

TelORB — El sistema operativo distribuido para comunicaciones

Lars Hennert y Alexander Larruy

Las plataformas de telecomunicaciones futuras deben cumplir tanto los requisitos tradicionales de disponibilidad y desempeño como los requisitos cada vez más rigurosos de apertura y escalabilidad.

TelORB es un sistema operativo distribuido para aplicaciones de tiempo real a gran escala incorporadas que requieran operación sin paradas. Se compone de un moderno núcleo (kernel) de SO, una base de datos en tiempo real, control de configuración de software, y un entorno de desarrollo asociado para escribir código de aplicación específico para las tareas. Un agente de petición de objetos que cumple CORBA y una máquina virtual Java corren en superposición con TelORB.

Los autores describen la plataforma del sistema operativo TelORB, sus singulares características, y las entidades de proceso. Éstas incluyen procesadores de dispositivos que controlan directamente el hardware con rigurosos requisitos de tiempo real; el sistema operativo TelORB, que controla la disponibilidad del tráfico y un funcionamiento flexible en tiempo real; y UNIX o Windows NT, que proporcionan entornos de programación estándar para funciones y aplicaciones de plataforma en tiempo real menos críticas.

MARCAS REGISTRADAS

Java™ es una marca registrada propiedad de Sun Microsystems Inc. en los Estados Unidos y otros países.

Windows NT es una marca registrada de Microsoft Corporation.

Introducción

Las futuras plataformas para uso en telecomunicaciones deben ofrecer una disponibilidad extremadamente alta, sus sistemas deben funcionar sin paradas, independientemente de los errores de hardware o software, y deben permitir a los operadores actualizar el hardware y el software en plena operación sin perturbar a las aplicaciones que funcionan en ellos. Estos rigurosos requisitos de robustez no deben afectar al funcionamiento del sistema.

En el campo de las telecomunicaciones, los requisitos relacionados con el desempeño se especifican a menudo en términos de estadísticas. Por ejemplo, debe ser posible, el 90% del tiempo, llevar a cabo una determinada operación dentro de un periodo especificado. Durante el 10% restante del tiempo el umbral estipulado puede ser excedido. Un funcionamiento en tiempo real

de este tipo, que a veces se denomina funcionamiento flexible en tiempo real, por lo general es suficiente para los sistemas operativos de telecomunicaciones.

Las plataformas deben ser también escalables en términos de capacidad: los operadores deben poder aumentar la capacidad del sistema simplemente enchufando nuevos equipos de procesamiento, en vez de tener que reemplazar los procesadores por otros más potentes en una planta.

Finalmente, hay una fuerte tendencia hoy día hacia los sistemas abiertos. La apertura, sin embargo, viene en muchas variedades:

- Plataforma de hardware abierta. Los operadores deben poder usar hardware disponible comercialmente en las estanterías. Este requisito garantiza que los sistemas puedan seguir el ritmo de los últimos avances en diseño de hardware y aprovecharse de ellos.
- Lenguajes de programación. Los operadores deben poder contratar desarrolladores experimentados que puedan ser productivos sin tener que asistir primero a cursos de larga duración.
- Interoperabilidad. Los sistemas abiertos deben soportar protocolos estándar para comunicarse con sistemas externos procedentes de toda una variedad de vendedores.
- Compatibilidad con el software de terceros. Las interfaces de programa de aplicación (application program interface - API) de los sistemas abiertos deben correr en sistemas operativos estándar.

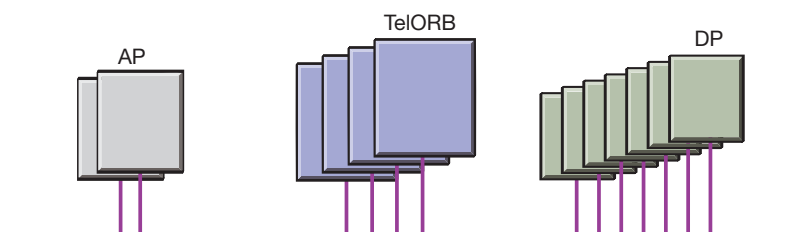
Los sistemas basados en TelORB soportan todos y cada uno de estos requisitos. Sin embargo, los procesadores que funcionan bajo el sistema operativo TelORB no cumplen el último requisito; éste es cumplido por los procesadores adjuntos del sistema, que usan UNIX o Windows NT.

Arquitectura general

En pocas palabras, TelORB proporciona un entorno para aplicaciones que controlan el tráfico y requieren una respuesta flexible en tiempo real, alto caudal, alta disponibilidad (tiempo fuera de servicio de un minuto al año), y escalabilidad (en el sentido de que la capacidad pueda ser incrementada añadiendo procesadores). De las aplicaciones que funcionan en TelORB se espera que presten servicio a numerosos usuarios finales del sistema. Se debe permitir que estas aplicaciones evolucionen o sean desarrolladas continuamente, ofreciendo así nuevos servicios a los usuarios. TelORB no proporciona un entorno para controlar dispositivos de hardware pequeños y de bajo coste que requieren una rigurosa respuesta en tiempo real (por ejemplo, comportamiento acotado de peor caso). Ni tampoco proporciona un entorno de programación estándar en el que pueda ser integrado software de terceros e introducir rápidamente adaptaciones a la medida — estos entornos son acomoda-

Figura 1

Los sistemas basados en TelORB pueden constar de procesadores funcionando bajo el sistema operativo TelORB, procesadores adjuntos bajo UNIX o Windows NT, y procesadores de dispositivo en los que corran sistemas operativos comerciales en tiempo real incorporados.



dos en la arquitectura del sistema por procesadores de dispositivo (device processors - DP) y procesadores adjuntos (adjunct processors - AP).

Los DP son típicamente procesadores de bajo coste que controlan un puñado de dispositivos de hardware. Mientras la huella de memoria del sistema operativo del DP puede ser un tema a considerar, los DP requieren pocos (si es que alguno) componentes de "middleware", y el software de aplicación es bastante estable y está bien definido. Para simplificar aún más el software del DP, los DP son propiedad de aplicaciones que funcionan en TelORB y están gestionados por ellas.

Los AP funcionan con sistemas operativos estándar, tales como UNIX o Windows NT. Albergan componentes de plataforma relacionados con la operación y el mantenimiento (O&M), tales como bases de datos fuera de línea para consultas complejas, facilidades de anotación, y modelos de gestión, pero también pueden albergar software de aplicación, tales como pilas de protocolo adquiridas o servicios especializados de usuario final. Un sistema puede estar compuesto por hasta varios centenares de DP, dos o más procesadores TelORB, y uno o más AP (Figura 1).

Dentro del sistema, una o más redes conectan los diversos procesadores entre sí.

Los sistemas TelORB del presente usan Ethernet dual, pero se puede usar ATM o prácticamente cualquier otra solución de red siempre y cuando se proporcione un ancho de banda adecuado. Incluso pueden coexistir diferentes redes con diferentes medios en el mismo sistema: los procesadores TelORB y los procesadores adjuntos podrían, por ejemplo, estar interconectados con Ethernet, mientras que se podría acceder a los procesadores a través de una red ATM.

