Universidad de las Ciencias Informáticas
Facultad 4

Título: Salva Automática Centralizada

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor: Juan Carlos Delgado Cruz

Tutor: Ing. Iósev Pérez Rivero

Co-tutor: Ing. Mayté Concepción Sigler

Ciudad de La Habana, Junio del 2008

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Dirección Técnica de la Infraestructura Productiva de la Universidad de las Ciencias Informáticas; así como a dicho centro, para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Juan Carlos Delgado Cruz

_____

Firma del Autor

Ing. Iósev Pérez Rivero

_____

Firma del Tutor

DATOS DE CONTACTO

Ing. Iósev Pérez Rivero, graduado en el año 2007 en la Universidad de las Ciencias Informáticas de Ingeniería en Ciencias Informáticas. Obtuvo una certificación el 18 de marzo del 2008 por el Cuba-India Knowledge Center en Enterprise Application Development (J2EE Technologies) Level 1. Su categoría docente actual es Adiestrado. Es miembro del Grupo de Arquitectura de Software y Tecnologías de la Dirección Técnica de la Infraestructura Productiva.

AGRADECIMIENTOS

A mi abuela (Julia) por quererme tanto.

A mi Blanca por siempre pensar en mí.

A mi papá (Juan Luis) por su confianza y paciencia.

A mi pito (Arel) y mi hermana (Dunia) por su amor.

A mi preciosa Isabel por sus horas de desvelo.

A mi Mily por malcriarme tanto.

A mi hermano mayor Brailys por sus consejos.

A mi Baby por sus ñoñerías.

A mi Nena por su cocina.

A mi Miriam por sus locuras.

A mi negrita (Mayté) por dejarme ser parte de su vida.

A mi familia toda, amigos y enemigos, por ayudarme a ser lo que soy.

A la Revolución y a Fidel por darme esta gran oportunidad.

DEDICATORIA

A la mujer que más amo en el mundo,

Que además de ser mi mejor amiga,

Es mi madre.

RESUMEN

La pérdida de datos puede ser un sinónimo de quiebra para muchas de las empresas que se han visto alejadas del mercado por este hecho. La Universidad de las Ciencias Informáticas, como empresa desarrolladora de software de corto, mediano y largo alcance, y con más de 5000 cómputos agregados a su red, no posee en la actualidad un mecanismo de defensa ante una situación de este tipo. El presente trabajo tiene como objetivo desarrollar un mecanismo de respaldo que se ajuste a las necesidades y características de la UCI.

En la actualidad no existe, en el campo del software libre, ningún sistema que cumpla con las exigencias de la Universidad. Sus productos no se encuentran lo suficientemente efectivos como para dar por sentado el asunto. Todos y cada uno de ellos presentas desventajas notables como son: la baja curva de aprendizaje, configuraciones complicadas, fuerte dependencia entre las aplicaciones que las componen, entre otras. Por tales razones, haciendo un estudio de los sistemas de respaldo existentes, se ha desarrollado un mecanismo genérico que se adapte, no solo a las características de la UCI, sino a las de cualquier empresa que desee proteger su material más preciado: la información. Además de proponerse un mecanismo de respaldo, también se ha desarrollado una propuesta de aplicación que lo ponga en práctica. Esta propuesta ha sido desarrollada en el lenguaje Java, apoyándose en algunos marcos de trabajo de la plataforma como Spring.

PALABRAS CLAVE

Pérdida de datos, Curva de aprendizaje, Mecanismo de respaldo, Información.

TABLA DE CONTENIDO

TABLA DE FIGURAS

INTRODUCCIÓN

La Universidad de las Ciencias Informáticas (UCI), ha sido creada con un nuevo concepto: universidad productiva. Una de sus misiones es la producción de software y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación, logrando una fuerte relación Universidad-Empresa. Esta universidad que convierte a la producción en sustento económico, político y social; es productora desde pequeñas hasta grandes soluciones informáticas y está comprometida a ser la vanguardia en el desarrollo de software en Cuba, de manera tal que convierta a la industria de software en un renglón fundamental de la economía del país.

La UCI tiene retos importantes, dados sus compromisos productivos reales, vinculando a la producción a todos sus estudiantes y a un alto por ciento de sus educadores, en proyectos de alto valor, tanto para el mercado nacional como el internacional.

Como entidad desarrolladora de software que pretende insertarse en el mercado internacional, la UCI debe lograr no solo el desarrollo de soluciones comerciales**,** siendo esta su principal fuente de ingresos, sino de herramientas internas para poder llevar a cabo estas soluciones, y a medida que vaya creciendo esta gama de recursos, poder obtener productos de mejor calidad en menor tiempo y con un mayor control.

Los proyectos productivos en la UCI se desarrollan actualmente a nivel de facultad, recayendo sobre esta todo el peso y control del mismo. Cada facultad organiza sus actividades productivas de acuerdo a sus necesidades, en uno o dos laboratorios, pudiéndose encontrar allí todo lo referente al producto en desarrollo.

Al solo encontrarse en el local donde se desarrolla el producto toda la información referente al mismo, se corren muchos riesgos de seguridad. Una catástrofe podría conllevar a un daño parcial y hasta total del mismo. La importancia de disponer de mecanismos preventivos para proteger los datos críticos, y poder reanudar las operaciones, en caso de desastre, con el menor impacto posible, debido a que por acción de virus, de usuarios, ataques malintencionados, fallos en el hardware, o simplemente por accidente (incendios, inundaciones, robo) o descuido, la información contenida en los servidores puede resultar dañada o incluso desaparecer.

Dada la siguiente situación, el **problema a resolver** queda formulado a modo de interrogante de la siguiente forma:
¿Cómo realizar salvas automáticas periódicamente desde los proyectos hacia servidores centrales?

El **objeto de estudio** lo constituyen los procesos de salvas automáticas centralizadas de datos, por lo que se especifica el siguiente **campo de acción:** salvas automáticas de datos desde los proyectos hacia servidores centrales en la UCI.

Con la puesta en práctica de este sistema de salvas, la información se encontraría en los servidores locales de cada proyecto y también estaría disponible en servidores centrales donde solo tenga acceso el personal adecuado. Teniendo una copia de seguridad de toda la información que nos interesa no hay daño que no pueda ser reparado.

La universidad, además de su servidor central destinado a las copias de seguridad ubicado en la Infraestructura Productiva (IP), cuenta con un servidor de respaldo a este ubicado en el Nodo Central, por lo que, para garantizar que el proceso de salva cumpla con las exigencias del desarrollo en la UCI, este debe dar la posibilidad de replicar la información del servidor central hacia otros servidores .El sistema de salva en general debe ser automatizado a través de un mecanismo que posea la capacidad de disminuir el esfuerzo que empleará la organización en la labor.

Con este mecanismo de respaldo implementado, la información de todos los productos de la universidad se encontraría centralizada y actualizada, lo que traería consigo varias mejoras dentro del proceso de toma de decisiones por parte de los directivos. Se podría conocer el avance real de cada proyecto si se cuenta con la autorización para hacerlo y sin necesidad de ir al laboratorio de desarrollo ni reunirse con los implicados con este. También sería una gran ventaja para el área de calidad pues podrían descargar el producto en cualquier momento, confiando en la actualidad del mismo, y realizar las pruebas pertinentes incluso antes de que el producto sea liberado por sus desarrolladores.

Dada estas condiciones se plantea lograr como **objetivo general**: diseñar un mecanismo que realice salvas automáticas hacia servidores centrales.

Según el objetivo planteado para la investigación se definen el siguiente conjunto de **tareas** a lograr para su cumplimiento:

- Construir un marco teórico sobre las herramientas tanto privativas como libres para realizar salvas.
- Diseñar un mecanismo que realice salvas automáticas hacia servidores centrales.
- Diseñar un mecanismo de restauración y acceso de los datos respaldados.
- Implementar una primera aproximación del mecanismo de salva automática que cumpla con los objetivos planteados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se brinda un estado del arte sobre los elementos que forman parte del problema, así como aquellos que son importantes a tener en cuenta para dar la solución que se requiere. Se plasma toda la información, conceptos y definiciones necesarios para llevar a cabo una investigación exhaustiva referente a las copias de seguridad, una breve reseña histórica de su evolución y una descripción de las últimas tecnologías por las que se ha podido ver influenciado. También se revisarán las herramientas más usadas actualmente en la realización de copias de seguridad, partiendo de los principales conceptos para lograr su entendimiento y necesidad. Se explican los principales conceptos de la mensajería así como la aplicación que puede tener en el campo del respaldo de datos.

1.1 La pérdida de datos

Dentro del desarrollo tecnológico en el que estamos inmersos, la información se ha convertido en el elemento más importante de cualquier persona, empresa, institución o gobierno. Su disponibilidad es una de las características fundamentales para la subsistencia del medio que la utilice, y su pérdida puede ocasionar grandes problemas, de los cuales es muy difícil recuperarse.

Debido a la gran cantidad de información con la que estamos relacionados en nuestros días, las copias de seguridad han pasado a un plano indispensable para el buen funcionamiento de cualquier tipo de empresa, sin tener en cuenta sus dimensiones o perfil de negocio; la capacidad de reanudar sus servicios lo antes posible ante cualquier situación de pérdida o corrupción de la información es de vital importancia. En la conferencia a cargo de Andrés Sánchez, en junio del 2007, explicó que el 40 porciento de las compañías que experimentan un desastre de gran magnitud quedan fuera del negocio si no pueden tener acceso a su información en 24 horas, también aportó, partiendo de una encuesta realizada a profesionales en almacenamiento de datos en Europa, que el 46 porciento usaba cintas abovedadas remotas. (1)

A lo largo de la historia, la humanidad ha sido víctima en determinados momentos de tragedias provocadas por la **ira de la naturaleza**, el 12 de julio del 2006, una universidad del sur de Chile, octava región, fue víctima del desborde del río Biobío, que además de verse obligado a cancelar sus actividades, se enfrentó al problema de perder todos los datos de un servidor, producto de la completa

inundación de la pieza donde se alojaba. Los computadores se encontraban sumergidos bajo el agua. (2)

Aunque siempre nos cuidemos de los desastres naturales, estos no son la principal causa de pérdida de información. Según un artículo publicado por Borrmart: "Las nuevas tendencias en respaldo y recuperación de datos" por Eduard Abad, director asociado de Esabe[1], el 56 porciento se le atribuye a los equipos, el 26 porciento a los errores humanos, 9 porciento para la corrupción del software, 4 porciento por virus, dejando solamente un 2 porciento para la naturaleza. (3)

Las **negligencias del hombre** han estado presentes en todas las ocasiones, aportando un gran porciento al incremento de este número de calamidades. Como ejemplo de lo antes planteado se tiene que poco antes de la media noche del sábado 12 de febrero de 2005 se declaró un incendio en el rascacielos Windsor, situado en la zona de AZCA, lo cual conllevó a su posterior derrumbe. El edificio, obra del estudio Alas y Casariego, fue construido en la segunda mitad de los años 70. La mayoría de sus plantas (más de 21 de un total de 32) estaban ocupadas por la consultora Deloitte[2]. Es considerado el mayor incendio en la historia de Madrid hasta ese momento. (4)

En las oficinas centrales del Credit Lyonnais, banco francés de mucho prestigio, un incendio hizo que los administradores entraran en el edificio en busca de las cintas de copias de seguridad, y todo porque no tenían copias fuera del lugar. (5)

En los últimos años, donde los avances tecnológicos han sido considerables en distintas ramas de la economía y la sociedad, se va haciendo cada vez más difícil el desarrollo de una empresa sin la vinculación con el **desarrollo tecnológico**, convirtiéndose este en la principal fuente de catástrofe. Según Yankee Group[3], el 40 porciento de los gestores tecnológicos de información no pueden recuperar los datos de sus copias de seguridad locales (6).

El 3 de enero del 2008 un servidor de correos electrónicos colapsó en TeliaSonera, la principal compañía telefónica y operador de redes móviles en Suecia y Finlandia. Consecuentemente se descubrió que la última copia de seguridad realizada era del 15 de diciembre de 2007, por lo que 300 000 cuentas fueron afectadas. El 23 de julio de 2007 un corte en el suministro eléctrico deja sin luz a unas 350.000 personas en Barcelona.

[1] Empresa dedicada a prestar servicios de seguridad informática, está fuertemente posicionada en seguridad para los contenidos y es líder en servicios de respaldo continuo (CDP) y auxiliares.

[2] Empresa que presta servicios en cuatro áreas funcionales: Auditoría, Consultoría, Asesoramiento Jurídico y Tributario.

[3] Primera tecnología independiente de investigación y empresa de consultoría.

Las negligencias y la naturaleza no son la única forma de atentar contra la sociedad, la **voluntad humana** también ha puesto su grano de arena. El Privacy Rights Clearinghouse[4] ha documentado 16 incidentes de pérdida o robo de cintas de seguridad entre 2005 y 2006, afectando a organizaciones como el Banco de América, Ameritrade, Citigroup, y Time Warner (7). El 21 de noviembre de 2007 el primer ministro del Reino Unido, Gordon Brown, tuvo que ofrecer disculpas al país por el extravío de 2 CD con información de 25 millones de ingleses al ser enviados por correo físico (8). El ejemplo más desbastador fue la bienvenida al nuevo milenio con los ataques del 11 de septiembre de 2001. La Reserva Federal redujo temporalmente sus contactos con bancos por la falta del equipo perdido en el distrito financiero de Nueva York; los índices bursátiles NYSE[5], Amex [6]y la Asociación Nacional de Distribuidores de valores de cotización del sistema automatizado no abrieron ese día, y permanecieron cerrados hasta el 17 de ese mismo mes. Los sistemas del NYSE no fueron dañados por el ataque, pero los daños en las líneas telefónicas del sistema financiero del World Trade Center impidieron que funcionara. Cuando los mercados reabrieron el 17 de septiembre de 2001, tras el mayor paro desde la Gran Depresión, el índice Dow Jones Industrial Average [7]cayó 684 puntos (7,1 porciento), hasta 8920, en su mayor caída en un solo día. Al final de la semana, el Dow Jones había perdido 1369,7 puntos (14,3 porciento), su mayor caída en una semana. (9)

Las copias de seguridad en los sistemas informáticos tienen por objetivo mantener cierta capacidad de recuperación de la información ante posibles pérdidas o desastres. Todas y cada una de las situaciones anteriormente planteadas ponen sobre la mesa la necesidad de poseer un **sistema de respaldo de datos**, y sobre todo, en un lugar geográficamente distante de donde se encuentran los posibles afectados, y así de esta manera poder restaurar las operaciones en el menor tiempo posible.

1.1.1 El costo de la pérdida de datos

Un episodio de pérdida de datos dará lugar a dos resultados posibles: o bien algunos datos son recuperables o se borran definitivamente. En el entorno actual, con numerosos sistemas de respaldo de datos y las distintas soluciones de recuperación, las empresas no tienen por qué sufrir episodios de

[4] Centro de Derechos de Privacidad dedicado a defender el derecho a la intimidad y la protección del consumidor contra el robo de identidad y otros crímenes de privacidad.

[5] Bolsa de cambio de Nueva York más conocido como "Big Board".

[6] Bolsa de cambio americana situada en Nueva York.

[7] Mercado de valores de índice.

datos irrecuperables, salvo en caso de negligencia o la mala planificación ante catástrofes graves. Se asume que para los efectos del presente documento, todos los datos pueden ser recuperados, sin embargo, como se ha demostrado en anteriores trabajos, el valor intrínseco de los datos pueden ser significativos, y como se señaló anteriormente, si los datos se borran definitivamente podría quedar en quiebra la organización. Dejando de lado esta posibilidad, vamos a centrarnos en los costes de recuperación de datos, así como en la pérdida de productividad y las ventas durante un episodio de inactividad por parte de los afectados.

En aproximadamente el 40 porciento de los casos - cuando no ha habido daños materiales en la unidad de disco duro - los datos pueden ser recuperados por una persona de apoyo técnico. Estos casos son a menudo causados por errores humanos, la corrupción del software, o los virus informáticos.

Además del costo de la contratación externa, la recuperación de datos, los usuarios y las empresas están sometidos a la pérdida de productividad. Cada usuario de computadoras ha experimentado un momento de frustración, y la correspondiente pérdida de productividad, cuando su equipo no está disponible para su uso. Cuando se produce la pérdida de datos, estos episodios pueden convertirse en prolongados y pueden llegar a ser bastante costosos. Con el fin de estimar los costes, los siguientes factores deben ser considerados: la productividad individual del usuario, la duración del tiempo de inactividad, y la medida en que una persona con pérdida de datos afecta a otros miembros de la organización.

La precisión en la estimación del efecto de contaminación dependerá de los factores que se han señalado anteriormente, pero una estimación conservadora de un típico episodio de pérdida de datos puede sugerir la idea de que una persona de clave acceso, que ha sufrido pérdida de datos, afectaría la productividad de otros tres compañeros de trabajo, reduciéndola en un 25 porciento.

La pérdida total debido a la ralentización de la productividad depende de manera crítica de la duración del tiempo de inactividad, que será determinada en forma considerable sobre si el ordenador tiene que ser enviado fuera de la empresa o si los datos pueden recuperarse sin el movimiento físico del equipo. Un estudio realizado en España arrojó que el tiempo promedio para la recuperación de datos, por técnicos internos de la institución, es de 8 horas, mientras que la cantidad mínima, si es necesario que el hardware salga hacia una empresa dedica a la recuperación, es de 5 días. (10)

1.2 Qué son las copias de seguridad

Existen varias definiciones escritas acerca de lo que son las copias de respaldo, aunque todas convergen en la misma idea. A continuación se muestran varias de estas definiciones obtenidas de diferentes fuentes:

- Copia de ficheros o datos de forma que estén disponibles en caso de que un fallo produzca la pérdida de los originales. Esta sencilla acción evita numerosos, y a veces irremediables, problemas si se realiza de forma habitual y periódica. (11)
- Hacer una copia de seguridad o copia de respaldo *(backup* en inglés, el uso de este anglicismo está ampliamente extendido) se refiere a la copia de datos de tal forma que estas copias adicionales puedan restaurar un sistema después de una pérdida de información. (12)
- Copia de seguridad es la actividad de copiar archivos o bases de datos a fin de que puedan ser conservadas en caso de fallo del equipo o de otra catástrofe. La copia de seguridad suele ser una parte rutinaria de la operación de grandes empresas con súper computadoras, así como los administradores de las empresas más pequeñas con menos recursos. Para los usuarios de computadoras personales, la copia de seguridad también es necesaria pero no se realiza muy a menudo. La recuperación de los archivos de la copia de seguridad se denomina: restablecimiento. (13)
- Archivo de copia de seguridad: copia de un archivo para efectos de la posterior reconstrucción del mismo, en caso de ser necesario. Nota: Un archivo de copia de seguridad puede utilizarse para preservar la integridad del archivo original y puede ser registrado por cualquier medio adecuado. Sinónimo de puestos de trabajo de recuperación de archivo de control. (14)
- Una copia de un programa o archivo que se almacena por separado del original. (15)

1.2.1 Historia de las copias de seguridad

Las copias de seguridad, hoy en día, son un campo rápido en vías de desarrollo. Las nuevas tendencias y soluciones aparecen, los métodos de respaldo y las tecnologías se convierten en más complejas. El respaldo es la copia de un archivo para efectos de la posterior reconstrucción del mismo. Así, podemos enfatizar que los primeros dos aspectos a tener en cuenta son: *el soporte lógico informático de almacenamiento para los datos* y *los depositarios para soporte lógico informático del respaldo.* Otro aspecto importante es *la necesidad del respaldo* causado por el desarrollo tecnológico y la expansión de volúmenes de datos.

Las primeras copias de seguridad estaban hechas de grandes carretes de cinta magnetofónica y hasta en papel, como las cintas perforadas. En la siguiente era se guardaron, en su mayor parte, en discos flexibles de tamaños diversos. Muchas de las computadoras de hoy no cuentan con unidades de disquete, sin mencionar dispositivos de procesamiento de tarjetas perforadas. Los respaldos son actualmente escritos en Discos Compactos, unidades de disco duro, dispositivos flash y hasta por vía remota. Pero algunas tecnologías, como cinta, aún son muy populares, continuándose su desarrollo y fabricación. (16)

1.2.2 Evolución de los dispositivos de almacenamiento

Tarjetas perforadas

En 1951, la primera generación de computación digital apareció cuando la **Computadora Universal Automática** (UNIVAC, por sus siglas en inglés) fue creada por Mauchly y Eckert. Usaba tubos al vacío como elementos lógicos principales, alternaba tambores magnéticos para el almacenamiento y tarjetas perforadas para introducir datos y almacenarlos externamente.

Así, las tarjetas perforadas pueden ser consideradas como los primeros dispositivos de almacenamiento de datos para respaldo. Por supuesto, no podemos hablar de integrales, métodos centralizados y estrategias en lo que se refiere a un respaldo por tarjetas, pero esencialmente son propios de la definición dada en el comienzo, porque las copias adicionales de tarjetas perforadas estaban hechas también para restaurar datos en caso de una pérdida. (16) (17)

Cintas Magnéticas

Aunque las tarjetas perforadas son tan simbólicas y han sido usadas por más de 200 años en diversos campos, realmente eran lentas, de baja capacidad y se requerían un montón de dispositivos, esfuerzos y tiempo para ir en procesión. Esta es la razón principal de por qué, durante los años 60, las tarjeta perforadas como medio primario fueron gradualmente desplazadas por la cinta magnetofónica, siendo esta mucho más eficiente. Desde el momento en que un rollo de cinta pudo almacenar la información equivalente a 10 000 tarjetas, se convirtió en la forma más popular de almacenamiento hasta mediado de los ´80. Las grandes y pequeñas compañías, y algunos usuarios atrevidos, comenzaron a crear respaldos de cinta. Los primeros respaldos y estrategias comenzaron a originarse a principios de 1960. Los respaldos de cinta fueron los más extendidos, por la fiabilidad,

dimensionalidad y bajo costo. Todas estas ventajas nivelan al respaldo de cinta como una solución atractiva, incluso en estos tiempos. (16) (17)

Los discos Duros

En 1956 IBM[8] introdujo el disco duro - IBM 305 RAMAC. A través de los años, la tecnología HDD[9] ha sido mejorada rápidamente. Desde 1983, con la introducción de la IBM PC/XT, la unidad de disco duro se ha convertido en un componente estándar para la mayoría de las computadoras personales.

