Sistemas Distribuidos

Modelo de Computo BOB **por Michel D. Schroender**
Adaptación y traducción al español: Guido

Los sistemas distribuidos son un paradigma popular de cómputo de alto desempeño. En este documento se exploraran las características, fortalezas y debilidades de los sistemas Distribuidos. En el mismo se describe un modelo para el "estado del arte" en los Sistemas Distribuidos el cual realiza un mejor trabajo para soportar el trabajo de computo de propósito general que aquellos realizados en los sistemas existentes. Este modelo de sistema combina las mejores características de los paradigmas en los sistemas centralizados y de las redes de computadoras con mejoras substanciales en la disponibilidad y la seguridad. El documento concluye subrayando los aspectos técnicos más promisorios para estructurar este tipo de sistemas.

1.- Características de los Sistemas Distribuidos

Un sistema distribuido puede considerarse como un grupo de computadoras realizando un trabajo en conjunto, conteniendo tres características primarias.

1) ***Múltiples computadoras***: Un sistema distribuido contiene más de una computadora física, cada una consistente de sus procesadores, memoria local y posiblemente algún almacenamiento permanente como los discos, además de rutas (puertos) de E/S para conectarse con el ambiente externo.
2) ***Interconectadas*** : Algunas de las rutas de E/S permitirán la interconexión entre las computadoras. Si éstas no pudiesen comunicarse entre ellas, entonces no obtendríamos un sistema distribuido.
3) ***Presentando un Estado de Compartición*** : Las computadoras cooperan para mantener algún estado de compartición. Visto de otra forma, si la operación correcta del sistema se describe en terminos de algunas invariantes globales, entonces el mantener estas invariantes requerirá de la operación coordinada y correcta de las múltiples computadoras.

El construir un sistema de este tipo interconectando computadoras requiere de 4 aspectos que deben ser tomados en cuenta.

1) ***Independencia de fallos:*** Debido a que existen diferentes computadoras involucradas cuando alguna de ellas falle existirán más que pueden estar activas. De ésta forma se podrá mantener el sistema en funcionamiento aún y después de que fallen algunas computadoras.
2) ***Comunicaciones interrumpidas***: Debido a que en la mayoría de los casos, la interconexión entre las computadoras no están expuestas a un ambiente cuidadosamente controlado, no funcionarán correctamente todo el tiempo. Las conexiones podrán no estar disponibles o los mensajes podrán perderse o llenarse de basura. Una computadora no puede contar con estar en posición de comunicarse claramente con las demás aun y cuando están activas y funcionando.
3) ***Comunicaciones inseguras***: Las interconexiones entre las computadoras pueden estar expuestas a operaciones no autorizadas y a la modificación de los mensajes.
4) ***Comunicaciones costosas***: La interconexión entre las computadoras usualmente proveen un ancho de banda menor, mayor latencia y un alto costo de comunicaciones que aquellos disponibles entre los procesos independientes con una sola computadora.

Un sistema centralizado que soporta múltiples procesos y permite alguna forma de comunicación interprocesos (como el sistema de tiempo compartido UNÍX) puede mostrar en una forma virtual las tres características primarias de un Sistema Distribuido. Un sistema centralizado puede también manifestar independencia de fallos, por ejemplo, el proceso demonio de control y transporte de correo electrónico puede inhibirse sin detener a los usuarios interactuando con algunos procesos ejecutándose en el mismo sistema. Debido a ello, las técnicas de diseño y de programación asociadas a la comunicación de procesos secuenciales

en Sistemas Centralizados forman parte de las técnicas básicas utilizadas para la construcción de los Sistemas Distribuidos. Sin embargo, los Sistemas Centralizados son suficiente exitosos sin tener que manejar la independencia de fallos y por lo general no confrontan comunicaciones interrumpidas, inseguras ni costosas. Si se piensa en un Sistema Distribuido entonces se tendrán que tomar en cuenta todos los aspectos anteriores.

El ejemplo canónico de un Sistema Distribuido de propósito general hoy en día es un sistema construido sobre redes de computadoras - un conjunto de estaciones de trabajo, computadoras personales y servidores interconectados sobre una subred de comunicaciones.

2.- Sistemas Centralizados vs. Redes de Computadoras.

Es fácil de entender él porque los sistemas de redes de computadoras son tan populares. Este tipo de sistemas permiten el compartir la información y los recursos sobre un área organizacional y geográfica esparcida y amplia. También permiten el utilizar computadoras pequeñas (con una buena relación costo beneficio) y permitir a los ciclos de procesamiento estar los más cercanos posibles a los datos. Estos sistemas pueden crecer poco a poco hasta alcanzar tamaños considerables, permitiendo una autonomía considerable entre sus múltiples versiones de programas, y adoptando múltiples políticas de administración. Finalmente, estos sistemas no necesariamente se inhiben simultáneamente por lo que, en las áreas de compartición, costo, crecimiento y autonomía. Los sistemas de redes de computadoras se consideran en ventaja contra los sistemas centralizados tradicionales.

Por otro lado, los sistemas centralizados realizan el trabajo de forma más eficiente que las redes de computadoras hoy en día, ya que todos los recursos y la información están disponibles de forma equitativa entre todos los usuarios, así como las funciones trabajan del mismo modo para todos y los objetos son identificados por el mismo nombre en cualquier punto del sistema centralizado. Además, los sistemas centralizados son mas fácilmente administrables. De esta forma y aun con las ventajas de las redes de computadoras, los sistemas centralizados son generalmente más fáciles de utilizar debido a su mayor accesibilidad, coherencia y administración.

