lo que necesita; Lo hace automaticamente)

Recursos

- Archivo de configuracion con:
  - Tiempo de sincronización
  - Directorio de almacenamiento de recursos

5. Cliente

Funciones del Cliente

- Hace multicast preguntando quien es el servidor
- Conectarse con el servidor una vez que responda a la pregunta de nodo activo 'eco'
- Enviar al servidor la información de recursos del nodo (CPU,RAM,HD).
- Sincronizarse con el Servidor en el tiempo establecido anteriormente por el Servidor
- responder KeepAlive (un eco propio, no es el keepAlive del socket)
- Establece su ID_HOST_CLIENTE_UNICO con la dirección MAC

Recursos a tener en cuenta lado cliente

- Host Servidor = IP
- ID Host = ID_Unico (dado por el servidor, sirve para que dos maquinas no puedan tener la misma ip y si tienen que se distingan cual de ellas es, o pone en fin enviando un mensaje para que se repare este error)
- Tiempo dado por el servidor para enviar los datos
- RAM, HD, CPU

6. Diagramas

6.1. Maquina de estados finitos del protocolo

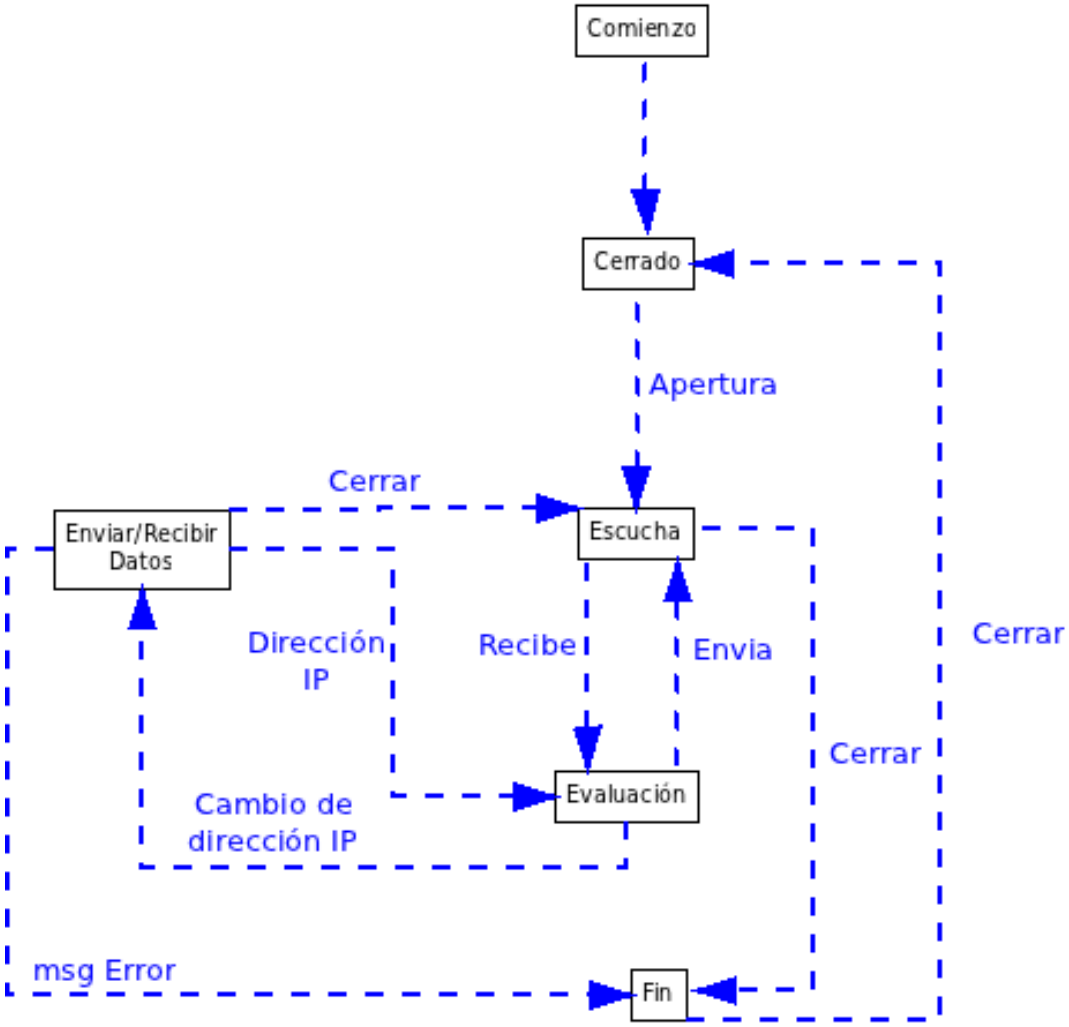

Figura 2: Maquina de estados finitos

6.2. Diagramas de flujos Cliete - Servidor

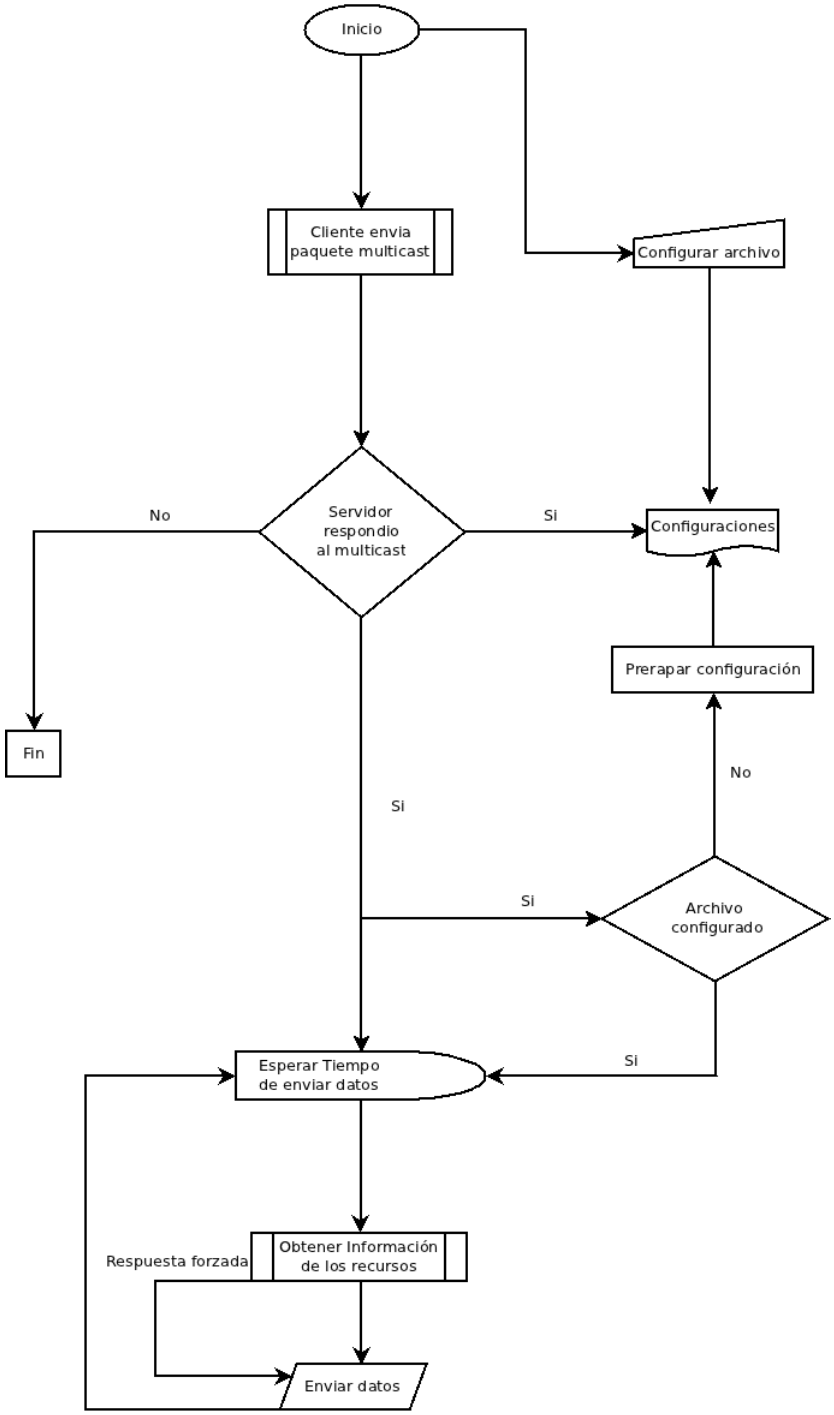

Figura 3: Diagrama de flujo del Cliente

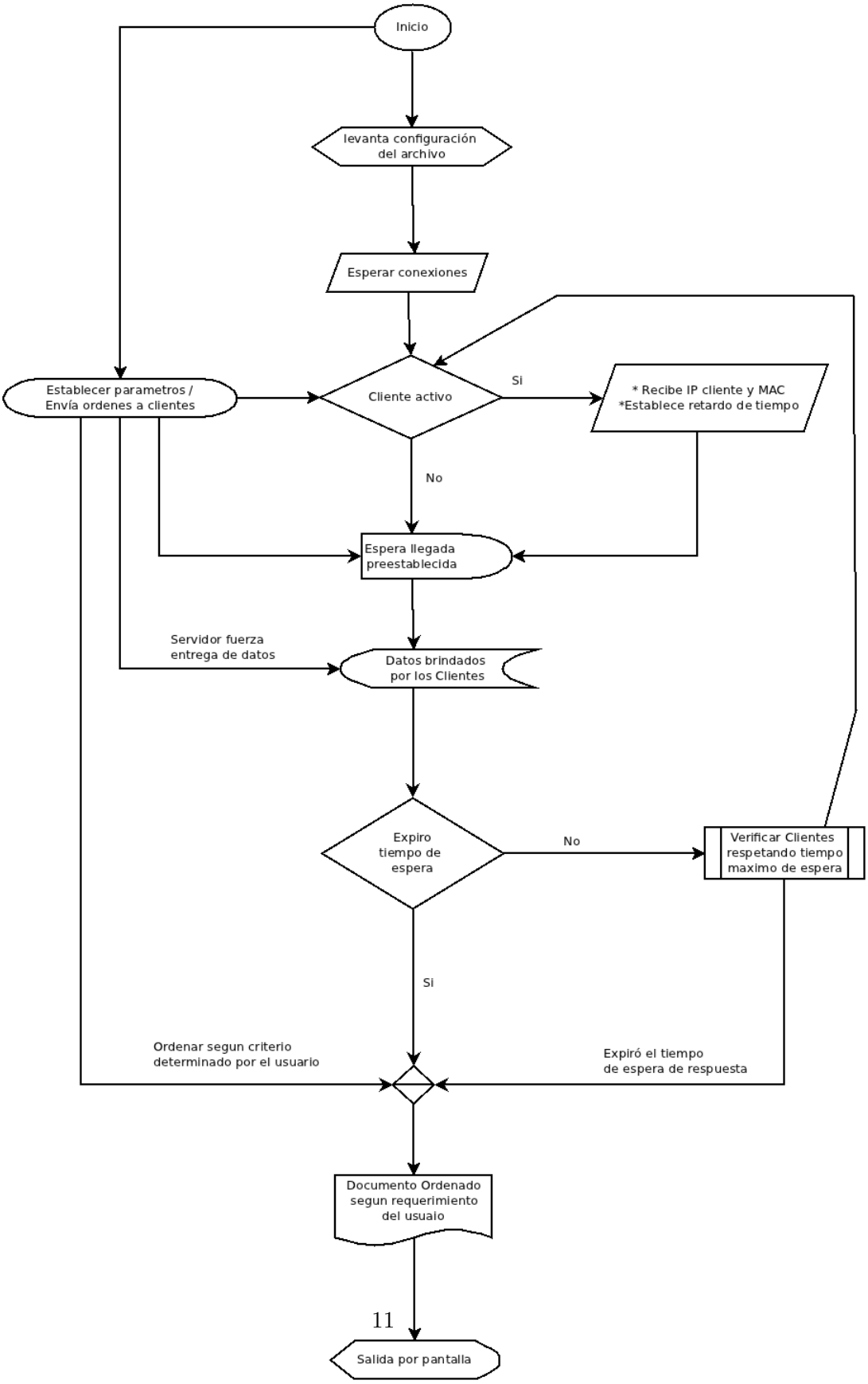

Figura 4: Diagrama de flujo del Servidor

7. Anexo I

7.1. Bases para entender XML

http://flanagan.ugr.es/xml/documento.htm

7.2. Librerias a usar

- **LIBGTOP**
  Paquetes desde el repositorio de Ubuntu
  - libgtop2.*
  - libgtop2.*-dev

  Para compilar es asi: EJ: gcc cpuinf.c -Wall -o cpuinf ‘pkg-config –cflags –libs glib-2.0 libgtop-2.0‘
- **LIBXML2**
  Paquetes desde el repositorio de Ubuntu
  - libxml2
  - libxml2-dev

  Sino se encuentran
  apt-get install libxml2 libxml2-dev

  Ejecutar manualmente estas órdenes:
  wget ftp://xmlsoft.org/libxml2/libxml2-sources-2.7.7.tar.gz
  tar xzf libxml2-sources-2.7.7.tar.gz
  cd libxml2-2.7.7
  ./configure –prefix=$HOME/libxml2-2.7.7-bin
  make
  make install

- **Como usar libxml**
  http://www.calcifer.org/documentos/librognome/xml.html
- **Compilar con la libreria**
  gcc -o enXML enXML.c -I/usr/include/libxml2/ -lxml2 -lz -lm