Otros vendedores también contribuyeron a este desarrollo. En 1982, Hitachi[10] envió la primera unidad lógica con más de 1 GB de almacenamiento. Otro acontecimiento importante fue la introducción, a principios de 1990, de la tecnología RAID (Sistema que incluye un número de disqueteras rígidas y permite el almacenaje seguro y el retorno rápido de información). Este esquema de almacenamiento de datos usa unidades de disco duro para compartir o reproducir datos entre ellos. Almacenar datos en discos duros se convirtió en una solución atractiva y conveniente gracias a estas mejoras.

Entre 1960 y 1970 los discos duros no eran adecuados para la realización de respaldos debido a su precio, gran tamaño y baja capacidad. Sin embargo, ya en los ´80 los discos duros comenzaron a considerarse para esta tarea. Hoy en día, la batalla entre la cinta y el respaldo del disco se mantiene viva. (16) (17)

Discos Flexibles

El primer disco flexible fue introducido en 1969. Fue un disco de 8 pulgadas, de solo lectura, que podría almacenar 80 KB de datos. Cuatro años más tarde, en 1973, un disco flexible similar con el mismo tamaño, podría almacenar 256 KB de datos, con la ventaja de poder copiar y/o borrar sus datos las veces que se deseaba. Desde entonces la tendencia ha sido la misma: discos flexibles más pequeños y con mayor capacidad de almacenamiento. A finales de los ´90 se podía almacenar 250 MB de datos en un disco de 3 pulgadas.

Los discos flexibles fueron considerados el soporte lógico informático más revolucionario para el transporte de datos de una computadora para otro. No contaban con la capacidad de almacenaje de

[8] International Business Machines Corporation, más conocido como "Big Blue".

[9] Dispositivo de almacenamiento digital que almacena datos codificados en rápida rotación de platos con superficies magnéticas.

[10] Empresa multinacional especializada en alta tecnología.

los discos duros, pero, siendo mucho más baratos y más flexibles, se pusieron muy de moda. Desde 1973, los discos de 8 pulgadas se volvieron muy comunes, y fueron usados para mover pequeñas cantidades de información, usándose ampliamente para realizar respaldos. Este tipo de respaldo no fue tan difundido como el de cinta, pero al ser muy baratos y convenientes, rápidamente se convirtieron en uno de los soportes lógicos informáticos más prevalecientes entre los usuarios y las pequeñas compañías. (16) (17)

CD-R/RW and DVD

Aunque el disco flexible de 3.5 pulgadas había tenido grandes ventajas para usuarios y pequeñas empresas que necesitaban respaldos, tenía relativamente poca capacidad. Este problema se soluciona con la nueva generación de soporte lógico informático de almacenamiento: los discos compactos grabables (CD-R) y los re-escribibles (CD-RW). El disco compacto, primer invento de Philips y Sony en 1979, alcanza el mercado asiático a finales del ´82, apareciendo en otros mercados al año siguiente. En junio de 1985, el CD-ROM (memoria de solo lectura) y en 1990 los CD-RW fueron reconocidos en el mundo entero, también desarrollado por Sony y Philips.

A principio de los años 90, los CD-R no eran muy usados para realizar respaldos debido a sus altos precios. Pero más tarde, cuándo el CD ROM se convirtió en un dispositivo usual para casi cada computadora y los precios de estos descendieron drásticamente, los CD de respaldo se pusieron muy de moda. Para el comienzo del nuevo milenio, los discos compactos habían apartado, prácticamente a la fuerza, a los discos flexibles. Después de 1995, la introducción del DVD con aproximadamente 4GB de capacidad, sólo ha fortalecido esta tendencia. (16) (17)

Dispositivos de destello (flash)

Los dispositivos de almacenamiento de destello (flash por su nombre popular), inventado en 1998, son bastante nuevos para el mundo del respaldo de datos, pero ya se han vuelto muy comunes. Los más pequeños de estos dispositivos almacenan 1000 veces más datos que un tradicional disco flexible de 3.5 pulgadas, algunos más poderosos pueden almacenar lo equivalente a un CD-ROM, incluso mucho más. Considerando el tamaño, poder y el costo de estos dispositivos, no es de extrañarse que se estén convirtiendo en una fuerza poderosa para la realización de respaldos en el mercado actual. (16) (17)

Discos Blu-ray y HDD-DVD

Los discos de Blu-laser usan tintes orgánicos, como el Sony Blu-ray y el HD-DVD de Toshiba (entre 23GB y 54GB), son el siguiente paso hacia la reducción del costo de los soportes lógicos extraíbles, con crecimiento en la mejora y la usabilidad. Aparecieron en el mercado en el 2006, y son ya considerados como prometedores dispositivos para respaldo de datos. (16) (17)

1.2.2.1 Soluciones On-line

Por otro lado, los respaldos de datos están estrechamente relacionados con la evolución de las redes. Cuando las redes locales aparecieron, se hicieron posibles las copias de seguridad remotas hacia otras computadoras. Las redes locales y globales posibilitaron el respaldo de grandes volúmenes de datos críticos por todo el mundo. Para quedar protegido en contra de desastres o algún otro problema, muchos usuarios prefieren enviar archivos de respaldo hacia bóvedas de seguridad**.** (16)

Redes de Área Local

Las primeras Redes de Área Local (LAN) fueron creadas a finales de los ´70, y fueron usadas para proveer alta velocidad entre varias computadoras centradas en un sitio geográfico específico. El surgimiento de la tecnología LAN fue la tendencia más significante en sistemas de almacenamiento, de finales de los ´80 y principio de los ´90, y también influenció significativamente el dominio de las copias de seguridad. (16)

Protocolo de Transferencia de Ficheros (FTP)

Aparecido en 1985. Conecta dos computadoras por la red a fin de que los usuarios puedan transferir archivos de una máquina a otra y ejecutarlos remotamente. Específicamente, FTP es un protocolo comúnmente usado para intercambiar archivos en cualquier red que soporte el protocolo del TCP/IP**.** (16)

Red de Almacenamiento Fija (Network Attached Storage, NAS)

A mediados de los ´80 aparecieron los sistemas NAS, diseñados para estar pegados a la red tradicional de datos. Desde la introducción del concepto del dispositivo NAS para el mercado en 1992, la tecnología fue ampliamente aceptada, y muchos de los fabricantes que lideraban el mercado del almacenamiento en esa época, añadieron estos dispositivos a sus productos incluyendo opciones diversas de respaldo**.** (16)

Red De Área de Almacenamiento (Storage Area Network, SAN)

Una red de área de almacenamiento (SAN), está diseñada para adherir dispositivos de almacenamiento como controladores de grandes colecciones de discos, y bibliotecas de cinta con servidores. Deja una computadora conectarse a tarjetas remotas, como discos y cintas, en una red. SAN puede servir para los propósitos de las copias de seguridad. Ofrece soluciones de respaldo de alta velocidad, inmediatas y programables para grandes empresas. (16)

Telaraña Mundial (World Wide Web, www)

La Internet fue el resultado de algunos pensadores videntes a principios de 1960, que vieron gran valor potencial en hacer compartir a las computadoras información, para la investigación y desarrollo en campos científicos y militares. Los precursores de Internet fueron: ARPANET [11](nacidos en 1969), NSFNet[12] (1983) y algunos otros. La Internet en su forma moderna apareció en 1990, cuando la primera página web fue publicada. La importancia del servicio de respaldo en línea, ha evolucionado dramáticamente en unos pocos años. Desde finales de los ´90, estos servicios no solo se han puesto accesibles a usuarios corporativos, sino al mundo entero**.** (16)

En la Figura 1 se muestra como han venido surgiendo los diferentes tipos de dispositivos anteriormente explicados, así como su utilización en el área del respaldo de datos. También se define el seguimiento de su vida útil y en el caso de las tarjetas perforadas y los discos flexibles, el momento en que quedaron desechados por sus desventajas en comparación con el rendimiento y capacidad de los demás dispositivos.

[11] Agencia de proyectos de investigación de red avanzada.

[12] Fundación nacional de científica de redes

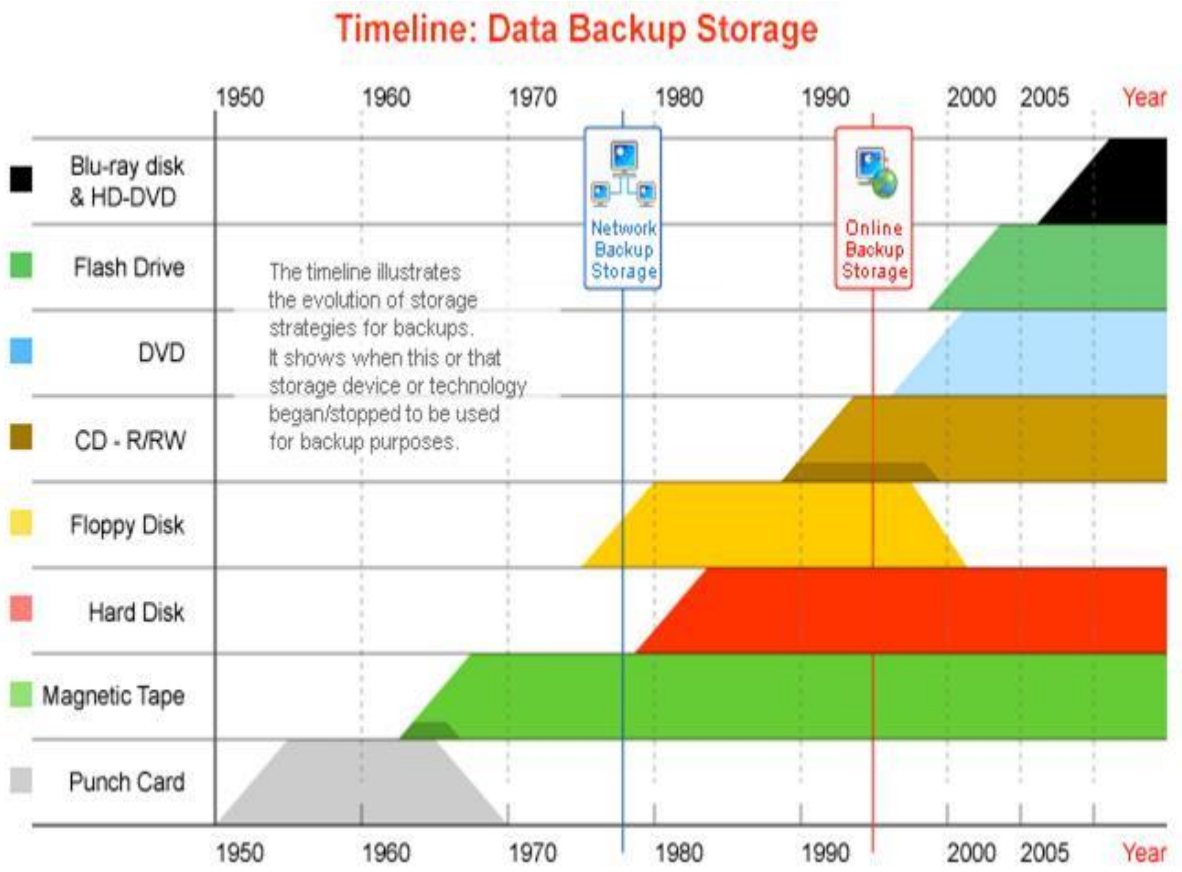

Figura 1 Resumen de la utilidad de los dispositivos en el tiempo (16)

1.3 Qué valorar en una herramienta de respaldo

Actualmente existen un sin fin de herramientas para realizar respaldos, sin embargo cuando las redes tienen un mínimo grado de complejidad, algunas herramientas pierden su fuerza y no responden adecuadamente a las exigencias que se les plantea. Existen políticas de respaldo de todo tipo y formas; las más sencillas suelen usar simples scripts y herramientas nativas del sistema operativo, estas soluciones se implementan bien a la hora de hacer copias de seguridad locales con poco volumen de datos o sistemas con pocos clientes. Pero, ¿qué pasa si necesitamos hacer un respaldo a nivel empresarial?, se nos haría tedioso y complicado crear un gran script, sin olvidar las limitaciones ocultas como el tiempo, volumen y soporte de hardware.

Existen herramientas a nivel empresarial que eliminan muchas de las restricciones anteriormente nombradas, aunque normalmente tienen un alto precio. Entonces, ¿qué se debe valorar principalmente

en un programa de respaldo para una empresa? Uno de los primeros aspectos es la capacidad de poder operar con distintas plataformas de sistema operativo. En la actualidad los entornos heterogéneos son bastante frecuentes, escenarios que surgen por el interés de apoyar los servicios en las plataformas más adecuadas en cuanto a robustez y rendimiento. En estos casos, uno de los principales retos que enfrenta un administrador está en consolidar una plataforma y a su vez un programa de respaldo que contribuya a la unificación de los servicios que presta o se utilizan en la empresa. Adquirir programas de respaldo independientes para cada plataforma, resulta una actitud opuesta con la sencillez de administración que se persigue con la consolidación y el punto de divergencia que plantea esta filosofía, hace que el programa de respaldo lejos de aportar soluciones, acabe siendo otra preocupación más para el administrador.

Esta integración que se persigue con los programas de respaldo, lleva a otra pregunta: ¿cómo debe ser la interfaz de usuario y la facilidad de manejo? Son aspectos importantes si se tiene en cuenta que es una operación que suele realizarse a horas en donde la empresa no cuenta con personal activo, de forma automática u operadores con la capacitación técnica limitada. La facilidad de manejo, una interfaz amigable y mensajes claros y adecuados al contexto de operación en el que se desenvuelve el técnico, evitan dudas o confusiones para realizar el proceso de respaldo, por ende, es un tema importante el programa que se elija para realizar dicho procedimiento. Igualmente a la hora de instalar un programa de respaldo, es imprescindible tener presente una proyección mínima del futuro y dejar las puertas abiertas a posibles variaciones imprevistas como la incorporación de nuevos servicios, el soporte a nuevos clientes, fusiones de compañía y muchos otros casos más.

Otra característica a valorar es la facilidad que incorpora el programa para la recuperación de los datos ante un fallo. El grado de tranquilidad que se puede alcanzar en la estrategia de respaldo, depende mucho de los mecanismos de recuperación que el programa disponga. (18) (19).

1.4 Lista de software para respaldo

En el mundo existen una gran variedad de estos tipos de sistemas dedicados al respaldo de datos. En la tabla de la Tabla 1 se reflejan las herramientas más destacadas de este tipo en el Software Libre así como la licencia bajo la cual fueron liberadas, y en la Tabla 2 se reflejan las más destacadas en el mundo privativo. (20)

1.4.1 Software Libre

Redes Extensas		Licencias
	Bacula	GPL
	BackupPc	GPL
	AMANDA	BSD
	Restore	GPL
Redes Pequeñas		
	RSync	GPL
	Duplicity	GPL
	Dump	GPL
	Tar	GPL
	CPIO	GPL
	DAR	GPL
	Rdiff-Backup	GPL
	Rsnapshot	GPL
	Dirvish	OSL
Sistemas Locales		
	Areca Backup	GPL
	Cobian Backup	MPL
	FlyBack	GPL
	Mondo	GPL

Tabla 1 Programas de respaldo de Software Libre

1.4.2 Software Privativo

Redes Extensas	
	Atempo TIMEnavigator
	Bakbone NetVault
	CommVault Systems Galaxy
	EMC Legato Networker
	HP OpenView Storage Data Protector and Archive Backup system
	IBM Tivoli Storage Manager (TSM)
	Microsoft Data Protection Manager
	Syncsort Backup Express
	SonicWALL SonicWALL CDP
	UltraBac Software
	Unitrends
	Symantec NetBackup
Redes Pequeñas	
	CommVault Systems
	EMC Corporation Retrospect
	Symantec Backup Exec
	Unitrends
	VisionWorks Solutions
Sistemas Locales	
	Mac Backup
	Backup4all
	Genie Backup Manager
	IBM Aggregate Backup And Recovery System
	Macrium Reflect
	Nero BackItUp
	Norton 360
	Roxio Toast
	Windows Live OneCare
	Windows Recovery Environment
	Ventis BackupSuite 2008

Tabla 2 Programas de respaldo de Software Privativo

1.4.3 Veritas Software Corp.

No se puede hablar de salva y recuperación de datos sin mencionar a la que fue una de las primeras compañías de renombre internacional en esta rama. Veritas fue fundada en 1983 como Tolerant System por Eli Alon y Dale Shipley (ambos de Intel), renombrada como Veritas Software Corp. en 1989. El 16 de diciembre del 2004 se anuncian los planes de fusión junto a Symantec en un trato de $13.5 billones, el más grande hasta ese momento en el mundo del software, no concretándose hasta el 2 de julio del siguiente año.

Entre los productos que se desarrollan actualmente bajo el nombre de Symantec se encuentra (21):

- NetBackup.
- Backup Reporter.
- Storage Foundation.
- CommandCentral Storage.
- Command Central Enterprise Reporter.
- Application Director.
- Configuration Manager.
- Cluster Server.
- Volume Replicator.

1.5 Descripción de algunas herramientas utilizadas para realizar respaldos

Rsnapshot: nace originalmente y se basa en el artículo llamado “Easy Automated Snapshot-Style Backups whit Linux and Rsync”, se compone de una colección de scripts escritos en Perl[13] que automatizan el proceso de crear copias de seguridad incrementales, la clave de este programa es que sólo copia el contenido que ha cambiado desde la última recopilación hecha por él. Rsnapshot utiliza rsync, un conocido y popular programa de sincronización y transferencia de archivos, del cual hablaremos más adelante. Trabaja en cualquier Sistema Operativo de la familia Unix (19) (22)

Glastree: este programa se basa en crear copias de seguridad mediante árboles y ramas por cada día que realiza el respaldo, esto hace que los usuarios que tengan acceso a él, puedan navegar

[13] Lenguaje de programación

directamente al pasado para recuperar los documentos más antiguos o recuperar archivos perdidos. Se compone de Scripts escritos en Perl (19) (23)

Storebackup: este programa crea copias de seguridad de un disco a otro, optimizando el espacio y mejorando el desempeño. Contiene las ventajas tradicionales de una copia de seguridad completa e incremental. Incluye herramientas para el análisis de datos y la restauración. (19) (24).

Amanda: Advanced Maryland Automatic Network Disk Archiver, es un sistema de copia de seguridad que permite al administrador realizar copias a distintos servidores almacenándolo todo en una unidad de cinta de gran capacidad. Amanda usa el servicio SAMBA[14] cuando necesita realizar copias de respaldo dentro de una red Microsoft Windows. (19) (25).

Afbackup: Se trata de un programa cliente – servidor, el cual centraliza todas las copias de respaldo de varias estaciones de trabajo en un servidor maestro, igualmente es posible realizar una copia de un solo computador. Tiene soporte para guardar la copia de seguridad en un dispositivo de cinta, la escritura se realiza en forma secuencial. Algunas características son: antes de tomar el control de un cliente o estación de trabajo, hace una autentificación de él, posee restricción de acceso para los dispositivos de cinta, hace una copia total o incremental, etc. (19) (26).

CDbackup: es un programa diseñado para facilitar al usuario la creación de copias de seguridad en CD-R (W). Fue diseñado bajo *dump/restore* usando *tar/cpi*o u otra utilidad de compresión. (19) (27).

Flexbackup: es una herramienta flexible de copia de seguridad que funciona bien para pequeñas y medianas tareas. Dicha herramienta ofrece una fácil configuración, utiliza herramientas tales como afio, dump, tar, cpio, star, pax, lhda hasta zip, realiza una copia total o incremental, maneja las copias remotas mediante SSH[15] y RSH[16], tiene soporte para dispositivos de cinta, mantiene una tabla de contenidos para que el administrador sepa donde se encuentran los archivos respaldados. (19) (28).

Rdiff-backup: es un script que realiza copias de seguridad de un directorio a otro. La idea de este script es combinar las mejores características de un espejo y una copia de seguridad incremental. Rdiff-backup preserva los subdirectorios, enlaces simbólicos, archivos especiales, permisos, uid / gid de propiedad, etc. Rdiff-backup opera de un modo más eficiente el ancho de banda que una tubería

[14] Implementación libre del protocolo de redes SMB/CIFS

[15] Un protocolo de red común para la administración remota de computadores Unix.

[16] Línea de comandos Unix con gran utilidad para la ejecución de comandos remotos.

como rsync. Así se puede utilizar rdiff-backup con SSH, la diferencia conceptual es la velocidad que adquiere la transmisión. (19) (29).

Slbackup: Es un sistema de copia de seguridad diseñado para realizar copias de varios clientes a un servidor maestro. Usa rdiff-backup como herramienta para crear las copias. El servidor se conecta a los clientes mediante SSH, para realizar las copias. (19) (30).

Rsync: Es un programa de respaldo muy fácil de utilizar, que puede respaldar archivos y directorios rápidamente. Esto se logra a través de una rutina muy inteligente para detectar cuándo los archivos han cambiado, para que solamente estos archivos se seleccionen para ser copiados. Rsync también puede utilizar una utilidad de compresión para agilizar el proceso de copiado. La desventaja más representativa que presenta es que el respaldo no puede verificar si el sistema destino tiene la partición montada, y puede no estar respaldando nada. (19)

Bacula: Es, sin duda, el sistema de respaldo de código abierto que se puede usar en entornos a gran escala para cubrir las necesidades profesionales. La herramienta de respaldo es adecuada para usarse con sistemas en producción, pero aún existen algunos elementos que deben mejorarse en el futuro. Respecto a la documentación, esta es bastante extensa y muy bien estructurada, se escribe antes de que exista el código y es posible encontrar características que todavía no se han implementado. Todas las tareas que realiza Bacula se han modularizado y repartido, entre varios demonios y servicios. Trabaja bajo Linux, la familia BSD, Solaris y Windows, tiene soporte para base de datos tanto MySQL, PostgreSQL y SQLite, y un sin fin de atributos más. (19) (31).