En las áreas de seguridad y disponibilidad, la comparación entre los sistemas de red y los centralizados no determina una ventaja clara para alguno de ellos ya que los dos presentan problemas a solucionar.

2.1.- Seguridad

En la practica habitual, tanto los sistemas centralizados como los sistemas de red ofrecen seguridad, pero por razones diferentes.

Los sistemas centralizados tienen un dominio de seguridad único bajo el control de una sola autoridad. La base de confianza en la capacidad de cómputo se determina por el único sistema operativo el cual se ejecuta en una sola computadora la cual puede activarse en un ambiente operativo seguro. Con todos los elementos de seguridad puestos en una sola cesta los usuarios pueden entender el nivel de confianza que se le asigna al sistema y a quien se deben de dirigir cuando surgen algunos problemas. Por otro lado, es notorio el nivel de dificultad para eliminar todos los hoyos de seguridad dentro de un sistema operativo o un ambiente operativo y con un solo dominio de seguridad si se explota tan solo un pequeño orificio de seguridad se puede tomar el control de todo el sistema.

Las redes de computadoras tienen múltiples dominios de seguridad y por lo tanto presentan el conjunto inverso de problemas y ventajas en las propiedades de seguridad que los sistemas centralizados. La base de confianza en la capacidad de cómputo se reparte entre muchos componentes los cuales operan en un ambiente

operativo con diferentes grados de seguridad física, políticas de seguridad, y posiblemente diferentes autoridades. La interconexión entre todas las computadoras puede ser físicamente insegura, así que es difícil el conocer que de todo es confiable y que no lo puede ser. Pero, debido a que el sistema contiene muchas computadoras, el explotar un orificio de seguridad en el software de alguna de ellas o en el ambiente de operación no necesariamente compromete la seguridad de todo el sistema.

2.2.-Disponibilidad

De forma similar se puede aplicar un análisis comparativo a la disponibilidad. Dentro de un sistema centralizado puede haber un espacio físico y un ambiente de operación controlado. Debido a que una alta proporción de los fallos del sistema son el resultado de factores fuera de control tanto operacionales como del ambiente, una administración cuidadosa del ambiente único puede producir una muy buena disponibilidad. Pero cuando algo anda mal y existe el riesgo de un fallo, el sistema corre el riesgo de detenerse completamente afectando a todos los usuarios que se encuentran trabajando en él.

Dentro de los sistemas en red los fallos entre computadoras se presentan de forma independiente, sin embargo, es común el caso en que varias computadoras deban estar activas concurrentemente para que un usuario pueda realizar su trabajo, por lo que la probabilidad de que falle la disponibilidad del sistema es mayor que aquella para que una sola computadora falle. Si comparamos este incremento en la probabilidad de no poder trabajar con la de los sistemas centralizados, obtenemos como resultado una baja disponibilidad por ignorar la independencia de fallos. La consecuencia se transforma en la definición Leslie Lamport para un sistema distribuido: “Usted sabe que tiene uno cuando el fallo total en una computadora de la cual usted jamás ha escuchado le impide la realización de cualquier trabajo en la suya”.

Por otro lado, la independencia de fallos en los Sistemas Distribuidos puede ser explotada para incrementar la disponibilidad y la confiabilidad. Cuando la independencia de fallos se aplica y se ajusta correctamente replicando las funciones entre componentes independientes, se requerirán múltiples fallos en estos componentes entes que se afecten la disponibilidad y la confiabilidad del sistema. La probabilidad de que el sistema falle puede considerarse entonces, menor que la probabilidad de fallo en un sistema centralizado.

Un sistema distribuido debe enfrentarse también con fallos en las comunicaciones. La interrupción en las comunicaciones no solamente contribuye a la falta de disponibilidad, sino que también conduce a funcionamientos incorrectos. Una computadora no puede distinguir confiablemente entre un vecino inactivo y un vecino sin conexión por lo tanto no podrá estar segura de que su vecino que no le puede responder se ha detenido.

Mantener estados globales invariantes en estas circunstancias puede ser riesgoso. Se requiere pues de un diseño cuidadoso para cumplir la meta de operar correctamente y de incrementar la disponibilidad utilizando componentes replicados.

2.3.- “Estado del arte” en los Sistemas Distribuidos.

Se puede observar posible y adecuado el desarrollo de sistemas distribuidos los cuales combinen las ventajas de accesibilidad, coherencia y facilidad de administración de los sistemas centralizados con las ventajas de compartición, crecimiento, costo y autonomía de las redes de computadoras. Si se agregan a la mezcla una seguridad efectiva y una alta disponibilidad entonces tendremos un sistema distribuido en el “estado del arte” para sistemas de computo de múltiples propósitos El cumplir estas características en un solo equipo es el reto principal que el soporte de computo de propósito general tiene para los Sistemas Distribuidos. Hasta el momento, no existe algún sistema que cumpla todas estas características, aunque hay algunos que se acercan bastante al modelo.

3.- El Modelo de Propiedades y de Servicios.

Podemos determinar ésta base de computo distribuida como el modelo BOB (del ingles “Best of Both” o lo mejor de ambos modelos, el centralizado y el de red), la cual se describe en términos de un modelo de propiedades y de servicios. A continuación se describen de forma técnica las metas que permiten una estructura la cual nos ayudara a entender los mecanismos para alcanzarlos.