El protocolo de comunicación entre procesadores (inter-process communication - IPC) de TelORB se usa para transportar datos entre procesadores TelORB. Una variación de ese protocolo se usa entre TelORB y los procesadores de dispositivo. El protocolo de datagrama de usuario/protocolo de Internet (user datagram protocol/Internet protocol - UDP/IP) estándar y el protocolo de control de transmisión/protocolo de Internet (transmission control protocol/Internet protocol - TCP/IP) se usan para transportar datos entre TelORB y los procesadores adjuntos. Las aplicaciones pueden usar CORBA o diálogos (cuando están dentro de TelORB) superpuestos a estos protocolos de transporte. TelORB se comunica directamente con los procesadores de dispositivo a través de la capa de transporte.

Características

Algunas propiedades clave de TelORB son sus características de tiempo real y su soporte de operación continua.

Tiempo real

TelORB está destinado a ser usado con aplicaciones flexibles de tiempo real; esto es, aplicaciones con comportamiento dependiente de la carga estadísticamente determinístico. El soporte para aplicaciones rigurosas en tiempo real (aplicaciones con comportamiento acotado de peor caso) afecta al desempeño y a la flexibilidad de la aplicación y por lo tanto se deja a los procesadores de dispositivo.

Prioridades y programación

Los procesos se ejecutan en uno de cuatro niveles de prioridad: alto, normal, bajo, y segundo plano. El nivel normal se destina a aplicaciones ordinarias de tráfico de telecomunicaciones, mientras que el nivel bajo se destina a mante-

CUADRO A, ABREVIACIONES Y TÉRMINOS

| | |
|--|--|
| AP Adjunct processor | MI Managed item |
| API Application program interface | MIB Managed information base |
| ATM Asynchronous transfer mode | NTP Network time protocol |
| Callback function Función que es llamada como resultado de un evento externo (por ejemplo, un mensaje entrante en un enlace de comunicación) | O&M Operation and maintenance |
| CORBA Common object request broker architecture | OMG Object management group |
| DBMS Database management system | ORB Object request broker |
| Delos Uno de los lenguajes de especificación de interfaz en TelORB | OS Operating system |
| delux El compilador Delos | OU Object unit |
| DOA Database object agent | Persistent object Database object |
| DP Device processor | SCC Source code component |
| DU Distribution unit | Scheduling queue Cola en el núcleo (kernel) que alberga procesos que están listos para ser ejecutados |
| Forlopp Cadena de procesos interconectados resultantes de un evento externo; se pueden asignar varios recursos durante esta cadena de procesos a fin de gestionar el evento originante | Supervisory mode Modo de ejecución en un microprocesador que contiene el juego de instrucciones completo; los procesos de aplicación se ejecutan en el modo de usuario, que tiene un juego de instrucciones ligeramente limitado |
| IDL Interface definition language | SWI Software interface |
| IDP Internal delivery package | TCP/IP Transmission control protocol/Internet protocol |
| IIOIP Internet inter-ORB protocol | TMN Telecommunication management network |
| IPC Inter-process communication | Trigger (database) Función opcional definida por el usuario que es llamada cuando tienen lugar determinados eventos (por ejemplo, cuando se crea o elimina un objeto de base de datos) |
| LM Load module | UDP User datagram protocol |
| LPC Linked procedure call | Zone Agrupación de procesadores TelORB interconectados |
| Managed object Objeto gestionado, Objeto al que se puede acceder desde el sistema O&M | |

nimiento. El nivel de alta prioridad se debe usar frugalmente para procesos que involucren la prestación de servicio al hardware. El nivel de segundo plano se podría usar para auditorías y pruebas de diagnóstico de hardware. TelORB usa una simple táctica de programación de tareas: el proceso con la prioridad más alta que esté listo para ejecutarse es autorizado a hacerlo durante como máximo un intervalo de tiempo (alrededor de dos milisegundos) hasta que es bloqueado, queda en reposo, o su intervalo de tiempo expira. Si su intervalo de tiempo expira, el proceso se pone en cola el último en su nivel de prioridad. Este procedimiento se repite entonces con el siguiente proceso de más alta prioridad que esté listo para ser ejecutado.

Obviamente, los retardos medios en la programación dependen de la carga del procesador. TelORB tiene un mecanismo de regulación de la carga que permite que las aplicaciones rechen partes de las operaciones de tráfico, a fin de sostener las prestaciones del tiempo real.

Núcleo interrumpible

Permitiendo que las operaciones dentro del núcleo del SO sean interrumpidas, el tiempo de respuesta a la interrupción pueda ser mantenido dentro de límites razonables. Sin embargo, para evitar que el núcleo llegue a ser excesivamente complejo y dado a errores, una rutina de servicio de interrupciones restringe el número

de operaciones a las que se permite correr en el núcleo. Una de esas operaciones es la programación de una rutina de servicio de interrupción retardada, que (debido a que está sincronizada con respecto a otras operaciones del núcleo) puede usar operaciones adicionales. Se aconseja que las aplicaciones hagan la mínima cantidad de trabajo en la rutina de servicio de interrupción real, posponiendo el resto del trabajo para la rutina de servicio de interrupción retardada.

Operación continua

Hoy día, cada vez más sistemas deben funcionar sin paradas — se espera de ellos que presten servicio durante todo el año, 24 horas al día, siete días a la semana. TelORB fue diseñado específicamente para este tipo de sistemas. Su hardware de protección de memoria protege a los sistemas de los fallos de aplicación corrientes, y su software tolerante a fallos permite las actualizaciones en servicio.

Protección de memoria

Los procesos de TelORB tienen su propio espacio de memoria, que no puede ser manipulada desde otros procesos. Los datos del núcleo de SO están también protegidos de la manipulación por procesos. Por lo tanto, los errores en cualquier proceso dado no pueden afectar a otros procesos. Esto reduce al mínimo el impacto que un error de software podría tener en el conjunto del sistema.

Tolerancia a fallos implementada en el software

Para gestionar los errores de hardware y los del núcleo del SO y recuperarse de ellos, TelORB reconfigura los procesos de un procesador que esté fallando a otros procesadores del sistema. En consecuencia, los sistemas TelORB no requieren un caro hardware tolerante a fallos (Cuadro B).

Redundancia de la red

Para gestionar situaciones catastróficas (terremotos y fuegos, por ejemplo), TelORB soporta la opción de tener un sistema redundante, geográficamente separado que funcione en reserva activa (standby). El sistema redundante es actualizado continuamente durante las operaciones normales y puede hacerse cargo de la operación inmediatamente sin ninguna pérdida de datos.