1.6 Funcionamiento básico de Bacula

Colección de programas o servicios que permiten al administrador de sistema administrar y recuperar las copias de seguridad. También se puede usar de forma local y puede realizar salvas para diferentes tipos de dispositivos como discos duros, cintas, etc.

Bacula se compone de 5 programas fundamentales:

- **Director**: Es el programa que supervisa todo el proceso de copia, restauración y verificación de todas las operaciones. El administrador usa el Director para configurar el horario en que se realizará cada tarea. Corre como un demonio o servicio en segundo plano. (31)
- **Consola**: programa que permite al administrador o a los usuarios comunicarse con el Director que se encuentra procesando en esos momentos. Actualmente se encuentra disponible en 3 versiones: basada en texto, GNOME, y como wxWidgets. La primera, y más simple, es ejecutar

la Consola en una terminal; la mayoría de los administradores encontrarán esto completamente adecuado. La segunda versión está realizada sobre GENOME, que aún falta bastante para que sea estable pero tiene la mayoría de las funcionalidades de la anterior. La tercera versión cuenta con un restaurador de ficheros interactivo y la mayoría de las funcionalidades de los 2 anteriores. Además, permite la ayuda por comandos y completamiento por tabulación. Ni la segunda ni la tercera versión se encuentran estables todavía. (31)

- **Fichero**: El servicio de ficheros de Bacula (también conocido como el Cliente), es el programa que se va a ejecutar en la máquina sobre la cual se desea realizar la copia de seguridad. Es su responsabilidad la compatibilidad con el sistema operativo en donde se está ejecutando y proveer los datos que son solicitados por el Director. También es responsable de la sección del sistema de ficheros que va ha ser restaurada así como sus datos. Este programa se ejecuta como un servicio en el cómputo que necesita ser respaldado. Además de los demonios existentes para Unix / Linux, cuenta con uno para Windows (normalmente distribuido en formato binario). El servicio para este último se puede ejecutar actualmente en NT, 2000, XP, 2003 y existen algunas versiones no muy confiables para Milenio y 98. (31)
- **Almacenamiento**: El servicio de almacenamiento de Bacula consiste en los programas de software que realizan el almacenamiento y recuperación de los atributos de archivos y datos a los medios de copia de seguridad física. En otras palabras, el demonio de almacenamiento se encarga de la lectura y la escritura de sus cintas (u otros medios de almacenamiento, por ejemplo: archivos). Los servicios de almacenamiento se ejecutan como un demonio en la máquina que tiene el dispositivo de seguridad (por lo general una unidad de cinta). (31)
- **Monitor**: El servicio de monitoreo es el programa que permite que el administrador o el usuario puedan ver la situación actual de los Directores, los Ficheros y los Almacenamientos. Actualmente, sólo se encuentra disponible en GTK +, que trabaja con GNOME, KDE, o cualquier gestor de ventanas que apoya la bandeja del sistema estándar (FreeDesktop.org). Para realizar una exitosa copia o restauración, los siguientes cuatro demonios deben estar configurados y funcionando: el Director, el del archivo, el de almacenamiento, y el catálogo de servicios (MySQL, PostgreSQL o SQLite). (31)

1.6.1 Desventajas

Después de haber realizado un esbozo de las características principales, así como el funcionamiento básico de Bacula, donde quedan plasmadas todas sus potencialidades, se revisarán las principales desventajas que afronta y las necesidades básicas que solicita dentro de su ambiente de ejecución:

- Necesidad de un puerto libre y abierto, controlado por cada uno de los demonios que interviene en el proceso.
- Cada cómputo que intervenga tiene que tener una dirección de IP estática o nombre de dominio definido y único.
- El Director tiene que conocer la localización física de los demás demonios y viceversa.
- Necesita una base de datos para su uso interno si se desea lograr un mejor rendimiento, la cual hay que crearla y configurarla a mano pues no todos los script que vienen en cada versión son totalmente confiables.
- Para revisar y monitorear el proceso a través de una consola remota, el Director tiene que exponer su IP.
- El Director solo puede realizar una tarea (Job) en cada momento, para balancear las cargas se puede configurar tantos Directores como se desee y adicionarle las tareas que realizarán, siempre que se separen estas funcionalidades para no crear conflictos de sincronización.
- Como proyecto de software libre no es el mejor ejemplo a seguir ya que existen algunos impedimentos entre los representantes y los desarrolladores dentro de la comunidad.

1.7 Mensajería

En la computación distribuida, muchos componentes de software en un sistema en funcionamiento pueden estar separados geográficamente. La necesidad de comunicación entre estos componentes es obvia y esa necesidad guía muchas de las decisiones en materia de diseño del software. Por ejemplo, podemos elegir usar CORBA[17], SOAP[18] o Middleware[19] orientado a mensajes para la comunicación entre componentes.

[17]Estándar definido por el Object Management Group (OMG) que permite a los componentes de software escritos en múltiples lenguajes y en funcionamiento en varios equipos a trabajar juntos.

Los ordenadores conectados a una red tienen interfaces de modo que pueden comunicarse entre ellos. Crear estas interfaces implica garantizar una conexión fiable entre el emisor y el receptor, una estructura de empaquetamiento, medias para posibles fallos, etc.

Entre los primeros métodos que se utilizaron para conectar ordenadores se encuentra el **RS232**. Un conector en serie estándar, utilizado en la actualidad para la comunicación con módems. Era un modo muy común de proporcionar comunicación basada en mensajes entre dos sistemas cualquiera durante los años setenta. La mayoría de protocolos que utilizan RS232 son asíncronos; el receptor tenía que estar recibiendo activamente al mismo tiempo que el emisor enviaba el mensaje. Esto es obviamente bastante restrictivo y provoca que las aplicaciones en comunicación se acoplen fuertemente.

El **Protocolo de Transferencia de Ficheros** (FTP) es un protocolo de transferencia de archivos entre sistemas conectados a una red TCP basado en la arquitectura cliente-servidor, de manera que desde un equipo cliente nos podemos conectar a un servidor para descargar archivos desde él o para enviarle nuestros propios archivos independientemente del sistema operativo utilizado en cada equipo. El Servicio FTP es ofrecido al usuario por la capa de Aplicación del modelo de capas de red TCP/IP, utilizando normalmente el puerto de red 20 y el 21. Un problema básico de FTP es que está pensado para ofrecer la máxima velocidad en la conexión, pero no la máxima seguridad, ya que todo el intercambio de información, desde el envío del nombre y contraseña de usuario al servidor, hasta la transferencia de cualquier archivo, se realiza en texto plano, sin ningún tipo de cifrado, con lo que un posible atacante lo tiene muy fácil para capturar este tráfico, acceder al servidor, o apropiarse de los archivos transferidos.

El **E-Mail** es más adecuado para la interacción humana que la transferencia de mensajes basada en equipos, debido a que no existe ningún estándar definido para reconocer la recepción, no existe forma alguna de saber si fue recibido. El protocolo Post Office Protocol (POP) proporciona mensajería asíncrona separando los requisitos que ambas máquinas necesitan para estar activas durante la transferencia del mensaje. Esta forma de intercambio de mensajes se define como débilmente acoplada.

Una aplicación que requiere tecnología de mensajería puede utilizar diferentes paquetes de protocolos, cada uno de ellos con sus ventajas e inconvenientes. Más recientemente, con la llegada de las

[18] Protocolo para el intercambio mensajes, basado en XML, a través de redes informáticas.

[19] Software informático que comunica componentes de software o aplicaciones.

tecnologías Message-Oriented Middleware (MOM), como MQSeries[20] de IBM, han permitido a las organizaciones implementar una capa de transporte de mensajes garantizada, segura, en la que pueden basarse más fácilmente las estrategias de integración de sistema de alcance empresarial.

Utilizando estas tecnologías de capa de transporte como base, muchas compañías han producido los llamados sistemas de software intermedio de "intermediario de mensaje" (basados en MOM), que ofrecen una combinación de transformación de datos y entrega de servicios construida sobre los componentes de mensajería de nivel inferior.

La mayoría de las organizaciones son conscientes en la actualidad del enorme valor de conectar aplicaciones distribuidas dispares (Enterprise Applicaction Integration, EAI[21]), y en conjunción con el crecimiento exponencial de empresas electrónicas, esto ha ampliado la importancia fundamental de la tecnología de mensajería más allá de las fronteras de la empresa y de la Web.

Arrastradas por este crecimiento, muchas compañías han cambiado a tecnologías de base Java. Aunque sus productos mantienen a menudo sus núcleos de base C o C++. Java ha sido adoptado como el lenguaje preferido para crear herramientas de mensajería y librerías.

Como parte del Java Community Process (JCP[22]), y en colaboración con la vanguardia de vendedores de mensajería de empresa, Sun se propuso proporcionar un API de mensaje de base Java que pudiera envolver la semántica genérica de entrega de mensajes ofrecida por los proveedores MOM establecidos. El resultado fue el API de Servicio de Mensajería de Java. Publicado en Agosto de 1998 por primera vez, proporciona un sistema que capacita el desarrollo en Java de aplicaciones portátiles basadas en mensajes. Estas aplicaciones, que son distribuidas por lo general, se comunican asincrónicamente a través de mensajes. (32)

1.7.1 Servicio de Mensaje Java (JMS)

El Servicio de Mensajes Java es un API Java que proporciona interfaces a las aplicaciones para que creen, reciban y lean mensajes utilizando cualquier implementación que se adapte al API. Esto es

[20] Familia de software de red de comunicación.

[21] Es definido como el uso de software y principios de arquitectura de sistemas computacionales para integrar un conjunto de aplicaciones informáticas empresariales.

[22] Proceso formalizado que permite a las partes interesadas a que participen en la definición de las futuras versiones y características de la plataforma Java.

posible puesto que la gestión y el empaquetamiento de nivel inferior de los mensajes dependen de la implementación, que restringe la interoperabilidad exclusivamente a las implementaciones cooperantes.

El API **JMS** (como los API **JNDI**[23] y **JDBC**[24]) prescribe solo interfaces. Queda para los terceros la provisión de implementaciones reales.

La intención subyacente a este enfoque era proporcionar un API mínimo, que maximizara la portabilidad pero todavía ofreciera un poderoso conjunto de funciones de mensajería. Un enfoque así reduce la dependencia respecto de una implementación concreta y reduce la curva de aprendizaje necesario para proporcionar funcionalidad básica. (32)

JMS permite un sistema de comunicación débilmente acoplado que es asíncrono y fiable:

- **Asíncrono:** El receptor no tiene que solicitar activamente los mensajes para recibirlos. Esto es comparable a cómo no vamos cada mañana a la *Oficina de Correos* para comprobar si tenemos algún envío; en cambio, solo proporcionamos una dirección para la entrega.
- **Fiable**: Se nos puede garantizar una, y solo una, entrega de mensajes; esto es esencial en sistemas modernos. Si un sistema de mensajería no fiable se utilizara para realizar los pedidos semanales de compras y la conexión de Internet se perdiera durante el pedido, asumiríamos que el pedido no se lleva a cabo. Si realizamos el pedido por segunda vez, podríamos perfectamente recibir un pedido doble.

1.7.2 Agente de mensajes (Message Broker)

El agente de mensaje es un programa intermediario que traduce un mensaje del protocolo formal de envío del remitente hacia el protocolo formal del aparato receptor en una red de telecomunicación, donde los programas se comunican intercambiando mensajes formales. (32)

Actualmente podemos encontrar una variada gama de empresas que se dedican al desarrollo de estos programas. La necesidad y funcionalidad de estos agentes son innegables y las empresas dedicadas

[23] Java Naming y Directory Interface: API para el servicio de directorios que permite a los clientes descubrir y buscar datos y objetos a través de un nombre.

[24] API para Java que define la manera en que un cliente pueda acceder a una base de datos. Proporciona los métodos de consulta y actualización de datos en una base de datos.

al software privativo como IBM, Oracle, Microsoft, entre otras muchas, no le han podido dar la espalda y han desarrollado sus propias soluciones empresariales.

Cuba ha tomado ventaja con el surgimiento y la amplia divulgación del software libre en el mundo, no como única solución de desarrollo sino como una alternativa al ya abarrotado mercado privativo, dando a países con bajo desarrollo la posibilidad de encontrar distribución para sus productos.

El software libre también cuenta con sus implementaciones de fuerza, entre las más destacadas se encuentra **ActiveMQ** de Apache Software Fundation. (33)

Apache ActiveMQ es el agente de mensajes de código abierto más popular y poderoso. Entre las características fundamentales se pueden destacar su rapidez, soporte para un elevado número de lenguajes como Java, Ruby, Perl, Python, PHP, soporte completo para JMS1.1 y J2EE 1.4 con persistencia, pasajeros, transacciones, soporte completo para patrones de integración empresarial (EIP) tanto en el cliente como en el agente, soporte para Spring, de esta forma ActiveMQ puede ser incrustado en él y configurado usando sus propios mecanismos de configuración basados en XML[25], OpenWire[26] para un alto rendimiento en clientes desarrollados en Java, C, C++, C#, soporte Stomp [27]para que los cliente puedan ser fácilmente escritos en C, Ruby, Perl, PHP, ActionScript/Flash y Smalltalk, características avanzadas como los grupos de mensajes, destinos virtuales, comodines y destinos compuestos, servidores J2EE populares interiores probados como Gerónimo, JBoss 4, GlassFish y WebLogic, [28]incluye adaptadores de recursos JCA 1.5 para entrada y salida de mensajes, así ActiveMQ puede ser desplegado en cualquier servidor J2EE 1.4, soporte para protocolos de transporte como in-VM, TCP, SSL, NIO, UDP, multicast, JGroups y JXTA, persistencia con un alto rendimiento usando JDBC, Ajax para el soporte de flujos web usando puro DHTML , permitiendo a los navegadores ser parte de la fabricación del mensaje, puede ser usado en memoria por proveedores JMS, ideal para pruebas unitarias, soporte CXF [29].

[25] Extensible Markup Language: Especificación de propósito general para la creación de lenguajes personalizados de marcas

[26] Protocolo binario diseñado para el trabajo con MOM.

[27] Protocolo para el trabajo con MOM.

[28] Servidores de Aplicaciones.

[29] Proyecto de código abierto para facilitar el uso de los servicios web.

En la última versión ActiveMQ 5.0, liberada bajo la licencia Apache 2.0, se incluyeron muchísimas más funcionalidades, entre las que se destacan los agentes de comando, las transformaciones de mensajes, control de los productores de flujo, las colas de imagen. (33)

Uno de los requerimientos más difundidos en estos días es el envío masivo de ficheros para ser procesados por sus consumidores. En esta versión se incluye el API BlobMessage que habilita el envío de los Binary Large Object dando funcionalidad a este requisito, el cual era soportado anteriormente por los ByteMessage y los StreamMessage. (33)

1.8 Spring Framework

Spring Framework es un framework de aplicación de código abierto que ayuda a hacer el desarrollo en JEE mucho más fácil. Ayuda a estructurar aplicaciones completas en una manera consistente y productiva para crear arquitecturas coherentes. (34)

Es el más popular y el más ambicioso de todos los framework de peso ligero. Es el único framework que interviene en todas las capas arquitectónicas de una aplicación JEE. Además está diseñado para facilitar una flexibilidad arquitectónica. (34)

Presenta varios módulos de los cuales los principales son (34):

- **Contenedor de Inversión de Control**: Permite una sofisticada administración de configuración para POJOs[30] y trabaja con otras partes de Spring para proveer servicios como la administración de la configuración.
- **Un framework para la Programación Orientada a Aspectos (AOP)**: Permite comportamientos que deberían ser esparcidos de otra manera a través de diferentes métodos para ser modularizado en un simple lugar. Spring usa la AOP[31] para implementar importantes servicios tales como administración de transacciones declarativas, y además es usado para implementar códigos estándar que debería de otra manera ser esparcido entre las clases de la aplicación.

[30] Objeto ordinario de java.

[31] Programación orientada a aspectos.

- **Comunicación remota de peso ligero (Lightweight remoting)**: Spring presenta comunicación remota basada en POJOs sobre un rango de protocolos, incluyendo RMI, IIOP, Hessian, Burlap, y otros protocolos de servicios web.
- **Soporte para Servicio de Mensajería de Java (Java Message Service, JMS)**: Spring presenta soporte para enviar y recibir mensajes de una forma mucho más simple que la que provee la especificación JEE.
- **Soporte para Java Management Extension (JMX)**: Spring presenta soporte para la administración JMX[32] de objetos de una aplicación para ser configurados.
- **Soporte para comprensivas estrategias de pruebas para desarrolladores de aplicación**: Spring no solamente ayuda a realizar un buen diseño, permitiendo efectivas pruebas de unidad, sino que presenta una comprensiva solución para pruebas de integración fuera de un servidor de aplicaciones.

Los principales valores de Spring, según Rod Johnson, en su libro: "Professional Java Development with the Spring Framework", se pueden resumir en:

- **No es un framework agresivo**: Este es el principal problema de los framework anteriores. Mientras que los framework tradicionales tales como EJB[33] o Apache Avalon[34] que forzaban al código de las aplicaciones a ser dependientes del framework, implementando interfaces específicas de estos framework o heredando clases específicas de estos; ayuda a reducir las dependencias del código de la aplicación sobre el framework.
- **Ayuda a promover la reusabilidad de código**: Ayuda a evitar la necesidad de tomar decisiones importantes y duras, como es si una aplicación alguna vez usará JTA[35] o JNDI; las abstracciones de Spring permitirán desarrollar el código en un diferente ambiente si tú alguna vez lo necesitas.
- **Facilita el diseño Orientado a Objetos (OO) en aplicaciones JEE**: Deberíamos preguntarnos: ¿Cómo una aplicación JEE, escrita en Java – un lenguaje OO – no es OO? En

[32] Tecnología Java que suministra herramientas para la gestión y el control de aplicaciones, sistema de objetos, dispositivos y servicios orientados a redes

[33] Arquitectura de componentes modulares del lado del servidor para la construcción de aplicaciones empresariales

[34] Proporcionar un marco de componentes reutilizables para contenedor de aplicaciones

[35] Especificación desarrollada bajo la JCP para la realización de transacciones distribuidas

realidad muchas aplicaciones JEE no merecen el nombre OO. Con Spring es más fácil eliminar los impedimentos del diseño OO en aplicaciones JEE tradicionales haciendo el código más coherente, con más bajo acoplamiento y más reusable.

- **Facilita buenas prácticas de programación, tales como la programación a interfaces**: El uso de un contenedor de Inversión de Control reduce grandemente la complejidad del código a interfaces, más que a clases. El uso de los objetos a través de estas interfaces protege los requerimientos, los cuales pudieran cambiar en el desarrollo de la aplicación.
- **Permite la extracción de valores de configuración desde el código java a archivos XML o archivos de propiedades**: Mientras que algunos valores de configuración pudieran ser programados en Java, todas las aplicaciones de mediana y alta complejidad necesitan algunas configuraciones externalizadas del código fuente, para permitir la administración sin recompilar las clases nuevamente.
- **Es consistente**: En diferentes ambientes de ejecución y en diferentes partes del framework, Spring usa un consistente acercamiento. Una vez que se aprenda una parte del framework, su conocimiento puede ser aplicado en muchas a otras áreas del mismo.
- **Promueve la selección arquitectónica**: Mientras Spring provee una columna vertebral arquitectónica, Spring apunta para facilitar el reemplazo de cada capa. Por ejemplo, con una capa media de Spring, se pudiera ser capaz de cambiar de un framework de mapeo objeto relacional (ORM[36]) a otro con un impacto mínimo sobre el código de la lógica de negocio, o cambiar de Struts[37] a Spring MVC o WebWork [38]sin algún impacto en la capa media.

[36] Técnica de programación para la conversión de datos entre sistemas incompatibles como las bases de datos relacionales y los lenguajes de programación orientados a objetos.

[37] Framework de Apache para el desarrollo de aplicaciones web en java.

[38] Framework de OpenSymphony para el desarrollo de aplicaciones web en java.

Conclusiones

A pesar de que el estudio de las tecnologías existentes en el Mercado y las alternativas que proporcionan las diferentes plataformas de desarrollo, brinden un marco teórico apropiado para la investigación en curso, no existe, en la actualidad, un mecanismo libre de respaldo de datos que cubra con todas las necesidades que exige la Universidad de las Ciencias Informáticas.

CAPÍTULO 2: CARACTERÍSTICAS DEL MECANISMO

Introducción

En el presente capítulo se exponen las características de despliegue del SAC[39], se profundiza en la distribución actual de los proyectos en la Universidad de las Ciencias Informáticas y los cambios venideros. Se identifican las características que debería cumplir un mecanismo de respaldo automatizado. Además, se presenta una adaptación del modelo de salva general estudiado en el capítulo anterior que satisface las necesidades de dicho mecanismo. Luego se argumentan a través de una primera aproximación al flujo de actividades necesarias para su implementación, así como las entidades identificadas en dicho flujo.