El modelo de propiedades y de servicios define a BOB como:

- Un conjunto heterogéneo de hardware, software y componentes de datos
- Cuyo tamaño y extensión geográfica puede variar en rangos amplios
- Conectados por una red para computadores
- Proveyendo un conjunto uniforme de servicios (nombramiento, invocación remota, registro de usuarios, sincronización del tiempo, uso de archivos, etc.)
- Con un grupo de propiedades globales (nombres, acceso, seguridad, administración y disponibilidad).

Debido a que nuestro enfoque se basa en la obtención de la capacidad de computo de propósito general, el modelo se define en los términos mas apropiados para ser entendido por los programadores que desarrollaran los componentes parte de ésta base y quienes desarrollaran las diversas aplicaciones que se ejecutarán sobre el sistema.

La coherencia que determina a BOB como un sistema íntegro en lugar de una colección de computadoras independientes es el resultado de sus servicios uniformes y de las propiedades globales. Estos servicios están disponibles del mismo modo que todas las partes del sistema y las propiedades le permiten a cada parte del sistema el observarse del mismo modo, sin importar el tamaño del mismo. Los diseñadores se cuidan mucho del cuidado que deben de tener para producir implementaciones que puedan soportar el crecimiento a tamaños realmente grandes (miles de nodos). Un reto similar existe para que éstos sistemas sean tan expandibles, ligeros y simples como para que puedan ajustarse a configuraciones pequeñas.

La coherencia de BOB no significa que todos los componentes del sistema deben ser los mismos. El modelo aplica a la colección de computadoras heterogéneas ejecutando diversas versiones de sistemas operativos. En breve, todas las plataformas deben de operar en este marco de trabajo, aun las computadoras y sistemas que sean producidas por diferentes proveedores y vendedores. La red de producción para el sistema distribuido puede ser entonces una colección de redes locales, puentes, enrutadores, switchers, compuertas de software y varios tipos de servicios de larga distancia con servicios de conectividad permitidos por diversos protocolos de transporte de mensajes.

3.1.- Propiedades de Permanencia:

- **Nombres Globales**: El mismo nombre funciona en todo el sistema.
- **Acceso Global**: Las mismas funciones son utilizables en cualquier punto.
- **Seguridad Global**: El mismo esquema de autentificación y de control de acceso del usuario.
- **Administración Global**: La misma autoridad puede manejar los componentes administrables desde cualquier punto.
- **Disponibilidad Global**: Los mismos servicios se presentan aun y cuando existan fallas.

3.2.- Servicios:

- Nombramiento: Proveé acceso a la base de datos replicada y distribuida de los nombres globales y sus valores asociados para los equipos, usuarios, archivos, listas de distribución, grupos de control de acceso y los

servicios. El servcio de nobraminbto es el componente primordial en el modelo BOB para administrar nombres globales, tomando en cuenta que el trabajo restante para implementar el nombramiento global deberá incluirse en los demas componentes del sistema distribuido. Por ejemplo, los sistemas operativos y las aplicaciones de los usuarios deberán incluir software que utilice el servicio de nombramiento de una forma consistente.

- Llamadas a procedimientos remotos: Proveé una forma estandar de definir e invocar de manera segura las interfaces de servicios. Esto permite a las instancias de los servicios el poder ser tanto locales como remotas. Los mecanismos de RPC pueden ser organizados para operar seleccionando una variedad de protocolos de transporte dinamicamente. El seleccionar los RPC's como el mecanismo estandar de invocación de servicios no obliga a usar la semantica de bloqueo en todos los programas. Los RPC's definen una forma de concordar los mensajes de respuestas con los de sus mensajes de petición adecuados, lo que permite que no se requiera de que el programa que hace la petición se bloque para esperar una respuesta.
- Registro de usuarios. Les permite a los usuarios el ser registrados, autentificados y certificados, asegurando su acceso a los recursos del sistema.
- Sincronización del tiempo. Distribuye señalizaciones de tiempo adecuada y cosnsistentemente de forma global.
- Uso de archivos: Proveé el acceso a sistemas de archivos distribuidos, replicados y globales. Cada equipo, como componente de BOB, puede compartir el espacio de almacenamiento y los archivos que en el almacena de forma local a través de su interface estandar. Los múltiples espacios con nombres de archivos se interconectan a través del servicio de nombramiento. La especificacion de los servicios de archivos debe de incluir presentaciones estandares para los diferentes tipos de archivos en sistemas tales como UNIX, VMS, NT, etc. Por ejemplo, todas las implementaciones deberán de soportar las vistas de los archivos como un arreglo de bytes.
- Administración: Proveé acceso a los datos de administración y a las operaciones para cada componente.

Agregando a los anteriores, BOB especifica otros servicios apropiados para la definición de las aplicaciones:

- Uso de registros.
- Impresión.
- Ejecución.
- Correo electrónico y mensajería personal.
- Terminales y/o puntos de acceso.
- Contabilidad del uso.

Se considera adecuado utilizar estándares de aceptación global existentes para la definición de la base de servicios adicionales. Cada servicio debe ser definido e implementado para proveer de las 5 propiedades de permanencia.

3.3.- Interfaces.