Actualización en servicio

El nuevo software puede ser cargado y puesto en funcionamiento mientras el sistema está funcionando y proporcionando servicio. Obviamente, esto requiere la cooperación entre TelORB y los programas de aplicación, que es facilitada por el marco de implementación de procesos. Debido a que se usan los mismos mecanismos para las actualizaciones en servicio y para reconfigu-

CUADRO B: NIVELES DE RECUPERACION EN TELORB

TelORB soporta los siguientes niveles de recuperación:

- Reanudación de transacción de base de datos. La aplicación es informada cuando una transacción es incapaz cumplirse con éxito.
- Aborto /reinicio de proceso. Si un proceso estático encuentra un error interno del cual no puede recuperarse (por ejemplo, división por cero) se reinicia. De forma similar, si un proceso dinámico encuentra un error interno del cual no puede recuperarse, se aborta.
- Recuperación Forlopp. TelORB informa a los procesos que están interconectados con enlaces de comunicación cuando el extremo remoto del enlace desaparece. Esto permite a la aplicación limpiar todos los recursos asignado a una determinada cadena de eventos.
- Recarga del procesador. TelORB intenta recargar el procesador después de que se ha producido un error en el hardware o en el software del código del núcleo.
- Recarga de zona. Si existe el riesgo de inconsistencia en el estado del sistema — por ejemplo, cuando dos o más procesadores fallan simultáneamente — TelORB recarga todo el grupo con el respaldo más reciente.

rar el sistema, los operadores pueden verificar fácilmente los aspectos de un programa de aplicación que vayan a ser actualizados.

Protección de sobrecarga

En situaciones raras, los eventos del mundo exterior crean una carga desproporcionada en los sistemas, agotando los recursos del mismo y dejando simultáneamente varios procesadores inútiles. TelORB protege al sistema de las situaciones de este tipo midiendo la longitud de la cola de programación de tareas y rechazando los intentos de preparación de diálogo cuando la longitud de la cola excede los límites prefijados. En este caso, la respuesta del sistema se ralentiza y se llevan a cabo menos operaciones con éxito.

Último recurso rápido

Aunque se han tomado muchas precauciones para proteger al sistema, siempre hay una remota posibilidad de que falle completamente. Por ejemplo, es concebible que varios procesadores pudieran fallar al mismo tiempo, haciendo a imposible que TelORB mantenga un estado consistente del sistema sin revertir a un estado previamente guardado. Por ello, como último recurso, TelORB recarga todos los procesadores con un respaldo de la base de datos. Para reducir al mínimo el tiempo fuera de servicio, TelORB usa un protocolo de carga de coerción múltiple (multicasting) que habilita la carga paralela de todos los procesadores en el sistema sin un cuello de botella en el medio de comunicación, y sin requerir que se pongan discos en cada procesador. Si se está usando la propiedad de redundancia de la red, entonces incluso las situaciones "catastróficas" pueden ser gestionadas sin pérdida de datos.

Entorno de programa

Los programas escritos para TelORB se ejecutan como uno ó más procesos cooperantes usando la base de datos para almacenar los parámetros de configuración y otros datos que se necesita que persistan. Los programas pueden usar varios servicios del sistema operativo a través de la API de TelORB. Los programas se escriben en C/C++ o Java estándar, con interfaces de interoperabilidad especificadas en el lenguaje de definición de interfaz (interface definition language - IDL) de CORBA. Delos, un lenguaje de especificación exclusivo, proporciona los elementos constructivos de que carecen los lenguajes de programación estándar para especificar procesos y objetos de base de datos (Figura 2).

Procesos

Especificaciones

Los procesos que corren en TelORB se especifican en Delos, que asigna un nombre y características a cada tipo de proceso. Desde el tipo de

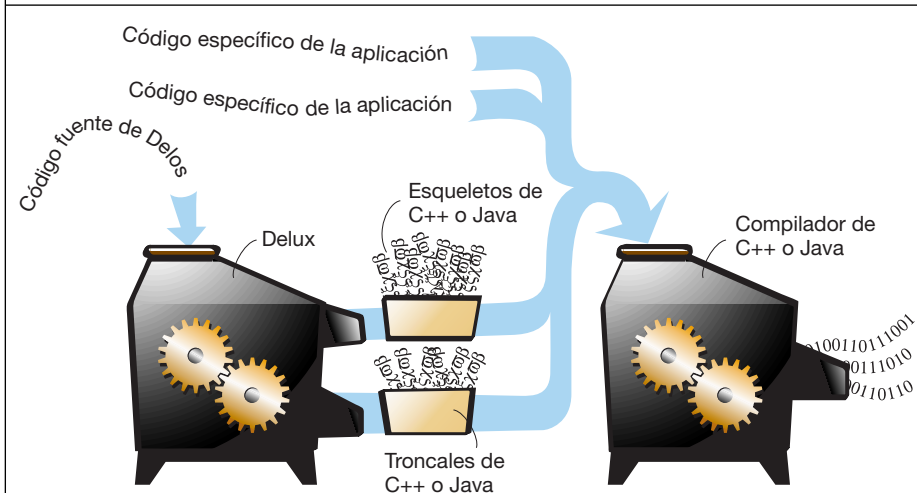
proceso, TelORB crea instancias de proceso como se define en las especificaciones.

Los procesos se declaran como estáticos o dinámicos. Las instancias de procesos estáticos, que se crean cuando se ponen en marcha el sistema o cuando se instala el proceso, se vuelven a crear después de un fallo. Las instancias de procesos dinámicos, que se crean cuando se dirige a ellos otro proceso, no se vuelven a crear después de un fallo.

La especificación de tipo de proceso indica si una instancia de proceso se ha de replicar o no en varios procesadores. Cuando se replica una instancia de proceso, TelORB dirige cualquier otro proceso que quiera cooperar con ella a su réplica (si existe una) en el mismo procesador. Si no existe una réplica, el proceso es dirigido a una réplica seleccionada arbitrariamente en algún otro procesador. Nota: las réplicas no conocen la existencia de la otra a menos que así lo requiera la aplicación.

Finalmente, la especificación del tipo de proceso indica cómo se han de diferenciar e instalar múltiples instancias del mismo tipo. Por ejemplo, como alternativa a ser simplemente instanciadas en cualquier sitio cuando se dirigen a ellas, las instancias de tipo de proceso dinámico pueden ser diferenciadas por una clave primaria (que es consistente con los correspondientes objetos de la base de datos). Por lo tanto TelORB puede dirigir varias llamadas con la misma clave a la misma instancia hasta que se termina. Para tipos de procesos estáticos, las instancias pueden ser creadas automáticamente cuando se pone en marcha el sistema, o el software de aplicación puede instalarlas mediante la API de TelORB. Esta opción está particular-

Figura 2
El código fuente de Delos se introduce en delux (el compilador de Delos), que genera archivos troncales y esqueletos en C++ / Java. Entonces se agrega manualmente el código específico de la aplicación a los archivos esqueleto antes de la compilación final (compilador C++ o Java) junto con otros códigos de aplicación. El código fuente IDL se maneja de forma similar.