2.1 Problemática de las salvas de los proyectos en la UCI

En el capítulo anterior se vieron reflejados algunos ejemplos de desastres donde se vio seriamente comprometida la información, llevando en alguno de los casos a la pérdida total de la misma. La Universidad de las Ciencias Informáticas a pesar de ser una empresa nueva ya ha tenido algunos percances que por suerte no han convergido en la pérdida de datos pero si ha estado bastante cerca. Como ejemplo de lo anteriormente planteado tenemos que el martes 29 de abril de 2008 uno de los servidores de la dirección técnica colapsó, el SALVASIP2. El mismo contenía toda la información salvada de Multimedia y de la Dirección de Comunicación Visual (Diseño). La totalidad de la información ocupaba un espacio de algo más de 400 Gb, 100 DVD como equivalencia. Por suerte, gracias al conocimiento de nuestros ingenieros, el servidor pudo volver a su estado habitual y no presenciamos ningún episodio de pérdida de datos que hubiera sido bastante lamentable debido a la cantidad de información involucrada.

La mayoría de los servidores de la UCI cuentan con más de 4 años de uso. El soporte técnico que se les ofrece no es el óptimo para su funcionamiento por lo que puede que en cualquier momento alguno de ellos deje de realizar las tareas diarias para lo que fue solicitado. Como una medida de seguridad se propone no solo la realización de salvas hacia un servidor central, como se ha venido explicando a lo largo de este documento, sino que también contar con servidores secundarios de respaldo de datos a estos mismos y que se encuentren, en la medida de lo posible, en lugares físicamente distantes.

[39] Nombre del producto propuesto (Salva Automática Centralizada).

2.2 Características propias de la salva centralizada de datos de respaldo

La problemática que implica el desempeño manual y aislado de las actividades de respaldo en la UCI, así como la importancia que tiene el soporte y mantenimiento de dicho sistema una vez implantado, crean la necesidad de automatizar dichas actividades a través de un mecanismo de respaldo de datos.

Dicho mecanismo debe cumplir con las siguientes características:

- Debido a la importancia de los datos a respaldar, se debe garantizar una alta **fiabilidad**, **eficiencia**, **integridad** y **confiabilidad**.
- El esfuerzo necesario para preparar entradas e interpretar las salidas debe ser lo más pequeño posible. Para lograr que las operaciones a realizar no involucren personal el mecanismo debe ejecutarse como un **servicio** o **demonio**.
- La UCI es una empresa en constante cambio y movimiento. Para una explotación real del mecanismo este debe ser lo más **portable** posible para poder afrontar las demandas necesarias y así ajustarse al entorno donde brindará servicios. Contar con un diseño **flexible** debido a los cambios estructurales que se avecinan en la universidad. Esto se abordará en epígrafes posteriores.
- Para un mejor conocimiento de las tareas que se encuentra desarrollando, así como la obtención de reportes de diferentes tipos, se debe contar con un **sistema de monitoreo**.
- Nadie está expenso a las fallas del hardware ni a las vicisitudes de la tecnología, la pérdida de la conexión es algo habitual para las personas que se desenvuelven dentro de una red, por lo que se debe contar con soporte para un envío **asincrónico** de datos.
- El mecanismo debe **retroalimentarse de información**, es decir, en todo momento debe informar el estado de la salva, de esta manera se puede medir el progreso del proceso completo e identificar los lugares puntuales donde la actualización no se pudo llevar a cabo.
- La comunicación debe realizarse por canales seguros y se deben aplicar medidas para garantizar la veracidad de los datos.

Estas características no son únicas del mecanismo de salva de la Universidad de las Ciencias Informáticas, sino que pueden ser retomadas en cualquier sistema que se enfrente a una problemática similar con los procesos de salva de datos de respaldo.

2.3 Modelo de respaldo de datos para los proyectos en la UCI

Se ha modelado un mecanismo de respaldo de datos que cumple con las necesidades y características expuestas anteriormente. Dicho modelo es representado en la Figura 2 a través de un diagrama de transición de estados por los que transita el cliente que pide la salva, el servidor que la gestiona y el servicio intermediario que se encarga de informar a cada una de las partes involucradas. El punto de partida es cuando al cliente funcional le llega la hora de realizar la copia de seguridad y pide una autorización.

La siguiente lista muestra todos los pasos del proceso en el modelo de respaldo de datos presentado:

- **Cliente Funcional**: El cliente se encuentra esperando el momento de comenzar el proceso de respaldo de datos. Este es el punto de partida del mecanismo.
- **Pide autorización**: El cliente le pide autorización al servidor para realizar la copia, se envían los datos que definen al cliente y el tipo de copia que desea realizar, completa o incremental, la primera incluiría una copia de todos los datos que se encuentren en el cliente mientras que la segunda solo procesaría los datos que han sido añadidos o modificados.
- **Servidor Informado:** El servidor, que se encontraba esperando el pedido inicial, es informado acerca del cliente que desea realizar la salva. Junto a esta petición le son entregados los datos que definen al cliente así como el tipo de respaldo a realizar. En caso de ser una copia incremental, se enviará una entidad que contenga el nombre de cada dato que se encuentra en el cliente así como su fecha de modificación o creación.
- **Procesa Información Inicial:** La información inicial es procesada por el servidor. Si la copia es incremental, se realiza una comparación entre los datos a salvar y los existentes para evitar la redundancia.
- **Autoriza:** Se envía la señal de autorización del servidor al cliente, en caso de ser denegada, se reiniciaría el proceso y todo volvería a empezar. Este es uno de los puntos de acceso seguro, como ya la información inicial ha sido procesada y revisada, se sabrá si realmente el cliente es quien dice ser y si tiene acceso al tipo de respaldo que solicita.
- **Autorizado:** El cliente es notificado acerca de la autorización, de ser denegada se reiniciaría el proceso; de ser aprobada se le enviaría además un conjunto de datos necesarios para realizar la copia.

- **Información Recopilada:** Si la copia a realizar es completa, se ordenarían todos los datos que posee el cliente, si esta fuera incremental, solo se ordenarán los datos que no se encuentren en el servidor o que hayan sido modificados.
- **Enviar Flujo de Datos***:* Estando todo en orden se procede al envío de los datos a respaldar desde el cliente al servidor.
- **Salvar Datos:** El servidor recibe los datos de respaldo del cliente y los salva de acuerdo a la información obtenida anteriormente para poder mantener una localización exacta de cada salva y poder restaurar los datos de cada cliente en caso de ser necesario.
- **Datos Publicados***:* Los datos salvados son publicados y se notifica a los demás servidores externos de respaldo que pueden acceder a ellos.
- **Servidor Externo Informado***:* Los servidores externos son informados y se preparan para realizar la copia. Este comunicado contiene información adicional de control de los datos publicados y se convertiría en otro punto de seguridad.
- **Datos Salvados***:* Se envía el flujo de datos a los respectivos servidores externos y estos realizan las copias de respaldo de acuerdo a la información recibida.

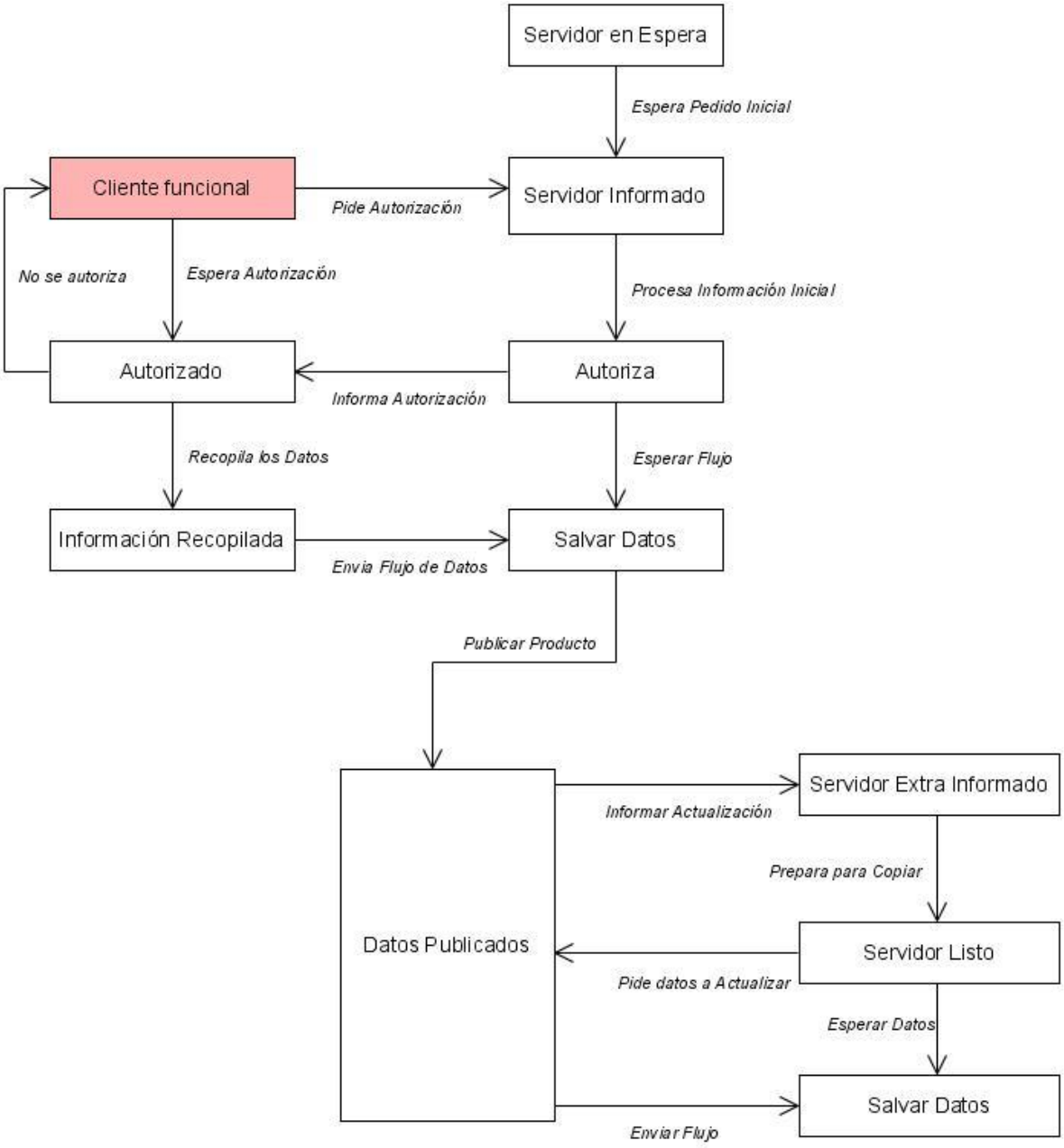

Figura 2 Comunicación entre los servicios.

Leyenda: El sombreado indica el inicio de los procesos en el diagrama.

2.4 Mecanismo de salvas automáticas centralizadas para los proyectos en la UCI

El mecanismo de respaldo de datos de la Universidad de las Ciencias Informáticas propuesto, cumple con las premisas del modelo de salva desarrollado en el epígrafe anterior y agrega nuevas funcionalidades al proceso.

En la confección del mecanismo se han dividido las funcionalidades en cuatro servicios informáticos con responsabilidades definidas de la siguiente forma:

- **Cliente**: Es el responsable de iniciar el proceso de salva. En él recae todo el peso de la recopilación y selección de los datos a respaldar.
- **Servidor**: Es el responsable de manipular y autorizar los pedidos de salva. También realiza la selección de datos para evitar redundancias y notifica el fin del proceso de respaldo de datos del cliente para dar paso a las réplicas.
- **Réplica**: Es el responsable de realizar las copias de respaldo del servidor central hacia servidores externos.
- **Monitoreo**: Es el responsable de recibir las notificaciones de los servicios anteriores, permitiendo visualizar cada tarea que se esté ejecutando en cada momento, así como la realización de reportes estadísticos para un mejor control y mantenimiento.

De conjunto, estos cuatro servicios colaboran para realizar actividades; estas han sido generalizadas en procesos que describen de manera general cómo funciona el mecanismo. Los procesos identificados son los siguientes:

1. Solicitar acceso al servidor para comenzar copia de respaldo.
2. Dar acceso a un cliente determinado para realizar copia de respaldo.
3. Recopilar información en el cliente y enviar datos generales.
4. Publicar notificación de datos actualizados.
5. Replicar datos a servidores secundarios.
6. Informar estado.

Los procesos enumerados necesitan una representación estructural de la información que procesan, generan, o utilizan, para intercomunicarse entre ellos. Por esta razón se identificaron entidades que describen, no solo la información, sino también el formato.

Las entidades identificadas se describen a continuación:

- **Notificación**: Objeto enviado desde el cliente que contiene, además de todos los datos referentes a este, un fichero con el **Listado de Recursos** que posee.
- **Listado de Recursos**: Fichero en formato XML que contiene un resumen todos los recursos que se encuentran en el cliente, así como su fecha de creación o modificación.
- **Acceso**: Objeto enviado desde el servidor que contiene, además del formato de comunicación con el mismo, una modificación del **Listado de Recursos** con la diferencia de datos entre el servidor y el cliente.
- **Tabla General de Respaldo**: Está contenida en el **Sistema de Monitoreo** y lista todos los respaldos que se encuentran en ejecución en esos momentos, así como de los que ya han sido respaldados, la hora y el modo. Es actualizada asincrónicamente por las aplicaciones **Cliente**, **Servidor** y **Réplica**.
- **Datos Generales**: Son todos los datos cuya descripción está contenida en el **Listado de Recursos** compactados en un fichero ZIP único.

Cada uno de estos subprocesos en los que ha sido dividido el mecanismo general para una mejor comprensión del mismo, se explicarán en los siguientes epígrafes.

2.4.1 Solicitar acceso al servidor para comenzar copia de respaldo

El primer evento que desencadena la secuencia de todos los procesos involucrados en la salva es la solicitud de permiso. El cliente se configura a través de una interfaz gráfica que le permite al jefe de proyecto hacer una selección de todos los datos que desea respaldar y en qué momento se realizarán estas copias. Después de haber configurado el cliente, este conocerá el momento en que él debe realizar la salva. Primero que todo realiza el **Listado de Recursos** que es un resumen de la información de los datos a respaldar, dando a conocer de cada dato la fuente, el camino a este y su fecha de creación o modificación. Luego se crea el objeto de **Notificación** que contendrá todos los datos que lo identifican como cliente de este y, anexado a esto, el resumen previamente hecho. Este último objeto es el que el cliente envía al servidor para ser procesado.

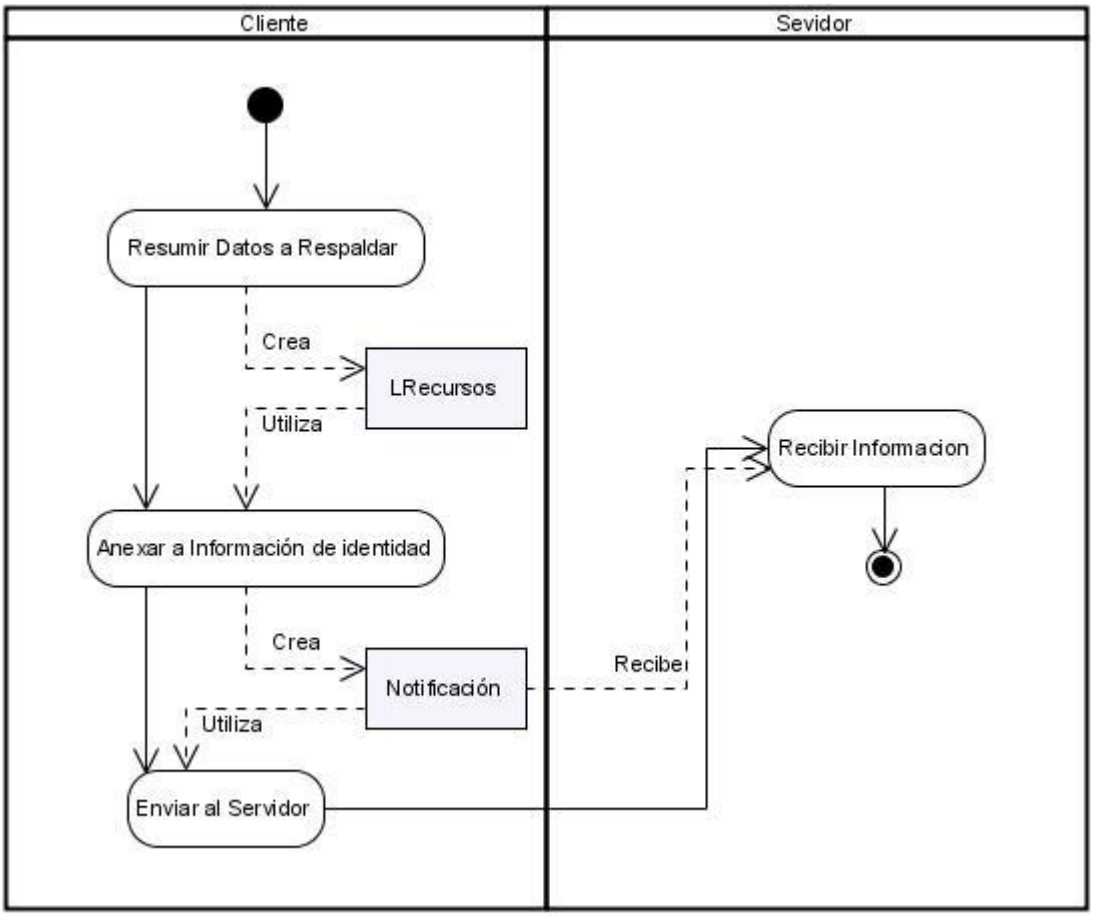

Figura 3 Proceso de solicitud de acceso al servidor para comenzar copia de respaldo

2.4.2 Dar acceso a un cliente determinado para realizar copia de respaldo

El servidor recibe la información que llega desde el cliente. Verifica la autenticidad de la petición y el acceso que este posee. De existir alguna duda en cuanto a la identidad se detiene el proceso inmediatamente y se le notifica la causa del rechazo. De tener acceso al servicio que solicita se procesa la información anexada, es decir, se realiza una selección de los datos que han sido creados o modificados en el cliente teniendo de referencia las salvas realizadas anteriormente. De no existir ninguna salva previa, se salva toda la información que posee el cliente. Se modifica el **Listado de Recursos** y se crea el **Objeto de Acceso**. Este objeto contendrá toda la información que el cliente necesita para realizar la salva y además se le anexará el **Listado de Recursos** modificado, es decir, con toda la información que no se encuentra actualizada en el servidor. El proceso termina con el envío del **Objeto de Acceso** para ser consumido por el cliente.

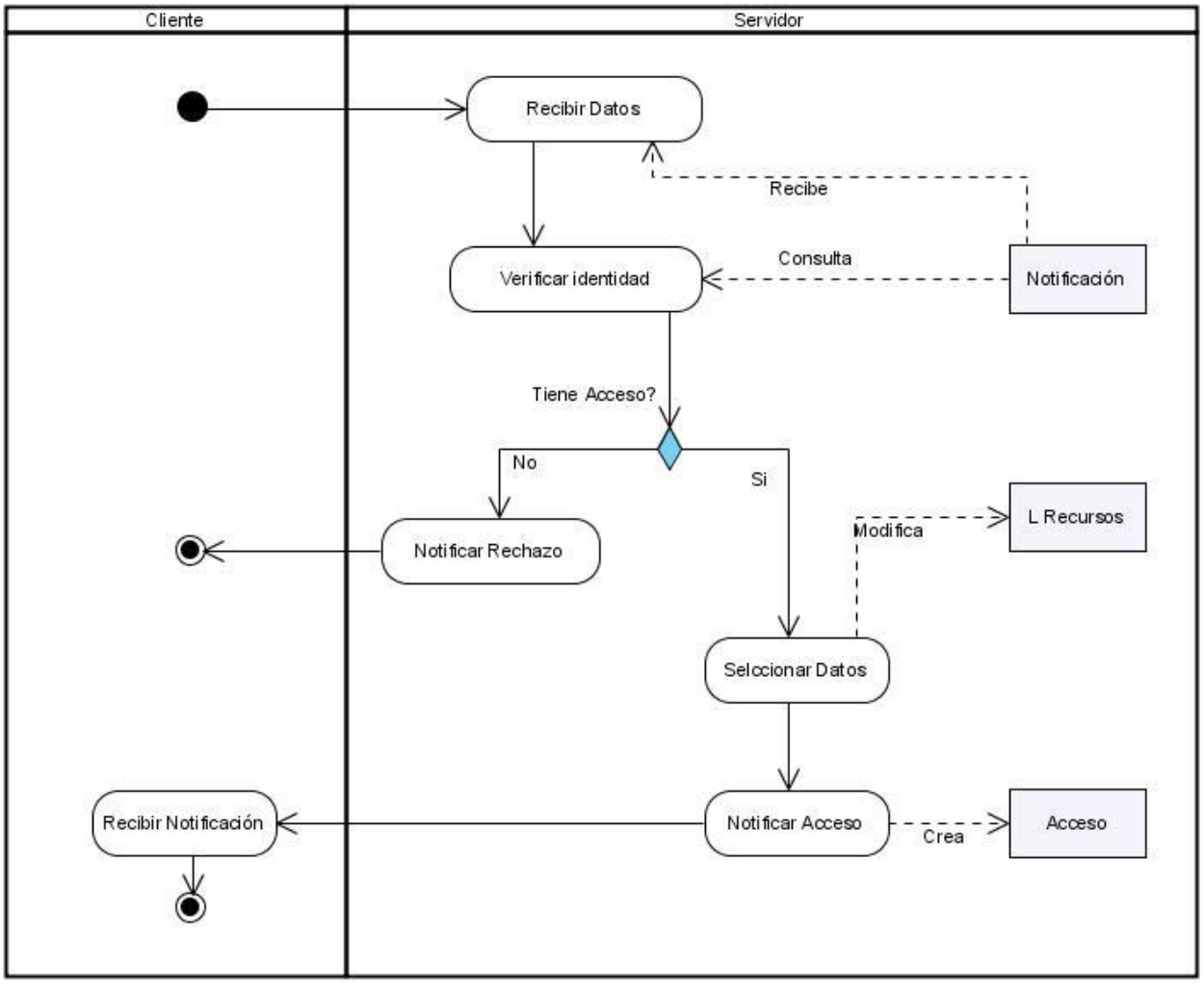

Figura 4 Proceso de acceso a un cliente determinado para realizar copia de respaldo

2.4.3 Recopilar información en el cliente y enviar datos generales

El servidor envía la información de **Acceso** al cliente, que contiene el **Listado de Recursos** con los datos que no se encuentran actualizados en el servidor y un conjunto de datos referentes al formato de comunicación con este. Los datos que se encuentran en el **Listado de Recursos** son buscados, organizados y compactados. Una vez empaquetados los datos bajo algún algoritmo de compresión, se procede a enviar este único archivo al servidor que lo recibirá y lo procesará.