Las interfaces son la clave para lograr un sistema BOB coherente y abierto. Cada uno de los servicios se define por su especificación de interfase la cual funciona como un contrato entre el servicio y sus clientes. La interfase define las operaciones que se van a proveer, los parámetros en cada una y los detalles semánticos de cada operación relacionados con el modelo de estados mantenido por el servicio. La especialización se representa normalmente como una definición de interfase RPC. Se deben proveer, además, algunos métodos de medición del desempeño de las operaciones. Una interfase definida precisamente permite la interacción entre el modelo, las versiones y los proveedores. Pueden existir muchas de las implementaciones para cada

interfase y esta variedad permite que los sistemas sean heterogéneos en sus componentes. En sus interfases, sin embargo son homogéneos.

Esta homogeneidad es la que hace al sistema presentar una conducta predecible en lugar de presentar una colección de componentes que puedan tan solo comunicarse. Si existe mas de una interfase para la misma función, será inevitable el que la función trabaje de modo diferente a través de las diferentes interfaces. El sistema consecuente mente será más complicado y menos confiable. Quizás algunos componentes, después de todo, no estarán en posibilidad de utilizar a otros porque no tengan una interfase común; algunos clientes y proveedores(vendedores) encontraran más difícil el configurar las colecciones de componentes que puedan interactuar y los programadores no conocerán de que servicios activos dependen para poder generar las aplicaciones correspondientes.

4.- Alcanzando las propiedades Globales.

La experiencia y la investigación sugieren un conjunto de modelos para cumplir con el nombramiento global, el acceso, la seguridad, la administración y la disponibilidad. Para cada una de estas propiedades de permanencia se deberá de considerar la meta general que sea mas prometedora.

4.1.- Modelo de Nombramiento.

Cada usuario y cada programa cliente debe observar el sistema a través de un árbol de nombres de objetos. Un nombre global se interpreta por el seguimiento de los nombres brincando entre los nodos del árbol comenzando desde la raíz general. Cada nodo tiene un camino para encontrar la copia de la raíz del árbol de nombramiento global.

Es muy común utilizar un espacio de nombres jerárquicos debido a que es la única estructura de nombramiento que conocemos que permite escalar apropiadamente, permite autonomía en la selección de nombres y es suficientemente flexible para permanecer por mucho tiempo. Se requiere de una raíz global en el espacio de nombramiento para otorgarle un nombre único a cada objeto el cual será reconocido desde cualquier punto del sistema. Se pueden agregar enlaces no jerárquicos en donde se requiera una estructura de nombres mas sofisticada.

Para cada tipo de objeto debe existir un servicio cuya interfase sea definida por BOB, la cual permita operaciones para crear y eliminar objetos de ese tipo y para leer y modificar su estado.

El servicio de nombramiento de BOB provee la parte alta del árbol de nombres y los objetos cercanos a la raíz del árbol son implementados por el servicio de nombramiento. Un nodo dentro del árbol de nombres, sin embargo, puede ser una integración entre el servicio de nombramiento y algún otro tipo de servicio, por ejemplo: el servicio de archivos. Una integración de objetos contiene:

- Un conjunto de servidores para el objeto nombrado.
- Reglas para escoger un nombre de servidor.
- El identificador de la interfase, por ejemplo: “Servicio de Archivos BOB 2.3”.
- Una serie de parámetros, por ejemplo: el identificador del volumen.

Para buscar un nombre a través de la integración se debe de escoger un servidor y llamar a la interfase del servicio a través del nombre y los parámetros del objeto. El servidor encontrara el resto del nombre.

Los servidores enumerados en una integración de objetos son designados como nombres globales. Para llamar a un servicio en el servidor, el cliente debe de convertir el nombre del servidor en algo mas útil, tal y como la dirección de red del servidor y la información de los protocolos que se utilizaran en esta llamada. Buscar un nombre de servidor dentro del árbol de nombres globales producirá un objeto servidor el cual contendrá:

- El nombre del equipo.
- El conjunto de nombres de los protocolos de comunicaciones.

Una búsqueda final del nombre hace un mapeo (global) del nombre del equipo a la dirección de red que será el destino del servicio para el RPC requerido.

La maquinaria de integración puede ser utilizada en diferentes niveles como se juzgue apropiado. La integración es una técnica poderosa para unificar múltiples implementaciones de los mecanismos de nombramiento por sobre un mismo espacio de nombramiento jerárquico.

Figura 1:

La figura anterior muestra un ejemplo acerca de cómo deben de lucir las partes altas del espacio global de almacenamiento utilizando como base el esquema de nombramientos del Internet Domain Service(DNS). Un servicio de X.500 o de LDAP también puede proveer esta parte del espacio de nombramiento (o todas estas se pudieran integrar). Una parte importante de la implementación de BOB es el definir la jerarquía de nombres actual a ser utilizada en los niveles altos del espacio de nombramiento global.

Considere algunos nombres de objetos como los siguientes: /com, /com/dec y /com/dec/ser son directorios implementados a través de los servicios de nombramiento de BOB, /com/dec/sre/adb es un usuario registrado y también un objeto implementado por el servicio de nombramiento. El objeto /com/dec/ser/adb contendrá una palabra clave de acceso encriptada (password), un conjunto de direcciones a buzones de correo electrónico y otro tipo de información asociada con el usuario en este sistema.