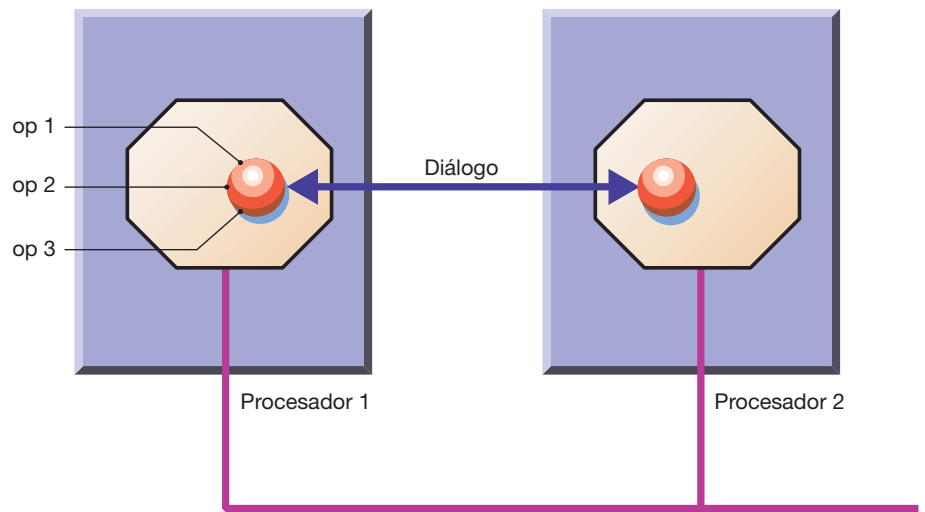


Figura 3
 Los procesos cooperan por medio de diálogos, que son especificados como un par de tipos de objetos que tiene operaciones (en este ejemplo, op1, op2 y op3 pueden ser llamadas desde el proceso en el procesador 1).

mente bien adecuada a instancias de procesos que estén directamente asociados con la configuración de hardware, que podrían variar de sitio a sitio y a medida que se extienden las plantas.

Marco de implementación

Los procesos especificados se implementan dentro de un marco dado por código de C++ o Java que fue generado desde la especificación de Delos. El marco da puntos de entrada para gestionar

- arranque (inicialmente, y después de caídas del sistema);
- la reconfiguración de instancias entre procesadores;
- actualizaciones de software en servicio; y
- terminación.

Cuando TelORB inicia una instancia de proceso estático, también indica (en la instancia) una de tres razones para ponerlo en marcha. O bien la instancia fue creada por primera vez, o ha sido recreada después de una caída del sistema. Alternativamente, la instancia fue recreada después de una recarga del sistema. En este caso, se produjo un serio error del sistema haciendo imposible preservar la consistencia de la base de datos. En consecuencia, el sistema fue recargado con un respaldo consistente de la base de datos, que por necesidad no contenía las actualizaciones más recientes. Por lo tanto, el estado del hardware podría ser inconsistente con el estado de la base de datos. No obstante, las aplicaciones que puedan diferenciar entre ambas cosas usualmente aceptan el estado del hardware.

Para facilitar las reconfiguraciones que no perturben y las actualizaciones en servicio, TelORB permite a los procesos dinámicos que corren en el procesador original (con el software original) terminar de manera natural en un periodo de tiempo. Los nuevos procesos (que corren el nuevo software) se crean en el nuevo procesador. Para procesos estáticos, TelORB crea otra instancia en el nuevo procesador (con el nuevo software) que es autorizado a correr en paralelo con la vieja instancia. Se da a la nueva instancia una referencia a la vieja, para que las dos se puedan comunicar — por medio de un protocolo de transferencia de estado específico de la aplicación. Otros procesos del sistema perciben el par como una sola instancia. No obstante, si no se debe perturbar la operación, se pueden necesitar ciertas provisiones especiales para la cooperación específica de la aplicación entre los procesos.

El paradigma de la ejecución

Toda la ejecución dentro de un proceso TelORB tiene lugar como la ejecución de funciones de retrollamada (lo más frecuentemente como funciones miembro de instancias de clase de C++ o Java) asociadas con eventos que son externos al proceso. Por lo común, el programa no puede elegir inhabilitar la gestión de eventos — éstos son ejecutados en el orden en el que tienen lugar. Por lo tanto, los programadores deben adaptarse a un mundo exterior inherentemente asíncrono. Sin embargo, la gestión de todos los eventos se serializa. Por lo tanto, los programadores no necesitan preocuparse de los problemas de la programación concurrente (una

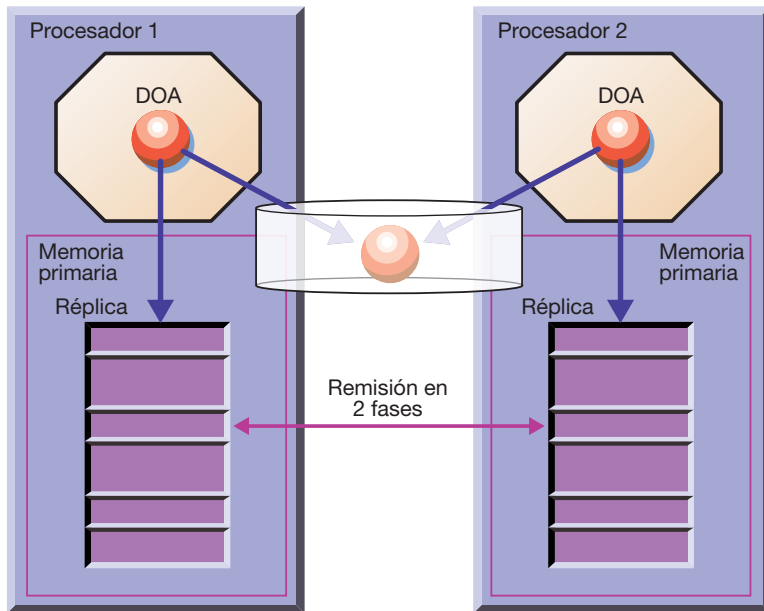


Figura 4
El agente de objetos de la base de datos (DOA) hace que los objetos de la base de datos aparezcan como objetos ordinarios pertenecientes a una base de datos global. Sin embargo, en realidad, cada objeto es almacenado como una réplica local en la memoria primaria del procesador correspondiente. El del sistema de gestión de la base de datos (database management system - DBMS) gestiona el acceso a los datos y mantiene la consistencia de los datos replicados en diferentes procesadores. Las réplicas de datos son actualizadas automáticamente usando un protocolo de remisión en dos fases.

función de retrollamada se finaliza antes de que se llame a la siguiente, lo que significa que de manera efectiva hay solamente una cadena de ejecución en un proceso TelORB). Además del paradigma de ejecución, los programadores son libres de hacer uso de cadenas livianas. Sin embargo, en Java, se soportan las cadenas múltiples como se describe en la especificación del lenguaje.

Como una excepción a este esquema, TelORB proporciona realmente mecanismos para bloquear la cadena de ejecución de solo proceso. En este estado, no puede ser desbloqueado excepto por un solo evento correspondiente o expiración de tiempo.

Cooperación de procesos

Los procesos cooperan mediante la invocación a operaciones remotas que están recopiladas en la noción de Delos de los diálogos. Un diálogo se especifica como un par de tipos de objetos que tiene operaciones, y a veces resultados. Las clases de C++ son generadas desde la especificación. Algunas clases ordenan y desordenan las operaciones con sus argumentos; otras proporcionan el marco para implementar las acciones de invocaciones dentro de los procesos (Figura 3).