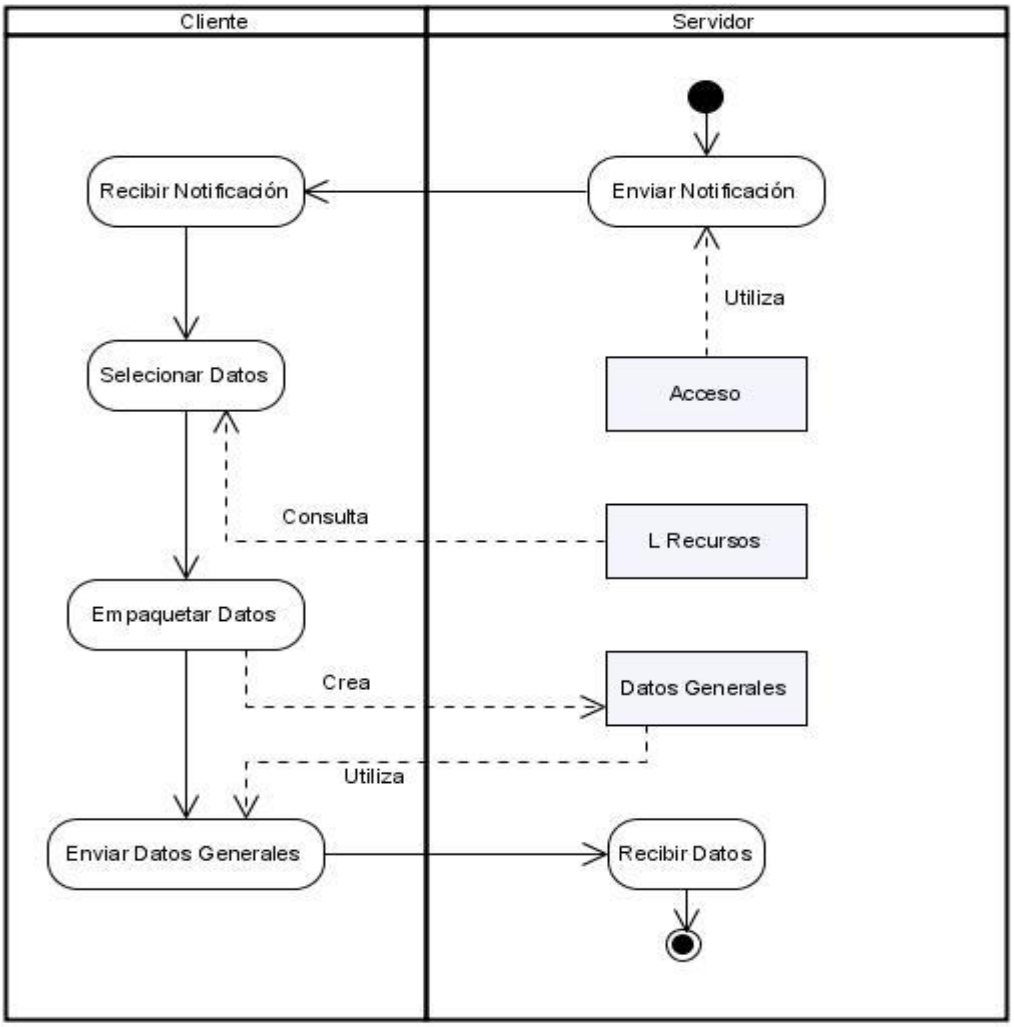

Figura 5 Proceso de recopilar información en el cliente y enviar datos generales

2.4.4 Publicar notificación de datos actualizados

Luego de encontrarse en el servidor todos los datos a respaldar, el mismo procede a organizarlos de acuerdo a la información obtenida al principio del proceso a través del objeto **Notificación**. De esta forma se pueden tener organizados los datos de respaldo de acuerdo al proyecto, tipos de datos, fecha en que se realizó la salva, etc. El próximo paso sería la publicación de un resumen de los datos actualizados, de esta forma los servidores de respaldo secundarios pueden realizar una comparación, entre los datos que ellos poseen y los publicados, y conocer realmente lo que ellos necesitan copiar.

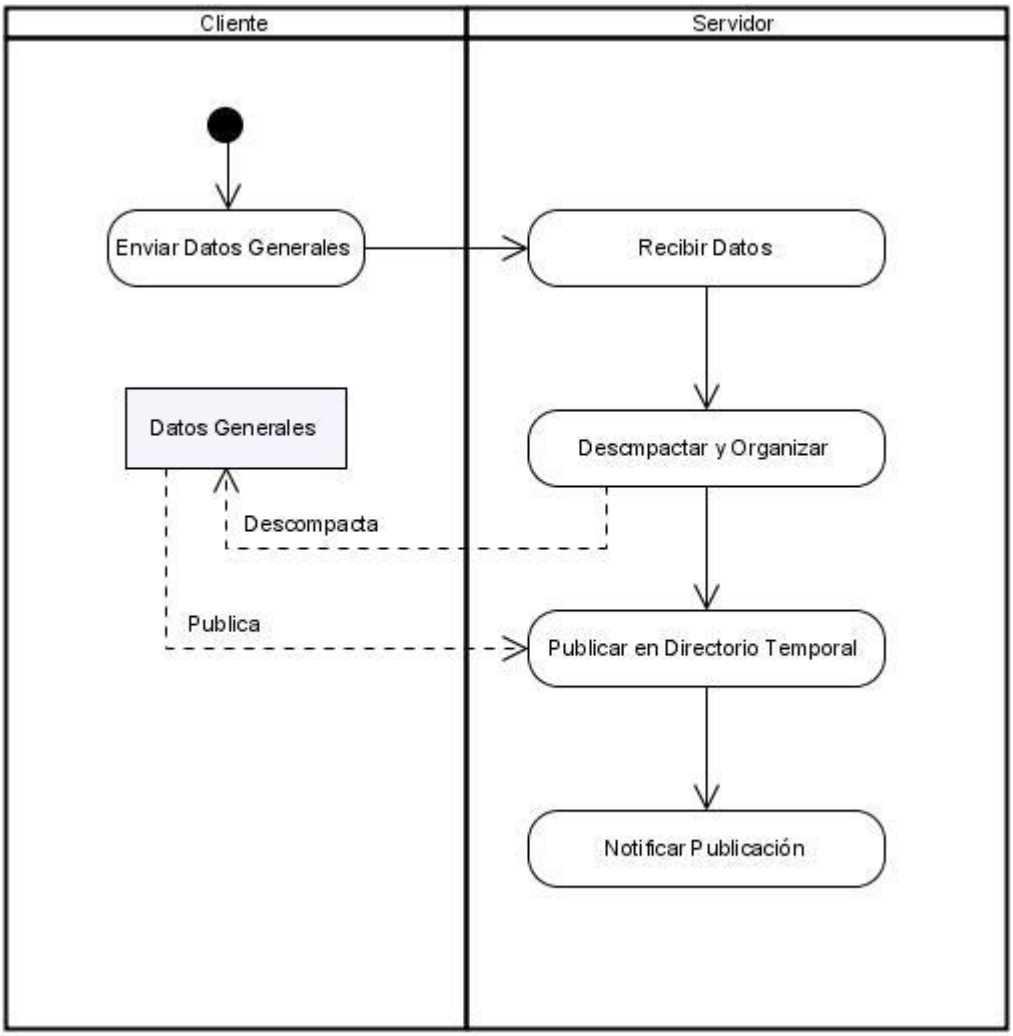

Figura 6 Proceso de publicar notificación de datos actualizados

2.4.5 Replicar datos a servidores secundarios

El proceso de réplica hacia servidores secundarios comienza con la notificación de datos actualizados que es recibida por todos los implicados. Esta notificación contendrá un objeto **Listado de Recursos** para que cada servidor pueda seleccionar los datos que no posee y de esta manera se asegura que la copia siempre sea incremental. Cada servidor secundario le envía al servidor central su **Listado de Recursos** modificado en dependencia a la necesidad de cada uno y un conjunto de datos que lo identifican como servidor secundario con acceso a la copia de los mismos. El servidor central verifica la veracidad de cada pedido y en caso de otorgar el acceso procede a notificar el formato para realizar la copia, quedando a responsabilidad del servidor secundario esta tarea.

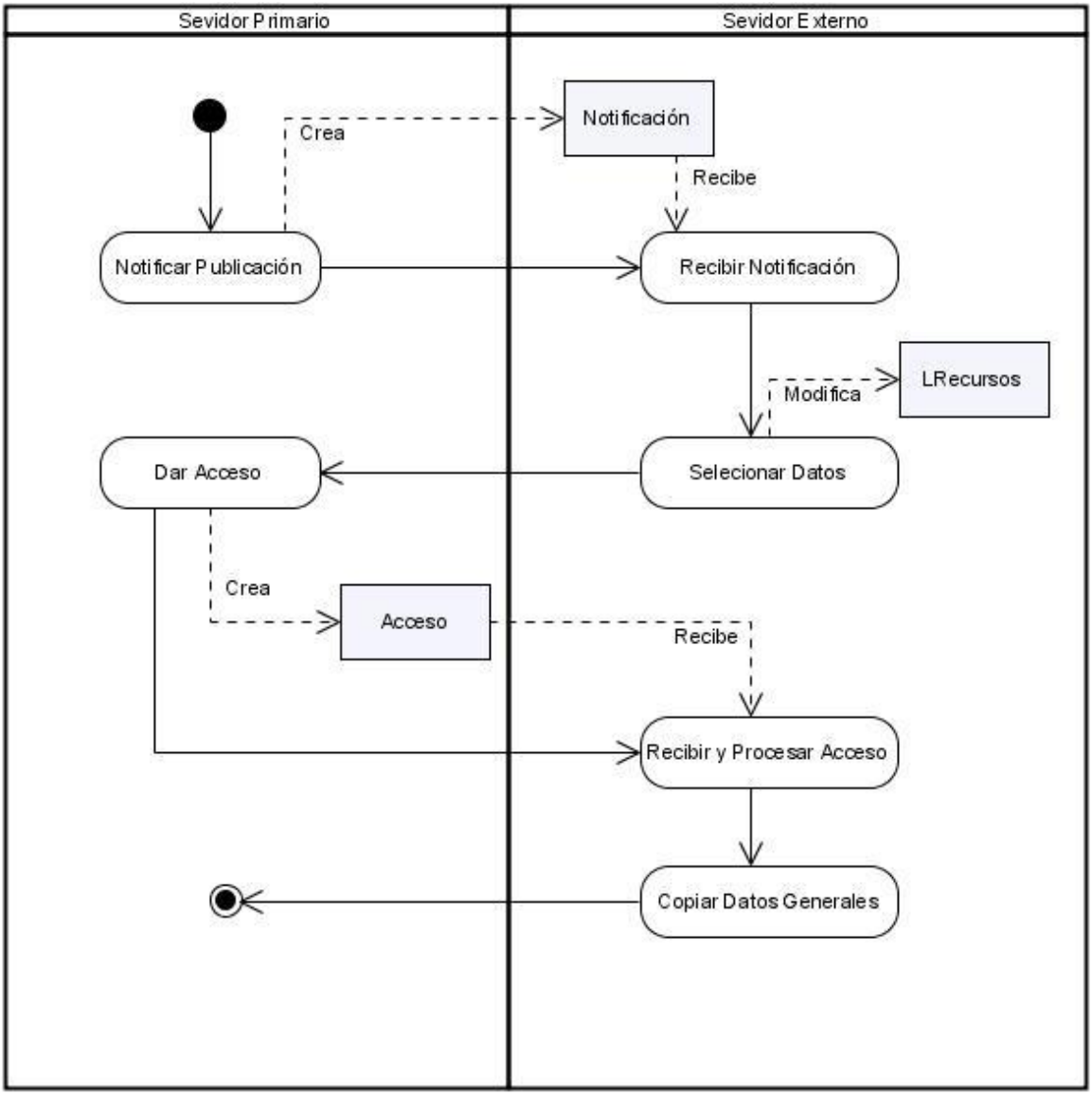

Figura 7 Proceso de replicar datos a servidores secundarios

2.4.6 Informar estado

En todo el proceso se han identificado puntos que pueden aportar información acerca del progreso del mismo. Cada uno de estos puntos reflejados en la Figura 8, notifican al sistema de monitoreo para tener un conocimiento concreto en cada momento y poder llevar un mejor control del proceso en general.

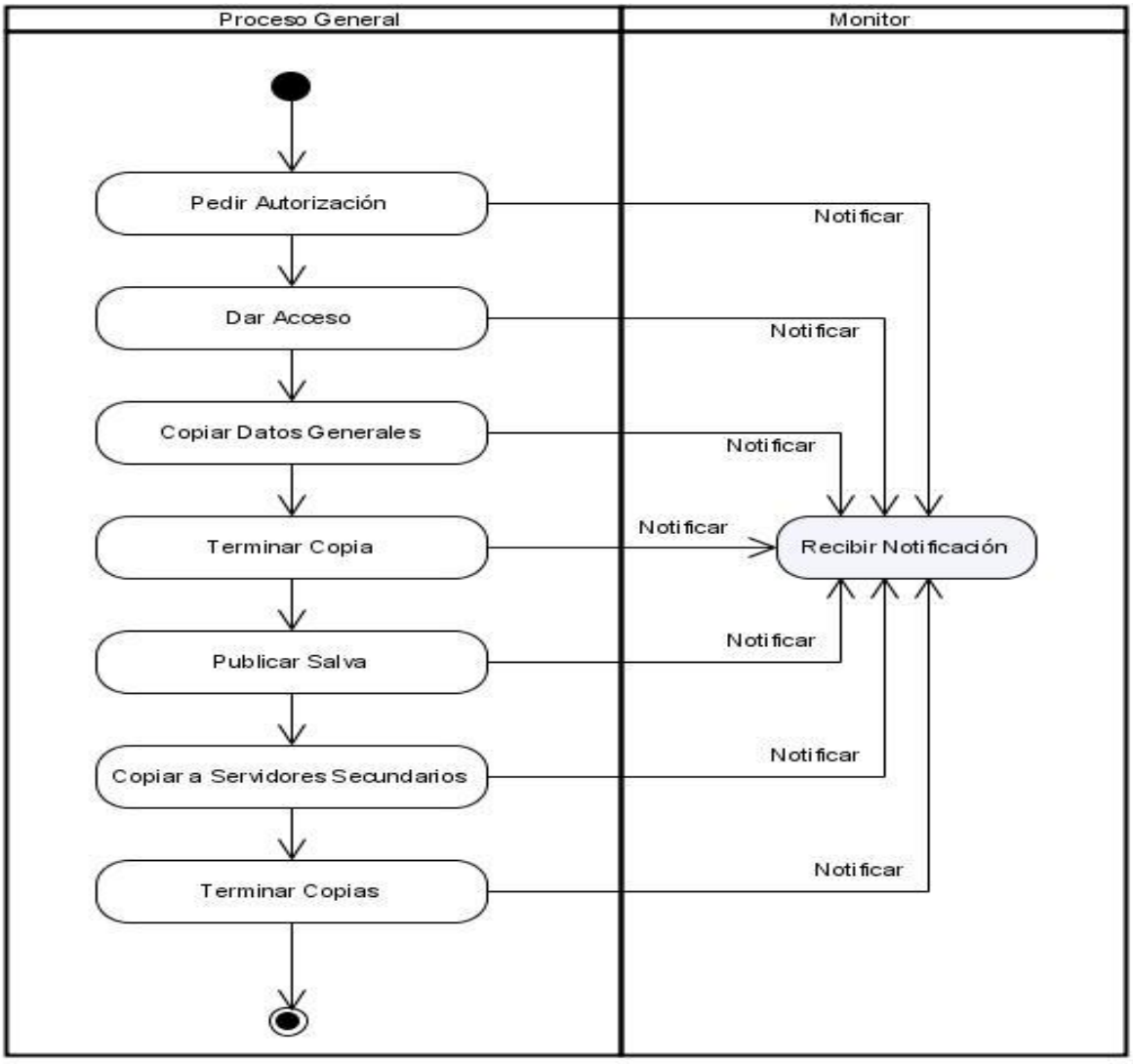

Figura 8 Proceso informar estado

2.5 Adaptación al medio

Una de las características principales de la Universidad de las Ciencias Informáticas es su constante dinámica de cambio, por lo que el mecanismo debe ser capaz de adaptarse a ellos y mantener su compatibilidad con las estructuras anteriores. En la actualidad cada proyecto es el responsable de guardar todo el código referente a los productos que desarrolla, quedando en sus laboratorios el repositorio de código y los servidores de bases de datos. Se puede encontrar un repositorio de código para cada proyecto configurado e instalado en las máquinas del mismo.

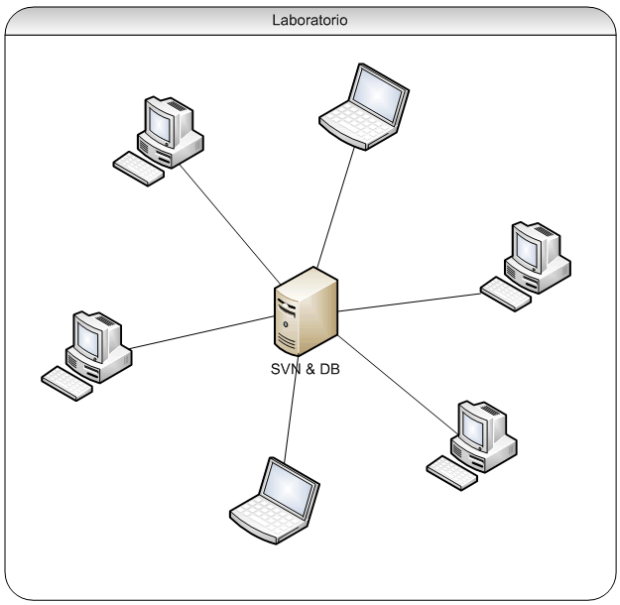

Figura 9 Distribución actual del repositorio de código y la base de datos

De acuerdo a esta estructura, el respaldo de datos sería de dos niveles, el primero a nivel de facultad, cada facultad posee una salva de todos los proyectos que se encuentran en su dominio, y el segundo a nivel de Universidad, los servidores centrales de IP respaldarán los datos que se encuentran en cada facultad.

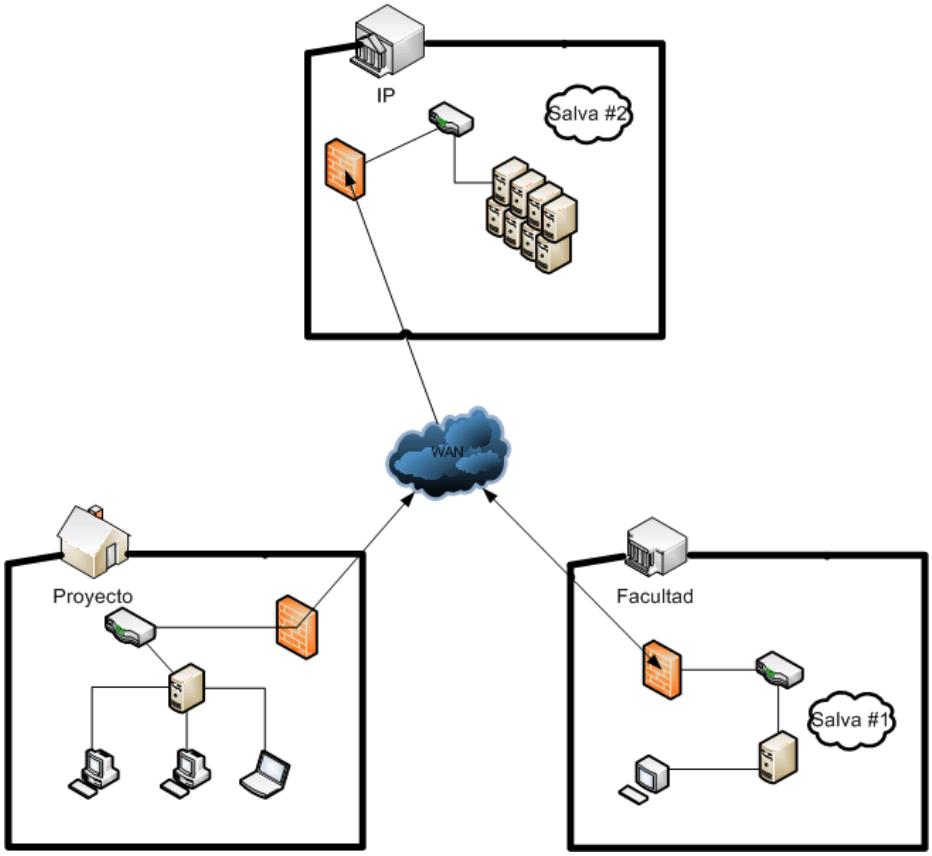

Figura 10 Distribución actual de los dos niveles de salva

Esta distribución puede estar sujeta a cambios estructurales. Para un mejor control y manejo por las facultades de los proyectos que se desarrollan, se podrían centralizar los repositorios de código, contando cada proyecto con una instancia a la cual solo tendría acceso el personal autorizado. Esta misma distribución podría extenderse a las bases de datos, quedando instalados en la facultad los sistemas de gestión de bases de datos autorizados por la Universidad y cada proyecto, a medida que es solicitado, poseer una instancia de la que utiliza.