Así mismo, /com/dec/ser/staff podría ser un grupo de nombres globales, BOB debe de proveer soporte a nombres de grupos de objetos para implementar aspectos como listas de distribución, listas de control del acceso y conjuntos de servidores /com/dec/ser/bin pudiese ser un volumen del sistema de archivos. Note que este objeto es una integración al servicio de archivos de BOB. La figura no muestra el contenido de la integración de este objeto, pero contiene un grupo de nombres del conjunto de servidores implementando este volumen de archivo y las reglas para seleccionar cual de ellos utilizar , por ejemplo, el primero que responda. Para buscar el nombre /com/dec/ser/bin/ls, por ejemplo, el sistema operativo en la maquina cliente trasladara la ruta /com/dec/ser/bin utilizando el servicio de nombramiento. El resultado en este punto es el contenido del objeto integrador, el cual le permitirá ahora al cliente el contactar un servidor de archivos para completar la búsqueda.

La autonomía de administración local permitida por el uso de nombres jerárquicos permite a los programadores y administradores de sistemas el construir y utilizar sus sistemas sin tener que esperar por el logro de un acuerdo de valides planetaria acerca de la infraestructura de los pocos y primeros niveles de la jerarquía global. Así, se puede construir un espacio de nombramiento jerárquico local que satisfaga las necesidades locales tratando la raíz global. Después, este espacio de nombramiento puede incorporarse como una subraiz dentro de un espacio de nombramiento mas grande.

Utilizando una variable que determina el nombre de la raíz operacional (colocada como nula al inicio) se podrá fácilmente migrar hacia un esquema de nombramiento superior (acortando los nombres que la gente usa actualmente). Otra técnica es la de inicializar el nombre de la raíz local con un identificador que no aparecerá

fácilmente en una futura raíz global, por lo que un enlace simbólico a la raíz global o una secuencia de código especial en la maquinaria de resolución de nombres locales pueden facilitar la transición.

4.2.- Modelo de Acceso.

El acceso global significa que un programa puede ejecutarse en cualquier punto de BOB (en computadoras y sistemas operativos compatibles binariamente) y obtener el mismo resultado, mientras que el desempeño puede variar dependiendo de la maquina que se seleccione. Debido a esto, un programa puede ser ejecutado tanto en la estación de trabajo del usuario o en un servidor de ejecución rápida de ciclos en el site de computo, mientras se están accesando concurrentemente los mismos archivos utilizando para ello los mismos nombres.

El lograr el acceso global requiere el permitir a todos los elementos dentro del ambiente computacional de los programas el poder acceder remotamente a la computadora donde se esta ejecutando el programa. Todos los servicios y los objetos requeridos por el programa para ejecutarse necesitan estar disponibles al programa ejecutándose en cualquier punto del sistema distribuido. Para un usuario en particular "cualquier punto" significa al menos:

- En la estación de trabajo del usuario.
- En una estación de trabajo publica o en el equipo servidor dentro del dominio del usuario.
- En una estación de trabajo publica dentro de otro dominio pero en la misma red local del usuario.
- En una estación de trabajo publica utilizando un enlace WAN con un bajo ancho de banda.

El desempeño en los primeros tres casos deberán de ser similares, pero en el cuarto caso esta limitado por las características definidas para el acceso sobre WAN's. Aun así, el uso de cache puede significar una pequeña diferencia en muchos casos.

Dentro de BOB, el nombramiento global y los servicios estándares que se exporten a través de mecanismos de RPC uniformes proveen la clave para lograr el acceso global. Todos los servicios en BOB deben de aceptar nombres globales para los objetos en los que se va a operar. Todos los servicios de BOB deberán estar disponibles a los clientes remotos. Así, cualquier objeto cuyo nombre global sea conocido puede accesarse remotamente.

Agregando, los programas deben de acceder a su ambiente utilizando exclusivamente los nombres globales de los objetos. Este ultimo paso requerirá un examen completo del ambiente de computo provisto por cada uno de los sistemas operativos para identificar todas las posibles formas en que los programas pueden acceder a su ambiente. Para cada forma identificada se debe diseñar un mecanismo para proveer el nombre global del objeto deseado.

Por ejemplo, en los sistemas UNÍX que operan como parte de BOB, la identidad de los directorios del sistema de archivos raíz, de trabajo y el de manejo de temporales (.tmp) dentro de un proceso deberán de ser especificados por su nombre global. Modificar VMS o UNÍX o cualquier otro sistema operativo para aceptar nombres globales desde cualquier punto deberá de ser un cambio muy importante.

Otro aspecto del acceso global es asegurar que los servicios en BOB tengan operaciones semánticas que presenten transparencia de localidad. Sin la transparencia de localidad en los servicios utilizados un programa no obtendrá los mismos resultados cuando se ejecuta en diferentes computadoras. Un servicio que permita compartir objetos con operaciones de lectura y escritura debe también de proveer un modelo de datos coherente. El modelo permite al programa cliente el mantener estados correctos en los objetos y comportarse de una forma que no sorprenda al usuario, sin importar donde se encuentren ejecutándose los clientes y los

servicios. Dependiendo de la naturaleza de los servicios, es posible el negociar el desempeño, la disponibilidad, la escala y la coherencia.