Para preparar un diálogo, un proceso se dirige a otro proceso por su tipo y, allí donde se necesiten, datos que diferencian una instancia particular. Después de que el diálogo ha sido preparado, se establece una conexión entre cada miembro, para permitir a los miembros señalar en el evento que uno de ellos muere. Cuando los procesos terminan, un procedimiento de

cierre de diálogo cierra la conexión de forma segura.

La base de datos

En tanto que los datos dentro de un proceso son volátiles (se pierden si hay una caída del proceso o del procesador en el que corre), los datos almacenados en la base de datos de TelORB persisten incluso después de que se produzca una caída del proceso o del procesador. Además de almacenar los datos, la base de datos comparte datos entre procesos. La base de datos de TelORB es una base de datos orientada a objetos en tiempo real que almacena datos en memoria primaria en el procesadores (Figura 4).

Especificaciones

Los datos de la base de datos constan de instancias de tipos de objetos persistentes (especificados en Delos) que tienen atributos de los datos almacenados persistentemente y un conjunto de métodos asociado que la aplicación pueden usar para manipular los datos.

Los atributos incluyen tipos de datos corrientes — entero, enumeración, registro, y así sucesivamente — así como un tipo de datos que es específico de la base de datos de TelORB, y que alberga referencias a otros objetos de la base de datos, muy parecido a los punteros en los lenguajes de programación. Estos atributos de referencia pueden tener un solo valor o un valor múltiple. Los tipos de objeto pueden estar derivados de otros tipos, en cuyo caso el comportamiento de los métodos se convierte en polimórfico; esto es, una aplicación puede abrir un objeto de un tipo básico y encontrar una instancia

de un tipo especializado. Cuando se invocan métodos, se ejecuta la implementación de la especialización. Delos permite también que se especifiquen ciertas propiedades para atributos; por ejemplo:

- atributo del tipo array puede tener una propiedad de acceso a elemento, lo que significa que en vez de extraer la totalidad del atributo, se pueden extraer de la base de datos elementos individuales del array;
- un atributo de tipo referencia puede tener una propiedad inversa (en este caso, un atributo de tipo referencia en el objeto referenciado debe volver a referirse al referente); y
- los atributos pueden ser opcionales, haciendo posible reducir el gasto almacenar los valores por omisión para atributos grandes.

Las clases de C++ y Java, que proporcionan a la base de datos las interfaces necesarias y el marco para implementar métodos mecanismos de activación (triggers), se generan desde las especificaciones de Delos.

Operaciones

Con los tipos de objetos así especificados, las aplicaciones pueden

- crear nuevos objetos;
- abrir objetos existentes;
- traer los valores de atributos y asignarlos nuevos valores;
- invocar métodos; y
- cerrar y borrar objetos.

La consistencia de la base de datos se mantiene agrupando las operaciones en transacciones atómicas: o se llevan a cabo todas las operaciones que hay dentro de una transacción o no se lleva a cabo ninguna operación. TelORB mantiene cierres sobre objetos para evitar que varios pro-

cesos cambien los mismos datos simultáneamente. También se permite a las aplicaciones introducir sus propias comprobaciones de integridad, implementando funciones de activación llamadas durante determinadas operaciones.

Replicación

Los datos de la base de datos se almacenan enteramente en la memoria primaria de los procesadores, lo que hace la operación muy rápida. Para sobrevivir a las caídas del sistema, los datos se replican en al menos dos procesadores. Para replicar los cambios rápidamente y con seguridad se emplea un protocolo de remisión de dos fases extendido y optimizado.

Redundancia de red

Los mecanismos para redundancia de red están estrechamente relacionado con la base de datos, ya que siempre sincronizan el contenido de la base de datos de la sección en reserva activa (standby) geográficamente separada con datos en un sistema activo.

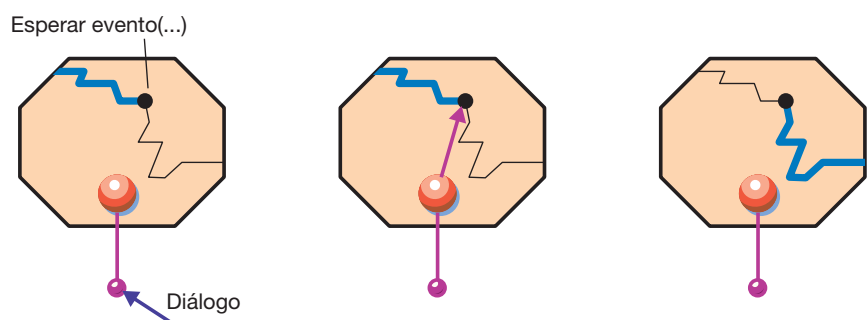
Búsqueda

Para que un objeto sea encontrado en la base de datos, debe o bien tener un atributo designado como clave primaria, que identifique de manera única una instancia de un tipo particular, o bien debe ser referenciado desde un atributo del tipo referencia de algún otro objeto de base de datos.

Los objetos de la base de datos pueden ser también encontrados por medio de iteradores, que extraen cada objeto de un tipo particular (que coincide con los criterios de selección definidos para el iterador). Los iteradores pueden ser aplicados a la totalidad de la base de datos o a un atributo de tipo de referencia de valor múltiple.

Figura 5

En ejemplo del uso de eventos. Un método que recibe un determinado mensaje en un diálogo puede inscribir un evento en una cadena que está monitorizando el evento. Cuando la cadena llama al método "WaitForEvent" (esperar evento), toma cualquier evento que ya haya sido inscrito o, si no lo hay, espera a que se inscriba uno. Esto proporciona un comportamiento síncrono al por otra parte inherentemente asíncrono modelo de programación de TelORB.



Cuando un iterador se aplica a la totalidad de la base de datos, solamente puede extraer tipos que tengan una clave primaria.

El modelo de O&M

Para O&M, TelORB requiere que el modelo de objeto especificado esté separado de la implementación actual de la aplicación. Este modelo de objeto consta de objetos CORBA, los cuales tienen atributos, acciones, y relaciones con otros objetos.

Los objetos gestionados de un sistema forma una base de información de gestión (management information base - MIB), que contiene información sobre el contenido de un objeto y sobre tipos de objetos. Esta información puede ser mostrada en pantalla mediante una aplicación gráfica.

El modelo de O&M también incluye notificaciones, que el sistema envía al operador, para informarle de eventos particulares. Una notificación está siempre asociada a un objeto de CORBA en particular. Un tipo especial de notificación es la alarma, que se envía cuando el sistema requiere la intervención del operador (un ejemplo típico es cuando una placa de procesador ha fallado y debe ser reparada).

Servicios

TelORB proporciona a las aplicaciones varios servicios, mediante

- la API de TelORB — por medio de clases de C++ y Java (allí donde las clases estándar de Java no son suficientes);
- Tipos de objetos Delos;
- Delos y el código generado desde las especificaciones; e
- IDL de Corba y el código generado desde las especificaciones.

Cadenas y eventos

Las cadenas livianas de TelORB están diseñadas para ser usadas con eventos especificados en Delos. Un evento de Delos es un tipo de mensaje que se envía desde un objeto a cualquier cadena que monitoree el evento. La intención ha sido proporcionar una manera de añadir o cambiar cadenas que ejecuten flujos de control sin tener que cambiar la implementación de objetos que representen recursos con un comportamiento más estático.