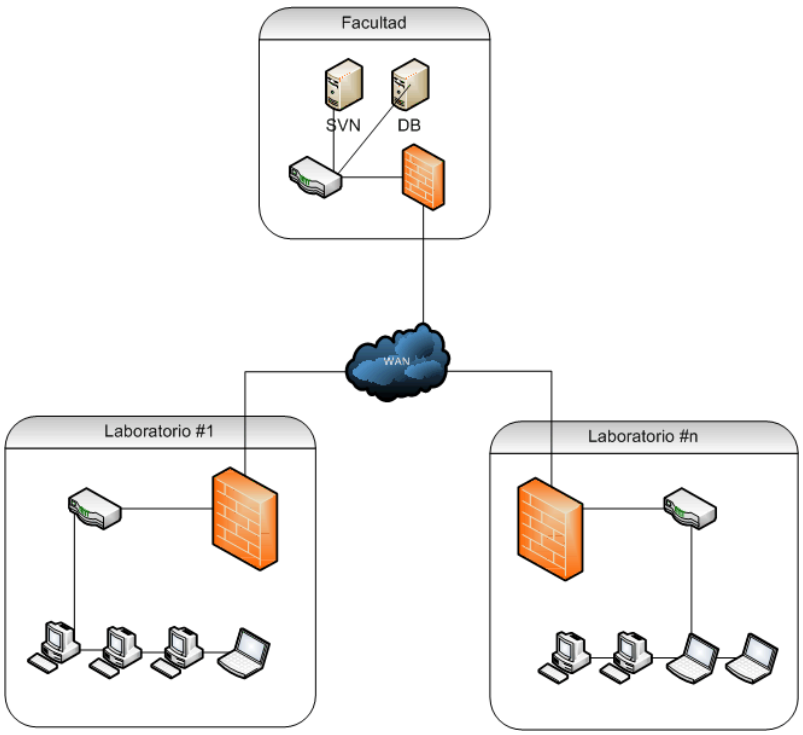

Figura 11 Servidores centrales por facultad

De acuerdo a esta estructura, el respaldo de datos sería de un nivel, como cada facultad posee toda la información referente a sus proyectos, el respaldo de datos se realizaría desde estos servidores hacia los de la universidad, acortando la separación existente entre los desarrolladores y la salva central.

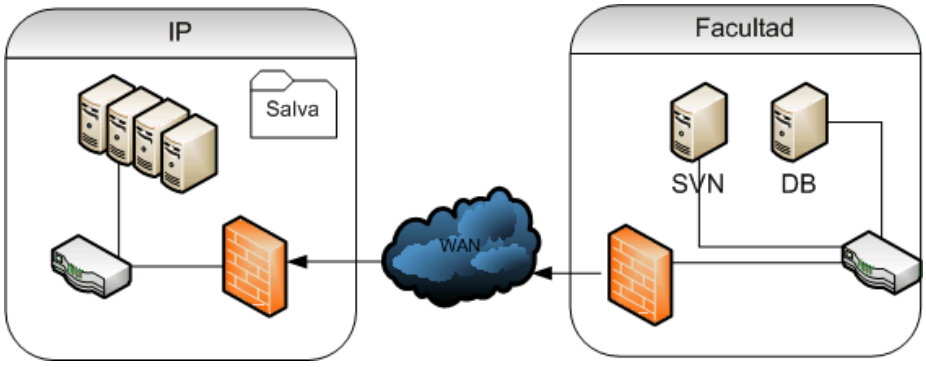

Figura 12 Distribución de un nivel

Al SAC le son indiferentes los niveles convergentes de datos que existan, su funcionamiento se abstrae para una mejor portabilidad. Esto se podría ver como un árbol, siendo la raíz la salva central de la institución y los hijos de esta los diferentes puntos de convergencia de datos que existan, quedando a responsabilidad de los nodos padres el respaldo efectivo de sus hijos.

En el primer caso se instalaría el servicio **Cliente** en los servidores centrales de cada proyecto para tener acceso a toda la información que será respaldada. El servicio **Servidor** será instalado en los servidores de cada facultad, siendo el punto de convergencia de datos. De esta manera cada facultad es la responsable de controlar y mantener las salvas referentes a sus proyectos, dejando a responsabilidad de la Universidad la salva referente a los datos que se encuentran en los servidores de cada facultad. Esta responsabilidad se materializa instalando en cada facultad el servicio **Cliente** y manteniendo en los servidores centrales de IP[40] el servicio **Servidor**.

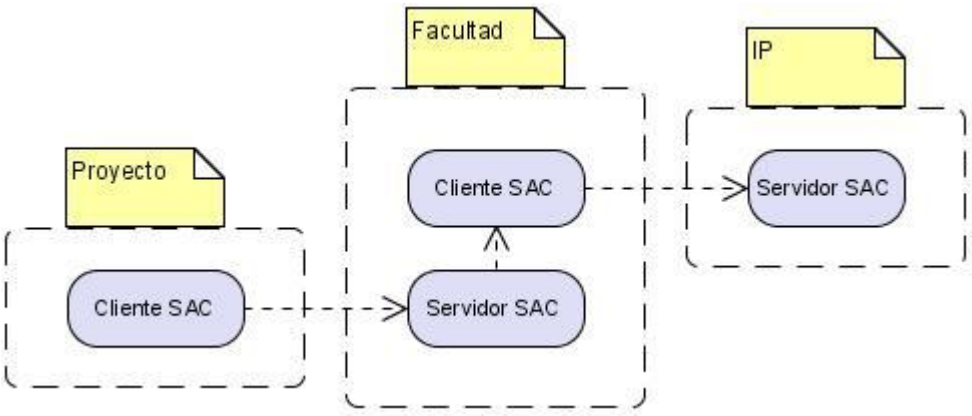

Figura 13 Distribución de los servicios del SAC (dos niveles)

Para el segundo de los casos se instalaría el servicio **Cliente** en los servidores de cada facultad, quedando a bajo su responsabilidad la extracción de datos de los repositorios de código y las bases de datos, y el servicio **Servidor** en los servidores centrales de la Universidad, manteniendo la lógica explicada en los epígrafes anteriores.

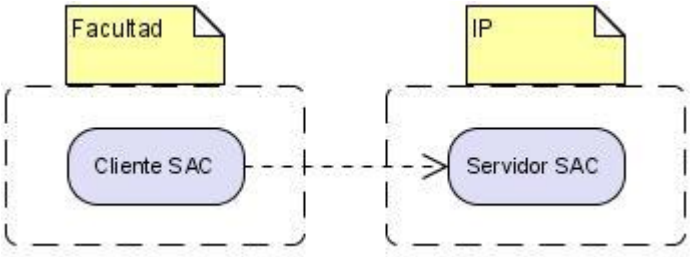

Figura 14 Distribución de los servicios del SAC (un nivel)

[40] Infraestructura Productiva

2.6 Mecanismo de restauración y acceso de los datos respaldados

Existen muchas maneras de realizar la restauración de datos, cada una con sus ventajas y desventajas de acuerdo a los protocolos que se utilicen. Se propone, para tener un control de los datos que han sido respaldados, la entrega a cada líder de proyecto, una vía de acceso a un sitio web donde encontrará registrada toda la información referente a todas las salvas del proyecto que lidera, de esta forma él podría elegir la salva más representativa de acuerdo a su necesidad y descargarla sin previa solicitud, quedando registrado cada movimiento realizado por este en el servidor. El protocolo usado será HTTPs, siendo este un canal seguro de comunicación. De esta manera queda involucrado al proceso la persona que más conocimiento tiene del tema y queda a su responsabilidad el uso que haga de los datos que considera conveniente recuperar.

Como cada proyecto tendrá sus salvas actualizadas diariamente, se propone que además del líder de proyecto, también tenga acceso a la información salvada el personal de calidad que atiende al proyecto por parte de la Universidad. De esta manera se podrían realizar pruebas al producto conjuntamente con el desarrollo del mismo, evitando la carga de revisar todo el producto de una vez y así poder contar con liberaciones de mejor calidad y confianza.

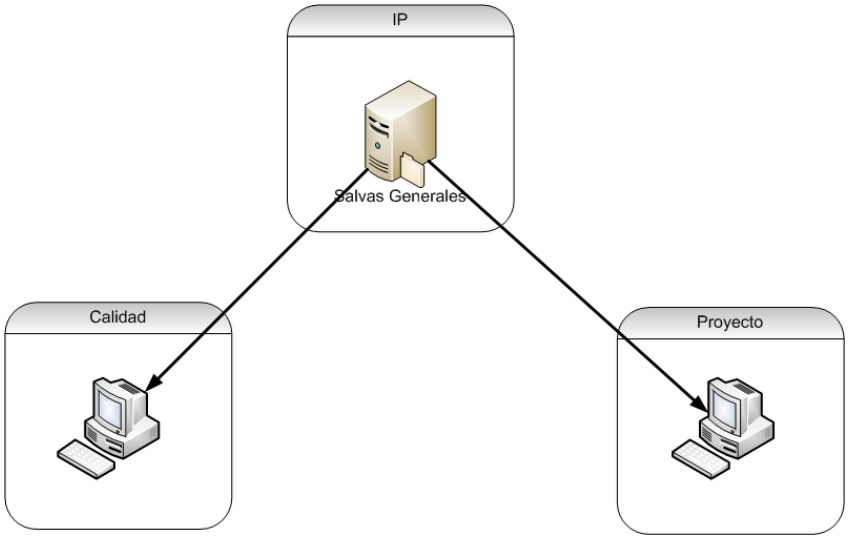

Figura 15 Descarga de las salvas

Conclusiones

En este capítulo se ha adaptado el modelo de respaldo de software estudiado en el Capítulo1 para dar solución a la problemática existente en la Universidad de las Ciencias Informáticas, además se introdujo el flujo de actividades necesarias que el mecanismo debe cumplir, así como las entidades identificadas en el mismo.

CAPÍTULO 3: CONSTRUCCIÓN DEL PROTOTIPO

Introducción

En el presente capítulo se expone cómo se ha implementado un prototipo inicial del mecanismo propuesto en el capítulo anterior. Se comienza hablando sobre las aplicaciones que se han construido y se describen las herramientas que utilizan. Se plasma el diseño de las clases de cada una de las aplicaciones construidas y una breve explicación de las más importantes. También se explica cómo interactúan todas estas aplicaciones para realizar el proceso de respaldo y cómo colaboran con un servidor de mensajería para transmitir los datos de una aplicación a otra. Como último tópico se abordan los beneficios de la mensajería y la justificación de su utilización.

3.1 Construcción del Prototipo

En el Capítulo 2 se explicaron cuatro aplicaciones necesarias para realizar el mecanismo de respaldo de datos, tratándose de un prototipo inicial solo se describirá el proceso de tres de ellas, dejando el sistema de monitoreo para posteriores iteraciones.

Las aplicaciones propuestas para el prototipo inicial son:

- Una aplicación del lado del cliente, que será la encargada de iniciar el proceso de salva. En ella recae todo el peso de la recopilación y selección de los datos a respaldar.
- Una aplicación del lado del servidor, que será el encargado de manipular y autorizar los pedidos de salva. También realiza la selección de datos para evitar redundancias y notifica el fin del proceso de respaldo de datos del cliente para dar paso a las réplicas y/o nuevas peticiones de respaldo.
- Una aplicación de réplica que será la encargada de realizar las copias de respaldo del servidor central hacia servidores externos.

Las tres aplicaciones están hechas sobre la plataforma Java (35) (36). El hecho de estar construidas en este lenguaje hace que estas sean multiplataforma, lo que quiere decir que el mismo software puede ser ejecutado en diferentes sistemas operativos sin que sea necesario cambiar ninguna línea de código. Por ejemplo, estas aplicaciones pueden ser ejecutadas en sistemas Windows como Windows Xp, Windows 2000, de la misma forma que en sistemas GNU/Linux, como Debian, Ubuntu, Gentoo, RedHat, entre otros.

Para la construcción de dichas aplicaciones se ha hecho uso del framework Spring. Un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Los frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. En el caso de Spring tenemos un framework que se integra con varios servidores de mensajería y en especial con ActiveMQ. Esta integración permite que el proceso sea mucho más simple, asignando las tareas de bajo nivel al framework, y dejando a cargo de la aplicación únicamente las tareas funcionales.

Como el cliente y el servidor se encuentran geográficamente distantes una de la otra, es necesario un medio seguro y confiable por el cual transferir los datos a respaldar, para esto se ha utilizado el servidor de mensajería Apache ActiveMQ. Se ha decidido utilizar este servidor de mensajería sobre los otros tantos que existen como Fiorano, Sonic, Joram, entre otros, por las siguientes razones:

- Simplicidad y sencillez de configuración. Esto hace posible que la puesta en marcha del servidor de mensajería sea un proceso rápido y seguro, ya que a medida que aumenta la complejidad de configuración, aumentan los posibles errores y disminuye la curva de aprendizaje.
- Soporte de una gran variedad de lenguajes. De esta forma podemos interactuar con ActiveMQ desde Java, C, C++, C#, Ruby, Perl, Python, PHP, lo cual posibilita que cualquiera de las aplicaciones propuestas sea desarrollada en alguno de estos lenguajes. Esto sería útil si en una futura versión del sistema se desea realizar un cambio tecnológico.
- Integración con Spring. ActiveMQ puede ser fácilmente embebido y configurado en una aplicación hecha con el marco de trabajo Spring lo cual aumenta la facilidad de configuración y la integración con el prototipo propuesto.
- Soporte de varios protocolos de comunicación como TCP, SSL, UDP, etc. De acuerdo con las características del sistema en general, es necesario que la información, y en este caso el envío de datos entre el servidor y los clientes, viaje por un medio seguro de comunicación. ActiveMQ brinda una manera fácil de enviar mensajes sobre SSL lo cual hace que los datos a respaldar viajen por un medio fiable y seguro. (33)
- Contiene varios sistemas de monitoreo, ayudando a la depuración de errores por parte de las aplicaciones y a la toma de decisiones en cuanto al desarrollo del producto.

Para un mejor entendimiento del proceso que se explicará a lo largo del capítulo, y siguiendo el orden de las actividades plasmadas en el capítulo anterior, se diseñó un diagrama donde se muestra el orden en que son enviados los mensajes.

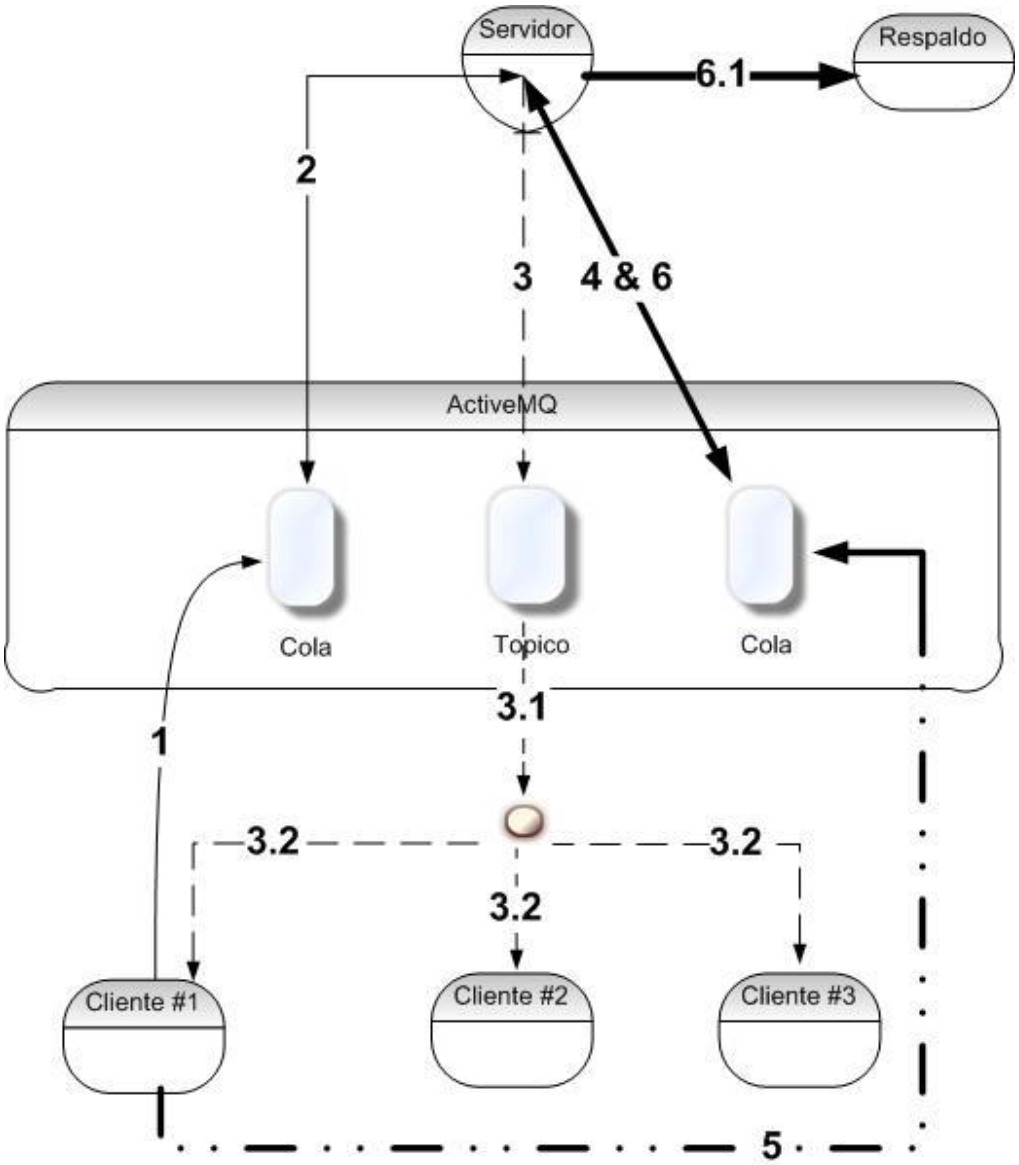

Figura 16 Modelo general del SAC

3.1.1 Cliente Gráfico

El proceso de salva contendrá sus regulaciones. Para poder optar por este servicio el líder de proyecto debe realizar el pedido y la justificación del mismo a los directivos de la universidad, de esta forma evitamos que se utilice de manera indebida. Si el pedido se aprueba, un representante del

servicio instalará el sistema como es debido y en los lugares donde sea requerido en las máquinas del cliente. Otra de las responsabilidades de dicho representante es generar el certificado de acceso e identificación a través de un programa dedicado a esta tarea. El certificado nos ayuda a controlar la seguridad y consiste en un objeto encriptado y serializado que contendrá entre otras cosas el nombre del proyecto, contraseña y una breve descripción como se muestra en la Figura 17.

Figura 17 Prototipo de interfaz para las propiedades de la aplicación Cliente (I)

También se definirán los archivos o directorios que serán respaldados. En iteraciones posteriores se agregará la posibilidad de extraer datos de los repositorios de código y los sistemas de bases de datos. El archivo generado, además de ubicarse en una dirección conocida por la aplicación cliente, es copiado hacia el servidor dentro de una carpeta con el nombre del proyecto, de esta manera el servidor solo le otorgará acceso a los clientes que contengan el mismo certificado que él tiene de ellos. Para cualquier modificación que se desee realizar en el certificado es obligatorio la presencia del responsable del servicio, de esta manera aseguramos que los archivos a respaldar sean realmente necesarios para el proyecto que solicita el respaldo. Como una comodidad en la interfaz gráfica se utilizó el componente **FileChooser** (37), que nos permite la selección de ficheros y directorios, quitando la posibilidad de error en cuanto a la escritura de los caminos a respaldar como se muestra en la Figura 18.

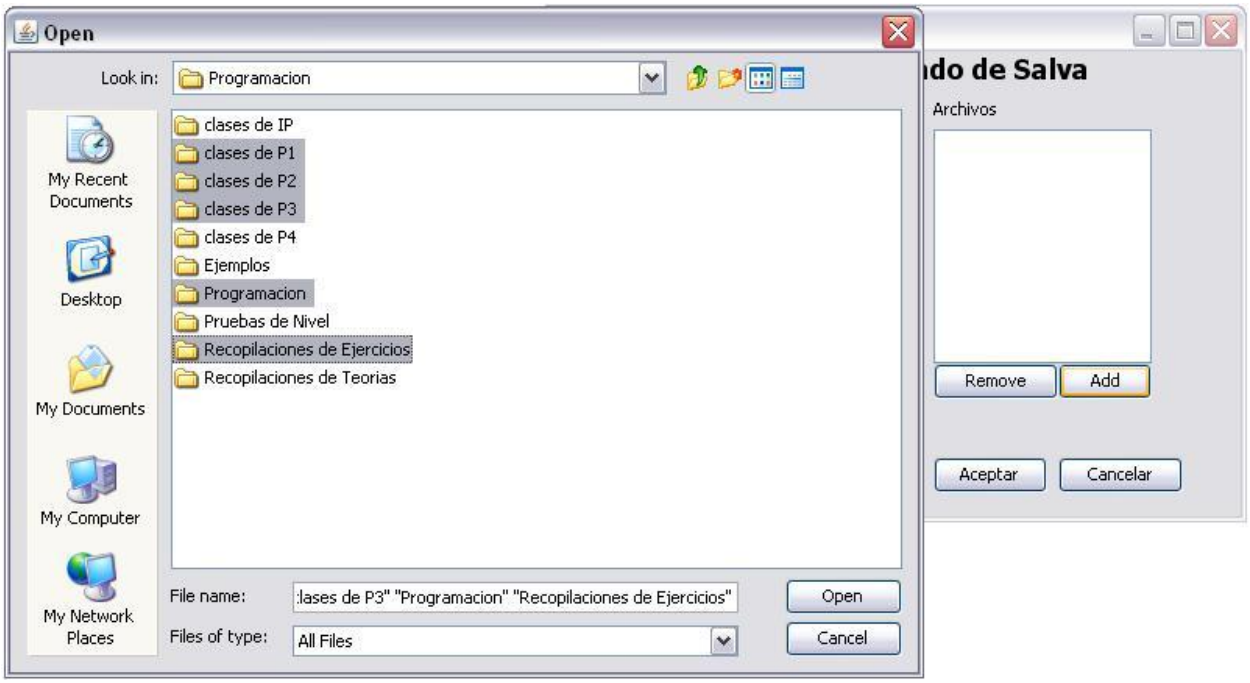

Figura 18 Prototipo de interfaz para las propiedades de la aplicación Cliente (II)

3.1.2 Aplicación Cliente

La aplicación cliente es la encargada de iniciar el proceso de salva. Su principal responsabilidad, como se explicó en el Capítulo 2, es reunir los datos que serán respaldados y enviarlos al agente de mensajes. Como las configuraciones necesarias para el correcto funcionamiento de la aplicación se insertaron por el responsable del servicio en etapas iniciales, la aplicación no necesita interfaz gráfica, y su manipulación está dada por un objeto derivado de la clase **TimerTask**. Heredar de la clase **TimerTask**, e implementar el método **run()**, nos da la posibilidad de crear una tarea programada para una sola vez o reiteradas ejecuciones por un temporizador. De esta manera se iniciaría el proceso una vez definido el horario o los intervalos de tiempo necesarios (38). Para empaquetar los ficheros que serán respaldados se ha utilizado la clase **ZipOutputStream**. Esta clase implementa un filtro de salida para escribir archivos en el formato de archivo ZIP e incluye soporte para entradas comprimidas y sin comprimir (39).