Dentro del servicio de nombramiento, por ejemplo, se considera apropiado el incrementar el desempeño, la disponibilidad, la escalabilidad a costa de la coherencia. Una actualización de un cliente para el nombre de la base de datos de los servicios puede realizarse conectándose a cualquier servidor. Después que la operación del cliente se complete, el servidor propagara la actualización a las replicas del objetos en los otros servidores. Hasta que la preparación se complete, diferentes clientes podrán leer diferentes valores para el objeto. Esta falta de coherencia produce algunas ventajas: incrementa el desempeño disminuyendo el tiempo de espera de los clientes hasta que se termine la actualización, incrementa la disponibilidad permitiéndole al cliente el realizar la actualización cuando solamente un servidor sea accesible, e incrementa la escalabilidad permitiendo un tiempo de latencia de propagación para el cliente separado del tiempo de latencia visible para la operación de actualización. Para los objetos que el servicio de nombramiento almacenara, esta falta de coherencia es extremadamente aceptable debido a los beneficios que produce. El modelo de datos coherentes para el servicio de nombramiento define las invariantes en la perdida de coherencia de los cuales depende el programador, así como para cumplir con los requerimientos del modelo de coherencia que son insensitivos para la localización del cliente y del servidor.

Por otro lado, el servicio de archivos de BOB necesita proveer una escritura compartida consistente, aun y cuando se presente un efecto negativo en el desempeño, la escala y la disponibilidad. Muchos programadores y usuarios están acostumbrados a utilizar el sistema de archivos como un canal de comunicaciones entre programas. Por ejemplo, un programador puede almacenar un código fuente para un modulo desde un editor ,y después activar la recompilación y relocalización de librerías en un servidor de ejecución de ciclos de computo. El usuario estará mas que sorprendido si se recompila el modulo desde un código fuente anterior debido a que el servidor de ejecución recupero con el mismo nombre de objeto un archivo con la versión del código no actual no almacenada en cache. La coherencia en la escritura y lectura de archivos es también importante entre elementos de computo distribuido ejecutándose en digamos múltiples equipos de computo en la misma red local. El modelo de coherencia del sistema de archivos debe de cubrir nombres de archivos y atributos así como archivos de datos.

Existe una diversidad de opiniones entre la comunidad de investigadores acerca del mejor modelo de consistencia para un sistema de archivos distribuidos de propósito general. Algunos opinan que un modelo de consistencia abierto / cerrado provee mejores características. Con este modelo los cambios realizados por cualquier cliente no se propagan hasta que el cliente cierra el archivo y los demás lo (re)abren. Algunos mas opinan que mantener compartidas las escrituras al nivel de byte es posible y deseable. Con este modelo los clientes comparten los archivos siempre y cuando se estén accesando en la misma área de memoria local. Se han construido sistemas exitosamente utilizando variantes de ambos modelos, por lo que BOB puede basarse exitosamente en ambos.

4.3.- Modelo de Seguridad

La seguridad se basa en tres notaciones:

- Autentificación
- Control del Acceso
- Auditoria

Para autentificar una petición cuando viaja desde un particular director, el sistema debe de determinar que efectivamente el director es el origen de la petición, y ésta no ha sido modificada en el camino hasta el destino. Se realiza la última parte estableciendo un canal seguro entre el sistema que origina la petición y aquellos que lo transportan. La práctica de seguridad en los sistemas distribuidos establece que se requiere de utilizar mecanismos de encriptación para asegurar los canales de comunicación. La encriptación no debe, por otro lado, alentar el tiempo de respuesta en las comunicaciones, debido a que en general es difícil de estar seguro cuando un mensaje en particular no requiere de esa encriptación. La arquitectura deberá de incluir

métodos de realizar la encriptación y decriptación al vuelo, ya que los datos fluyen desde las computadoras hacia las redes y de regreso.

Para determinar quien ha originado la petición es necesario el conocer quien está del otro lado del canal seguro. Generalmente esto se hace asegurándonos de tener al director del otro lado demostrando que conoce algún elemento secreto (como una clave secreta de acceso) y después comprobando de manera confiable el nombre del director que dice conocer el elemento secreto. La arquitectura de seguridad en BOB requiere de especificar como se realizan ambas casas. Esto se mejora si el director puede demostrar que conoce el elemento secreto sin tener que mostrarlo, de lo contrario alguien en el sistema posteriormente puede "suplantar" al director Los esquemas basados en claves secretas de acceso (passwords). Revelan el secreto, pero los esquemas basados en encriptación pueden no hacerlo.
Es deseable el autentificar a los usuarios por la posesión de un dispositivo. El cual conoce el elemento secreto y puede demostrarlo utilizando encriptación. Este tipo de dispositivos se conocen como "tarjetas inteligentes" (smart cards) requiriendo lectores especiales para su lectura. Una alternativa no tan sofisticada es pedirle al director que de a conocer (tecleando) el elemento secreto a algún agente autentificador del acceso. Para autentificar un sistema de cómputo, se requiere de asegurar que se ha cargado adecuadamente con una imagen certificada del sistema operativo. En BOB se deben especificar los métodos para asegurar que lo anterior se realice.
La seguridad depende del nombramiento, desde que el control de acceso identifica al director que está requiriendo acceso a través de su nombre. La seguridad en la práctica también depende del soporte a grupos de principales. Ambas facilidades en los servicios deberán de ser provistas por el servicio de nombres. Para asegurar que los nombres y los grupos están definidos confiablemente se pueden utilizar firmas digitales para certificar la infamación dentro del servicio de nombramiento. Las firmas digitales se generan por

una autoridad competente y certificadora la cual se define para ofrecer una alta confiabilidad y de modo oculto, quizás en un lugar privado y asegurado, cuando no se requiera. La autentificación depende solamente del pequeño sub-árbol del árbol completo de nombramiento el cual incluye la definición para peticiones del principal y los recursos. Las autoridades de certificación que se encuentren lo más remotas del sistema se asumen que son las menos confiables.