Una nueva cadena se crea simplemente instanciando una clase de C++ derivada desde una clase de base de cadena en la API de TelORB. El programa principal para la nueva cadena consta de una función de miembro virtual sustituido. El programa ejecutado por la cadena típicamente monitoriza varios eventos desde diferentes objetos u, donde sea apropiado, espera cualquier evento de un conjunto selecto de ellos, adoptando las medidas apropiadas en respuesta al evento recibido (Figura 5). Estas cadenas livianas y eventos se usan solamente para código

C++. El lenguaje Java proporciona sus propias cadenas y primitivas de sincronización.

Temporizadores

Un proceso puede usar temporizadores de una vez así como periódicos, que pueden ser ajustados para que expiren en pasos de cinco milisegundos, desde diez milisegundos hasta alrededor de 20 días. Cuando el temporizador expira, TelORB ejecuta una función de retrollamada. Un temporizador puede ser cancelado en cualquier momento antes de que la función de retrollamada haya sido ejecutada.

Reloj y calendario

TelORB proporciona funciones de reloj de tiempo real y de calendario para convertir el formato interno a un formato de calendario estandarizado. Se ha preparado soporte para calendarios locales, incluyendo ajustes para la hora de ahorro con luz solar, pero no está totalmente implementado. El reloj de tiempo real está sincronizado con diferentes procesadores en virtud de una adaptación de TelORB del protocolo de tiempo de red (network time protocol - NTP).

Instalación de proceso estático

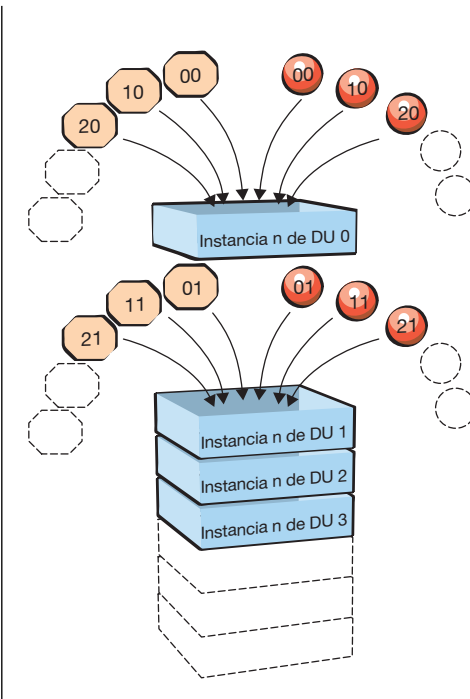
Como se indicó anteriormente, las instancias de procesos estáticos pueden ser instaladas mediante software de aplicación a través de la API de TelORB. Dado que estas instancias se relacionan típicamente con la configuración de hardware, las aplicaciones deben hacer comprobaciones a menudo para ver en qué procesador debe correr un proceso. Por esta razón, la aplicación instala una instancia de proceso para creación dentro de un grupo de procesadores. TelORB proporciona ciertos grupos de procesadores predefinidos, tales como "todos los procesadores", pero, por lo general, cada aplicación debe proporcionar sus propios grupos de procesadores apropiados y registrarlos mediante la API de TelORB.

Gestión de los errores de software

Básicamente, cualquier proceso en el que se haya detectado un error se considera un proceso malo que debe ser terminado inmediatamente — dando un volcado de error en el sistema para la localización del fallo. TelORB termina automáticamente los procesos con errores detectados por el hardware (referencias de punteros cero, por ejemplo). Pero para errores detectados por el software de aplicación, TelORB proporciona una interfaz mediante la cual se puede adjuntar información sobre el error al volcado. Después de la terminación, se recrea una instancia de proceso estático de la manera usual.

Los procesos que cooperan con un proceso malo son notificados por medio de indicaciones de aborto de diálogo. TelORB no terminará automáticamente un proceso que reciba una indicación de aborto de diálogo, pero deja al pro-

Figura 6
 Para los procesos dinámicos que se identifican mediante una clave, las instancias de varios procesos y los correspondientes objetos de base de datos pueden ser agrupados en una instancia de unidad de distribución (distribution unit - DU). TelORB garantiza que el contenido de la DU se mantenga intacto independientemente de las reconfiguraciones, asegurando así que el procesador siempre tiene acceso local desde uno de estos procesos al objeto de base de datos correspondiente.



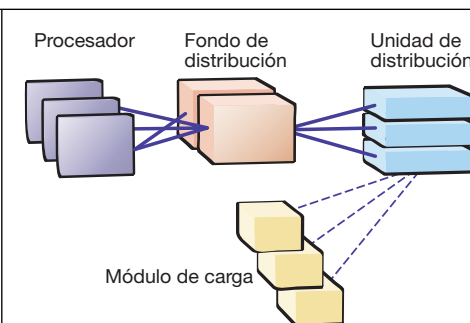
grama de aplicación seleccionar una acción o respuesta apropiada.

Drivers

Los drivers proporcionan el control a bajo nivel del hardware. Son programas que se ejecutan en el modo supervisor del procesador, que tiene el mismo modo privilegiado que el núcleo. Las aplicaciones pueden usar sus propios drivers para controlar hardware específico de la aplicación. La API de TelORB proporciona servicios para implementar drivers y para acceder a ellos desde los procesos.

TelORB da a los drivers la siguiente funcionalidad: temporizadores, instalación de rutinas de servicio de interrupciones, control de nivel de interrupción, gestión de memoria, acceso seguro a datos del proceso, el uso de otros drivers, y la capacidad de enviar señales a usuarios de dri-

Figura 7
 Adjuntando unidades de distribución a fondos de distribución y asignando procesadores a los fondos en la configuración, TelORB puede determinar qué código se necesita para ejecutar las unidades de distribución en un procesador en el fondo.



vers en procesos. Los drivers pueden ser abiertos, cerrados, y controlados desde un proceso.

Distribución

Para sacar el mayor partido de la plataforma del procesador distribuido, los procesos y los datos deben estar distribuidos de una manera que equilibre la carga del procesador y reduzca al mínimo la comunicación entre procesadores. En lo que se relaciona con TelORB, dos enfoques simplistas son o bien distribuir todas las instancias de objetos de procesos bases de datos arbitrariamente, o bien dejar que el ingeniero de la planta especifique la distribución de cada instancia individual. El primer enfoque probablemente daría como resultado un mal desempeño, mientras que el segundo enfoque impondría requisitos no realistas a los ingenieros de la planta. Además, cada uno de ellos llenaría la memoria de tablas con las direcciones de diferentes instancias.