Un patrón de diseño es una solución a un problema de diseño. Estos pretenden proporcionar catálogos de elementos reusables en el diseño de sistemas software evitando la reiteración en la búsqueda de

soluciones a problemas ya conocidos y solucionados anteriormente. A través de ellos se puede facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

Entre los patrones de diseño más relevantes que se utilizaron en la aplicación cliente se encuentran:

- **Facade (Fachada):** Patrón estructural que provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema. (40)
- **Command (Orden):** Patrón de comportamiento que encapsula una operación en un objeto, permitiendo ejecutar dicha operación sin necesidad de conocer el contenido de la misma. (40)
- **Strategy (Estrategia):** Patrón de comportamiento que permite disponer de varios métodos para resolver un problema y elegir cuál utilizar en tiempo de ejecución. (40)
- **Singleton (Instancia única):** Patrón de creación que garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. (40)

En la aplicación se definieron siete responsabilidades centrales que realizarán las tareas fundamentales de la misma:

- **Common:** Módulo que posee todas las funcionalidades comunes dentro del software. En él se definen los ficheros de configuración necesarios que serán cargados por la aplicación para la realización de sus labores, el paquete de utilidades y las clases del dominio.
- **Properties:** Módulo encargado de la obtención de propiedades del software, ya sean el certificado de acceso e identificación como las colas que necesita conocer para el buen funcionamiento de la aplicación.
- **Flow:** Módulo encargado del envío de los datos que serán respaldados desde el cliente hacia el agente de mensajes.
- **Notification:** Módulo encargado de pedir y obtener el acceso al proceso de respaldo de datos.
- **Summary:** Módulo encargado de recopilar los datos que serán respaldados. También es su responsabilidad la de unificarlos en un solo fichero para un único envío a través de un mecanismo de compactado.
- **Time:** Módulo encargado de la programación de tareas. Aquí se definirán los momentos en que la aplicación comenzará el proceso de respaldo de datos y la periodicidad del mismo.

- **Communication:** Módulo encargado del envío y la recepción de toda la información entre la aplicación cliente y el agente de mensajes.

3.1.2.1 Diseño de clases de la aplicación Cliente

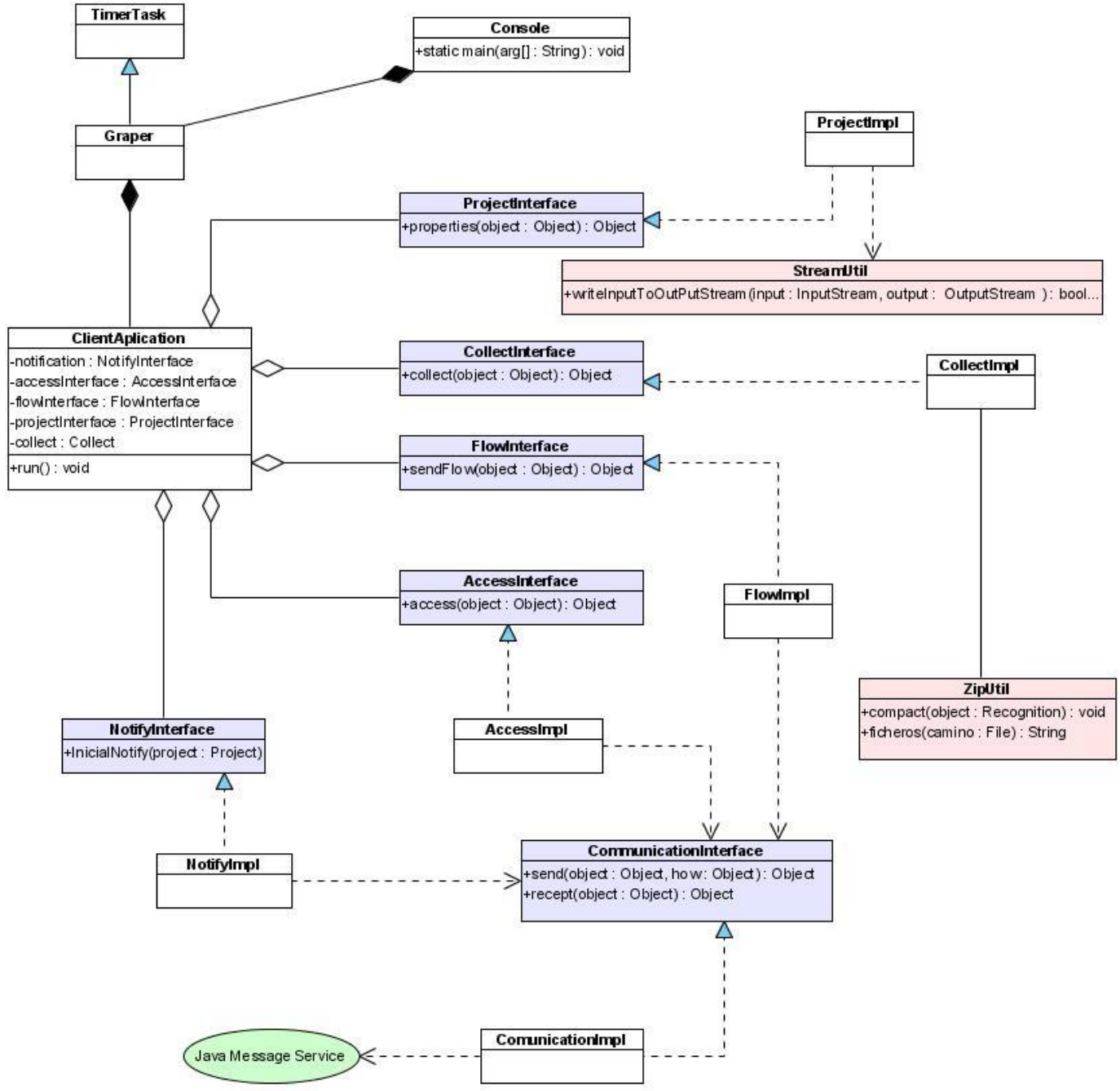

Figura 19 Diseño de clases de la aplicación Cliente

3.1.2.2 Descripción de las clases más importantes

A la hora de realizar el respaldo todas las clases son importantes, aunque algunas solo existen para una mejor portabilidad y mantenimiento del software. El uso de las interfaces para separar las responsabilidades modularmente permite que cualquier cambio que exista en las próximas iteraciones no interfiera en las capas superiores.

Las clases más representativas son:

- **Graper:** Clase encargada de velar por que se cumplan las tareas en el tiempo previamente configurado. Hereda de la clase **TimerTask** [4].
- **ProjectImpl:** Clase encargada de cargar todas las propiedades del **Cliente**. También es la responsable de reconocer cuándo ha ocurrido un cambio en el fichero de certificado y actualizarse en caso de que sea necesario.
- **CollectImpl:** Clase encargada de recopilar todos los datos que serán respaldados. Para un mejor funcionamiento de la misma se implementaron especialistas en cada una de las áreas destinadas al almacenamiento de datos como cinta magnética, ficheros locales y remotos, sistemas de bases de datos, repositorios de código, etc. Esta clase solo es responsable de reunir la información de cada especialista en un solo fichero, compactarlo; para luego ser respaldado.
- **NotifyImpl:** Clase encargada de realizar el pedido inicial de respaldo al servidor. Es la responsable de, luego que la información se encuentre reunida y en orden, enviarla al destino configurado inicialmente para que sea procesada por el servidor.
- **AccessImpl:** Clase encargada de manipular la información de acceso al proceso de respaldo. Su responsabilidad recae en la espera de la autorización, en caso que sea aprobado, procesar la información y notificar a la aplicación el formato elegido por el **Servidor**.
- **FlowImpl:** Clase encargada del envío de los datos a respaldar. Es la responsable de que el fichero auxiliar previamente compactado, que contiene todos los datos que serán respaldados, se envíe correctamente al agente de mensajes y luego sea eliminado.
- **CommunicationImpl:** Clase encargada de la comunicación entre el cliente y el agente de mensajes. Es responsable de que toda la información que salga de la máquina hacia el agente encuentre su destino y viceversa.

- **ZipUtil:** Clase de utilidad. Su responsabilidad es manipular toda la información que desea ser compactada o descompactada.
- **StreamUtil:** Clase de utilidad. Su responsabilidad es copiar grandes flujos de bytes de un lugar a otro.

3.1.3 Aplicación Servidor

La aplicación servidor es la encargada de recibir el flujo de datos que se desea respaldar. Consiste en un servicio que siempre estará activo y listo para procesar cualquier petición. Entre sus responsabilidades se encuentra la de dar acceso al proceso de salva después de la verificación por certificado explicada en el epígrafe anterior y realizar el respaldo si el cliente obtuvo el acceso de forma satisfactoria. A pesar de ser un servicio que siempre debe estar corriendo, se puede detener su funcionamiento sin comprometerse el proceso de salva. Al reanudar sus tareas se encargará de procesar toda la información que fue guardada en el agente de mensajes mientras no se encontraba activo.

El proceso burocrático de petición de respaldo por parte del líder de proyecto concluye cuando es creada una carpeta en el servidor de respaldo con el nombre del proyecto y copiando dentro de la misma el certificado de identidad de este.

Los patrones de diseño utilizados en el servidor fueron los mismos explicados en el epígrafe anterior. En el caso del patrón **Estrategia** se utilizó para proporcionar una vía de respaldar los datos hacia diferentes dispositivos de almacenamiento sin necesidad de cambiar el código previamente implementado.

En la aplicación se definieron cinco responsabilidades centrales que realizarán las tareas fundamentales de la misma:

- **Common:** Módulo que posee todas las funcionalidades comunes dentro del software. En él se definen los ficheros de configuración necesarios que serán cargados por la aplicación para la realización de sus labores, el paquete de utilidades y las clases del dominio.
- **Receive:** Módulo encargado de recibir las peticiones iniciales. Su responsabilidad es escuchar y procesar todas las peticiones que lleguen a la cola pública del agente de mensajes y notificar al módulo de acceso de la nueva petición.

- **Access:** Módulo encargado de procesar y otorgar el acceso a un cliente determinado al proceso de respaldo de datos.
- **Flow:** Módulo encargado del recibo de los datos que serán respaldados desde el agente de mensajes hacia el servidor.
- **Communication:** Módulo encargado del envío y la recepción de toda la información entre la aplicación servidora y el agente de mensajes.

3.1.3.1 Diseño de clases de la aplicación Servidor

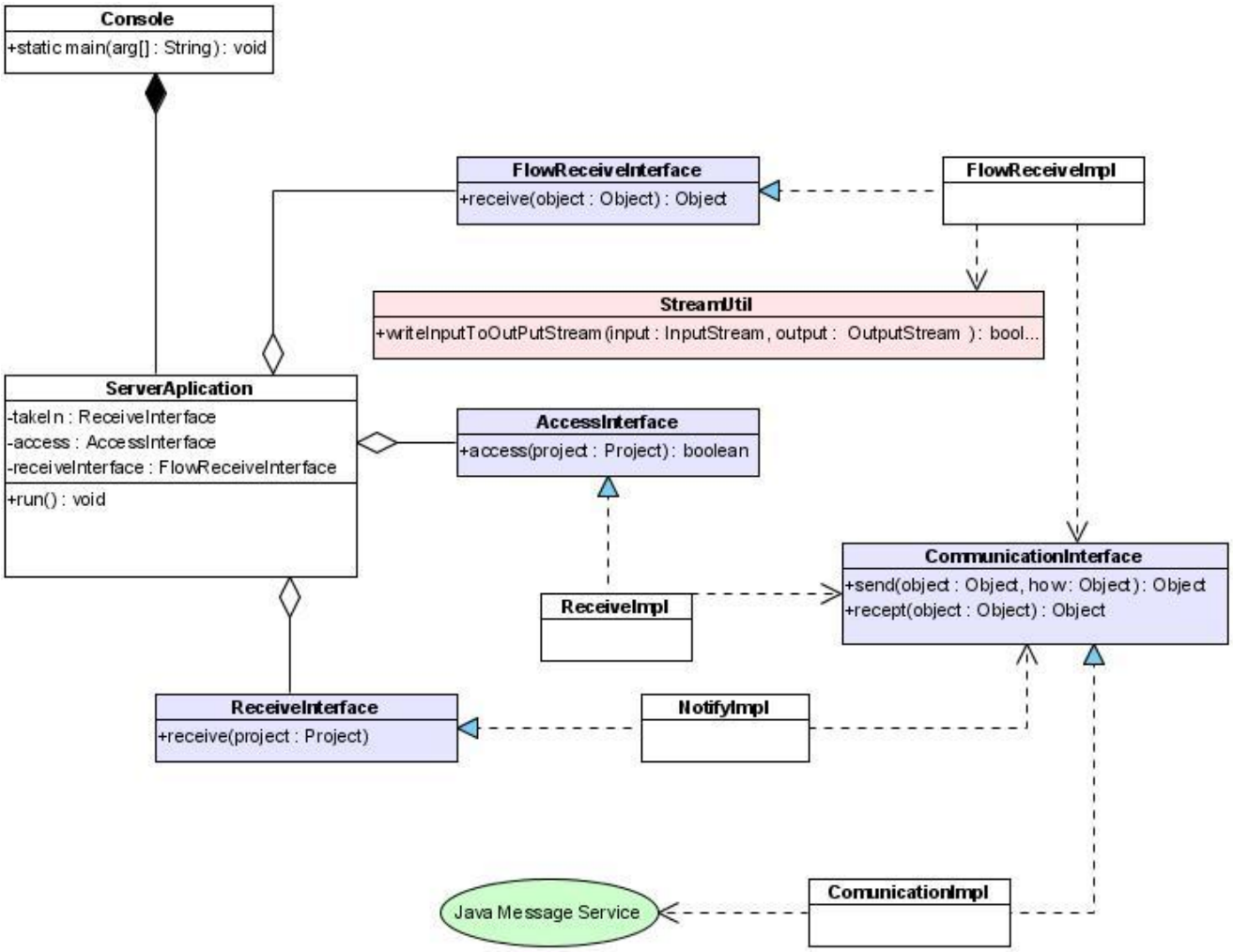

Figura 20 Diseño de la aplicación Servidor

3.1.3.2 Descripción de las clases más importantes

En el servidor se mantuvo la misma línea de implementación que en el cliente haciendo gran énfasis en la separación atómica de las responsabilidades para un menor impacto a la hora de realizar modificaciones.

Las clases más representativas son:

- **ReceiveImpl:** Clase encargada de escuchar las peticiones iniciales. Su responsabilidad recae en la obtención de los pedidos de respaldo y notificar a la clase de acceso para su procesamiento. De esta forma comienza el proceso de respaldo por parte del servidor.
- **AccessImpl:** Clase encargada de manipular la información de acceso al proceso de respaldo. Su responsabilidad recae en procesamiento de la petición inicial y la comparación de los certificados para otorgar la autorización en caso que sea aprobado. Es uno de los nodos de seguridad de la aplicación.
- **FlowImpl:** Clase encargada del recibo de los datos a respaldar. Es la responsable de recibir el fichero compactado enviado por el agente de mensajes y ubicarlo en su correcta localización en el servidor de recursos.
- **CommunicationImpl:** Clase encargada de la comunicación entre el cliente y el agente de mensajes. Es responsable de que toda la información que salga de la máquina hacia el agente encuentre su destino y viceversa.
- **StreamUtil:** Clase de utilidad. Su responsabilidad es copiar grandes flujos de bytes de un lugar a otro.

3.1.4 Réplica y Monitoreo

La implementación del sistema de réplica, así como el de monitoreo, se dejará para posteriores iteraciones del producto. En las siguientes figuras se desea plasmar las ideas centrales de cada uno utilizando las mismas actividades que se plasmaron en el capítulo anterior. Proponer que el sistema de Monitoreo, para un mejor rendimiento y alcance, se desarrolle vía WEB[41], que su interfaz gráfica se realice utilizando la tecnología Ajax (40), el framework Dojo (41) en específico por sus altas prestaciones y facilidades con el manejo de JavaScript[8] y los envíos asincrónicos de datos.

[41] Red Global Mundial es un sistema de documentos de hipertexto y/o hipermedios enlazados y accesibles a través de Internet.

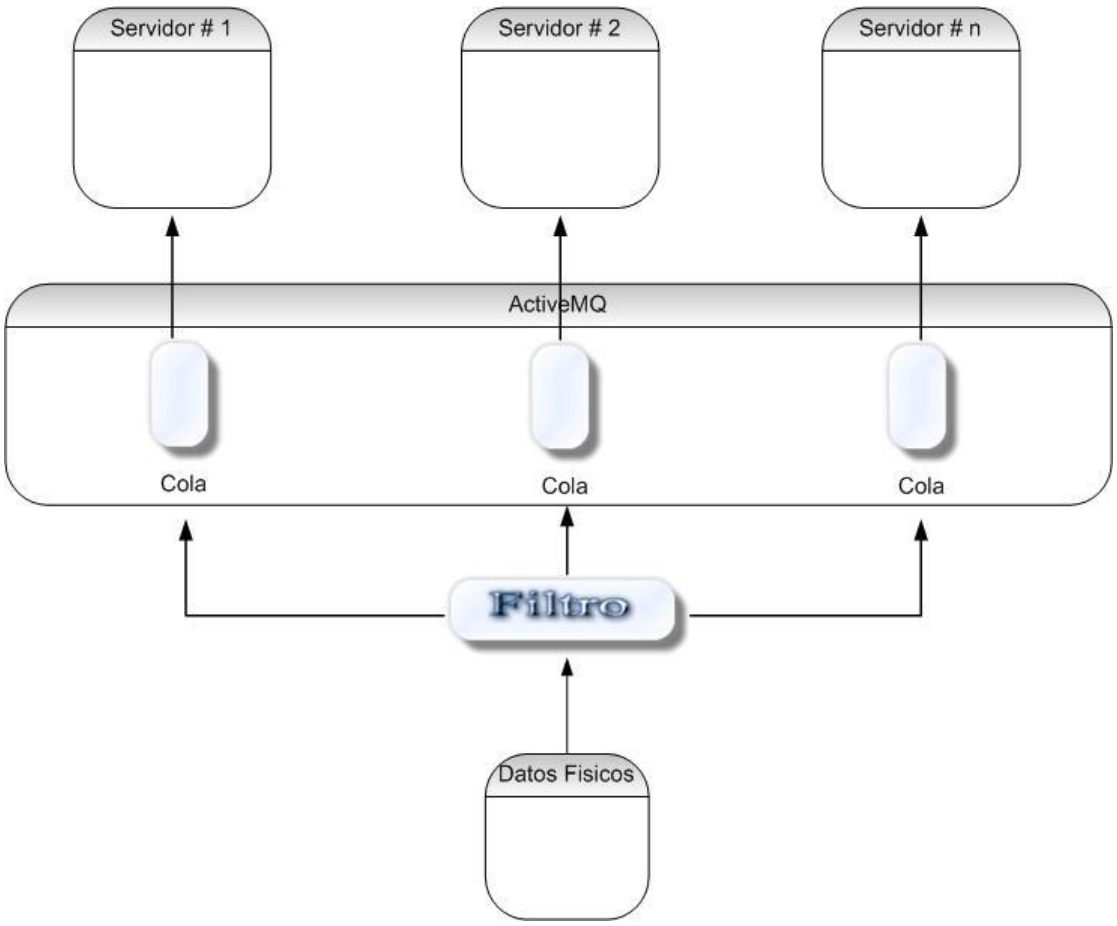

Figura 21 Diagrama de réplica

El sistema de réplica estará compuesto por dos aplicaciones, una aplicación que se instalará en el servidor central donde se respaldan los datos de los cliente según los flujos explicados hasta el momento, que se comportará como un cliente del SAC, y una aplicación que se instalará en cada máquina hacia donde se desee replicar los datos, la cual se comportará como un servidor de SAC. Por cada servidor de respaldo se creará una cola única en el agente de mensajes. No se proponen los apartados por su lentitud en el envío de datos demasiado grandes, como los que se podrán manejar con este sistema, y para poder darle prioridad a cada respaldo. Para abstraer al servidor que desea ser respaldado de los destinos se creó una capa de abstracción que consiste en filtros, utilizando las funcionalidades del paquete **java.io** (42) , especializados en lectura, escritura y manejo de ficheros, que será el encargado de distribuir las copias cargando los destinos de un fichero de configuración. Las copias siempre serán incrementales para un mejor rendimiento de la red y el software en general.