La seguridad también depende del tiempo. En la autentificación el control del acceso y los canales de seguridad requieren de marcas de tiempo y una especificación sincronizada del tiempo (relojes). El servicio de distribución de la referencia al tiempo debe de hacerlo de forma segura también.

4.4.- Modelo de Administración

El modelo de administración es el ajuste del estado del sistema por el administrador humano. La administración se requiere cuando no se pueden incluir algoritmos satisfactorios para un ajuste automático - o cuando se requiere un ajuste a juicio humano. El problema en un sistema distribuido a gran escala es el proveer a cada administrador del sistema con los medios para monitorear y ajustar una gran colección de componentes distribuidos de diferente tipo y sobre un área geográfica muy amplia. De las cinco propiedades de permanencia en BOB, la administración global es la única de la cual no se entiende con claridad como obtenerla. Sin embargo, se pueden establecer algunos lineamientos estructurales para la obtención de este servicio.
El modelo de administración en BOB se basa en el concepto de dominio. Cada componente dentro del sistema distribuido se asigna a un dominio. Cada dominio tiene un administrador del sistema responsable. En la versión más simple del modelo, los dominios están separados y los administradores son independientes, aunque se pueden generar convenios complejos cuando sea posible. Idealmente los dominios no dependerán entre ellos para su correcto funcionamiento.
Se requiere de tener una flexibilidad adecuada en la disponibilidad para definir dominios. Debido a que diferentes convenios podrán ser efectivos en diferentes instalaciones.
Ejemplos de dominio son:

- Componentes utilizados por un grupo de gente con metas comunes

- Componentes que un grupo de usuarios esperan encontrar para trabajar
- Un número grande de componentes dependientes de un administrador
- Un número arbitrario de componentes que no sea muy grande

Como un aspecto práctico, los clientes requerirán de guías al trabajo para la definición de un criterio de administración efectivo para el dominio.
BOB requiere que cada componente defina y exporte una interfase de administración utilizando RPC si es posible. Cada componente se administra a través de las llamadas a su interfase desde las herramientas interactivas ejecutadas por 1os administradores humanos. algunos requerimientos para la internase de administración del componente son:

- Acceso Remoto
- Interfase del Programa
- Relevancia
- Uniformidad

La administración de interfases y las herramientas hacen práctico para una persona el administrar dominios de diversos tamaños. Una herramienta interactiva de administración puede invocar la interfase de administración de todos los componentes dentro del dominio. Además provee formas ajustables para desplegar y correlacionar los datos, y de cambiar los estados de administración relevantes de los componentes. Las herramientas de administración son capaces de realizar el mismo cambio de estado en un grupo numeroso de componentes similares dentro del dominio a través de llamadas interactivas. Para permitir la flexibilidad de desarrollar nuevas operaciones de administración algunas herramientas de administración soportan la construcción de programas que llaman a las interfaces de administración de los componentes dentro del dominio.

4.5.- Modelo de Disponibilidad

Para alcanzar una alta disponibilidad del servicio deben de existir servidores múltiples para cualquier servicio. Si estos servidores se estructuran para soportar la independencia de fallos, entonces se puede alcanzar cualquier nivel deseado de disponibilidad tan solo ajustando el grado de replicación.
El esquema más practico para la replicación de servicios en BOB es la replicación primaria / respaldo, en la cual un cliente utiliza un servidor a la vez y los servidores cuando ocurre un fallo se las arreglan (tanto como sea posible) de que ocurra de la forma más benéfica posible, por decir, deteniéndose. Los métodos alternativos, denominados replicaciones activas, permiten que el cliente realice cada operación en múltiples servidores. La replicación activa consume más recursos que la replicación primara / respaldo, pero no tiene retrasos en la ocurrencia de fallos así como puede tolerar una conducta con ocurrencia aleatoria de fallos en los servidores.

Para observar como trabaja la replicación primaria / respaldo debemos de retomar la discusión para el modelo de nombramiento en la que el objeto que representa un servicio incluye un conjunto de servidores y algunas reglas para su selección. Si el servidor seleccionado (el primario) presenta una falla, entonces el cliente puede buscar un nuevo servidor (respaldo) y repetir la operación. El cliente asume la ocurrencia del fallo si expira el tiempo de tolerancia para obtener la respuesta a una operación mientras espera. El tiempo de tolerancia deberá de ser tan pequeño para que la latencia de una operación que falla sea comparable a la latencia usual de espera. En la práctica, es difícil establecer una cantidad adecuada para el tiempo de tolerancia así como establecer tiempos fijos puede no ser lo adecuado. Si los clientes pueden realizar operaciones que cambien estados de largo periodo entonces el servidor primario debe de mantener a todos los servidores de respaldo actualizados.
Para asegurar la transparencia de fallos desde el punto de vista de los programas clientes, el conocimiento de los múltiples servidores deberá de ser encapsulado dentro de un agente en la computadora del cliente. En éste documento nos referiremos al agente como un “despachador”. El despachador puede exportar lógicamente servicios locales y centralizados a los programas clientes, aún y cuando la implementación del servicio, determinado sea distribuida, replicada y remota. El software del despachador puede tener muchas relaciones estructurales diferentes con sus clientes. En el caso más simple puede ser una librería de ejecución activa