TelORB permite al desarrollador de aplicaciones para agrupar instancias individuales en un número manejable de unidades de distribución (distribution units - DU). Este agrupamiento se hace típicamente por tipo de proceso o tipo de objeto de base de datos, asociando una especificación de tipo de unidad de distribución de Delos con la especificación del proceso o tipo de objeto de base de datos. El tipo de unidad de distribución especifica el número de unidades de distribución sobre el cual se deben extender las instancias. Cuando más de una unidad de distribución ha sido especificada para el tipo, el programa de aplicación debe suministrar una función que establezca la relación entre las instancias y las unidades de distribución. TelORB permite también que las instancias de procesos y los objetos de base de datos sean agrupados o colocados en la misma unidad de distribución. Esto reduce al mínimo la comunicación entre los procesadores que se necesitan para manipular los objetos de la base de datos (Figura 6).

Cuando el ingeniero de la planta configure la misma, debe agrupar procesadores en fondos de distribución y asignar tipos de unidad de distribución a fondos. TelORB configura (en tiempo de ejecución) a procesadores dentro del fondo unidades de distribución individuales a partir de los tipos, y las reconfigura como sea necesario; por ejemplo, cuando un procesador sufre una caída, cuando el sistema se expande mediante la adición de un nuevo procesador, y cuando se cambia la asignación de un procesador a un fondo (Figura 7).

Gracias a las unidades de distribución, los tamaños de la tabla permanecen dentro de magnitudes manejables. La experiencia ganada hasta ahora indica que el diseñador de aplicaciones debe decidir también qué fondos usar y qué tipos de unidades de distribución se deben asignar a ellos. Por lo tanto, esta información pertenece al paquete de entrega interno (internal delivery package - IDP).

Modelo de memoria

TelORB divide el espacio de direccionamiento lógico del procesador en tres partes principales:

- espacio de memoria de instrucciones (alberga el código que se ha de ejecutar);
- espacio de memoria de datos del núcleo (alberga los datos para el núcleo de TelORB y los drivers); y
- espacio de memoria de datos de proceso (alberga los datos para el proceso que está actualmente en ejecución).

Todos los procesos comparten el código en el espacio de memoria de instrucciones y pueden leer (pero no escribir) datos del espacio de memoria de datos del núcleo. Este espacio se usa para acelerar el acceso a la base de datos, y podría ser también explotado por los drivers.

El espacio de datos del proceso se mapea a diferente memoria física cuando se despachan diferentes instancias de proceso. De esta forma, los procesos quedan aislados entre sí, de manera que un error en un proceso no puede afectar a otro proceso (Figura 8).

Separación de módulos de carga

El código se carga en módulos de carga que no están directamente relacionados con tipos de procesos. Aunque un proceso se implementa en un módulo de carga, también puede ejecutar código en cualquier número de otros módulos de carga por medio de un mecanismo denominado llamada a procedimiento enlazado (linked procedure call - LPC). El mecanismo LPC se usa para implementar métodos de tipo de objeto de base de datos y activadores (Figura 9).

Comunicaciones externas

Para la comunicación con el mundo exterior, TelORB proporciona protocolos comunes de Internet y los medios de implementar protocolos específicos de aplicación, según se necesite.

Protocolos de Internet

TelORB implementa TCP/IP y UDP/IP para comunicarse con otros sistemas.

Protocolos de CORBA

TelORB soporta el protocolo de interoperabilidad de Internet (Internet inter-operability - IIOP), que se transporta mediante TCP/IP.

Entorno de desarrollo

TelORB viene con un entorno de desarrollo que corre en una estación de trabajo UNIX e incluye un modelo de estructura de software, soporte de construcción, y herramientas para configurar una planta. Los programas de aplicación, que se pueden correr en el anfitrión en procesadores simulados, usa un depurador a nivel de código fuente y otras herramientas para encontrar fallos en el código.

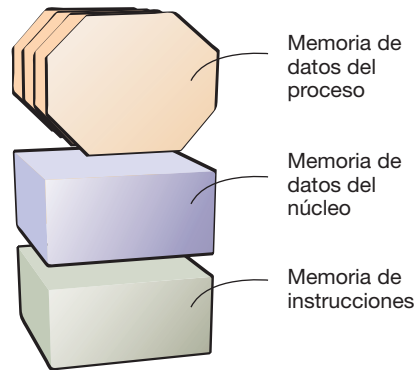


Figura 8
Aunque cada instancia de proceso tiene su propio espacio de memoria, todas las instancias de proceso comparten la memoria de instrucciones y la memoria de datos del núcleo. Los datos del núcleo solo pueden ser escritos mediante llamadas del SO.

Modelo de estructura de software

El modelo de estructura de software está basado en elementos gestionados (managed items - MI), que tienen atributos (por ejemplo, una identidad y versión) y relaciones con otros MI. También hay una estructura de archivos asociada con cada tipo de MI.

Elementos gestionados básicos

Los tipos más básicos de MI son

- el módulo de carga (load module - LM);
- el paquete de envío interno (internal delivery package - IDP);
- la unidad objeto (object unit - OU);
- el componente de código fuente (source code component - SCC); y
- la interfaz de software (software interface - SWI).

El módulo de carga contiene el archivo de código

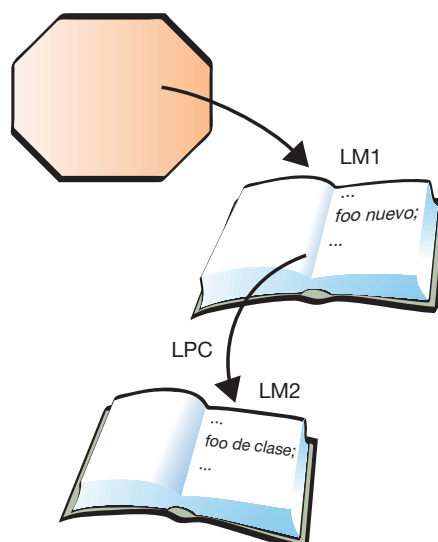


Figura 9
Por medio del mecanismo de llamada enlazada de procedimiento (linked procedure call - LPC), un proceso que ejecuta código en un módulo de carga (load module - LM) puede crear y usar objetos definidos en otro LM. Si el nuevo LM es actualizado, la nueva versión será usada sin tener que volver a enlazar (fuera de línea) el código que la usa.

go objeto a ser cargado en un procesador, donde es reubicado y ejecutado. Tiene una relación con varios unidades objeto cuyo código objeto se ha de incluir en el módulo de carga.

El paquete de envío interno recopila varios módulos de carga relacionados que deben ser usados juntos en el sistema meta, pero que podrían ser cargados en diferentes procesadores. También contiene información sobre los fondos de distribución a usar cuando se especifica la distribución de las DU que contiene.

Una unidad objeto contienen código fuente que se ha de incluir en tan solo un módulo de carga. En contraste, un componente de código fuente contiene código fuente de biblioteca cuyo código objeto se ha de enlazar a cualquier número de módulos de carga. Las unidades objeto y los componentes de código fuente proporcionan y usan interfaces de software.

Una interfaz de software contiene especificaciones de interfaz (especificaciones de Delos y archivos de cabecera de IDL y C/C++) que son compartidos por varios elementos gestionados. Cuando esté suministrada por una unidad objeto, una interfaz de software puede ser usada por cualquier número de unidades objeto, componentes de código fuente, u otras interfaces de software (Figura 10).

Soporte de construcción

Cuando el código fuente está estructurado de acuerdo con el modelo de elementos gestionados, puede automáticamente ser construido mediante el soporte de construcción que se envía

con TelORB. En particular, el soporte de construcción

- genera código C++ y Java a partir de especificaciones de Delos e IDL de CORBA y lo archiva en los directorios apropiados;
- compila el código generado y el código fuente de la aplicación;
- ensambla los módulos de carga (por ejemplo, determina que LM de un SCC han de ser incluidas); y
- reúne la información que necesita la herramienta de configuración de planta para asignar módulos de carga a los procesadores.

Lenguajes y compiladores

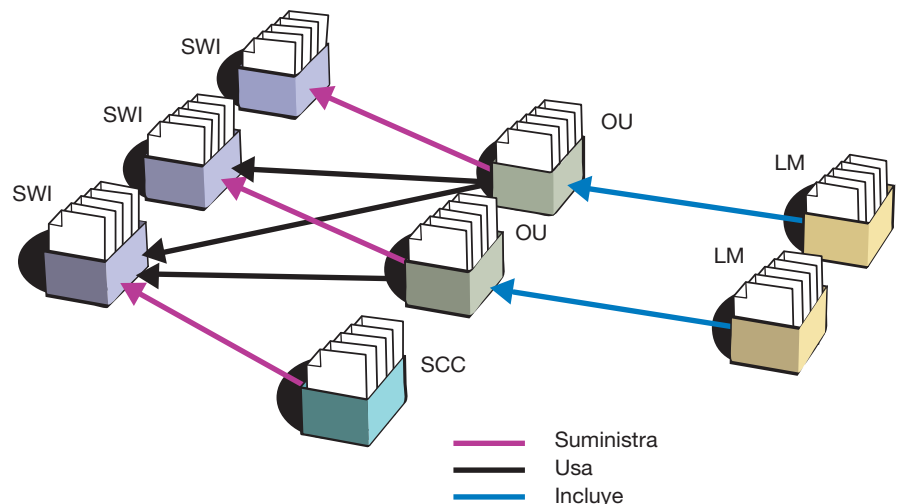
C/C++

Los programas de un sistema TelORB están escritos en su mayoría en C o C++ estándar. La implementación actual del compilador no soporta excepciones de C++. Se desaconseja el uso de plantillas hasta que se encuentre una forma aceptable para tratar con ellas en el soporte de construcción. El código C y C++ se compila con el compilador de C GNU, versión 2.7.

Java

TelORB proporciona un entorno de tiempo de ejecución para programas Java. TelORB soporta un subconjunto de Java con API añadidas para permitir a los programas Java hacer uso de los servicios de TelORB. En esencia, el subconjunto es el equivalente del Java 2 estándar con el soporte gráfico eliminado.

Figura 10
Relaciones entre los tipos más comunes de elementos gestionados.



IDL de CORBA

El IDL de CORBA se puede usar para especificar interfaces dentro del sistema TelORB y entre TelORB y otros sistemas.

Delos

Delos es un lenguaje de especificación exclusivo usado para especificar procesos y objetos de base de datos para los cuales C++ no tiene construcciones. Delos fue originalmente una familia de lenguajes usada para expresar interfaces, comportamiento (lenguaje de codificación), estructura de software, y distribución. Por lo tanto, TelORB es capaz de expresar interfaces y distribución. Con Delos, los desarrolladores pueden expresar tipos de datos, diferentes categorías de tipos de objetos, diálogos, notificaciones, tipos de procesos, y tipos de unidades de distribución.

Las especificaciones de Delos se compilan con el compilador de Delos, *delux*. El compilador se divide en una parte frontal, que analiza las especificaciones y produce un formato intermedio, y una parte trasera, que toma el formato intermedio y genera código C++ y Java más alguna otra información requerida por el sistema objeto.

Bibliotecas de programación

Para simplificar el desarrollo de aplicaciones, TelORB incluye las siguientes bibliotecas de programación:

- una biblioteca C estándar (menos unas pocas funciones que no encajaban en el entorno TelORB);
- una biblioteca de clase C++, que proporciona listas, colas, colecciones, y números aleatorios;
- las clases de bibliotecas que contienen implementaciones de tipos de datos de Delos, tales como los tipos de cadena de caracteres y cadena de octetos y otro soporte para código generado a partir de las especificaciones de Delos.

Configuración de planta

Los ingenieros configuran la planta pasando un programa llamado *epct*, que toma un archivo de entrada que describe la configuración y saca una estructura de archivo que puede ser transferida al sistema de archivos de TelORB. El archivo de entrada

- lista los paquetes de envío interno con software a ser ejecutado en el sistema;
- define los fondos de distribución;
- asigna los tipos de unidad de distribución a los fondos de distribución;
- crea representaciones de los procesadores del sistema; y
- asigna los procesadores a los fondos de distribución.

La estructura del archivo de salida contiene los

LM a cargar, una tabla de carga inicial para los primeros procesadores a cargar, y archivos que especifican qué operaciones de objeto gestionado se necesitan para poner el sistema en funcionamiento.

Depuración

Vega — un entorno simulado de procesador — es la principal herramienta para poner en marcha aplicaciones. Es un proceso UNIX con una capa de adaptación de “hardware” que le hace comportarse como un procesador corriente. Múltiples procesos Vega pueden ser interconectados para verificar los aspectos de distribución.

Se usan varias herramientas para la depuración, incluyendo la siguientes:

- *gdb*. El depurador de GNU para depuración de alto nivel de código C o C++. La herramienta puede ser extendida con reiniciación cíclica de gráficos, tales como *xemacs*. En la actualidad, se está desarrollando para TelORB un depurador distribuido de alto nivel y encadenamiento múltiple que soporta C++ y Java.
- *sysview*. Otra aplicación gráfica con la que los procesos pueden ser examinados según corren, y desde la cual se pueden iniciar y presentar los seguimientos.
- una herramienta de inspección basada en Telnet. Esta herramienta permite a los desarrolladores examinar los contenidos de la base de datos desde una ubicación remota.

Conclusión

El mercado para soluciones de red inteligente está creciendo rápidamente, lo que significa que habrá un incremento en la necesidad de plataformas con tiempo fuera de servicio cero que soporten una cantidad masiva de procesamiento orientado a las transacciones. Con frecuencia, los nodos de una red inteligente necesitan bases de datos ultrarápida para gestionar las solicitudes procedentes de un gran número de usuarios. La plataforma TelORB es muy adecuada para cumplir estos requisitos. Además, su excepcional escalabilidad permite a los operadores expandir gradualmente sus sistemas a medida que surge la necesidad.

Debido a que el sistema usa placas de procesador disponibles comercialmente “en las estanterías”, los operadores pueden sacar siempre ventaja de los más recientes adelantos en el diseño de hardware.

Las aplicaciones se construyen usando lenguajes bien conocidos, tales como C++ y Java, y se proporciona interoperabilidad mediante el agente de petición de objetos incorporado.

TelORB es una plataforma realmente abierta cuyas características, en términos de robustez y configuraciones flexibles, no tienen paralelo. En la actualidad está dispuesta como base para la plataforma Jambala.