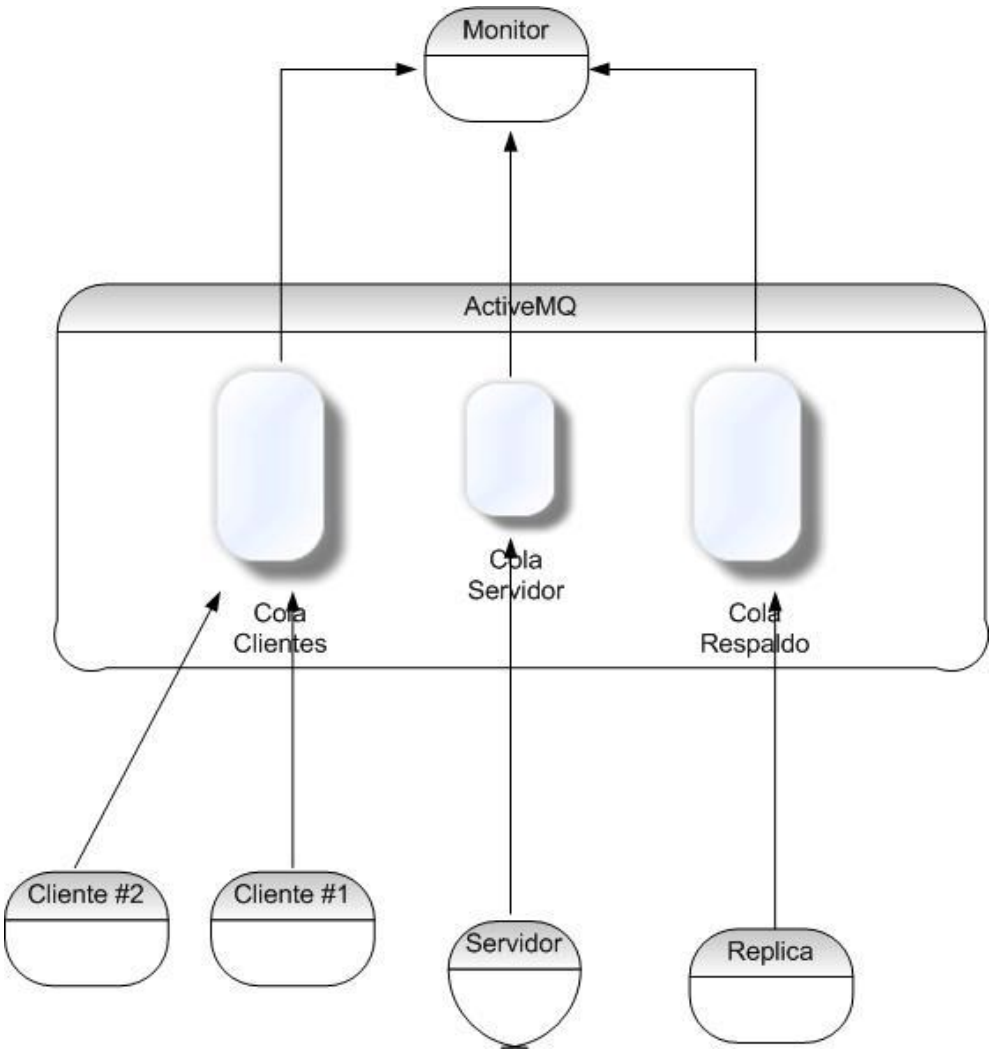

Figura 22 Diagrama de monitoreo

El sistema de monitoreo estará compuesto principalmente por una implementación del patrón **Observador** (40). Este patrón define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.

Este patrón también se conoce como el patrón de **publicación-inscripción** o **modelo-patrón**. Estos nombres sugieren las ideas básicas del patrón, que son bien sencillas: el objeto de datos, llamémoslo "Objeto", contiene atributos mediante los cuales cualquier objeto observador o vista se puede suscribir a él pasándole una referencia a sí mismo. El **Sujeto** mantiene así una lista de las referencias a sus observadores.

Los observadores a su vez están obligados a implementar unos métodos determinados mediante los cuales el **Sujeto** es capaz de notificar a sus observadores **suscritos** los cambios que sufre para que todos ellos tengan la oportunidad de refrescar el contenido representado. De manera que cuando se produce un cambio en el **Sujeto**, ejecutado, por ejemplo, por alguno de los observadores, el objeto de datos puede recorrer la lista de observadores avisando a cada uno (43).En Java se puede heredar la funcionalidad del patrón **Observador** de la clase **Observable**, implementando los observadores la interfaz **Observer** (44), lo que nos da una gran ventaja ya que este fue el lenguaje escogido para la implementación del producto.

Cada una de las partes implicadas contendrá un observador que notificará las actividades que van ocurriendo en ellas, de esta forma se puede mantener actualizado al **Monitor** y llevar un control real de las mismas. Al tener un conocimiento real de lo que está sucediendo en todo el proceso de respaldo, nos da la posibilidad de ubicar errores de implementación y una vez corregidos y depurados, monitorear el proceso en general.

3.2 Cooperación entre las aplicaciones

En este epígrafe se verá cómo cooperan las dos aplicaciones centrales para poner en funcionamiento el mecanismo de respaldo de datos propuesto anteriormente. El intercambio de información y la transferencia de datos en todo el proceso entre las aplicaciones se ha realizado mediante mensajería asíncrona (45), utilizando para ello el servidor ActiveMQ expuesto anteriormente en este capítulo y el anterior.

Para poder usar un servidor de mensajería y por tanto enviar mensajes a través de este, es necesaria la configuración de determinados objetos, y uno de estos son las colas de mensajes. Estas colas tienen el objetivo de almacenar los mensajes enviados por un emisor hasta que el servidor de mensajería haya entregado los mensajes al receptor correspondiente (45).

En la Figura 23 se muestra la configuración de las colas que se han configurado en el servidor de mensajería Apache ActiveMQ los cuales son utilizados para el intercambio de datos.

3.2.1 Descripción de las colas de mensajería

Existen dos tipos de dominio en la especificación de JMS, el punto-a-punto y el publicar/subscribir. El primero utiliza como destino una cola, que no es más que un almacén persistente. Las colas obedecen a la regla FIFO (First In First Out); es decir, los mensajes abandonan la cola en el mismo orden en que han sido enviados. Todos los mensajes son enviados a una cola específica, donde permanecen hasta que expiran o hasta que son consumidos por el receptor. El emisor no necesita tener conocimiento del receptor y el receptor no necesita saber dónde se ha producido el mensaje. Sin embargo, debido a la naturaleza precisa de la mensajería punto-a-punto, es común que ambas partes sean conocidas por la aplicación. La cola de mensajes siempre es el punto común de intercambio (32).

La segunda utiliza los tópicos, que también se puede tratar como un almacén persistente aunque no sea su mejor funcionamiento. Su ventaja principal es que puede tener múltiples recipientes por mensaje. Esto no es posible en un dominio punto-a-punto y esta es la principal diferencia entre los dos dominios. No existen colas en el dominio pub/sub, en su lugar los mensajes son publicados y enviados a apartados. Los apartados son similares a las colas excepto que múltiples consumidores pueden compartir un único mensaje. Al igual que las colas, los apartados son FIFO una vez que el consumidor se ha suscrito, igual que en el dominio punto-a-punto, varios productores pueden publicar mensajes para un único apartado. Los apartados retienen los mensajes durante tanto tiempo como tardan en distribuirlos a los consumidores. Los suscriptores de un dominio pub/sub son habitualmente, pero no estrictamente, anónimos, es decir son desconocidos para el que publica (32).

A continuación se describirán los dominios utilizados por las aplicaciones para el intercambio de información.

Tópicos:

- **All.Notification.Topic**: Apartado donde se envía la notificación de acceso al respaldo de datos.

Colas:

- **Public.Queue**: Cola donde los **Clientes** envían el pedido de respaldo y de la cual el **Servidor** se encuentra esperando por dicho pedido.
- **Transfer.Queue**: Cola donde el **Cliente** envía los datos a respaldar y de la cual el **Servidor** se encuentra escuchando para procesar la información que reciba.

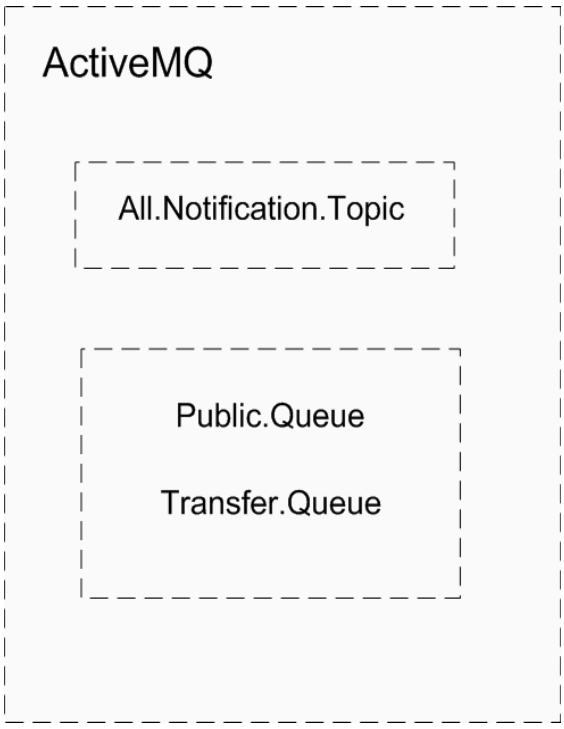

Figura 23 Descripción de las colas de mensajería

3.3 Beneficios de la mensajería

En la implementación del mecanismo propuesto se ha utilizado la mensajería como vía para intercambiar los datos entre las aplicaciones. La mensajería además de permitir a estas aplicaciones intercambiar información, resuelve otra serie de problemas derivados de las características del mecanismo propuesto. A continuación se verán algunas características del mecanismo y cómo son resueltas con la mensajería:

- **Comunicación remota:** La mensajería posibilita que aplicaciones separadas se puedan comunicar y transferir datos. No importa lo distante que estén una de otra, el sistema de mensajería siempre se asegura que los mensajes lleguen a su destino. También las aplicaciones se pueden comunicar con el sistema de mensajería haciendo uso de varios protocolos de comunicación, como, TCP, SSL, NIO, UDP, etc. Esto es muy importante puesto que los clientes y el servidor de respaldo pueden estar en lugares lejanos geográficamente.

- **Comunicación asíncrona:** La mensajería posibilita una estrategia enviar-y-olvidar a la comunicación. El emisor no tiene que esperar por el receptor a que reciba y procese el mensaje; incluso no tiene que esperar a que el sistema de mensajería entregue el mensaje. El

emisor solo tiene que esperar a que el mensaje sea enviado; a que el mensaje sea almacenado satisfactoriamente en el canal por el sistema de mensajería. Una vez que el mensaje es almacenado, el emisor es libre para hacer otra tarea mientras el mensaje es transmitido en segundo plano. El receptor puede querer enviar un acuse de recibo o resultado de vuelta al emisor, lo cual requiere otro mensaje. Debido a que los ficheros de respaldo pueden llegar a ser de un tamaño significativo este tipo de comunicación es muy valiosa porque así el cliente no tiene que esperar a que el servidor reciba los datos para continuar haciendo otras tareas.

- **Comunicación fiable:** La mensajería proporciona la entrega confiable que una llamada a procedimiento remoto (RPC) no puede. Esta razón está dada por el uso de la estrategia Almacena-y-Reenvía para transmitir mensajes. Los datos son empaquetados como mensajes, los cuales son unidades atómicas independientes. Cuando el emisor envía un mensaje, el sistema de mensajería lo almacena. El sistema entonces entrega el mensaje reenviándolo al receptor donde es almacenado nuevamente. Almacenar el mensaje en el emisor y en el receptor es asumido de una forma confiable (para hacerlo más confiable, el mensaje puede ser almacenado en disco en lugar de en la memoria). El envío directo entre los dos extremos implicados pierde confiabilidad debido a que el receptor o la red pueden no estar funcionando correctamente. El sistema de mensajería soluciona esto mediante el reenvío del mensaje hasta que sea exitoso. Esta forma automática de reintento posibilita al sistema de mensajería solucionar problemas con la red de tal forma que el emisor y el receptor no tengan que preocuparse sobre estos detalles. De esta forma se le da tanto al **Servidor** como al **Cliente** la seguridad de que una vez que envíen un mensaje pueden estar seguros que será entregado a su destino sin tener que preocuparse de ello.

- **Integración plataforma/lenguaje:** Cuando se conectan múltiples sistemas de computadoras vía comunicación remota, estos sistemas comúnmente usan diferentes lenguajes, tecnologías y plataformas, quizás porque fueron desarrollados a través del tiempo por diferentes equipos y situaciones. Integrar tales tipos de aplicaciones divergentes puede requerir una zona desmilitarizada de middleware para negociar entre las aplicaciones, a menudo usando el menor denominador común, tal como datos planos con formato oscuro. En estas circunstancias un sistema de mensajería puede ser un traductor universal entre las aplicaciones que trabajan cada una con su lenguaje y plataforma en sus propios términos y aun así permite que todas se comuniquen a través de un paradigma común de mensajería. Esto puede ser útil, si por alguna

razón se hace algún cliente en otro lenguaje entonces no se tendría que cambiar nada en el mecanismo.

- **Regulación de flujo:** Un problema con la llamada a procedimientos remotos es que demasiadas a un mismo receptor al mismo tiempo puede causar una sobrecarga en el receptor. Esto puede causar degradación en el rendimiento e incluso que el receptor deje de funcionar. La comunicación asíncrona posibilita al receptor controlar la velocidad a la cual consumir las peticiones, de tal forma que no se sobrecargue debido a demasiadas peticiones simultáneas. El efecto adverso de esto en los emisores es minimizado debido a la comunicación asíncrona, ya que el emisor no está bloqueado esperando por el receptor. Esto es deseable en los actualizadores que estarán recibiendo la actualización y podrán controlar la velocidad con que reciben los mensajes, que de otra forma podrían colapsar.

Conclusiones

El desarrollo de este capítulo ha permitido una mejor comprensión del mecanismo de respaldo que se propone en este trabajo y de cómo es viable su desarrollo como producto real. Se han construido dos aplicaciones centrales para implementar el mismo obteniéndose muy buenos resultados. Se ha implementado un **Cliente** para recopilar, compactar y enviar la información a respaldar con una interfaz gráfica amigable para su configuración y un **Servidor** para recibirla y ordenarla. También se ha mostrado la idea a seguir en cuanto al desarrollo de las aplicaciones **Monitor** y **Réplica**. Se ha plasmado cómo se comunican estas aplicaciones y cómo se han solucionado los diversos problemas que acarrean las características intrínsecas del mecanismo.

CONCLUSIONES

Como resultado del trabajo realizado se concibió la creación de un mecanismo de respaldo de datos. Dicho mecanismo puede ser integrado al plan de contingencia de la Universidad, previendo así la menor cantidad de pérdida de información en caso de catástrofe. Con la materialización del mismo, se contaría con un producto propio de nuestro país, completamente automatizado y con la documentación suficiente para darle el desarrollo y soporte necesario. Por lo que se puede arribar a la conclusión de que el objetivo principal de este trabajo se ha materializado con la realización de las tareas investigativas planteadas.

RECOMENDACIONES

- Aplicar los conocimientos adquiridos durante el transcurso de esta investigación en otros trabajos enmarcados en la misma línea investigativa.
- Estudiar la viabilidad de aplicar el mecanismo propuesto en otros proyectos de la Universidad.
- Agregar funcionalidad al mecanismo propuesto, así como perfeccionarlo y probarlo en ambientes reales para garantizar que la implementación cumpla con los parámetros de rendimiento deseados.

BIBLIOGRAFÍA

1. *HP Storage Mirroring.* **Sanchez, Andrés.** 2007.

2. Radio Cooperativa. [Online] Julio 12, 2006. [Cited: mayo 20, 2008.] http://www.cooperativa.cl/p4_noticias/antialone.html?page=http://www.cooperativa.cl/p4_noticias/site/artic/20060712/pags/20060712053348.html.

3. **Abad, Eduard.** *Borrmart,S.A.* [Online] 2005. [Cited: abril 28, 2008.] http://www.borrmart.es/articulo_redseguridad.php?id=1537.

4. **El Mundo.** *Arde el Windsor.* 2005.

5. **Hoy.** Trama Urbana. [Online] enero 13, 2000. [Cited: abril 15, 2008.] http://pdf.diariohoy.net/2000/01/13/pdf/u0405.pdf.

6. **Comisión Nacional de Energía (CNE).** *Nota informativa sobre el incidente eléctrico acaecido el 23 de julio en Barcelona.* Barcelona : s.n., 2007.

7. *Privacy Rights.* [Online] 2007. [Cited: mayo 8, 2008.] http://www.privacyrights.org.

8. *Enter 2.0.* [Online] noviembre 21, 2007. [Cited: mayo 20, 2008.] http://www.enter.com.co/enter2/ente2_tele/ente2_tele/ARTICULO-WEB-NOTA_INTERIOR_2-3710046.html.

9. Atentados del 11 de septiembre de 2001. *Wikipedia.* [Online] marzo 11, 2008. [Cited: mayo 20, 2008.] http://es.wikipedia.org/wiki/Atentados_del_11_de_septiembre_de_2001.

10. *DeepSpar.* [Online] 2006. [Cited: mayo 15, 2008.] http://www.deepspar.com/wp-data-loss.html.

11. **Definicion.org.** 2008.

12. Copia de Seguridad. *Wikipedia.* [Online] 2008. [Cited: mayo 15, 2008.] http://es.wikipedia.org/wiki/Copia_de_seguridad.

13. *SearchStorage.* [Online] marzo 7, 2007. [Cited: abril 20, 2008.] http://searchstorage.techtarget.com/sDefinition/0,,sid5_gci211633,00.html.

14. *Glossary.* [Online] agosto 23, 1996. [Cited: abril 15, 2008.] http://glossary.its.bldrdoc.gov/fs-1037/dir-004/_0509.htm.

15. *The Free Dictionary.* [Online] 2006. [Cited: mayo 20, 2008.] http://www.thefreedictionary.com/backup.

16. **Yurin, Maxim.** *The History of Backup.* [Online] http://www.backuphistory.com/.

17. Backup. *Wikipedia.* [Online] 2007. http://en.wikipedia.org/wiki/Backup.

18. Respaldo y recuperación de datos. [Online] julio 2000. http://www.ccee.edu.uy/ensenian/catcomp/material/respyrec.pdf.

19. Herramientas para respaldo. *Linea Pelle Magazine.* [Online] 2005. http://kill-9.debianchile.cl/misc/Material/publicacion.pdf.

20. List of backup software. *Wikipedia.* [Online] 2008. http://en.wikipedia.org/wiki/List_of_backup_software.

21. Veritas Software. *Wikipedia.* [Online] 2008. http://en.wikipedia.org/wiki/Veritas_Software.

22. *rsnapshot.* [Online] 2007. http://www.rsnapshot.org/.

23. Codus operandi. *Igmus.* [Online] febrero 14, 2006. http://igmus.org/code/.

24. storebackup. *Source Forge.* [Online] 2008. [Cited: marzo 15, 2008.] http://sourceforge.net/projects/storebackup.

25. *Amanda.* [Online] abril 1ro, 2008. [Cited: mayo 8, 2008.] http://www.amanda.org/.

26. afbackup. *Source Forge.* [Online] 2008. [Cited: marzo 15, 2008.] http://sourceforge.net/projects/afbackup/.

27. CD Backup. *Stefan´s Muempf Seiten.* [Online] 2008. http://www.muempf.de.

28. Flexbackup. *Edwin´s Page.* [Online] octubre 13, 2003. http://www.edwinh.org/flexbackup.

29. rdiff-backup. [Online] enero 3, 2008. http://www.nongnu.org/rdiff-backup.

30. *Skolelinux.* [Online]

31. *Bacula.* [Online] 2008. http://www.bacula.org.

32. *Programación Java Server con J2EE Edición 1.3.*

33. *ActiveMQ.* [Online] mayo 7, 2008. http://activemq.apache.org/.

34. **JOHNSON, R.** *Professional Java Development with the Spring Framework.* 2005.

35. Java Programming Language. *Java Sun.* [Online] 2005. http://java.sun.com/javase/6/docs/technotes/guides/language/index.html.

36. **W. Clay Richardson, Donald Avondolio, Scot Schrager.** *Professional Java JDK 6.*

37. Package javax.swing.filechooser. *Java Sun.* [Online] 2008. http://java.sun.com/j2se/1.4.2/docs/api/javax/swing/filechooser/package-summary.html.

38. Class TimerTask. *Java Sun.* [Online] http://java.sun.com/j2se/1.4.2/docs/api/java/util/TimerTask.html.

39. Class ZipOutputStream. *Java Sun.* [Online] http://java.sun.com/j2se/1.4.2/docs/api/java/util/zip/ZipOutputStream.html.

40. **Erich Gamma, Richard Helm, Ralph Johnson, Jhon Vlissides.** *Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : ISBN: 0-201-63361-2.

41. **Nicholas C. Zakas, Jeremy McPeak, Joe Fawcett.** *Professional Ajax.* 2007. ISBN: 978-0-471-77778-6.

42. Dojo. *Dojotoolkit.* [Online] 2008. http://dojotoolkit.org/.

43. **Harold, Elliote Rusty.** *Java I/O.* s.l. : O´Reilly, 1999. ISBN: 1-56592-485-1.

44. Interface Observer. *Java Sun.* [Online] http://java.sun.com/j2se/1.4.2/docs/api/java/util/Observer.html.

45. **Hohpe, G.** *Enterprise Integration Patterns.* 2003.