dentro del espacio de direcciones del cliente e invocada a través de llamadas a procedimientos locales. Puede ser también un espacio de direccionamientos en la misma computadora que el cliente e invocada a través de un mecanismo de IPC o de RPC o de llamadas de retorno del sistema operativo dentro del mismo equipo. O hasta puede incluirse en los servicios del mismo sistema operativo.
La interfase del despachador no debe de ser la misma, que la interfase del servidor. De hecho, la interfase del servidor usualmente deberá de ser significativamente más compleja. Agregando a la implementación de la selección del servidor, el despachador debe de proveer el uso de cache y de escritura retardada para incrementar el nivel de desempeño, así como puede implementar la inclusión de operaciones que lean, escriban o modifiquen los estados de los servidores. Ejemplos simples del uso de caché, un despachador del servicio de nombramiento debe de recordar los resultados de las recientes búsquedas de nombres de objetos y mantener conexiones abiertas a los servidores de nombres con más frecuencia de utilización; o un despachador del servicio de archivos debe de usar caché para almacenar los resultados de búsquedas recientes de archivos y de las operaciones de lectura de datos. La escritura retardada te permite al despachador el procesar por lotes varias actualizaciones como una sola operación del servicio, lo cual puede realizarse con conexiones asincrónicas entre los servidores. De este modo se reduce la latencia en las operaciones realizadas con la internase del despachador. La implementación de una operación de lectura - modificación escritura en un cliente puede requerir la utilización de estrategias complejas de reintentos por parte del despachador involucrando algunas operaciones con los servidores cuando ocurre una falla en alguno de los pasos intermediarios.

Como un ejemplo do cómo un despachador enmascara la existencia de múltiples servidores, considere las acciones involucradas en el listado del contenido de /com/dec/ser/udir/bvwl/Mail/inbox, un directorio dentro del sistema distribuido de archivos de BOB. El programa cliente presenta una ruta completa del nombre al despachador de servicio de archivos. El despachador localiza el servidor de nombres que almacena el directorio raíz y le presenta el nombre completo. Este servidor puede almacenar los directorios inferiores, por decir, /com/dec. La entrada del directorio para /com/dec/ser puede indicar otro conjunto de servidores. De este modo la primera operación de búsqueda regresará el nuevo conjunto de servidores con una respuesta acerca del nombre de la ruta pendiente a resolver. El despachador entonces contactará un servidor dentro del nuevo conjunto de servidores y le presentará una nueva búsqueda a partir de la dirección pendiente de resolver sre/udir/bwMail/inbox. Este servidor contactado descubrirá que sre/udir es una integración a un volumen del sistema de archivos, así que regresará la información de la integración y el nombre de la ruta inconclusa udir/bwl/Mail/inbox. Finalmente, el despachador utiliza la información de la integración para contactar a un servidor de archivos, el cual dentro do éste ejemplo almacenará el directorio determinado y responderá con el contenido de este directorio. Así, lo que parece ser una operación simple por parte del programa cliente involucrará realmente llamadas a procedimientos remotos en tres diferentes servidores por el despachador.

El ejemplo de la resolución de nombres se podría completar con la realización o no de algunas operaciones en los servidores remotos si el despachador presenta un cache de servicios que contenga la información necesaria. En la práctica, el uso de caché es una parte importante de muchas implementaciones de despachadores y la mayoría de las operaciones se agilizan considerablemente si se presentan caches de datos para los servicios.

Se pueden analizar otras características en la implementación de servicios con alto nivel de disponibilidad con estados de largo periodo. Los servidores deben de cooperar entre ellos para mantener un estado consistente, de este modo un respaldo se puede realizar razonablemente rápido. Los problemas que se deben de solucionar incluyen un convenio en el cual no se deben de perder las escrituras durante la presencia de fallos entre los servidores y de que un servidor que se encuentre inactivo pueda recobrar su estado actual cuando sé reestablezca. Combinando estos requerimientos con caché y escrituras retardadas para obtener un buen nivel de desempeño, sin sacrificar la consistencia de la compartición, puede significar en la implementación y construcción de un servicio con alta disponibilidad con muchos retos.

5.- Conclusiones

Este documento cubre las fortalezas inherentes de los sistemas de computo centralizados y en red. Remarca la estructura y las propiedades de BOB, un sistema de computo distribuido en el "estado del arte" base para el soporte de cómputo de propósito general. Este sistema combina las mejores características de los sistemas centralizados y de red con los más recientes avances en los aspectos de seguridad y de disponibilidad para producir un ambiente de cómputo poderoso, de buena relación costo - eficiencia y de fácil utilización.

La obtención de sistemas como el descrito por BOB en el ámbito de uso general se debe de considerar una tarea difícil. Una vez determinado el "estado del arte" en la tecnología de los sistemas distribuidos, es posible la construcción de un prototipo como prueba del concepto planteado. Sin embargo, el único método práctico de obtener varios sistemas con las características planteadas de uso general y de amplia cobertura es el de encontrar las formas prácticas para alcanzar las metas propuestas en los sistemas de red actuales modificándolos para que soporten los puntos no incluidos y mejorándolos en los puntos que si incluyan pero no cumplan con las metas propuestas. Para ello se requerirá producir una secuencia de cambios útiles y palpables para conducir el nuevo diseño hasta alcanzar las metas.

6.- Reconocimientos

El material utilizado en éste documento fue desarrollado en conjunto por Michael Schroender, Butler Lampson y Andrew Birell. Las ideas aquí exploradas se basan en años de experiencia de los diseñadores, constructores, programadores y usuarios de sistemas distribuidos. Algunos colegas en el SRC y autores de libros de Sistemas Distribuidos nos han proporcionado sugerencias para la presentación.

N.del.T.: